International Conference on Educational Data Mining (EDM) 2017
Proceedings of the 10th International Conference on Educational Data Mining
Xiangen Hu, Tiffany Barnes, Arnon Hershkovitz, Luc Paquette(eds)
Wuhan, China, June 25-28, 2017

Preface

The 10th International Conference on Educational Data Mining (EDM 2017) is held under the auspices of the International Educational Data Mining Society at the Optics Velley Kingdom Plaza Hotel, Wuhan, Hubei Province, in China. The conference, held June 25th - June 28th, 2017, follows the nine previous editions (Raleigh 2016, Madrid 2015, London 2014, Memphis 2013, Chania 2012, Eindhoven 2011, Pittsburgh 2010, Cordoba, 2009 and Montréal 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 two invited talks by: Dr. Jie Tang, Associate Professor with the Department of Computer Science and Technology at Tsinghua University; and Dr. Ron Cole, President of Boulder Learning Inc. Together with the Journal of Educational Data Mining (JEDM), the EDM 2017 conference supports a JEDM Track that provides researchers with 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; five papers have been accepted to the track and will be presented at the conference. The abstract for 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 122 submissions (71 full, 47 short, 4 industry). We accepted 18 full papers (25% acceptance rate) and 32 short papers for oral presentation (42% acceptance rate) and an additional 39 for poster presentations, 3 demonstrations. The industry track includes all 4 submitted industry papers and 1 paper initially submitted as a full paper.

The EDM conference provides opportunities for young researchers, and particularly Ph.D. 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 includes 3 workshops: Graph-based Educational Data Mining (G-EDM 2017); Sharing and Reusing Data & Analytics Methods with LearnSphere; Deep Learning with Educational Data, and 2 tutorials: Why Data Standards are Critical for EDM and AIED; and Principal Stratification for EDM Experiments.

We thank the sponsors of EDM 2017 for their generous support: 17Zuoye, Coursera, Learnta, and the Prof. Ram Kumar Memorial Foundation. 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.

Xiangen Hu	Tiffany Barnes	Arnon Hershkovitz	Luc Paquette
University of Memphis	North Carolina State	University of Tel Aviv	University of Illinois at
Central China Normal University	University	Program Chair	Urbana-Champaign
Conference Chair	Conference Chair		Program Chair

Organization

Conference Chairs

Xiangen Hu	University of Memphis, USA, Central China Normal University, China
Tiffany Barnes	North Carolina State University, USA

Program Chairs

Arnon Hershkovitz	Tel Aviv University, Israel
Luc Paquette	The University of Illinois at Urbana-Champaign, USA

Workshop and Tutorials Chairs

Ran Liu	Carnegie Mellon University, USA
Michael Eagle	Carnegie Mellon University, USA

Poster Chairs

Mariluz Guenaga	Deusto University, Spain
Pedro J. Muñoz-Merino	Universidad Carlos III de Madrid, Spain
Rinat Rosenberg-Kima	Tel Aviv University, Israel

Industry Track Chairs

Giora Alexandron	Center for Educational Technology, Israel
Qian Zhou	Tsinghua University, China

Doctoral Consortium Chair

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

JEDM Track Chair

Radek Pelanek	Masaryk University, Czech Republic
Agathe Merceron	Beuth University of Applied Sciences Berlin, Germany

Student Volunteers Chair

Ying Fang	University of Memphis, USA

Sponsorship Chair

Steve Ritter	Carnegie Learning, USA

Web Chairs

Alexandra Andres	Ateneo de Manila University, Philippines
Yun Tang	Central China Normal University, China

Local Arrangement Chair

Qinglin Cao	Central China Normal University, China

Social Media Chair

Sharon Hsiao	Arizona State University, USA

Proceedings Chair

Paul Salvador Inventado	Carnegie Mellon University, USA

Steering Committee / IEDMS Board of Directors

Mykola Pechenizkiy	Eindhoven University of Technology, Netherlands
Mingyu Feng	SRI International, USA
Rakesh Agrawal	Data Insights Laboratories, USA
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
Kalina Yacef	University of Sydney, Australia

Senior Program Committee

Vincent Aleven	Human-Computer Interaction Institute, Carnegie Mellon University, USA
Roger Azevedo	North Carolina State University, USA
Joseph Beck	Worcester Polytechnic Institute, USA
Matthew Berland	University of Wisconsin Madison, USA
Sidney D'Mello	University of Notre Dame, USA
Dragan Gasevic	University of Edinburgh, UK
Art Graesser	University of Memphis, USA
Kenneth Koedinger	Carnegie Mellon University, USA
Agathe Merceron	Beuth University of Applied Sciences Berlin, Germany
Ma. Mercedes T. Rodrigo	Department of Information Systems and Computer Science, Ateneo de Manila University, Philippines
Cristobal Romero	Department of Computer Sciences and Numerical Analysis, University of Cordoba, Spain
Carolyn Rose	Carnegie Mellon University, USA
George Siemens	UT Arlington, USA
John Stamper	Carnegie Mellon University, USA

Program Committee

Lalitha Agnihotri	McGraw Hill Education, USA
Esma Aimeur	University of Montreal, Canada
Carlos Alario-Hoyos	Universidad Carlos III de Madrid, Spain
Giora Alexandron	Massachusetts Institute of Technology, USA
Ma. Victoria Almeda	Teachers College Columbia University, USA
Aitor Almeida	DeustoTech Deusto Institute of Technology, Spain
Ivon Arroyo	Worcester Polytechnic Institute, USA
Juan I. Asensio-Prez	Collaborative and Intelligent Systems Group, University of Valladolid, Spain
Mirjam Augstein	Upper Austria University of Applied Sciences, Communication and Knowledge Media, Austria
Costin Badica	University of Craiova, Software Engineering Department, Romania
Gautam Biswas	Vanderbilt University, USA
Mary Jean Blink	TutorGen, Inc., USA
Nigel Bosch	University of Illinois Urbana-Champaign, USA
Miguel L. Bote-Lorenzo	Universidad de Valladolid, Spain
Jesus G. Boticario	UNED, Spain

Franois Bouchet	LIP6 - Universit Pierre et Marie Curie, France
Alex Bowers	Teachers College, Columbia University, USA
Kristy Elizabeth Boyer	University of Florida, USA
Javier Bravo Agapito	Universidad Autonoma de Madrid, Spain
Keith Brawner	United States Army Research Laboratory, USA
Manuel Caeiro Rodrguez	University of Vigo, Spain
Renza Campagni	Universit degli Studi di Firenze, Italy
Alberto Cano	Virginia Commonwealth University, USA
Zhongzhou Chen	MIT physics, USA
Min Chi	North Carolina State University, USA
Mihaela Cocea	School of Computing, University of Portsmouth, UK
Miguel ngel Conde	University of Len, Spain
Scott Crossley	Georgia State University, USA
Juan Cruz-Benito	GRIAL Research Group. University of Salamanca, Spain
Cynthia D'Angelo	SRI International, USA
Rosanna De Rosa	University of Naples Federico II, Italy
Hendrik Drachsler	Open University, The Netherlands
Bruno Emond	National Research Council Canada, Canada
Maka Eradze	Tallinn University, Estonia
Stephen Fancsali	Carnegie Learning, Inc., USA
Vladimir A. Fomichov	School of Business Informatics,National Research University Higher School of Economics, Russia
Davide Fossati	Emory University, ISA
Kobi Gal	Ben Gurion University, Israel
April Galyardt	University of Georgia, USA
Carlos Garca-Martnez	Computing and Numerical Analysis Dept. Univ. of Crdoba, Spain
Eva Gibaja	Department of Computer Science and Numerical Analysis, University of Cordoba, Spain
Daniela Godoy	ISISTAN Research Institute, Argentina
Ilya Goldin	2U, Inc., USA
Eduardo Gmez-Snchez	University of Valladolid, Spain
Yue Gong	CS department, Worcester Polytechnic Institute, USA
Jos Gonzlez-Brenes	Chegg, USA
Joseph Grafsgaard	North Carolina State University, USA
Mariluz Guenaga	Deusto Institute of Technology - University of Deusto, Spain
Philip Guo	UC San Diego, USA
Jiangang Hao	Educational Testing Service, USA
ngel Hernndez-Garca	Universidad Politcnica de Madrid, Spain
Davinia Hernandez-Leo	Universitat Pompeu Fabra, Barcelona, Spain
Andrew Hicks	North Carolina State University, USA
Sharon Hsiao	Arizona State University, USA
Stephen Hutt	University Of Notre Dame, USA
Seiji Isotani	University of Sao Paulo, Brazil
Vladimir Ivanevi	University of Novi Sad, Faculty of Technical Sciences, Serbia
Yang Jiang	Teachers College, Columbia University, USA
Jelena Jovanovic	University of Belgrade, Serbia
Mike Joy	University of Warwick, UK
Ralf Klamma	RWTH Aachen University, Germany
Irena Koprinska	The University of Sydney, Australia
Sotiris Kotsiantis	Univerisity of Patras, Greece
Sbastien Lall	University of British Columbia, Canada
Charles Lang	Teachers College, Columbia University, USA

Mikel Larraaga	University of the Basque Country, Spain
Sunbok Lee	MIT, USA
Young-Jin Lee	University of Kansas, USA
James Lester	North Carolina State University, USA
Innar Liiv	Tallinn University of Technology, Estonia
Ran Liu	Carnegie Mellon University, USA
Martn Llamas-Nistal	Universidad de Vigo, Spain
Vanda Luengo	Laboratoire d'informatique de Paris, LIP6, Universit Pierre et Marie Curie, France
Ivan Lukovi	University of Novi Sad, Faculty of Technical Sciences, Serbia
J. M. Luna	Dept. of Computer Science and Numerical Analysis, University of Cordoba, Spain
Mihai Lupu	Vienna University of Technology, Austria
Maria Luque	University of Cordoba, Spain
Collin Lynch	North Carolina State University, USA
Christopher Maclellan	Carnegie Mellon University, USA
Juan Jose Martins	University of Deusto, Spain
Noboru Matsuda	Texas A&M University, USA
Victor Menendez	Universidad Autnoma de Yucatn, Mexico
Donatella Merlini	Universit di Firenze, Italy
Cristian Mihaescu	University of Craiova, Romania
Piotr Mitros	edX/MIT, USA
Shirin Mojarad	McGraw Hill Education, USA
Carlos Monroy	Rice University, USA
Behrooz Mostafavi	North Carolina State University, USA
Bradford Mott	North Carolina State University, USA
Mario Muoz-Organero	Carlos III University of Madrid, Spain
Tristan Nixon	University of Memphis, USA
Roger Nkambou	Universit du Qubec Montral (UQAM), Canada
Cristian Olivares-Rodrguez	Universidad Andres Bello, Chile
Andrew Olney	University of Memphis, USA
Shai Olsher	University of Haifa, Israel
Abelardo Pardo	The University of Sydney, Australia
Zach Pardos	UC Berkeley, USA
Philip I. Pavlik Jr.	University of Memphis, USA
Radek Pelnek	Masaryk University Brno, Czech Republic
Niels Pinkwart	Humboldt-Universitt zu Berlin, Germany
Paul Stefan Popescu	University of Craiova, Faculty of Automation, Conputers an Electronics, Craiova, Romania
Thomas Price	North Carolina State University, USA
David Pritchard	MIT, USA
Arti Ramesh	University of Maryland, College Park, USA
Martina Rau	University of Wisconsin - Madison, Department of Educational Psychology, USA
Steven Ritter	Carnegie Learning, Inc., USA
Jos Ral Romero	University of Cordoba, Spain
Susana Romero	Universidad de Deusto, Spain
Salvador Ros	UNED, Spain
Yigal Rosen	Harvard University, USA
Adolfo Ruiz Calleja	Tallinn University, Estonia
Vasile Rus	The University of Memphis, USA
Shaghayegh Sahebi	University at Albany - SUNY, USA

Maria Ofelia San Pedro	Teachers College, Columbia University, USA
Olga C. Santos	aDeNu Research Group (UNED), Spain
Erica Snow	Arizona State University, USA
Angela Stewart	University of Notre Dame, USA
Jun-Ming Su	Department of Information and Learning Technology, National University of Tainan, Taiwan
Ling Tan	Australian Council for Educational Research, Australia
Mike Tissenbaum	Massachusetts Institute of Technology, USA
Stefan Trausan-Matu	University Politehnica of Bucharest, Romania
Peter Van Rosmalen	Open University, The Netherlands
Sebastin Ventura	Department of Computer Sciences and Numerical Analysis, University of Cordoba, Spain
Katrien Verbert	KU Leuven, Belgium
Lucian Vintan	"Lucian Blaga" University of Sibiu, Romania
Feng-Hsu Wang	Ming Chuan University, Taiwan
Stephan Weibelzahl	Private University of Applied Sciences Gttingen, Germany
Fridolin Wild	Oxford Brookes University, UK
Michael Yudelson	Carnegie Mellon University, USA
Amelia Zafra Gmez	Department of Computer Sciences and Numerical Analysis, University of Cordoba, Spain
Alfredo Zapata Gonzlez	Universidad Autonoma de Yucatan, Mexico
Diego Zapata-Rivera	Educational Testing Service, USA
Marta Zorrilla	University of Cantabria, Spain

Awards

Best papers and exemplary paper selection

The two program chairs selected 5 best paper nominees based on the reviews and meta-reviews for each of those paper. The nominees were then sent to the members of the best paper awards committee. Each committee member read and ranked each one of the nominees. Ranking was compiled and the best paper award was attributed to the most highly ranked paper. The best student paper award was attributed to the most highly ranked paper that was also authored by a student. The winner of the best paper award was not eligible to also win the best student paper award.

Best paper/best student papers committee:

Ryan Baker	Michel Desmarais	Zach Pardos
Cristobal Romero	Danielle McNamara	Didith Rodrigo

Award winners

Best paper	Efficient Feature Embeddings for Student Classification with Variational Auto-encoders *Severin Klingler, Rafael Wampfler, Tanja Kser, Barbara Solenthaler and Markus Gross*
Best student paper	Generalizability of Face-Based Mind Wandering Detection Across Task Contexts *Angela Stewart, Nigel Bosch and Sidney DMello*
Best paper nominees	Zone out no more: Mitigating mind wandering during computerized reading *Sidney D'Mello, Caitlin Mills, Robert Bixler and Nigel Bosch*
	Efficient Feature Embeddings for Student Classification with Variational Auto-encoders *Severin Klingler, Rafael Wampfler, Tanja Kser, Barbara Solenthaler and Markus Gross*
	Generalizability of Face-Based Mind Wandering Detection Across Task Contexts *Angela Stewart, Nigel Bosch and Sidney D'Mello*
	Towards Closing the Loop: Bridging Machine-induced Pedagogical Policies to Learning Theories *Guojing Zhou, Jianxun Wang, Collin Lynch and Min Chi*
	The Misidentified Identifiability Problem in Bayesian Knowledge Tracing *Shayan Doroudi and Emma Brunskill*
Best Poster Award	Identifying the relationships Between Students' Questions Type and Their Behavior *Fatima Harrak, Franois Bouchet and Vanda Luengo*

Table of Contents

Short Papers

Posters

Industry Track

Workshops

Invited Talks (abstracts)

Can AI help MOOCs?

Jie Tang
Department of Computer Science and Technology at Tsinghua University
jietang@tsinghua.edu.cn

ABSTRACT

Massive open online courses (MOOCs) boomed in recent years and have attracted millions of users worldwide. It is not only transforming higher education but also provides fodder for scientific research. In this talk, I am going to first introduce the major MOOC platforms in China, for example, XuetangX.com, a similar platform to Coursear and edX, is offering thousands of courses to more than 7,000,000 registered users. I will also introduce how we leverage AI technologies to help enhance student engagement on MOOCs.

The evolution of virtual tutors, clinician, and companions: A 20-year perspective on conversational agents in real-world applications

Ronald Cole
Boulder Learning Inc.
rcole@boulderlearning.com

ABSTRACT

The talk will present an overview of research projects initiated in 1997 and continue today in 2017, in which 3-D computer characters interact with children and adults with the aim of improving their language communication skills, educational achievement, and/or personal well-being. The talk examines how advances in human language and character animation technologies, and research leading to a deeper understanding of how to apply these technologies to optimize engagement and learning, led to positive experiences and learning outcomes similar to experienced teachers and clinicians,individuals from 5 to 80 years of age, The talk concludes with a consideration of how recent advances in machine learning algorithms, coupled with cloud-based delivery of automated assessment and instruction, delivered by virtual agents, can save teachers millions of hours of time annually, and provide EDM researchers with vast amounts of speech and language data that can be mined to improve students' learning experiences and outcomes.

JEDM Track Journal Papers (abstracts)

Identifiability of the Bayesian Knowledge Tracing Model

Junchen Feng
17zuoye.com
Greenland Center Tower B 16th Floor
Beijing China
junchen.feng@17zuoye.com

ABSTRACT

The three "unidentified" model specifications proposed by Beck and Chang (2007) are identified by the Bayesian Knowledge Tracing model with a non-informative Dirichlet prior distribution and an observed sequence that is longer than three periods. Although these specifications have the same observed learning curve, they generate different likelihood given the same data. The paper further shows that the observed learning curve is not the sufficient statistics of the data generating process stipulated by the Bayesian Knowledge Tracing model. Therefore, it cannot be used in parameter inference of the Bayesian Knowledge Tracing model.

RiPLE: Recommendation in Peer-Learning Environments Based on Knowledge Gaps and Interests

Hassan Khosravi
University of Queensland
h.khosravi@uq.edu.au

Kendra Cooper
Independent Scholar
kendra.m.cooper@gmail.com

Kirsty Kitto
University of Technology Sydney
kirsty.kitto@uts.edu.au

ABSTRACT

Various forms of Peer-Learning Environments are increasingly being used in post-secondary education, often to help build repositories of student generated learning objects. However, large classes can result in an extensive repository, which can make it more challenging for students to search for suitable objects that both reflect their interests and address their knowledge gaps. Recommender Systems for Technology Enhanced Learning (RecSysTEL) offer a potential solution to this problem by providing sophisticated filtering techniques to help students to find the resources that they need in a timely manner. Here, a new RecSysTEL for Recommendation in Peer-Learning Environments (RiPLE) is presented. The approach uses a collaborative filtering algorithm based upon matrix factorization to create personalized recommendations for individual students that address their interests and their current knowledge gaps. The approach is validated using both synthetic and real data sets. The results are promising, indicating RiPLE is able to provide sensible personalized recommendations for both regular and cold-start users under reasonable assumptions about parameters and user behavior.

Keywords

Peer-Learning Environments, Recommender Systems, Knowledge Gaps

Modeling Wheel-spinning and Productive Persistence in Skill Builders

Shimin Kai
Teachers College Columbia University,
525 W125th Street, New York, NY 10027
+1 212-678-3000
smk2184@tc.columbia.edu

Ma. Victoria Almeda
Teachers College Columbia University,
525 W125th Street, New York, NY 10027
+1 212-678-3000
mqa2000@tc.columbia.edu

Ryan S. Baker
Graduate School of Education, University of Pennsylvania
3700 Walnut St., Philadelphia, PA 19104
+1 877-736-6473
ryanshaunbaker@gmail. com

Nicole Shechtman
Center for Technology in Learning, SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
+1 650-859-2000
nicoleshechtman@sri.com

Cristina Heffernan
Worcester Polytechnic Institute
100 Institute Rd, Worcester, MA 01609
+1 508-831-5000
cristina.heffernan@gmail.com

Neil Heffernan
Worcester Polytechnic Institute
100 Institute Rd, MA 01609
+1 508-831-5000
nth@wpi.com

ABSTRACT

Research on non-cognitive factors has shown that persistence in the face of challenges plays an important role in learning. However, recent work on wheel-spinning, a type of unproductive persistence where students spend too much time struggling without achieving mastery of skills, show that not all persistence is uniformly beneficial for learning. For this reason, it becomes increasingly pertinent to identify the key differences between unproductive and productive persistence toward informing interventions in computer-based learning environments. In this study, we attempt to address this by using classification models to distinguish between productive persistence and wheel-spinning in ASSISTments, an online math learning platform. Our results indicate that wheel-spinning is associated with shorter delays between solving problems of the same skill, more attempts to answer problems, and the heavy use of bottom out hints except for the first problem. These findings suggest that encouraging students to engage in spaced practice and avoid over-using bottom-out hints is likely helpful to reduce their wheel-spinning and improve learning. These findings also provide insight on which students are struggling and how to make students' persistence more productive.

Modeling MOOC Student Behavior With Two-Layer Hidden Markov Models

Chase Geigle
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois, USA
geigle1@illinois.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois, USA
czhai@illinois.edu

ABSTRACT

Massive open online courses (MOOCs) provide educators with an abundance of data describing how students interact with the platform, but this data is highly underutilized today. This is in part due to the lack of sophisticated tools to provide interpretable and actionable summaries of huge amounts of MOOC activity present in log data. To address this problem, we propose a student behavior representation method alongside a method for automatically discovering those student behavior patterns by leveraging the click log data that can be obtained from the MOOC platform itself. Specifically, we propose the use of a two-layer hidden Markov model (2L-HMM) to extract our desired behavior representation, and show that patterns extracted by such a 2L-HMM are interpretable, meaningful, and unique. We demonstrate that features extracted from a trained 2L-HMM can be shown to correlate with educational outcomes.

Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant Number DGE-1144245.

Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains

Ran Liu
Human-Computer Interaction Institute
Carnegie Mellon University
ranliu@cmu.edu

Kenneth R. Koedinger
Human-Computer Interaction Institute
Carnegie Mellon University
koedinger@cmu.edu

ABSTRACT

As the use of educational technology becomes more ubiquitous, an enormous amount of learning process data is being produced. Educational data mining seeks to analyze and model these data, with the ultimate goal of improving learning outcomes. The most firmly grounded and rigorous evaluation of an educational data mining discovery is whether it yields better student learning when applied. Such an evaluation has been referred to as "closing the loop", as it completes cycle of system design, deployment, data analysis, and discovery leading back to design. Here, we present an instance of “closing the loop” on an automated cognitive modeling improvement discovered by Learning Factors Analysis (Cen, Koedinger, & Junker, 2006). We discuss our findings from a process in which we interpret the automated improvements yielded by the best-fitting cognitive model, validate the interpretation on novel data, use it to make changes to classroom-deployed educational technology, and show that the changes lead to significant learning gains relative to a control condition.

Full Papers

Zone out no more: Mitigating mind wandering during computerized reading

Sidney K. D'Mello, Caitlin Mills, Robert Bixler, & Nigel Bosch
University of Notre Dame
118 Haggar Hall
Notre Dame, IN 46556, USA
sdmello@nd.edu

ABSTRACT

Mind wandering, defined as shifts in attention from task-related processing to task-unrelated thoughts, is a ubiquitous phenomenon that has a negative influence on performance and productivity in many contexts, including learning. We propose that next-generation learning technologies should have some mechanism to detect and respond to mind wandering in real-time. Towards this end, we developed a technology that automatically detects mind wandering from eye-gaze during learning from instructional texts. When mind wandering is detected, the technology intervenes by posing just-in-time questions and encouraging re-reading as needed. After multiple rounds of iterative refinement, we summatively compared the technology to a yoked-control in an experiment with 104 participants. The key dependent variable was performance on a post-reading comprehension assessment. Our results suggest that the technology was successful in correcting comprehension deficits attributed to mind wandering (d = .47 sigma) under specific conditions, thereby highlighting the potential to improve learning by "attending to attention."

Keywords

Mind wandering; gaze tracking; student modeling; attention-aware.

1. INTRODUCTION

Despite our best efforts to write a clear and engaging paper, chances are high that within the next 10 pages you might fall prey to what is referred to as zoning out, daydreaming, or mind wandering [45]. Despite your best intention to concentrate on our paper, at some point your attention might drift away to unrelated thoughts of lunch, childcare, or an upcoming trip. This prediction is not based on some negative or cynical opinion of the reader/reviewer (we read and review papers too), but on what is known about attentional control, vigilance, and concentration while individuals are engaged in complex comprehension activities, such as reading for understanding.

One recent study tracked mind wandering of 5,000 individuals from 83 countries with a smartphone app that prompted people with thought-probes at random intervals throughout the day [24]. People reported mind wandering for 46.9% of the prompts, which confirmed lab studies on the pervasiveness of mind wandering (see [45] for a review). Mind wandering is more than merely incidental; a recent meta-analysis of 88 samples indicated a negative correlation between mind wandering and performance across a variety of tasks [34], a correlation which increases with task complexity. When compounded with its high frequency, mind wandering can have serious consequences on the performance and productivity of society at large.

Mind wandering is also unfortunately an under-addressed problem in education and is yet to be deeply studied in the context of learning with technology. Traditional learning technologies rely on the assumption that students are attending to the learning session, although this is not always the case. For example, it has been estimated that students mind wander approximately 40% of the time when engaging with online lectures [38], which are an important component of MOOCs. Some advanced technologies do aim to detect and respond to affective states like boredom, but evidence for their effectiveness is still equivocal (see [9] for a review). Further, boredom is related to but not the same as attention [12]. There are technologies that aim to prevent mind wandering by engendering a highly immersive learning experience and have achieved some success in this regard [40, 41]. But what is to be done when attentional focus inevitably wanes as the session progresses and the novelty of the system and content fades?

Our central thesis is that next-generation learning technologies should include mechanisms to model and respond to learners' attention in real-time [8]. Such attention-aware technologies can model various aspects of learner attention (e.g., divided attention, alternating attention). Here, we focus on detecting and mitigating mind wandering, a quintessential signal of waning engagement. We situate our work in the context of reading because reading is a common activity shared across multiple learning technologies, thereby increasing the generalizability of our results. Further, students mind wander approximately 30% of the time during computerized reading [44]. And although mind wandering can facilitate certain cognitive processes like future planning and divergent thinking [2, 28], it negatively correlates with comprehension and learning (reviewed in [31, 45]), suggesting that it is important to address mind wandering during learning.

Towards this end, we developed and validated a closed-loop attention-aware learning technology that combines a machine-learned mind wandering detector with a real-time interpolated testing and re-study intervention. Our attention-aware technology works as follows. Learners read a text on a computer screen using a self-paced screen-by-screen (also called page-by-page) reading paradigm. We track eye-gaze during reading using a remote eye tracker that does not restrict head movements. We focus on eye-gaze for mind wandering detection due to decades of research suggesting a tight coupling between attentional focus and eye movements during reading [36]. When mind wandering is detected, the system intervenes in an attempt to redirect attentional focus and correct any comprehension deficits that might arise due to mind wandering. The interventions consist of asking comprehension question on pages where mind wandering was detected and providing opportunities to re-read based on learners' responses. In this paper, we discuss the mind wandering

detector, intervention approach, and results of a summative evaluation study[1].

1.1 Related Work

The idea of attention-aware user interfaces is not new, but was proposed almost a decade ago by Roda and Thomas [39]. There was even an article on futuristic applications of attention-aware systems in educational contexts [35]. Prior to this, Gluck, et al. [15] discussed the use of eye tracking to increase the bandwidth of information available to an intelligent tutoring system (ITS). Similarly, Anderson [1] followed up on some of these ideas by demonstrating how particular beneficial instructional strategies could only be launched via a real-time analysis of eye gaze.

Most of the recent work has been on leveraging eye gaze to increase the bandwidth of learner models [22, 23, 29]. Conati, et al. [5] provide an excellent review of much of the existing work in this area. We can group the research into three categories: (1) offline-analyses of eye gaze to study attentional processes, (2) computational modeling of attentional states, and (3) closed-loop systems that respond to attention in real-time. Offline-analysis of eye movements has received considerable attention in cognitive and educational psychology for several decades [e.g., 16, 19], so this area of research is relatively healthy. Online computational models of learner attention are just beginning to emerge [e.g., 6, 11], while closed-loop attention-aware systems are few and far between (see [7, 15, 42, 48] for a more or less exhaustive list). Two known examples, GazeTutor and AttentiveReview, are discussed below.

GazeTutor [7] is a learning technology for biology. It has an animated conversational agent that provides spoken explanations on biology topics which are synchronized with images. The system uses a Tobii T60 eye tracker to detect inattention, which is assumed to occur when learners' gaze is not on the tutor agent or image for at least five consecutive seconds. When this occurs, the system interrupts its speech mid utterance, directs learners to reorient their attention (e.g., "I'm over here you know"), and repeats speaking from the start of the current utterance. In an evaluation study, 48 learners (undergraduate students) completed a learning session on four biology topics with the attention-aware components enabled (experimental group) or disabled (control group). The results indicated that GazeTutor was successful in dynamically reorienting learners' attentional patterns towards the interface. Importantly, learning gains for deep reasoning questions were significantly higher for the experimental vs. control group, but only for high aptitude learners. The results suggest that even the most basic attention-aware technology can be effective in improving learning, at least for a subset of learners. However, a key limitation is that the researchers simply assumed that off-screen gaze corresponded to inattention, but did not test this assumption (e.g., students could have been concentrating with their eyes closed and this would have been perceived as being inattentive).

AttentiveReview [32] is a closed-loop system for MOOC learning on mobile phones. The system uses video-based photoplethysmography (PPG) to detect a learners' heart rate from the back camera of a smartphone while they view MOOC-like lectures on the phone. AttentiveReview ranks the lectures based on its estimates of learners' "perceived difficulty," selecting the most difficult lecture for subsequent review (called adaptive review). In a 32-participant between-subjects evaluation study, the authors found that learning gains obtained from the adaptive review condition were statistically on par with a full review condition, but were achieved in 66.7% less review time. Although this result suggests that AttentiveReview increased learning efficiency, there is the question as to whether the system should even be considered to be an "attention-aware" technology. This is because it is arguable if the system has anything to do with attention (except for "attention" appearing in its name) as it selects items for review based on a model of "perceived difficulty" and not on learners' "attentional state." The two might be related, but are clearly not the same.

1.2 Novelty

Our paper focuses on closing the loop between research on educational data and learning outcomes by developing and validating the first (in our view) real-time learning technology that detects and mitigates mind wandering during computerized reading. Although automated detection of complex mental states with the goal of developing intelligent learning technologies that respond to the sensed states is an active research area (see reviews by [9, 18]), mind wandering has rarely been explored as an aspect of a learner's mental state that warrants detection and corrective action. And while there has been some work on modeling the locus of learner attention (see review by [5]), mind wandering is inherently different than more commonly studied forms of attention (e.g., selective attention, distraction), because it involves more covert forms of involuntary attentional lapses spawned by self-generated internal thought [45]. Simply put, mind wandering is a form of "looking without seeing" because the eyes might be fixated on the appropriate external stimulus, but very little is being processed as the mind is consumed by stimulus-independent internal thoughts. *Offline* automated approaches to detect mind wandering have been developed (e.g., [3, 11, 27, 33]), but these detectors have not yet been used to trigger *online* interventions. Here, we adapt an offline gaze-based automated mind wandering detector [13] to trigger real-time interventions to address mind wandering during reading. We conduct a randomized control trial to evaluate the efficacy of our attention-aware learning technology in improving learning.

2. MIND WANDERING DETECTION

We adopted a supervised learning approach for mind wandering detection. Below we provide a high-level overview of the approach; readers are directed to [3, 13] for a detailed discussion of the general approach used to build gaze-based detectors of mind wandering.

2.1 Training Data

We obtained training data from a previous study [26] that involved 98 undergraduate students reading a 57-page text on the surface tension of liquids [4] on a computer screen for an average of 28 minutes. The text contained around 6500 words, with an average of 115 words per page, and was displayed on a computer screen with Courier New typeface. We recorded eye-gaze with a Tobii TX300 eye tracker set to a sampling frequency of 120 Hz.

[1] This paper reports updated results of an earlier version [10] presented as a "Late-Breaking Work" (LBW) poster at the *2016 ACM CHI conference*. LBW "Extended Abstracts" are not included in the main conference proceedings and copyright is retained by the authors.

Participants could read normally and were free to move or gesture as they pleased.

Participants were instructed to report mind wandering (during reading) by pressing a predetermined key when they found themselves "thinking about the task itself but *not the actual content of the text*" or when they were "thinking about *anything else besides the task*." This is consistent with contemporary approaches (see [45]) that rely on self-reporting because mind wandering is an internal conscious phenomena. Further, self-reports of mind wandering have been linked to predictable patterns in physiology [43], pupillometry [14], eye-gaze [37], and task performance [34], providing validity for this approach.

On average, we received mind wandering reports for 32% of the pages (*SD* = 20%), although there was considerable variability among participants (ranging from 0% to 82%). Self-reported mind wandering negatively correlated (r = -.23, p < .05) with scores on a subsequent comprehension assessment [26], which provides evidence for the predictive validity of the self-reports.

2.2 Model Building

The stream of eye-gaze data was filtered to produce a series of fixations, saccades, and blinks, from which *global eye gaze* features were extracted (see Figure 1). Global features are independent of the words being read and are therefore more generalizable than so-called local features. A full list of 62 global features along with detailed descriptions is provided in [13], but briefly the features can be grouped into the following four categories: (1) Eye movement descriptive features (n = 48) were statistical functionals (e.g., min, median) for fixation duration, saccade duration, saccade amplitude, saccade velocity, and relative and absolute saccade angle distributions; (2) Pupil diameter descriptive features were statistical functionals (n = 8) computed from participant-level z-score standardized estimates of pupil diameter; (3) Blink features (n = 2) consisted of the number of blinks and the mean blink duration; (4) Miscellaneous gaze features (n = 4) consisted of the number of saccades, horizontal saccade proportion, fixation dispersion, and the fixation duration/saccade duration ratio. We proceeded with a subset of 32 features after eliminating features exhibiting multicollinearity.

Features were calculated from only a certain amount of gaze data from each page, called the *window*. The end of the window was positioned 3 seconds before a self-report so as to not overlap with the key-press. The average amount of time between self-reports and the beginning of the page was 16 seconds. We used this time point as the end of the window for pages with no self-report. Pages that were shorter than the target window size were discarded, as were pages with windows that contained fewer than five gaze fixations as there was insufficient data to compute some of the features. There were a total of 4,225 windows with sufficient data for supervised classification.

We experimented with a number of supervised classifiers on window sizes of 4, 8, and 12 seconds to discriminate positive (pages with a self-report = 32%) from negative (pages without a self-report) instances of mind wandering. The training data were downsampled to achieve a 50% base rate; testing data were unaltered. A leave-one-participant-out validation approach was adopted where models were built on data from *n-1* participants and evaluated on the held-out participant. The process was repeated for all participants. Model validation was conducted in a way to simulate a real-time system by analyzing data from every page. When classification was not possible due to a lack of valid gaze data and/or because participants did not spend enough time on the page, we classified the page as a positive instance of mind wandering. This was done because analyses indicated that participants were more likely to be mind wandering in those cases (but see [13] for alternate strategies to handle missing instances).

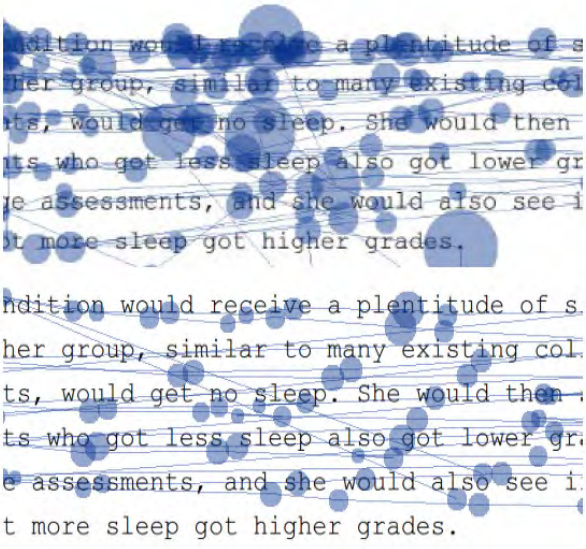


Figure 1: Gaze fixations during mind wandering (top) and normal reading (bottom)

2.3 Detector Accuracy

The best model was a support vector machine that used global features and operated on a window size of 8-seconds. The area under the ROC curve (AUC or AUROC or A') was .66, which exceeds the 0.5 chance threshold [17].

We assigned each instance as mind wandering or not mind wandering based on whether the detector's predicted likelihood of mind wandering (ranges from 0 to 1) was below or above 0.5 We adopted the default 0.5 threshold as it led to a higher rate of true positives while maintaining a moderate rate of true negatives. This resulted in the following confusion matrix shown in Table 1. The model had a weighted precision of 72.2% and a weighted recall of 67.4%, which we deemed to be sufficiently accurate for intervention.

Table 1: Proportionalized confusion matrix for mind wandering detection

	Predicted mind wandering (MW)	
Actual MW	*yes*	*no*
yes	0.715 (hit)	0.285 (miss)
no	0.346 (false positive)	0.654 (correct rejection)

3. Intervention to Address Mind Wandering

Our intervention approach is grounded in the basic idea that learning of conceptual information involves creating and maintaining an internal model (*mental model*) by integrating information from the text with prior knowledge from memory [25]. This integration process relies on attentional focus and breaks down during mind wandering because information from the external environment is no longer being integrated into the internal mental model. This results in an impaired model which leads to less effective suppression of off-task thoughts. This increase in mind wandering further impairs the mental model,

resulting in a vicious cycle. Our intervention targets this vicious cycle by redirecting attention to the primary task and attempting to correct for comprehension deficits attributed to mind wandering. Based on research demonstrating the effectiveness of interpolated testing [47], we propose that asking questions on pages where mind wandering is detected and encouraging re-reading in response to incorrect responses will aid in re-directing attention to the text and correct knowledge deficits.

3.1 Intervention Implementation

Our initial intervention was implemented for the same text used to create the mind wandering detector (although it could be applied to any text). The text was integrated into the computer reading interface. Mind wandering detection occurred when the learner navigated to the next page using the right arrow key. In order to address ambiguity in mind wandering detection, we used the detector's mind wandering likelihood to probabilistically determine when to intervene. For example, if the mind wandering likelihood was 70%, then there was a 70% chance of intervention on any given page (all else being equal). We did not intervene for the first three pages in order to allow the learner to become familiar with the text and interface. To reduce disruption, there was a 50% reduced probability of intervening on adjacent pages, and the maximum number of interventions was capped at 1/3 × the number of pages (19 for the present 57-page text). Table 2 presents pseudo code for when to launch an intervention.

Table 2: Pseudo code for intervention strategy

```
launch_intervention:
    if current_page >= WAITPAGES
    and
        total_interventions < MAXINTRV)
    and
        gaze_likelihood > random(0,1)
    and
        (!has_intervened(previous_page)
        or 0.5 < random (0,1)):
            do_intervention()
    else:
        show_next_page()

do_intervention:
    answer1 = show_question1()
    if answer1 is correct:
        show_positive_feedback()
        show_next_page()
    else:
        show_neg_feedback()
        suggest_rereading()
        if page advance detected:
            answer2 = show_question2();
            show_next_page()
```

Figure 2 presents an outline of the intervention strategy. The intervention itself relied on two multiple choice questions for each page (screen) of the text. When the system decided to intervene, one of the questions (randomly selected) was presented to the learner. If the learner answered this *online question* correctly, positive feedback was provided, and the learner could advance to the next page. If the learner answered incorrectly, negative feedback was provided, and the system encouraged the learner to re-read the page. The learner was then provided with a second (randomly selected) online question, which could either be the same or the alternate question for that page. Feedback was not provided and the learner was allowed to advance to the next page regardless of whether the second question was answered correctly, so as not to be overly burdensome.

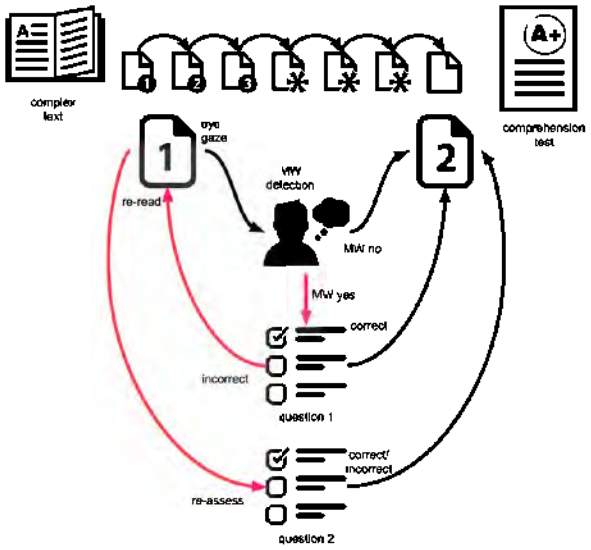


Figure 2: Outline of intervention strategy

3.2 Iterative Refinement

The technology was refined through multiple rounds of formative testing with 67 participants, recruited from the same institution used to build the detector. Participants were observed while interacting with the technology, their responses were analyzed, and they were interviewed about their experience. We used the feedback gleaned from these tests to refine the intervention parameters (i.e., when to launch, how many interventions to launch, whether to launch interventions on subsequent pages), intervention questions themselves, and instructions on how to attend to the intervention. For example, earlier versions of the intervention used a fixed threshold (instead of the aforementioned probabilistic approach) to trigger an intervention. Despite many attempts to set this threshold, the end result was that some participants received many interventions while others received almost no interventions. This issue was corrected by probabilistically rather than deterministically launching the intervention. Additional testing/refinement of the comprehension questions used in the intervention was done using crowdsourcing platforms, specifically Amazon's Mechanical Turk (MTurk).

4. Evaluation Study

We conducted a randomized controlled trial to evaluate the technology. The experiment had two conditions: an intervention condition and a yoked control condition (as described below). The yoked control was needed to verify that any learning benefits are attributed to the technology being sensitive to mind wandering and not merely to the added opportunities to answer online questions and re-read. This is because we know that interpolated testing itself has beneficial comprehension effects [47].

4.1 Method

Participants (N = 104) were a new set of undergraduate students who participated to fulfill research credit requirements. They were recruited from the same university used to build the MW detector and for the iterative testing and refinement cycles.

We did not use a pretest because we expected participants to be unfamiliar with the topic. Participants were not informed that the interface would be tracking their mind wandering (until the

debriefing at the end), Instead, they were instructed as follows: "While reading the text, you will occasionally be asked some questions about the page you just read. Depending on your answer, you will re-read the same page and you will be asked another question that may or may not be the same question."

Participants in the intervention condition received the intervention as described above (i.e., based on detected mind wandering likelihoods). Each participant in the yoked control condition was *paired* with a participant in the intervention condition. He or she received an intervention question on the same pages as their paired intervention participant regardless of mind wandering likelihood. For example, if participant A (i.e., intervention condition) received questions on pages 5, 7, 10, and 25, participant B (i.e., yoked control condition) would receive intervention questions on the same pages. However, if the yoked participant answered incorrectly, then (s)he had the opportunity to re-read and answer another question regardless of the outcome of their intervention-condition partner.

After reading, participants completed a 38-item multiple choice comprehension assessment to measure learning. The questions were randomly selected from the 57 pages (one per page) with the exception that a higher selection priority was given to pages that were re-read on account of the intervention. Participants in the yoked control condition received the same posttest questions as their intervention condition counterparts.

4.2 Results

Participants received an average of 16 (min of 7 and max of 19) interventions. They spent an average of 27.5 seconds on each screen prior to receiving an intervention. There was no significant difference across conditions (p = .998), suggesting that reading time was not a confound. In what follows, we compared each intervention participant to his/her yoked control with a two-tailed paired-samples t-test and a 0.05 criteria for statistical significance.

Mind wandering detection. The detector's likelihood of mind wandering was slightly higher for participants in the yoked-control condition (M = .431; SD = .170) compared to the intervention condition (M = .404; SD = .112), but the difference was not statistically significant (p = .348). This was unsurprising as participants in both groups received the same interventions, which itself was expected to reduce mind wandering. Importantly, mind wandering likelihoods were negatively correlated with performance on the online questions (r = -.296, p = .033) as well as on posttest questions (r = -.319, p = .021). This provides evidence for the validity of the mind wandering detector when applied to a new set of learners and under different conditions (i.e., reading interspersed with online questions compared to uninterrupted reading).

Comprehension assessment. There was some overlap between the online questions and the posttest questions. To obtain an unbiased estimate of learning, we only analyzed performance on previously unseen posttest questions. That is, questions that were used as part of the intervention were first removed before computing posttest scores.

There were no significant condition differences on overall posttest scores (p = .846). The intervention condition answered 57.6% (SD = .157) of the questions correctly while the yoked control condition answered 58.1% (SD = .129) correctly. This finding was not surprising as both conditions received the exact same treatment except that the interventions were triggered based on detected mind wandering in the intervention condition but not the control condition.

Next, we examined posttest performance as a function of mind wandering during reading. Each page was designated as a low or high mind wandering page based on a median split of mind wandering likelihoods (medians = .35 and .36 on a 0 to 1 scale for intervention and control conditions, respectively). We then analyzed performance on posttest questions corresponding to pages with low vs. high likelihoods of mind wandering (during reading). The results are shown in Table 3.

We found no significant posttest differences on pages where both the intervention and control participants had low (p = .759) or high (p = .922) mind wandering likelihoods (first and last rows in Table 3, respectively). There was also no significant posttest difference (p = .630) for pages where the intervention condition had high mind wandering likelihoods but the control condition had low mind wandering likelihoods (row 3). However, the intervention condition significantly (p = .003, d = .47 sigma) outperformed the control condition for pages where the intervention participants had low likelihoods of mind wandering but control participants had high mind wandering likelihoods (row 2). These last two finding suggests that the intervention had the intended effect of reducing comprehension deficits attributable to mind wandering because it led to equitable performance when mind wandering was high and improved performance when it was low.

Table 3: Posttest performance (proportion of correct responses) as a function of mind wandering during reading. Standard deviations in parenthesis.

	Mind wandering		**Posttest scores**	
N	*Int.*	*Cntrl.*	*Int.*	*Cntrl.*
43	Low	Low	.604 (.288)	.623 (.287)
40	**Low**	**High**	**.643 (.263)**	**.489 (.298)**
43	High	Low	.535 (.295)	.566 (.305)
45	High	High	.522 (.312)	.515 (.291)

Note. Int. = intervention. Cntrl. = control. Bolded cells represent a statistically significant difference. N = number of pairs (out of 52) in each analysis. It differs slightly across analyses as not all participants were assigned to each mind wandering group.

After-task interview. We interviewed a subset of the participants in order to gauge their subjective experience with the intervention. A few key themes emerged. Participants reported paying closer attention to the text after realizing they would be periodically answering multiple-choice questions. This was good. However, participants also reported that they adapted their reading strategies in one of two ways in response to the questions. Since the questions targeted factual information (sometimes verbatim) from the text, some participants paid more attention to details and precise wordings instead of the broader concepts being discussed in the text. More discouragingly, some participants reported adopting a preemptive skimming strategy in that they would only look for keywords that they expected to appear in a subsequent question.

Participants were encouraged to re-read text when they answered incorrectly before receiving another question (or the same question in some cases). Many participants reported simply scanning the text (when re-reading) to locate keywords from the question before moving on. Since the scanning strategy was often

successful to answer the subsequent question, participants reported that the questions were too easy and it took relatively little effort to locate the correct answer compared to re-reading. They suggested that it may have been better if the questions had targeted key concepts rather than facts.

Finally, participants reported difficulties with re-engaging with the text after answering an online question because the text was cleared when an intervention question was displayed; an item that can be easily corrected in subsequent versions.

5. Discussion

We developed the first educational technology capable of real-time mind wandering detection and dynamic intervention during computerized reading. In the remainder of this section, we discuss the significance of our main findings, limitations, and avenues for future work.

5.1 Significance of Main Findings

We have three main findings. First, we demonstrated that a machine-learned mind wandering detector built in one context can be applied to a different (albeit related) interaction context. Specifically, the detector was trained on a data set involving participants silently reading and self-reporting mind wandering, but was applied to an interactive context involving interpolated assessments, which engendered different reading strategies. Further, self-reports of mind wandering were *not* collected in this interactive context, which might have influenced mind wandering rates in and of itself. Despite these differences, we were able to demonstrate the predictive validity of the detector by showing that it negatively correlated with both online and offline comprehension scores when evaluated on new participants.

Second, we showed promising effects for our intervention approach despite a very conservative experimental design, which ensured that the intervention and control groups were equated along all respects, except that the intervention was triggered based on the mind wandering detector (key manipulation). Further, we used a probabilistic approach to trigger an intervention, because the detector is inherently imperfect. As a result, participants could have received an intervention when they were not mind wandering and/or could have failed to receive one when they were mind wandering. Therefore, it was essential to compare the two groups under conditions when the mind wandering levels differed. This more nuanced analysis revealed that although the intervention itself did not lead to a boost in overall comprehension (because it is remedial), it equated comprehension scores when mind wandering was high (i.e., scores for the intervention group were comparable when the control group was low on mind wandering). It also demonstrated the cost of not intervening during mind wandering (i.e., scores for the intervention group were greater when the control group was high on mind wandering). In other words, the intervention was successful in mitigating the negative effects of mind wandering.

Third, despite the advantages articulated above, the intervention itself was reactive and engendered several unintended (and presumably suboptimal) behaviors. In particular, students altered their reading strategies in response to the interpolated questions, which were a critical part of the intervention. In a sense, they attempted to "game the intervention" by attempting to proactively predict the types of questions they might receive and then adopting a complementary reading strategy consisting of skimming and/or focusing on factual information. This reliance on surface- rather than deeper-levels of processing was incongruent with our goal of promoting deep comprehension.

5.2 Limitations

There are a number of methodological limitations with this work that go beyond limitations with the intervention (as discussed above). First, we focused on a single text that is perceived as being quite dull and consequently triggers rather high levels of mind wandering [26]. This raises the question of whether the detector will generalize to different texts. We expect some level of generalizability in terms of features used because the detector only used content- and position- (on the screen) free global gaze features. However, given that several supervised classifiers are very sensitive to differences in base rates, the detector might over- or under- predict mind wandering when applied to texts that engender different rates of mind wandering. Therefore, retraining the detector with a more diverse set of texts is warranted.

Another limitation is the scalability of our learning technology. The eye tracker we used was a cost-prohibitive Tobii TX300 that will not scale beyond the laboratory. Fortunately, commercial-off-the-shelf (COTS) eye trackers, such as Eye Tribe and Tobii EyeX, can be used to surpass this limitation. It is an open question as to whether the mind wandering detector can operate with similar fidelity with these COTS eye trackers. Our use of global gaze features which do not require high-precision eye tracking holds considerable promise in this regard. Nevertheless, replication with scalable eye trackers and/or scalable alternatives to eye tracking (e.g., facial-feature tracking [46] or monitoring reading patterns [27]) is an important next step (see Section 5.3).

Our use of surface-level questions for both the intervention and the subsequent comprehension assessment is also a limitation as is the lack of a delayed comprehension assessment. It might be the case that the intervention effects manifest as richer encodings in long-term memory, a possibility that cannot be addressed in the current experiment that only assessed immediate learning.

Other limitations include a limited student sample (i.e. undergraduates from a private Midwestern college) and a laboratory setup. It is possible that the results would not generalize to a more diverse student population or in more ecological environments (but see below for evidence of generalizability of the detector in classroom environments). Replication with data from more diverse populations and environments would be a necessary next step to increase the ecological validity of this work.

5.3 Future Work

Our future work is progressing along two main fronts. One is to address limitations in the intervention and design of the experimental evaluation as discussed above. Accordingly, we are exploring alternative intervention strategies, such as: (a) tagging items for future re-study rather than interrupting participants during reading; (b) highlighting specific portions of the text as an overt cue to facilitate comprehension of critical information; (c) asking fewer intervention questions, but selecting inference questions that target deeper levels of comprehension and that span multiple pages of the text; and (d) asking learners to engage in reflection by providing written self-explanations of the textual content. We are currently evaluating one such redesigned intervention – open-ended questions targeting deeper levels of comprehension (item c). Our revised experimental design taps both surface- and inference-level comprehension and assesses comprehension immediately after reading (to measure learning) and after a one-week delay (to measure retention).

We are also developing attention-aware versions of more interactive interfaces, such as learning with an intelligent tutoring

system called GuruTutor [30]. This project also addresses some of the scalability concerns by replacing expensive research-grade eye tracking with cost-effective COTS eye tracking (e.g., the Eye Tribe or Tobii EyeX) and provides evidence for real-world generalizability by collecting data in classrooms rather than the lab. We recently tested our implementation on 135 students (total) in a noisy computer-enabled high-school classroom where eye-gaze of entire classes of students was collected during their normal class periods [20]. Using a similar approach to the present work, we used the data to build and validate a student-independent gaze-based mind wandering detector. The resultant mind wandering detection accuracy (F_1 of 0.59) was substantially greater than chance (F_1 of 0.24) and outperformed earlier work on the same domain [21]. The next step is to develop interventions that redirect attention and correct learning deficiencies attributable to mind wandering and to test the interventions in real-world environments. By doing so, we hope to advance our foundational vision of developing next-generation technologies that enhance the process and products of learning by "attending to attention."

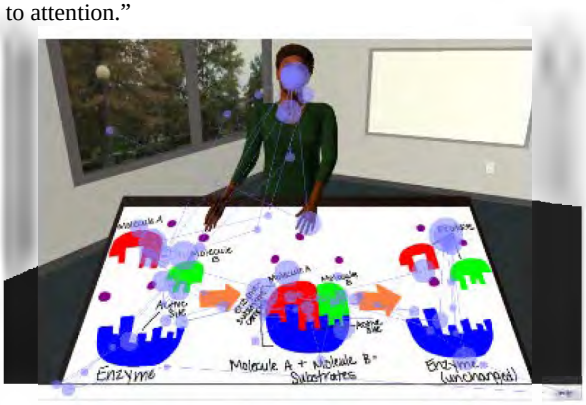


Figure 3: Guru Tutor interface overlaid with eye-gaze obtained via the EyeTribe

6. Acknowledgements

This research was supported by the National Science Foundation (NSF) (DRL 1235958 and IIS 1523091). The authors are grateful to Kris Kopp and Jenny Wu for their contributions to the study. Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

7. REFERENCES

[1] Anderson, J.R. 2002. Spanning seven orders of magnitude: A challenge for cognitive modeling. *Cognitive Science*, *26* (1), 85-112.

[2] Baird, B., Smallwood, J., Mrazek, M.D., Kam, J.W., Franklin, M.S. and Schooler, J.W. 2012. Inspired by distraction mind wandering facilitates creative incubation. *Psychological Science*, *23* (10), 1117-1122.

[3] Bixler, R. and D'Mello, S.K. 2016. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling & User-Adapted Interaction*, *26*, 33-68.

[4] Boys, C.V. 1895. *Soap bubbles, their colours and the forces which mold them.* Society for Promoting Christian Knowledge.

[5] Conati, C., Aleven, V. and Mitrovic, A. 2013. Eye-Tracking for Student Modelling in Intelligent Tutoring Systems. In Sottilare, R., Graesser, A., Hu, X. and Holden, H. eds. *Design Recommendations for Intelligent Tutoring Systems - Volume 1: Learner Modeling*, Army Research Laboratory, Orlando, FL.

[6] Conati, C. and Merten, C. 2007. Eye-tracking for user modeling in exploratory learning environments: An empirical evaluation. *Knowledge-Based Systems*, *20* (6), 557-574.

[7] D'Mello, S., Olney, A., Williams, C. and Hays, P. 2012. Gaze tutor: A gaze-reactive intelligent tutoring system. *International Journal of human-computer studies*, *70* (5), 377-398.

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

[9] D'Mello, S.K., Blanchard, N., Baker, R., Ocumpaugh, J. and Brawner, K. 2014. I feel your pain: A selective review of affect-sensitive instructional strategies. In Sottilare, R., Graesser, A., Hu, X. and Goldberg, B. eds. *Design Recommendations for Adaptive Intelligent Tutoring Systems: Adaptive Instructional Strategies (Volume 2)*, US Army Research Laboratory, Orlando, FL.

[10] D'Mello, S.K., Kopp, K., Bixler, R. and Bosch, N. 2016. Attending to attention: Detecting and combating mind wandering during computerized reading In *Extended Abstracts of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2016)*, ACM, New York.

[11] Drummond, J. and Litman, D. 2010. In the zone: Towards Detecting student zoning out using supervised machine learning. In Aleven, V., Kay, J. and Mostow, J. eds. *Intelligent Tutoring Systems.*, Springer-Verlag, Berlin / Heidelberg.

[12] Eastwood, J.D., Frischen, A., Fenske, M.J. and Smilek, D. 2012. The unengaged mind: Defining boredom in terms of attention. *Perspectives on Psychological Science*, *7* (5), 482-495.

[13] Faber, M., Bixler, R. and D'Mello, S.K. in press. An automated behavioral measure of mind wandering during computerized reading. *Behavior Research Methods*.

[14] Franklin, M.S., Broadway, J.M., Mrazek, M.D., Smallwood, J. and Schooler, J.W. 2013. Window to the Wandering Mind: Pupillometry of Spontaneous Thought While Reading. *The Quarterly Journal of Experimental Psychology*, *66* (12), 2289-2294.

[15] Gluck, K.A., Anderson, J.R. and Douglass, S.A. 2000. Broader Bandwidth in Student Modeling: What if ITS Were "Eye" TS? In Gauthier, C., Frasson, C. and VanLehn, K. eds. *Proceedings of the 5th international conference on intelligent tutoring systems*, Springer, Berlin.

[16] Graesser, A., Lu, S., Olde, B., Cooper-Pye, E. and Whitten, S. 2005. Question asking and eye tracking during cognitive disequilibrium: Comprehending illustrated texts on devices when the devices break down. *Memory and Cognition*, *33*, 1235-1247.

[17] Hanley, J.A. and McNeil, B.J. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, *143* (1), 29-36.

[18] Harley, J.M., Lajoie, S.P., Frasson, C. and Hall, N.C. in press. Developing Emotion-Aware, Advanced Learning Technologies: A Taxonomy of Approaches and Features. *International Journal of Artificial Intelligence In Education*.

[19] Hegarty, M. and Just, M. 1993. Constructing mental models of machines from text and diagrams. *Journal of Memory and Language*, *32* (6), 717-742.

[20] Hutt, S., Mills, C., Bosch, N., Krasich, K., Brockmole, J.R. and D'Mello, S.K. in review. Out of the Fr-Eye- ing Pan: Towards Gaze-Based Models of Attention during Learning with Technology in the Classroom.

[21] Hutt, S., Mills, C., White, S., Donnelly, P.J. and D'Mello, S.K. 2016. The Eyes Have It: Gaze-based Detection of Mind Wandering during Learning with an Intelligent Tutoring System. In *Proceedings of the 9th International Conference on Educational Data Mining (EDM 2016)*, International Educational Data Mining Society.

[22] Jaques, N., Conati, C., Harley, J.M. and Azevedo, R. Year. Predicting Affect from Gaze Data during Interaction with an Intelligent Tutoring System. In *Intelligent Tutoring Systems*, (2014), Springer, 29-38.

[23] Kardan, S. and Conati, C. 2012. Exploring gaze data for determining user learning with an interactive simulation. In Carberry, S., Weibelzahl, S., Micarelli, A. and Semeraro, G. eds. *Proceedings of the 20th International Conference on User Modeling, Adaptation, and Personalization (UMAP 2012)*, Springer, Berlin.

[24] Killingsworth, M.A. and Gilbert, D.T. 2010. A wandering mind is an unhappy mind. *Science*, *330* (6006), 932-932.

[25] Kintsch, W. 1998. *Comprehension: A paradigm for cognition*. Cambridge University Press, New York.

[26] Kopp, K., D'Mello, S. and Mills, C. 2015. Influencing the occurrence of mind wandering while reading. *Consciousness and Cognition*, *34* (1), 52-62.

[27] Mills, C. and D'Mello, S.K. 2015. Toward a Real-time (Day) Dreamcatcher: Detecting Mind Wandering Episodes During Online Reading. In Romero, C., Pechenizkiy, M., Boticario, J. and Santos, O. eds. *Proceedings of the 8th International Conference on Educational Data Mining (EDM 2015)*, International Educational Data Mining Society.

[28] Mooneyham, B.W. and Schooler, J.W. 2013. The costs and benefits of mind-wandering: A review. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, *67* (1), 11.

[29] Muir, M. and Conati, C. 2012. An analysis of attention to student–adaptive hints in an educational game. In Cerri, S.A., Clancey, W.J., Papadourakis, G. and Panourgia, K. eds. *Proceedings of the International Conference on Intelligent Tutoring Systems*, Springer, Berlin.

[30] Olney, A., D'Mello, A., Person, N., Cade, W., Hays, P., Williams, C., Lehman, B. and Graesser, A. 2012. Guru: A computer tutor that models expert human tutors. In Cerri, S., Clancey, W., Papadourakis, G. and Panourgia, K. eds. *Proceedings of the 11th International Conference on Intelligent Tutoring Systems*, Springer-Verlag, Berlin/Heidelberg.

[31] Olney, A., Risko, E.F., D'Mello, S.K. and Graesser, A.C. 2015. Attention in educational contexts: The role of the learning task in guiding attention. In Fawcett, J., Risko, E.F. and Kingstone, A. eds. *The Handbook of Attention*, MIT Press, Cambridge, MA.

[32] Pham, P. and Wang, J. 2016. Adaptive Review for Mobile MOOC Learning via Implicit Physiological Signal Sensing. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction (ICMI 2016)*, ACM, New York, NY.

[33] Pham, P. and Wang, J. 2015. AttentiveLearner: improving mobile MOOC learning via implicit heart rate tracking. In *International Conference on Artificial Intelligence in Education*, Springer, Berlin Heidelberg.

[34] Randall, J.G., Oswald, F.L. and Beier, M.E. 2014. Mind-wandering, cognition, and performance: A theory-driven meta-analysis of attention regulation. *Psychological Bulletin*, *140* (6), 1411-1431.

[35] Rapp, D.N. 2006. The value of attention aware systems in educational settings. *Computers in Human behavior*, *22* (4), 603-614.

[36] Rayner, K. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, *124* (3), 372-422.

[37] Reichle, E.D., Reineberg, A.E. and Schooler, J.W. 2010. Eye movements during mindless reading. *Psychological Science*, *21* (9), 1300.

[38] Risko, E.F., Buchanan, D., Medimorec, S. and Kingstone, A. 2013. Everyday attention: mind wandering and computer use during lectures. *Computers & Education*, *68* (1), 275-283.

[39] Roda, C. and Thomas, J. 2006. Attention aware systems: Theories, applications, and research agenda. *Computers in Human Behavior*, *22* (4), 557-587.

[40] Rowe, J., Mott, B., McQuiggan, S., Robison, J., Lee, S. and Lester, J. Year. Crystal island: A narrative-centered learning environment for eighth grade microbiology. In *Workshop on Intelligent Educational Games at the 14th International Conference on Artificial Intelligence in Education, Brighton, UK*, (2009), 11-20.

[41] Shute, V.J., Ventura, M., Bauer, M. and Zapata-Rivera, D. 2009. Melding the power of serious games and embedded assessment to monitor and foster learning: Flow and grow. In Ritterfeld, U., Cody, M. and Vorderer, P. eds. *Serious games: Mechanisms and effects*, Routledge, Taylor and Francis, Mahwah, NJ.

[42] Sibert, J.L., Gokturk, M. and Lavine, R.A. 2000. The reading assistant: eye gaze triggered auditory prompting for reading remediation. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, ACM, New York, NY.

[43] Smallwood, J., Davies, J.B., Heim, D., Finnigan, F., Sudberry, M., O'Connor, R. and Obonsawin, M. 2004. Subjective experience and the attentional lapse: Task engagement and disengagement during sustained attention. *Consciousness and Cognition*, *13* (4), 657-690.

[44] Smallwood, J., Fishman, D.J. and Schooler, J.W. 2007. Counting the cost of an absent mind: Mind wandering as an underrecognized influence on educational performance. *Psychonomic Bulletin & Review*, *14* (2), 230-236.

[45] Smallwood, J. and Schooler, J.W. 2015. The science of mind wandering: empirically navigating the stream of consciousness. *Annu. Rev. Psychol*, *66*, 487-518.

[46] Stewart, A., Bosch, P., Chen, H., Donnelly, P.J. and D'Mello, S.K. 2016. Where's Your Mind At? Video-Based Mind Wandering Detection During Film Viewing. In Aroyo, L., D'Mello, S., Vassileva, J. and Blustein, J. eds. *Proceedings of the 2016 ACM on International Conference on User Modeling, Adaptation, & Personalization (ACM UMAP 2016)*, ACM, New York.

[47] Szpunar, K.K., Khan, N.Y. and Schacter, D.L. 2013. Interpolated memory tests reduce mind wandering and improve learning of online lectures. *Proceedings of the National Academy of Sciences*, *110* (16), 6313-6317.

[48] Wang, H., Chignell, M. and Ishizuka, M. 2006. Empathic tutoring software agents using real-time eye tracking. In *Proceedings of the 2006 symposium on Eye tracking research &applications*, ACM, New York.

Measuring Similarity of Educational Items Using Data on Learners' Performance

Jiří Řihák
Faculty of Informatics
Masaryk University
Brno, Czech Republic
thran@mail.muni.cz

Radek Pelánek
Faculty of Informatics
Masaryk University
Brno, Czech Republic
pelanek@mail.muni.cz

ABSTRACT

Educational systems typically contain a large pool of items (questions, problems). Using data mining techniques we can group these items into knowledge components, detect duplicated items and outliers, and identify missing items. To these ends, it is useful to analyze item similarities, which can be used as input to clustering or visualization techniques. We describe and evaluate different measures of item similarity that are based only on learners' performance data, which makes them widely applicable. We provide evaluation using both simulated data and real data from several educational systems. The results show that Pearson correlation is a suitable similarity measure and that response times are useful for improving stability of similarity measures when the scope of available data is small.

1. INTRODUCTION

Interactive educational systems offer learners items (problems, questions) for solving. Realistic educational systems typically contain a large number of such items. This is particularly true for adaptive systems, which try to present suitable items for different kinds of learners. The management of a large pool of items is difficult. However, educational systems collect data about learners' performance and the data can be used to get insight into item properties. In this work we focus on methods for computing item similarities based on learners' performance data, which consists of binary information about the answers (correct/incorrect).

Automatically detected item similarities are the first and necessary step in further analysis such as clustering of the items, which is useful in several ways, with one particular application being learner modeling [9]. Learner models estimate knowledge and skills of learners and are the basis of adaptive behavior of educational systems. A learner's models requires a mapping of items into knowledge components [17]. Item clusters can serve as a basis for knowledge component definition or refinement. The specified knowledge components are relevant not only for modeling, but they are typically directly visible to learners in the user interface of a system, e.g., in a form of open learner model visualizing the estimated knowledge state, or in a personalized overview of mistakes, which is grouped by knowledge components.

Information about items is also very useful for management of the content of educational systems – preparation of new items, filtering of unsuitable items, preparation of explanations, and hint messages. Information about item similarities and clusters can be also relevant for teachers as it can provide them an inspiration for "live" discussions in class. This type of applications is in line with Baker's argument [1] for focusing on the use of learning analytics for "leveraging human intelligence" instead of its use for automatic intelligent methods.

Item similarities and clusters are studied not only in educational data mining but also in a closely related area of recommender systems. The setting of recommender systems is in many aspects very similar to educational systems – in both cases we have users and items, just instead of "performance" (the correctness of answers, the speed of answers) recommender systems consider "ratings" (how much a user likes an item). Item similarities and clustering techniques have thus been also considered in the recommender systems research (we mention specific techniques below). There is a slight, but important difference between the two areas. In recommender systems item similarities and clusterings are typically only auxiliary techniques hidden within a "recommendation black box". In educational system, it is useful to make these results explicitly available to system developers, curriculum production teams, or teachers.

There are two basic approaches to dealing with item similarities and knowledge components: a "model based approach" and an "item similarity approach". The basic idea of the model based approach is to construct a simplified model that explains the observed data. Based on a matrix of learners' answers to items we construct a model that predicts these answers. Typically, the model assigns several latent skills to learners and uses a mapping of items to corresponding latent factors. This kind of models can often be naturally expressed using matrix multiplication, i.e., fitting a model leads to matrix factorization. Once we fit the model to data, items that have the same value of a latent factor can be denoted as "similar". This approach leads naturally to multiple knowledge components per skill. The model is typically computed

using some optimization technique that leads only to local optima (e.g., gradient descent). It is thus necessary to address the role of initialization, and parameter setting of the search procedure. In recommender systems this approach is used for implementation of collaborative filtering; it is often called "singular value decomposition" (SVD) [18]. In educational context many variants of this approach have been proposed under different names and terminology, e.g., Q-matrix [3], non-negative matrix factorization techniques [8], sparse factor analysis [19], or matrix refinement [10].

With the item similarity approach we do not construct an explicit model of learners' behavior, but we compute directly a similarity measure for each pairs of items. These similarities are then used to compute clusters of items, to project items into a plane, or for other analysis (e.g., for each item listing the 3 most similar items). This approach naturally leads to a mapping with a single knowledge component per item (i.e., different kind of output from most model based methods). One advantage of this approach is easier interpretability. In recommender system research this approach is called neighborhood-based methods [11] or item-item collaborative filtering [7]. Similarity has been used for clustering of items [23, 24] and also for clustering of users [29]. In educational setting item similarity has been analyzed using correlation of learners' answers [22] and problem solving times [21], and also using learners' wrong answers [25].

So far we have discussed methods that are based only on data about learners' answers. Often we have some additional information about items and their similarities, e.g., a manual labeling or data based on syntactic similarity of items (text of questions). For both model based and item similarity approaches previous research has studied techniques for combination of these different types of inputs [10, 21].

In this work we focus on the item similarity approach, because in the educational setting this approach is less explored than the model based approach. We discuss specific techniques, clarify details of their usage, and provide evaluation using both data from real learners and simulated data. Simulated data are useful for evaluation of the considered unsupervised machine learning tasks, because in the case of real-world data we do not know the "ground truth".

The specific contributions of this work are the following. We provide guidelines for the choice of item similarity measures – we discuss different options and provide results identifying suitable measures (Pearson, Yule, Cohen); we also demonstrate the usefulness of "two step similarity measures". We explore benefits of the use of response time information as supplement to usual information of correctness of answer. We use and discuss several evaluation methods for the considered tasks. We specifically consider the issue of "how much data do we need". This is often practically more important than the exact choice of a used technique, but the issue is rather neglected in previous work.

2. MEASURES OF ITEM SIMILARITY

Figure 1 provides a high-level illustration of the item similarity approach. This approach consist of two steps that are to a large degree independent. At first, we compute an item similarity matrix, i.e., for each pair of items i, j we compute similarity s_{ij} of these items. At second, we can construct clusters or visualizations of items using only the item similarity matrix.

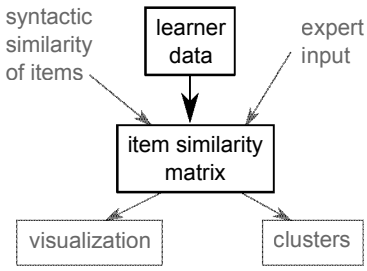


Figure 1: High-level illustration of the general approach to item analysis based on item similarities.

Experience with clustering algorithms suggests that the appropriate choice of similarity measure is more important than choice of clustering algorithm [13]. The choice of similarity measure is domain specific and it is typically not explored in general research on clustering. Therefore, we focus on the first step – the choice of similarity measure – and explore it for the case of educational data.

2.1 Basic Setting

In this work we focus on computing item similarities using learners' performance data. As Figure 1 shows, the similarity computation can also utilize information from domain experts or automatically determined information based on the inner structure of items (e.g., text of questions or some available meta-data).

We discuss different possibilities for computation of item similarities. Note that in our discussion we consistently use "similarity measures" (higher values correspond to higher similarity), some related works provide formulas for dissimilarity measures (distance of items; lower values correspond to higher similarity). This is just a technical issue, as we can easily transform similarity into dissimilarity by subtraction.

The input to item similarity computation are data about learner performance, i.e., a matrix $L \times I$, where L is the number of learners and I is the number of items. The matrix values specify learners' performance. The matrix is typically very sparse (many missing values). The output of the computation is an item similarity matrix, which specifies similarity for each pair of items.

Note that in our discussion we mostly ignore the issue of learning (change of learners skill as they progress through items). When learning is relatively slow and items are presented in a randomized order, learning is just a reasonably small source of noise and does not have a fundamental impact on the computation of item similarities. In cases where learning is fast or items are presented in a fixed order, it may be necessary to take learning explicitly into account.

2.2 Correctness of Answers

The basic type of information available in educational systems is the correctness of learners' answers. So we start with

similarity measures that utilize only this type of information, i.e., dichotomous data (correct/incorrect) on learners' answers on items. The advantage of these measures is that they are applicable in wide variety of settings.

With dichotomous data we can summarize learners' performance on items i and j using an agreement matrix with just four values (Table 1). Although we have just four values to quantify the similarity of items i and j, previous research has identified large number of different measures for dichotomous data and analyzed their relations [5, 12, 20]. For example Choi et al. [5] discuss 76 different measures, albeit many of them are only slight variations on one theme. Similarity measures over dichotomous data are often used in biology (co-occurrence of species) [14]. A more directly relevant application is the use of similarity measures for recommendations [30]. Recommender systems typically use either Pearson correlation or cosine similarity for computation of item similarities [11], but they consider richer than binary data.

Table 1: An agreement matrix for two items and definitions of similarity measures based on the agreement matrix ($n = a + b + c + d$ is the total number of observations).

		item i	
		incorrect	correct
item j	incorrect	a	b
	correct	c	d

Yule	$S_y = (ad - bc)/(ad + bc)$
Pearson	$S_p = (ad - bc)/\sqrt{(a+b)(a+c)(b+d)(c+d)}$
Cohen	$S_c = (P_o - P_e)/(1 - P_e)$ $P_o = (a + d)/n$ $P_e = ((a + b)(a + c) + (b + d)(c + d))/n^2$
Sokal	$S_s = (a + d)/(a + b + c + d)$
Jaccard	$S_j = a/(a + b + c)$
Ochiai	$S_o = a/\sqrt{(a + b)(a + c)}$

Table 1 provides definitions of 6 measures that we have chosen for our comparison. In accordance with previous research (e.g., [5, 14]) we call measures by names of researchers who proposed them. The choice of measures was done in such a way as to cover measures used in the most closely related work and measures which achieved good results (even if the previous work was in other domains). We also tried to cover different types of measures.

Pearson measure is the standard Pearson correlation coefficient evaluated over the dichotomous data. In the context of dichotomous data it is also called Phi coefficient or Matthews correlation coefficient. *Yule* measure is similar measure, which achieved good results in previous work [30]. *Cohen* measure is typically used as a measure of inter-rater agreement (it is more commonly called "Cohen's kappa"). In our setting it makes sense to consider this measure when we view learners' answers as "ratings" of items. Relations between these three measures are discussed in [32].

Ochiai coefficient is typically used in biology [14]. It is also equivalent to cosine similarity evaluated over dichotomous data; cosine similarity is often used in recommender systems for computing item similarity, albeit typically over interval data [7]. *Sokal* measure is also called Sokal-Michener or "simple matching". It is equivalent to accuracy measure used in information retrieval. Together with *Jaccard* measure they are often used in biology, but they have also been used for clustering of educational data [12].

Note that some similarity measures are asymmetric with respect to 0 and 1 values. These measures are typically used in contexts where the interpretation of binary values is presence/absence of a specific feature (or observation). In the educational context it is more natural to use measures which treat correct and incorrect answers symmetrically. Nevertheless, for completeness we have included also some of the commonly used asymmetric measures (Ochiai and Jaccard). In these cases we focus on incorrect answers (value a as opposed to d) as these are typically less frequent and thus bear more information.

2.3 Other Data Sources

The correctness of answers is the basic source of information about item similarities, but not the only one. We can also use other data. The second major type of performance data is response time (time taken to answer an item). The basic approach to utilization of response time is to combine it with the correctness of an answer. Given the correctness value $c \in \{0, 1\}$, a response time $t \in \mathbb{R}^+$, and the median of all response times τ, we combine them into a single score r. Examples of such transformations are: linear transformation for correct answers only ($r = c \cdot max(1 - t/2\tau,\ 0)$); exponential discounting used in MatMat [28] ($r = c \cdot min(1,\ 0.9^{t/\tau - 1})$); linear transformation inspired by *high speed, high stakes scoring rule* used in Math Garden [16] ($r = (2c - 1) \cdot max(1 - t/2\tau,\ 0)$). The first approach was used in our experiment due to its simplicity and high influence of response time information.

The scores obtained in this way are real numbers. Given the scores it is natural to compute similarity of two items using Pearson correlation coefficient of scores (over learners who answered both items). It is also possible to utilize specific wrong answers for computation of item similarity [25].

It is also possible to combine performance based measures with other types of data. For example we may estimate item similarity based on analysis of the content of items (syntactical similarity of texts), or collect expert opinion (manual categorization of items into several groups). The advantage of the similarity approach (compared to model based approach) is that different similarity measures can be usually combined in straightforward way by using a weighted average of different measures.

2.4 Second Level of Item Similarity

The basic computation of item similarities computes similarity of items i and j using only data about these two items. To improve a similarity measure, it is possible to employ a

"second of level of item similarity" that is based on the computed item similarity matrix and uses information on all items. Examples of such a second step is Euclidean distance or correlation. Similarity of items i and j is given by the Euclidean distance or Pearson correlation of rows i and j in the similarity matrix. Note that Euclidean distance may be used implicitly when we use standard implementation of some clustering algorithms (e.g., k-means).

With the basic approach to item similarity, we consider items similar when performance of learners on these items is similar. With the second step of item similarity, we consider two items similar when they behave similarly with respect to other items. The main reason for using this second step is the reduction of noise in data by using more information. This may be useful particularly to deal with learning. Two very similar items may have rather low direct similarity, because getting a feedback on the first item can strongly influence the performance on the second item. However, we expect both items to have similar similarities to other items.

A more technical reason to using the second step (particularly the Euclidean distance) is to obtain a measure that is a distance metric. The measures described above mostly do not satisfy triangle inequality and thus do not satisfy the requirements on distance metric; this property may be important for some clustering algorithms.

3. EVALUATION

In this work we focus on item similarity, but we keep the overall context depicted in Figure 1 in mind. The quality of a visualization is to a certain degree subjective and difficult to quantify, but the quality of clusters can be quantified and thus we can use it to compare similarity measures. From the large pool of existing clustering algorithms [15] we consider k-means, which is the most common implementation of centroid-based clustering, and hierarchical clustering. We used agglomerative or "bottom up" approach where items are successively merged to clusters using Ward's method as linkage criteria.

3.1 Data

We use data from real educational systems as well as simulated learner data. Real-world data provide information about the realistic performance of techniques, but the evaluation is complicated by the fact that we do not know the "ground truth" (the "correct" similarity or clusters of items). Simulated data provide a setting that is in many aspects simplified but allows easier evaluation thanks to the access to the ground truth.

For generating simulated data we use a simple approach with minimal number of assumptions and ad hoc parameters. Each item belongs to one of k knowledge components. Each knowledge component contains n items. Each item has a difficulty generated from the standard normal distribution $d_i \sim \mathcal{N}(0, 1)$. Skills of learners with respect to individual knowledge components are independent. Skill of a learner l with respect to knowledge component j is generated from the standard normal distribution $\theta_{lj} \sim \mathcal{N}(0, 1)$. We assume no learning (constant skills). Answers are generated as Bernoulli trials with the probability of a correct answer given by the logistic function of the difference of a relevant skill and an item difficulty (a Rasch model): $p = exp(\theta_{lj} - d_i)^{-1}$. This approach is rather standard, for example Piech at al. [26] use very similar procedure and also other works use closely related procedures [4, 12]. In the experiment reported below the basic setting is 100 learners, 5 knowledge components with 20 items each.

Table 2: Data used for analysis.

	learners	items	answers
Czech 1 (adjectives)	1 134	108	62 613
Czech 2	4 567	210	336 382
MatMat: numbers	6 434	60	67 753
MatMat: addition	3 580	135	20 337
Math Garden: addition	83 297	30	881 994
Math Garden: multiplic.	97 842	30	1 233 024

To evaluate techniques on realistic educational data, we use data from three educational systems. Table 2 describes the size of the used data sets.

Umíme Česky (`umimecesky.cz`) is a system for practice of Czech spelling and grammar. We use data only from one exercise from the system – simple "fill-in-the-blank" questions with two options. We use only data on the correctness of answers (response time is available, but since it depends on the text of a particular item its utilization is difficult). We focus particularly on one subset of items: questions about the choice between i/y in suffixes of Czech adjectives. For this subset we have manually determined 7 groups of items corresponding to Czech grammar rules.

MatMat (`matmat.cz`) is a system for practice of basic arithmetic (e.g., counting, addition, multiplication). For each item we know the underlying construct (e.g., "13" or "7 + 8") and also the specific form of questions (e.g., what type of visualization has been used). We use data on both correctness and response time. We selected the two largest subsets: multiplication and numbers (practice of number sens, counting).

Math Garden is another system for practice of basic arithmetic [16]. This system is more widely used than MatMat, but we do not have direct access to the system and detailed data. For the analysis we reuse publicly available data from previous research [6]. The available data contain both correctness of answers and response times, but they contain information only about 30 items without any identification of these items.

3.2 Comparison of Similarity Measures

To evaluate similarity measures we consider several types of analysis. With simulated data, we analyze the similarity measures with respect to the ground truth while for real-world data we evaluate correlations among similarity measures. We also compare the quality of subsequent clusterings using adjusted Rand index (ARI) [27, 31], which measures the agreement of two clusterings (with a correction for agreement due to chance). Typically, we use the adjusted Rand index to compare the clustering with a ground truth (available for simulated data) or with a manually provided

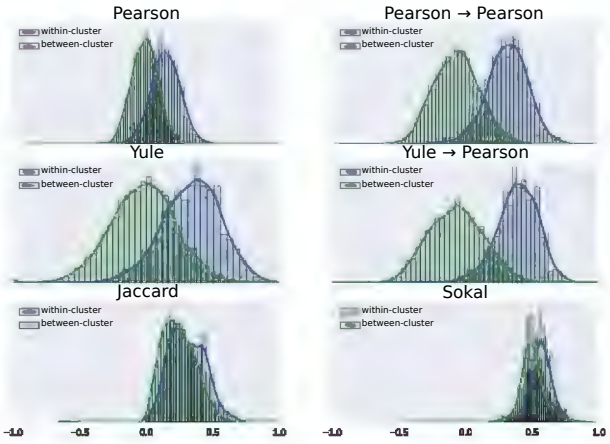


Figure 2: Differences between similarity values inside knowledge components and between them. Simulated data set with the basic setting were used.

classification (available for the Czech 1 data set). It can be also used to compare two detected clusterings (clusterings based on two different algorithms or clusterings based on two independent halves of data).

As a first step in the evaluation of similarity measures, we consider experiments with simulated data where we can utilize the ground truth. In clustering we expect high within-cluster similarity values and low between-cluster similarity values. Figure 2 shows distribution of the similarity values for selected measures and suggest which measures separate within-cluster and between-cluster values better and therefore which measures will be more useful in clustering. The results show that for Jaccard and Sokal measures the values overlap to a large degree, whereas Pearson and Yule measures provide better results. Adding the second step – Pearson correlation in this example – to the similarity measure separates within-cluster and between-cluster values better. That suggests that extending similarities in this way is not only necessary step for some subsequent algorithms such as k-means but also a useful technique with better performance.

For data coming from real systems we do not know the ground truth and thus we can only compare the similarity measures to each other. To evaluate how similar two measures are we take all similarity values for all item pairs and computed correlation coefficient. Figure 3 shows results for two data sets which are good representatives of overall results. Pearson and Cohen measures are highly correlated (> 0.98) across all data sets and have nearly the same values (although not exactly the same). Larger differences (but only up to 0.1) can be found typically when one of the values in the agreement matrix is small and that happens only for poorly correlated items with the resulting similarity value around 0. The second pair of highly correlated measures is Ochiai and Jaccard, which are both asymmetric with respect to the agreement matrix. The correlation between these two pairs of measures vary depending on data set and in some cases drops up to 0.5. Because of this high correlation within these pairs we further report results only

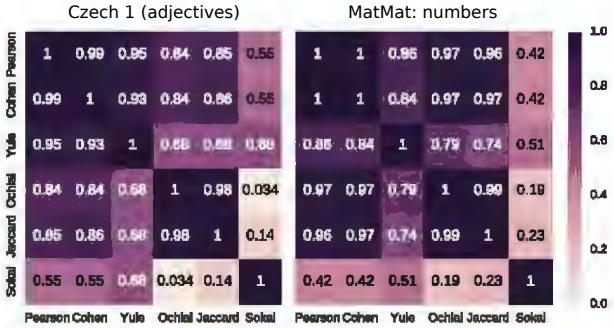


Figure 3: Correlations of similarity measures.

for Pearson and Jaccard measures. Yule measure is usually similar to Pearson measure (correlation usually around 0.9). The main difference is that the Yule measure spreads values more evenly across the interval [-1, 1]. Sokal is the most outlying measure with no correlation or small correlation (usually < 0.6) with all other measures.

Figure 4 shows the effect of the second levels of item similarity on the Pearson measure (results for other measures are analogical). The Euclid distance as second level similarity brings larger differences (lower correlation) than Pearson correlation. The correlations for large data sets such as Math Garden are usually high (> 0.9) and conversely the lowest correlations are found in results for small data sets. This suggests that the second level of similarity is more significant, and thus potentially more useful, where only limited amount of data is available.

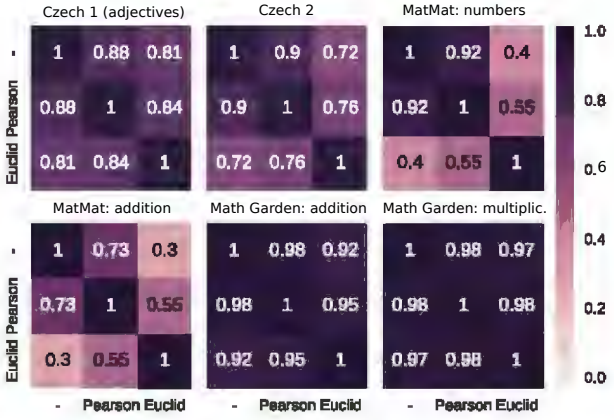


Figure 4: Correlations of Pearson measure and Pearson with different second levels.

Finally, we evaluate the quality of the similarity measures according to the performance of the subsequent clustering. From the two considered clustering methods we used the hierarchical clustering in this comparison because it naturally works with similarity measure and does not require metric space. The other two methods have similar result with same conclusions. Table 3 and Figure 5 show results. Although the results are dependent on the specific data set and the used clustering algorithm, there is quite clear general conclusion. Pearson and Yule measures provide better results

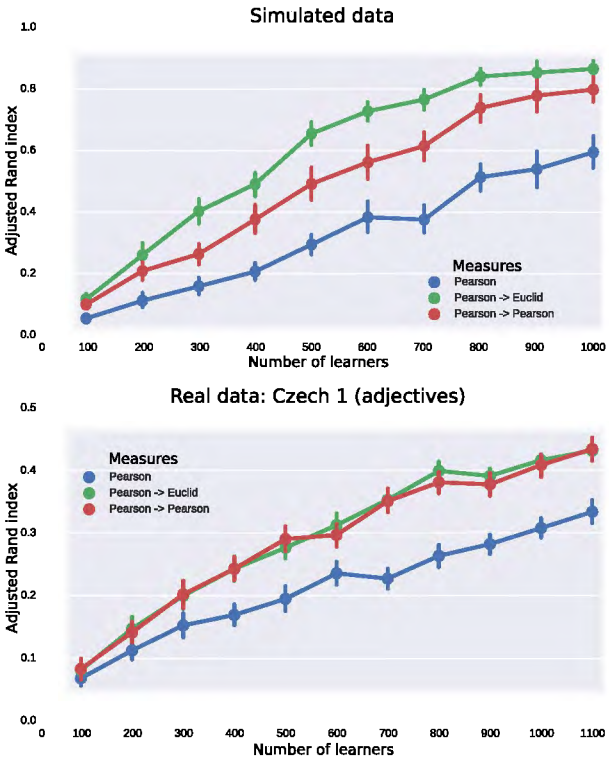


Figure 5: The quality of clustering for different measures used in the second step of item similarity. Top: Simulated data with 5 correlated skills. Bottom: Czech grammar with 7 manually determined clusters.

than Jaccard and Sokal, i.e., for the considered task the later two measures are not suitable. The Pearson is usually slightly better than Yule but the choice between them seems not to be fundamental (which is not surprising given that they are highly correlated). The results also show that the "second step" is always useful. The result for simulated data favor Euclidean distance over Pearson but there are almost no differences for real-world data.

3.3 Do We Have Enough Data?

In machine learning the amount of available data often is more important than the choice of a specific algorithm [2]. Our results suggest that once we choose a suitable type of similarity measure (e.g., Pearson, Cohen, or Yule), the differences between these measures are not fundamental, the more important issue becomes the size of available data.

Specifically, for a given data set we want to know whether the data are sufficiently large so that the computed item similarities are meaningful and stable. This issue can be explored by analyzing confidence intervals for computed similarity values. As a simple approach to analysis of similarity stability we propose the following approach: We split the available data into two independent halves (in a learner stratified manner), for each half we compute the item similarities, and we compute the correlation of the resulting item similarities.

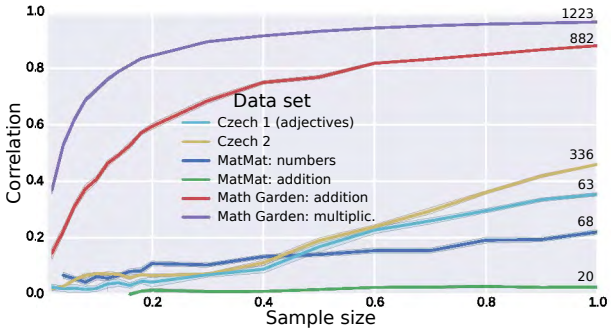


Figure 6: Stability of similarity measure (Yule) for real-world data sets. Data set was sampled, split to halves and Pearson correlation was computed for similarity values. Numbers on the right side indicate thousands of answers in data sets.

We can also perform this computation for artificially reduced data sets – this shows how the stability of results increases with the size of data. Figure 6 shows this kind of analysis for our data (real-world data sets). We clearly see large differences among individual data sets. Math Garden data set contains large number of answers and only a few items, the results show excellent stability, clearly in this case we have enough data to analyze item similarities. For the Czech grammar data set we have large number of answers, but these are divided among relatively large number of items. The results show a reasonably good stability, the data are usable for analysis, but clearly more data can bring improvement. For MatMat data the stability is poor, to draw solid conclusions about item similarities we need more data.

3.4 Response Time Utilization

The incorporation of response time information to similarity measure can change the meaning of similarity. Figure 7 gives such example and shows projection of items from MatMat practicing number sense. Similar items according to measures using only correctness of answers tend to be items with the same graphical representation in the system. On the other hand, similar items according to measures using also response time are usually items practicing close numbers.

We used this method also on data sets from Math Garden, which are much larger. In this case the use of response times has only small impact on the computed item similarities (correlations between 0.9 and 0.95). However, the use of response times influences how quickly does the computation converge, i.e., how much data do we need. To explore this we consider as the ground truth the average of computed similarity matrices with and without response times for the whole data set. Then we used smaller samples of the data set, used them to compute item similarities and checked the agreement with this ground truth. Figure 8 shows the difference between speed of convergence of measure with and without response time utilization. Results shows that the measure which use addition information from response time converges to ground truth much faster. This result suggests that the use of response time can improve clustering or visualizations when only small number of answers are available.

Table 3: Comparison of similarity measures for one real-world data (with sampled students) set and simulated data sets with c knowledge components and l learners. The values provide the adjusted Rand index (with 0.95 confidence interval) for a hierarchical clustering computed based on the specific similarity measure. The top result for every data set is highlighted.

	Czech 1 (c=7)	$l = 50, c = 5$	$l = 100, c = 5$	$l = 200, c = 5$	$l = 100, c = 2$	$l = 100, c = 10$
Pearson	0.32 ± 0.02	0.26 ± 0.04	0.48 ± 0.05	0.84 ± 0.05	0.77 ± 0.12	0.34 ± 0.04
Jaccard	0.31 ± 0.03	0.06 ± 0.03	0.15 ± 0.04	0.29 ± 0.08	0.32 ± 0.18	0.09 ± 0.02
Yule	0.31 ± 0.03	0.19 ± 0.04	0.43 ± 0.05	0.77 ± 0.07	0.60 ± 0.15	0.31 ± 0.03
Sokal	0.15 ± 0.06	0.11 ± 0.02	0.18 ± 0.03	0.25 ± 0.05	0.12 ± 0.11	0.14 ± 0.02
Pearson → Euclid	$\mathbf{0.43} \pm 0.01$	$\mathbf{0.45} \pm 0.05$	$\mathbf{0.80} \pm 0.06$	$\mathbf{0.98} \pm 0.01$	$\mathbf{0.95} \pm 0.03$	$\mathbf{0.67} \pm 0.04$
Yule → Euclid	0.32 ± 0.02	0.36 ± 0.05	0.65 ± 0.07	0.94 ± 0.04	0.89 ± 0.11	0.43 ± 0.03
Pearson → Pearson	0.41 ± 0.03	0.39 ± 0.05	0.73 ± 0.06	0.96 ± 0.02	0.92 ± 0.03	0.55 ± 0.04
Yule → Pearson	0.32 ± 0.03	0.38 ± 0.05	0.72 ± 0.06	0.97 ± 0.02	0.94 ± 0.04	0.55 ± 0.05

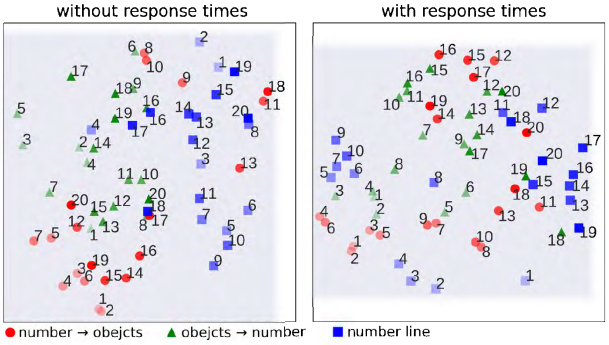


Figure 7: Projection of items practicing number sense from MatMat system. Left: Measure based only correctness. Right: Measure using response time. Opacity corresponds to the number value of the item and color corresponds to the graphical representation of the task.

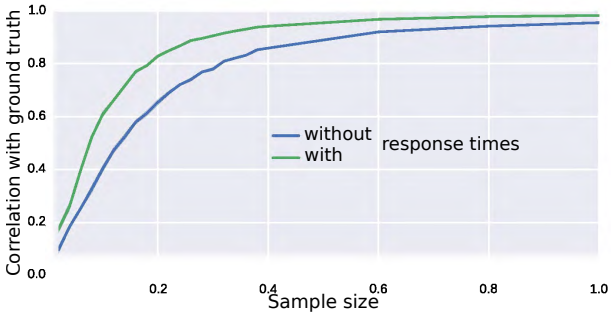


Figure 8: The speed of convergence to ground truth for measures with and without response time on Math Garden addition data set.

4. DISCUSSION

Our focus is the automatic computation of item similarities based on learners' performance data. These similarities can be then used in further analysis of an item relations such as an item clustering or a visualization. This outlines direction for future work in which methods using the item similarities should be studied in more detail. Compared to alternative approaches that have been proposed for the task (e.g., matrix factorizations, neural networks), the item similarity approach is rather straightforward, easy to realize, and it can be easily combined with other sources of information about items (text of items, expert opinion). For these reasons the item similarity approach should be used at least as a baseline in proposals for more complex methods like deep knowledge tracing [26].

The most difficult step in this approach is the choice of a similarity measure. Once we make a specific choice, the realization of the approach is easy. Our results provide some guidelines for this choice. Pearson, Yule, and Cohen measures lead to significantly better results than Ochiai, Sokal, and Jaccard measures. It is also beneficial to use the second step of item similarity (e.g., the Euclidean distance over vectors of item similarities). The exact choice of details does not seem to make fundamental difference (e.g., Pearson versus Yule in the first step, the Euclidean distance versus Pearson correlation in the second step). The Pearson correlation coefficient is a good "default choice", since it provides quite robust results and is applicable in several settings and steps. It also has the pragmatic advantage of having fast, readily available implementation in nearly all computational environments, whereas measures like Yule may require additional implementation effort.

The amount of data available is the critical factor for the success of automatic analysis of item relations. A key question for practical applications is thus: "Do we have enough data to use automated techniques?" In this work we used several specific methods for analysis of this question, but the issue requires more attention – not just for the item similarity approach, but also for other methods proposed in previous work. For example previous work on deep knowledge tracing [26], which studies closely related issues, states only that deep neural networks require large data without providing any specific quantification what 'large' means. The necessary quantity of data is, of course, connected to the quality of data – some data sources are more noisy than other, e.g., answers from voluntary practice contain more noise than answers from high-stakes testing. An important direction for future work is thus to compare model based and item simi-

larity approaches while taking into account the 'amount and quality of data available' issue.

5. REFERENCES

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

[2] M. Banko and E. Brill. Scaling to very very large corpora for natural language disambiguation. In *Proc. of Association for Computational Linguistics*, pages 26–33, 2001.

[3] T. Barnes. The q-matrix method: Mining student response data for knowledge. In *Educational Data Mining Workshop*, 2005.

[4] W.-H. Chen and D. Thissen. Local dependence indexes for item pairs using item response theory. *Journal of Educational and Behavioral Statistics*, 22(3):265–289, 1997.

[5] S.-S. Choi, S.-H. Cha, and C. C. Tappert. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8(1):43–48, 2010.

[6] F. Coomans, A. Hofman, M. Brinkhuis, H. L. van der Maas, and G. Maris. Distinguishing fast and slow processes in accuracy-response time data. *PloS one*, 11(5):e0155149, 2016.

[7] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[8] M. C. Desmarais. Mapping question items to skills with non-negative matrix factorization. *ACM SIGKDD Explorations Newsletter*, 13(2):30–36, 2012.

[9] M. C. Desmarais and R. S. 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.

[10] M. C. Desmarais, B. Beheshti, and P. Xu. The refinement of a q-matrix: Assessing methods to validate tasks to skills mapping. In *Proc. of Educational Data Mining*, pages 308–311, 2014.

[11] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[12] H. Finch. Comparison of distance measures in cluster analysis with dichotomous data. *Journal of Data Science*, 3(1):85–100, 2005.

[13] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[14] D. A. Jackson, K. M. Somers, and H. H. Harvey. Similarity coefficients: measures of co-occurrence and association or simply measures of occurrence? *American Naturalist*, pages 436–453, 1989.

[15] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[16] S. Klinkenberg, M. Straatemeier, and H. Van der Maas. Computer adaptive practice of maths ability using a new item response model for on the fly ability and difficulty estimation. *Computers & Education*, 57(2):1813–1824, 2011.

[17] 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.

[18] Y. Koren and R. Bell. Advances in collaborative filtering. *Recommender Systems Handbook*, pages 145–186, 2011.

[19] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *Journal of Machine Learning Research*, 15(1):1959–2008, 2014.

[20] S.-F. M. Liang and L.-W. Tzeng. Assessing suitability of similarity coefficients in measuring human mental models. In *Network of Ergonomics Societies Conference*, pages 1–5. IEEE, 2012.

[21] J. Nižnan, R. Pelánek, and J. Řihák. Using problem solving times and expert opinion to detect skills. In *Proc. of Educational Data Mining*, pages 434–434, 2014.

[22] J. Nižnan, R. Pelánek, and J. Řihák. Student models for prior knowledge estimation. In *Proc. of Educational Data Mining*, pages 109–116, 2015.

[23] M. O'Connor and J. Herlocker. Clustering items for collaborative filtering. In *Proc. of the ACM SIGIR Workshop on Recommender Systems*, volume 128. UC Berkeley, 1999.

[24] Y.-J. Park and A. Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proc. of Recommender systems*, pages 11–18. ACM, 2008.

[25] R. Pelánek and J. Řihák. Properties and applications of wrong answers in online educational systems. In *Proc. of Educational Data Mining*, 2016.

[26] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.

[27] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

[28] J. Rihák. Use of time information in models behind adaptive system for building fluency in mathematics. In *Proc. of Educational Data Mining*, 2015.

[29] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proc. of Computer and Information Technology*, volume 1, 2002.

[30] E. Şenyürek and H. Polat. Effects of binary similarity measures on top-n recommendations. *Anadolu University Journal of Science and Technology – A Applied Sciences and Engineering*, 14(1):55–65, 2013.

[31] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proc. of Machine Learning*, pages 1073–1080. ACM, 2009.

[32] M. J. Warrens. On association coefficients for 2×2 tables and properties that do not depend on the marginal distributions. *Psychometrika*, 73(4):777–789, 2008.

Adaptive Sequential Recommendation for Discussion Forums on MOOCs using Context Trees

Fei Mi Boi Faltings
Artificial Intelligence Lab
École polytechnique fédérale de Lausanne, Switzerland
firstname.lastname@epfl.ch

ABSTRACT

Massive open online courses (MOOCs) have demonstrated growing popularity and rapid development in recent years. Discussion forums have become crucial components for students and instructors to widely exchange ideas and propagate knowledge. It is important to recommend helpful information from forums to students for the benefit of the learning process. However, students or instructors update discussion forums very often, and the student preferences over forum contents shift rapidly as a MOOC progresses. So, MOOC forum recommendations need to be adaptive to these evolving forum contents and drifting student interests. These frequent changes pose a challenge to most standard recommendation methods as they have difficulty adapting to new and drifting observations. We formalize the discussion forum recommendation problem as a sequence prediction problem. Then we compare different methods, including a new method called context tree (CT), which can be effectively applied to online sequential recommendation tasks. The results show that the CT recommender performs better than other methods for MOOCs forum recommendation task. We analyze the reasons for this and demonstrate that it is because of better adaptation to changes in the domain. This highlights the importance of considering the adaptation aspect when building recommender system with drifting preferences, as well as using machine learning in general.

Keywords

MOOCs forum recommendation, context tree, model adaptation

1. INTRODUCTION

With the increased availability of data, machine learning has become the method of choice for knowledge acquisition in intelligent systems and various applications. However, data and the knowledge derived from it have a timeliness, such that in a dynamic environment not all the knowledge acquired in the past remains valid. Therefore, machine learning models should acquire new knowledge incrementally and adapt to the dynamic environments. Today, many intelligent systems deal with dynamic environments: information on websites, social networks, and applications in commercial markets. In such evolving environments, knowledge needs to adapt to the changes very frequently. Many statistical machine learning techniques interpolate between input data and thus their models can adapt only slowly to new situations. In this paper, we consider the dynamic environments for recommendation task. Drifting user interests and preferences [3, 11] are important in building personal assistance systems, such as recommendation systems for social networks or for news websites where recommendations need be adaptive to drifting trends rather than recommending obsolete or well-known information. We focus on the application of recommending forum contents for massive open online courses (MOOCs) where we found that the adaptation issue is a crucial aspect for providing useful and trendy information to students.

The rapid emergence of some MOOC platforms and many MOOCs provided on them has opened up a new era of education by pushing the boundaries of education to the general public. In this special online classroom setting, sharing with your classmates or asking help from instructors is not as easy as in traditional brick-and-mortar classrooms. So discussion forums there have become one of the most important components for students to widely exchange ideas and to obtain instructors' supplementary information. MOOC forums play the role of social learning media for knowledge propagation with increasing number of students and interactions as a course progresses. Every member in the forum can talk about course content with each other, and the intensive interaction between them supports the knowledge propagation between members of the learning community.

The online discussion forums are usually well structured via the different threads which are created by students or instructors; they can contain several posts and comments within the topic. An example of the discussion forum from a famous "Machine Learning" course by Andre Ng on Coursera[1] is shown in Figure 1. The left figure shows various threads and the right figure illustrates some replies within the last thread ("Having a problem with the Collaborative Filtering Cost"). In general, the replies within a thread are related to the topic of the thread and they can also refer to some other threads for supplementary information, like the link in the second reply. Our goal is to point the students towards useful forum threads through effectively mining forum visit patterns.

Two aspects set forum recommendation system for MOOCs apart from other recommendation scenarios. First, student interests and preferences drift fast during the span of a course, which is influenced by the dynamics in forums and the content of the course; second, the pool of items to be recommended and the items them-

[1] https://www.coursera.org/

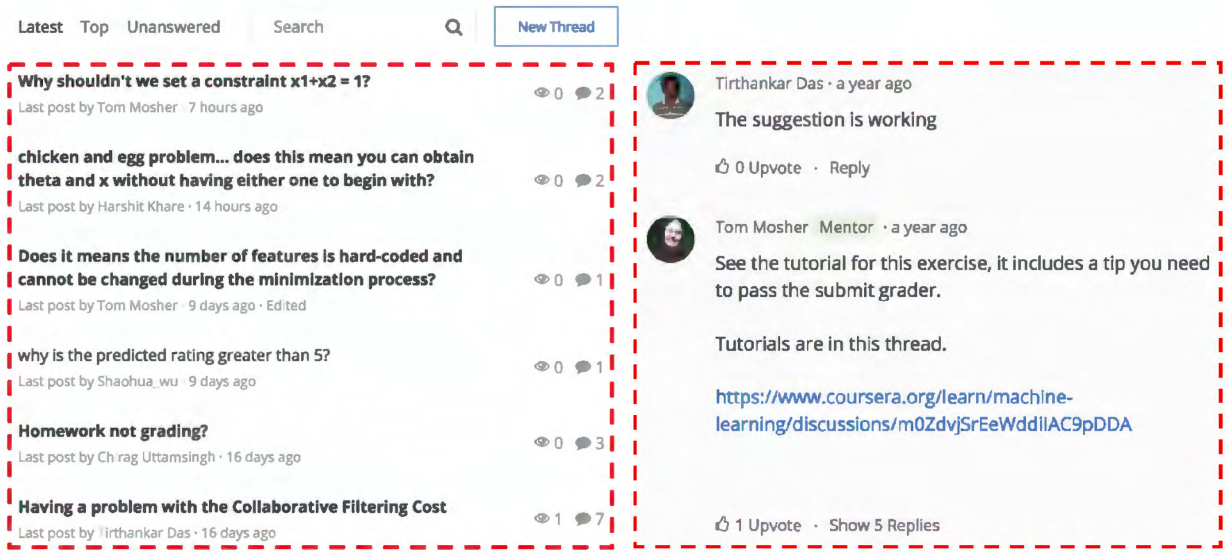


Figure 1: An sample discussion forum. Left: sample threads. Right: replies within the last thread ("Having a problem with the Collaborative Filtering Cost").

selves are evolving over time because forum threads can be edited very frequently by either students or instructors. So the recommendations provided to students need to be adaptive to these drifting preferences and evolving items. Traditional recommendation techniques, such as collaborative filtering and methods based on matrix factorization, only adapt slowly, as they build an increasingly complex model of users and items. Therefore, when a new item is superseded by a newer version or a new preference pattern appears, it takes time for recommendations to adapt. To better address the dynamic nature of recommendation in MOOCs, we model the recommendation problem as a dynamic and sequential machine learning problem for the task of predicting the next item in a sequence of items consumed by a user. During the sequential process, the challenge is combining old knowledge with new knowledge such that both old and new patterns can be identified fast and accurately. We use algorithms for sequential recommendation based on variable-order Markov models. More specifically, we use a structure called context tree (CT) [21] which was originally proposed for lossless data compression. We apply the CT method for recommending discussion forum contents for MOOCs, where adapting to drifting preferences and dynamic items is crucial. In experiments, it is compared with various sequential and non-sequential methods. We show that both old knowledge and new patterns can be captured effectively through context activation using CT, and that this is why it is particularly strong at adapting to drifting user preferences and performs extremely well for MOOC forum recommendation tasks.

The main contribution of this paper is fourfold:

- We applied the context tree structure to a sequential recommendation tasks where dynamic item sets and drifting user preferences are of great concern.
- Analyze how the dynamic changes in user preferences are followed in different recommendation techniques.
- Extensive experiments are conducted for both sequential and non-sequential recommendation settings. Through the experimental analysis, we validate our hypothesis that the CT recommender adapts well to drifting preferences.
- Partial context matching (PCT) technique, built on top of the standard CT method, is proposed and tested to generalize to new sequence patterns, and it further boosts the recommendation performance.

2. RELATED WORK

Typical recommender systems adopt a static view of the recommendation process and treat it as a prediction problem over all historical preference data. From the perspective of generating adaptive recommendations,we contend that it is more appropriate to view the recommendation problem as a sequential decision problem. Next, we mainly review some techniques developed for recommender systems with temporal or sequential considerations.

The most well-known class of recommender system is based on collaborative filtering (CF) [19]. Several attempts have been made to incorporate temporal components into the collaborative filtering setting to model users' drifting preferences over time. A common way to deal with the temporal nature is to give higher weights to events that happened recently. [6, 7, 15] introduced algorithms for item-based CF that compute the time weightings for different items by adding a tailored decay factor according to the user's own purchase behavior. For low dimensional linear factor models, [11] proposed a model called "TimeSVD" to predict movie ratings for Netflix by modeling temporal dynamics, including periodic effects, via matrix factorization. As retraining latent factor models is costly, one alternative is to learn the parameters and update the decision function online for each new observation [1, 16]. [10] applied the online CF method, coupled with an item popularity-aware weighting scheme on missing data, to recommending social web contents with implicit feedbacks.

Markov models are also applied to recommender systems to learn the transition function over items. [24] treated recommendation as a univariate time series problem and described a sequential model with a fixed history. Predictions are made by learning a forest of decision trees, one for each item. When the number of items is big, this approach does not scale. [17] viewed the problem of generating recommendations as a sequential decision problem and they con-

sidered a finite mixture of Markov models with fixed weights. [4] applied Markov models to recommendation tasks using skipping and weighting techniques for modeling long-distance relationships within a sequence. A major drawback of these Markov models is that it is not clear how to choose the order of Markov chain.

Online algorithms for recommendation are also proposed in several literatures. In [18], a Q-learning-based travel recommender is proposed, where trips are ranked using a linear function of several attributes and the weights are updated according to user feedback. A multi-armed bandit model called LinUCB is proposed by [13] for news recommendation to learn the weights of the linear reward function, in which news articles are represented as feature vectors; click-through rates of articles are treated as the payoffs. [20] proposed a similar recommender for music recommendation with rating feedback, called Bayes-UCB, that optimizes the nonlinear reward function using Bayesian inference. [14] used a Markov Decision Process (MDP) to model the sequential user preferences for recommending music playlists. However, the exploration phase of these methods makes them adapt slowly. As user preferences drift fast in many recommendation setting, it is not effective to explore all options before generating useful ones.

Within the context of recommendation for MOOCs, [23] proposed an adaptive feature-based matrix factorization framework for course forum recommendation, and the adaptation is achieved by utilizing only recent features. [22] designed a context-aware matrix factorization model to predict student preferences for forum contents, and the context considered includes only supplementary statistical features about students and forum contents. In this paper, we focus on a class of recommender systems based on a structure, called context tree [21], which was originally used to estimate variable-order Markov models (VMMs) for lossless data compression. Then, [2, 12, 5] applied this structure to various discrete sequence prediction tasks. Recently it was applied to news recommendation by [8, 9]. The most important property of online algorithms is the no-regret property, meaning that the model learned online is eventually as good as the best model that could be learned offline. According to [21], the no-regret property is achieved by context trees for the data compression problem. Regret analysis for CT was conducted through simulation by [5] for stochastically generated hidden Markov models with small state space. They show that CT achieves the no-regret property when the environment is stationary. As we focus on dynamic recommendation environments with time-varying preferences and limited observations, the no-regret property can be hardly achieved while the model adaptation is a bigger issue for better performance.

3. CONTEXT TREE RECOMMENDER

Due to the sequential item consumption process, user preferences can be summarized by the last several items visited. When modeling the process as a fixed-order Markov process [17], it is difficult to select the order. A variable-order Markov model (VMM), like a context tree, alleviates this problem by using a context-dependent order. The context tree is a space efficient structure to keep track of the history in a variable-order Markov chain so that the data structure is built incrementally for sequences that actually occur. A local prediction model, called expert, is assigned to each tree node, it only gives predictions for users who have consumed the sequence of items corresponding to the node. In this section, we first introduce how to use the CT structure and the local prediction model for sequential recommendation. Then, we discuss adaptation properties and the model complexity of the CT recommender.

3.1 The Context Tree Data Structure

In CT, a sequence $\mathbf{s} = \langle n_1, \ldots, n_l \rangle$ is an ordered list of items $n_i \in N$ consumed by a user. The sequence of items viewed until time t is $\mathbf{s}_t$ and the set of all possible sequences $\mathcal{S}$.

A context $S = \{\mathbf{s} \in \mathcal{S} : \xi \prec \mathbf{s}\}$ is the set of all possible sequences in $\mathcal{S}$ ending with the suffix ξ. ξ is the suffix ($\prec$) of $\mathbf{s}$ if last elements of $\mathbf{s}$ are equal to ξ. For example, one suffix ξ of the sequence $\mathbf{s} = \langle n_2, n_3, n_1 \rangle$ is given by $\xi = \langle n_3, n_1 \rangle$.

A context tree $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ and edges $\mathcal{E}$ is a partition tree over all contexts of $\mathcal{S}$. Each node $i \in \mathcal{V}$ in the context tree corresponds to a context S_i. If node i is the ancestor of node j then $S_j \subset S_i$. Initially the context tree $\mathcal{T}$ only contains a root node with the most general context. Every time a new item is consumed, the active leaf node is split into a number of subsets, which then become nodes in the tree. This construction results in a variable-order Markov model. Figure 2 illustrates a simple CT with some sequences over an item set $\langle n_1, n_2, n_3 \rangle$. Each node in the CT corresponds to a context. For instance, the node $\langle n_1 \rangle$ represents the context with all sequences end with item n_1.

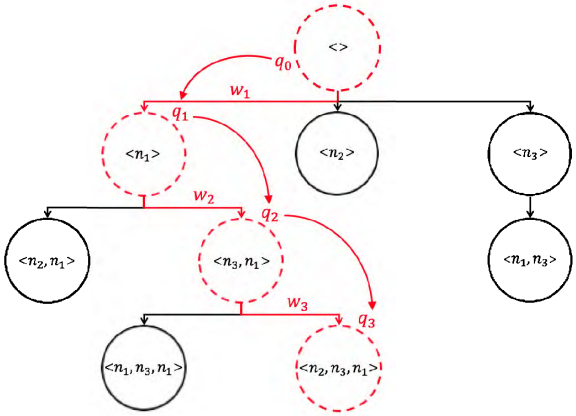


Figure 2: An example context tree. For the sequence $\mathbf{s} = \langle n_2, n_3, n_1 \rangle$, nodes in red-dashed are activated.

3.2 Context Tree for Recommendation

For each context S_i, an expert μ_i is associated in order to compute the estimated probability $\mathbb{P}(n_{t+1}|\mathbf{s}_t)$ of the next item n_{t+1} under this context. A user's browsing history $\mathbf{s}_t$ is matched to the CT and identifies a path of matching nodes (see Figure 2). All the experts associated with these nodes are called *active*. The set of *active* experts $\mathcal{A}(\mathbf{s}_t) = \{\mu_i : \xi_i \prec \boldsymbol{s}_t\}$ is the set of experts μ_i associated to contexts $S_i = \{\mathbf{s} : \xi_i \prec \mathbf{s}_t\}$ such that ξ_i are suffix of $\mathbf{s}_t$. $\mathcal{A}(\mathbf{s}_t)$ is responsible for the prediction for $\mathbf{s}_t$.

3.2.1 *Expert Model*

The standard way for estimating the probability $\mathbb{P}(n_{t+1}|\mathbf{s}_t)$, as proposed by [5], is to use a Dirichlet-multinomial prior for each expert μ_i. The probability of viewing an item x depends on the number of times α_{xt} the item x has been consumed when the expert is active until time t. The corresponding marginal probability is:

$$\mathbb{P}_i(n_{t+1} = x|\mathbf{s}_t) = \frac{\alpha_{xt} + \alpha_0}{\sum_{j \in \mathcal{N}} \alpha_{jt} + \alpha_0} \quad (1)$$

where α_0 is the initial count of the Dirichlet prior

3.2.2 Combining Experts to Prediction

When making recommendation for a sequence $\mathbf{s}_t$, we first identify the set of contexts and active experts that match the sequence. The predictions given by all the active experts are combined by mixing the recommendations given by them:

$$\mathbb{P}(n_{t+1} = x|\mathbf{s}_t) = \sum_{i \in \mathcal{A}(\mathbf{s}_t)} u_i(\mathbf{s}_t)\mathbb{P}_i(n_{t+1} = x|\mathbf{s}_t) \quad (2)$$

The mixture coefficient $u_i(\mathbf{s}_t)$ of expert μ_i is computed in Eq. 3 using the weight $w_i \in [0, 1]$. Weight w_i is the probability that the chosen recommendation stops at node i given that the it can be generated by the first i experts, and it can be updated in using Eq.5.

$$u_i(\mathbf{s}_t) = \begin{cases} w_i \prod_{j:S_j \subset S_i}(1 - w_j), & \text{if } \mathbf{s}_t \in S_i \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

The combined prediction of the first i experts is defined as q_i and it can be computed using the recursion in Eq. 4. The recursive construction that estimates, for each context at a certain depth i, whether it makes better prediction than the combined prediction q_{i-1} from depth $i - 1$.

$$q_i = w_i\mathbb{P}_i(n_{t+1} = x|\mathbf{s}_t) + (1 - w_i)q_{i-1} \quad (4)$$

The weights are updated by taking into account the success of a recommendation. When a user consumes a new item x, we update the weights of the active experts corresponding to the suffix ending before x according to the probability $q_i(x)$ of predicting x sequentially via Bayes' theorem. The weights are updated in closed form in Eq. 5, and a detailed derivation can be found in [5].

$$w_i' = \frac{w_i\mathbb{P}_i(n_{t+1} = x|\mathbf{s}_t)}{q_i(x)} \quad (5)$$

3.2.3 CT Recommender Algorithm

The whole recommendation process first goes through all users' activity sequences over time incrementally to build the CT; the local experts and weights updated using Equations 1 and 5 respectively. As users browse more contents, more contexts and paths are added and updated, thus building a deeper, more complete CT. The recommendation for an activity or context in a sequence is generated using Eq. 2 continuously as experts and weights are updated. At the same time, a pool of candidate items is maintained through a dynamically evolving context tree. As new items are added, new branches are created. At the same time, nodes corresponding to old items are removed as soon as they disappear from the current pool.

The CT recommender is a mixture model. On the one hand, the prediction $\mathbb{P}(n_{t+1} = x|\mathbf{s}_t)$ is a mixture of the predictions given by all the activated experts along the activated path so that it's a mixtures of local experts or a mixture of variable order Markov models whose oder are defined by context depths. On the other hand, one path in a CT can be constructed or updated by multiple users so that it's a mixture of users' preferences.

3.3 Adaptation Analysis

Our hypothesis, which is validated in later experiments, is that the CT recommender can be applied elegantly to domains where adaptation and timeliness are of concern. Two properties of the CT methods are crucial to the goal. First, the model parameter learning process and recommendations generated are online such that the model adapts continuously to a dynamic environment. Second, adaptability can be achieved by the CT structure itself as knowledge is organized and activated by context. New items or paths are recognized in new contexts, whereas old items can still be accessed in their old contexts. It allows the model to make predictions using more complex contexts as more data is acquired so that old and new knowledge can be elegantly combined. For new knowledge or patterns added to an established CT, they can immediately be identified through context matching. This context organization and context matching mechanism help new patterns to be recognized to adapt to changing environments.

3.4 Complexity Analysis

Learning CT uses the recursive update defined in Eq. 4 and recommendations are generated by weighting the experts' predictions along the activated path given by Eq. 2. For trees of depth D, the time complexity of model learning and prediction for a new observation are both $O(D)$. For input sequence of length T, the updating and recommending complexity are $O(M^2)$, where $M = min(D, T)$. Space complexity in the worst case is exponential to the depth of the tree. However, as we do not generate branches unless the sequence occurs in the input, we achieve a much lower bound determined by the total size of the input. So the space complexity is $O(N)$, where N is the total number of observations. Compared with the way that Markov models are learned, in which the whole transition matrix needs to be learned simultaneously, the space efficiency of CT offers us an advantage for model learning. For tasks that involve very long sequences, we can limit the depth D of the CT for space and time efficiency.

4. DATASET AND PROBLEM ANALYSIS

4.1 Dataset Description

In this paper, we work with recommending discussion forum threads to MOOC students. A forum thread can be updated frequently and it contains multiple posts and comments within the topic. As we mentioned before that the challenge is adapting to drifting user preferences and evolving forum threads as a course progresses. For the experiments elaborated in the following section, we use forum viewing data from three courses offered by École polytechnique fédérale de Lausanne on Coursera. These three courses include the first offering of "Digital Signal Processing", the third offering of "Functional Program Design in Scala", and the first offering of "'Reactive Programming'. They are referred to *Course 1*, *Course 2* and *Course 3*. Some discussion forum statistics for the three courses are given in Table 1. From the number of forum participants, forum threads, and thread views, we can see that the course scale increase from *Course 1* to *Course 3*. A student on MOOCs often accesses course forums many times during the span of a MOOC. Each time the threads she views are tracked as one *visit session* by the web browser. The total number of visit sessions and the average session lengths for three courses are presented in Table 1. The length of a session is the number of threads she viewed within a visit session. The thread viewing sequences corresponding to these regular visit sessions are called *separated* sequences in our later experiments and they treat threads in one visit session as one sequence. Models built using separated sequences try to catch short-term patterns within one visit session and we do not differentiate the patterns from different students. Another setting, called *combined* sequences, concatenates all of a student's visit sessions into one longer sequence so that models built using combined sequences try to learn long-term patterns across students. The average length of combined sequences is the average session length times the average number of sessions per student. From *Course 1* to *Course 3*, average lengths for separated and combined sequences both increase.

	Course 1	*Course 2*	*Course 3*
# of forum participants	5,399	12,384	13,914
# of forum threads	1,116	1,646	2,404
# of thread views	130,093	379,456	777,304
# of sessions	19,892	40,764	30,082
avg. session length	6.5	9	25.8
avg. # of sessions per student	3.7	3.3	2.2

Table 1: Course forum statistics for three datasets.

Another important issue that we can discover from the statistics is that thread viewing data available for sequential recommendation is very sparse. For example in *Course 1*, the average session length is 6.5 and the number of threads is around 1116. Then the complete space to be explored will be $1116^{6.5}$, which is much larger than the size of observations (130,093 thread views). The similar data sparsity issue is even more severe in the other two datasets.

4.2 Forum Thread View Pattern

Next, we study the thread viewing pattern which highlights the significance of adaptation issues for thread recommendation. Figure 3 illustrates the distribution of thread views against *freshness* for three courses. The freshness of an item is defined as the relative creation order of all items that have been created so far. For example, when a student views a thread t_m which is the m-th thread created in the currently existing pool of n threads, then *freshness* of t_m is defined as:

$$freshness = \frac{m}{n} \tag{6}$$

We can see from Figure 3 that there is a sharp trend that the new forum threads are viewed much more frequently than the old ones for all three courses. It is mainly due to the fact that fresh threads are closely relevant to the current course progress. Moreover, fresh threads can also supersede the contents in some old ones to be viewed. This tendency to view fresh items leads to drifting user preferences. Such drifting preferences, coupled with the evolving nature of forum contents, requires recommendations adaptive to drifting or recent preferences.

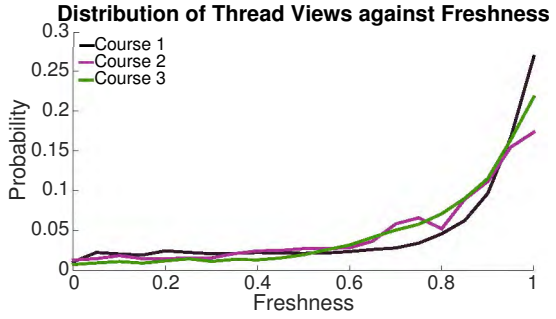


Figure 3: Thread viewing activities against freshness

A further investigation through those views on old threads leads us to a classification of threads into two categories: *general threads* and *specific threads*. Some titles of the general and specific threads are listed in Table 2. We could see the clear difference between these two classes of threads as the general ones corresponds to broad topics and specific ones are related to detailed course contents or exercises. We also found that only a very small part of the old threads are still rather active to be viewed and they are mostly general ones. Different from general threads, specific threads that subject to a fine timeliness are viewed very few times after they get old. In general, sequential patterns are observed more often within specific threads as some specific follow-up threads might be related and useful to the one that you are viewing. So the patterns learned could be used to guide your forum browsing process. On the contrary, sequential patterns on general threads are relatively random and imperceptible.

General Threads	**Specific Threads**
"Using GNU Octave"	"Homework Day 1 / Question 9"
"Any one from INDIA??"	"Quiz for module 4.2"
"Where is everyone from?	"quiz -1 Question 04"
"Numerical Examples in pdf"	'Homework 3, Question 11"
"How to get a certificate"	"Week 1: Q10 GEMA problem"

Table 2: Sample thread titles of general and specific threads.

5. RESULTS AND EVALUATION

In this section, we compare the proposed CT method against various baseline methods in both non-sequential and sequential settings. The results show that the CT recommender performs better than other methods under different setting for all three MOOCs considered. Through the adaptation analysis, we validate our hypothesis that the superior performance of CT recommender comes from the adaptation power to drifting preferences and trendy patterns in the domain. In the end, a regularization technique for CT, called partial context matching (PCT), is introduced. It is demonstrated that PCT helps better generalize among sequence patterns and further boost performance.

5.1 Baseline Methods

5.1.1 Non-sequential Methods

Matrix factorization methods proposed by [23, 22] are the state-of-the-art for MOOCs course content recommendation. Besides the user-based MF given in [23], we also consider item-based MF that generates recommendations based on the similarity of the latent item features learned from standard MF. In our case, each entry in the user-item matrix of MF contains the number of times a student views a thread. We also test a version where the matrix had a 1 for any number of views, but the performance was not as good, so the development of this version was not taken any further. MF models considered here are updated periodically (week-by-week). To enable a fair comparison against non-sequential matrix factorization techniques, we implemented versions where the CT model is updated at fixed time intervals, equal to those of the MF models. In the "One-shot CT" version, we compute the CT recommendations for each user based on the data available at the time of the model update, and the user then receives these same recommendations at every future time step until the next update. This mirrors the conditions of user-based MF. To compare with item-based MF, the "Slow-update CT" version updates the recommendations, but not the model, at each time point based on the sequential forum viewing information available at that time.

5.1.2 Sequential Methods

Sequential methods update model parameters and recommendations continuously as items are consumed. The first two simple methods are based on the observation and heuristic that fresh threads are viewed much frequently than old ones. *Fresh_1* recommends the last 5 *updated* threads, and *Fresh_2* recommends the last 5 *created* threads. Another baseline method, referred as *Popular*, recommends the top 5 threads among the last 100 threads viewed before the current one. We also consider an online version of MF [10] that

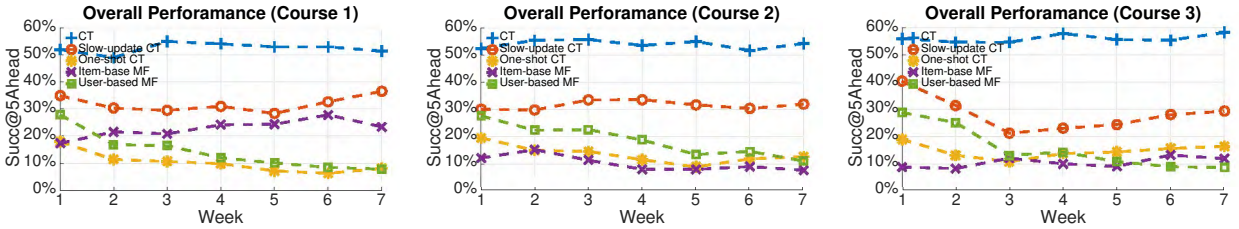


Figure 4: Overall performance comparison of CT and non-sequential methods

is currently the state-of-the-art sequential recommendation method, referred to "online-MF", in which the corresponding latent factor of the item i and user u are updated when a new observation R_{ui} arrive. The model optimization is implemented based on element-wise Alternating Least Squares. The number of latent factors is tuned to be 15, 20, 25 for three datasets, and the regularization parameter is set as 0.01. Moreover, the weight of a new observation is the same as old ones during optimization for achieving the best performance. Furthermore, the proposed CT recommender refers to the full context tree algorithm with a continuously updated model.

5.2 Performance and Adaptation Analysis

5.2.1 Evaluation Metrics

In our case, all methods recommend top-5 threads each time. Two evaluation metrics are adopted in the following experiments:

- **Succ@5**: the mean average precision (MAP) of predicting the immediately next thread view in a sequence.
- **Succ@5Ahead**: the MAP of predicting the future thread views within a sequence. In this case, a recommendation is successful even if it is viewed later in a sequence.

5.2.2 Comparison of Non-sequential Methods

Figure 4 shows the performance comparison between different versions of methods based on MF and CT on three datasets. "CT" is the sequential method with a continuously updated model, and all other methods Figure 4 are non-sequential versions. *Combined* sequences are used for the CT methods here to have a parallel comparison against MF. We found that a small value of the depth limit of the CTs hurts performances, yet a very large depth limit does not increase performance at the cost of computation and memory. Through experiments, we tune depths empirically and set them as 15, 20, 30 for three datasets.

Among non-sequential methods, one-shot CT and user-based MF perform the worst for all three courses, which means that recommending the same content for the next week without any sequence consideration is ineffective. Slow-update CT performs consistently the best among non-sequential methods, and it proves that adapting recommendations through context tree helps boost performance although the model itself is not updated continuously. Compared to slow-update CT, item-based MF performs much worse. They both update model parameters periodically and the recommendations are adjusted given the current observation. However, using the contextual information within a sequence and the corresponding prediction experts of slow-update CT are much more powerful than just using latent item features of item-base MF. Moreover, we can clearly see that the normal CT with continuous update outperforms all other non-sequential methods by a large margin for three datasets. It means that drifting preferences need to be followed though continuous and adaptive model update, so sequential methods are better choices. Next, we focus on sequential methods, and we validate our hypothesis that the CT model has superior performances because it better handles drifting user preferences.

5.2.3 Comparison of Sequential Methods

The results presented in Table 3 show the performance of the full CT recommender compared with other sequential baseline methods under different settings and evaluation metrics. Each result tuple contains the performance on the three datasets. We also consider a *tail performance* metric, referred to *personalized* evaluation, where the most popular threads (20, 30, and 40 for three courses) are excluded from recommendations. The depth limits of CTs using *separated* sequences are set to 8, 10, and 15 for three courses.

We notice that the online-MF method, with continuous model update, performs much worse compared with the CT recommender for all three datasets. This result shows that matrix factorization, which is based on interpolation over the user-item matrix, is not sensitive enough to rapidly drifting preferences with limited observations. The performances of two versions of the *Fresh* recommender are comparable with online-MF, and *Fresh_1* even outperforms online-MF in many cases, especially for **Succ@5Ahead**. It means that simply recommending fresh items even does a better job than online-MF for this recommendation task with drifting preferences. We can see that the CT recommender outperforms all other sequential methods under various settings, except for using non-personalized Succ@5Ahead for *Course 2*. The *Popular* recommender is indeed a very strong contender when using non-personalized evaluation since there is a bias that students can click a "top threads" tag from user interface to view popular threads which are similar to the ones given by *Popular* recommender. From the educational perspective, the setting using separated sequences and personalized evaluation is the most interesting as it reflects shot-term visiting patterns within a session over those specific and less popular forum threads. We could see from the upper right part of Table 3 that the CT recommender outperforms all other methods by a large margin under this setting.

	Non-personalized		*Personalized*	
	Succ@5	Succ@5Ahead	Succ@5	Succ@5Ahead
Separated Sequences				
CT	[**25**, **23**, **21**]%	[**48**, 53, **52**]%	[**19**, **14**, **16**]%	[**41**, **37**, **42**]%
online-MF	[15, 12, 8]%	[33, 29, 23]%	[10, 7 ,6]%	[27, 25, 20]%
Popular	[15, 20, 16]%	[40, **61**, 51]%	[9, 8 ,8]%	[34, 31, 36]%
Fresh_1	[12, 14, 10]%	[37, 43, 41]%	[10, 10, 8]%	[33, 31, 37]%
Fresh_2	[9, 8, 6]%	[31, 31, 29]%	[8, 7, 6]%	[30, 30, 28]%
Combined Sequences				
CT	[**21**, **20**, **20**]%	[**55**, 55, **56**]%	[**16**, **13**, **14**]%	[**46**, **39**, **46**]%
online-MF	[9, 8, 7]%	[34, 27, 23]%	[7,6,6]%	[29, 24, 20]%
Popular	[13, 14, 14]%	[52, **62**, 58]%	[9, 8, 7]%	[45, 36, 43]%
Fresh_1	[10, 12, 9]%	[48, 44, 44]%	[8, 9, 8]%	[44, 34, 42]%
Fresh_2	[7, 6, 6]%	[43, 34, 32]%	[6, 6, 6]%	[42, 32, 31]%

Table 3: Performance comparison of sequential methods

5.2.4 Adaptation Comparison

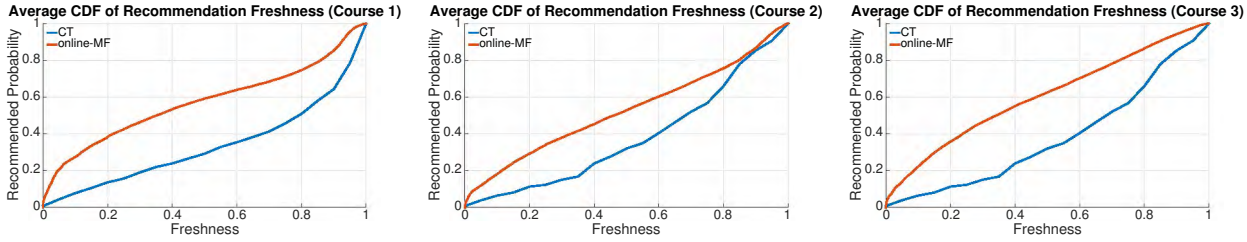


Figure 5: Distribution of recommendation freshness of CT and online-MF

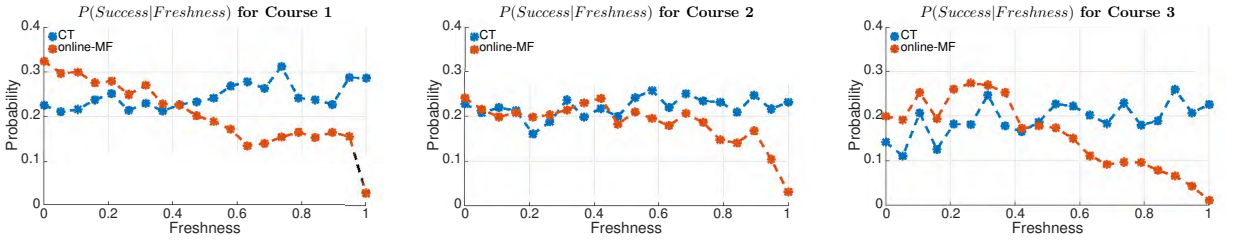


Figure 6: Conditional success rate of CT and online-MF

After seeing the superior performance of the CT recommender, we move to an insight analysis of the results. To be specific, we compare CT and online-MF in terms of their adaptation capabilities to new items. Figure 5 illustrates the cumulative density function (CDF) of the threads recommended by different methods against thread freshness. We can see that the CDFs of CT increase sharply when thread freshness increases, which means that the probability of recommending fresh items is high compared to online-MF. In other words, CT recommends more fresh items than online-MF. As we mentioned before that a large portion of fresh threads are specific ones, instead of general ones, so CT recommends more specific and trendy threads to students while methods based on matrix factorization recommend more popular and general threads.

Other than the quantity of recommending fresh and specific threads, the quality is crucial as well. Figure 6 shows the conditional success rate $P(Success|Freshness)$ across different degrees of freshness for three courses. $P(Success|Freshness)$ is defined as the fraction of the items successfully recommended given the item freshness. For instance, if an item with freshness 0.5 is viewed 100 times throughout a course, then $P(Success|Freshness = 0.5) = 0.25$ means it is among the top 5 recommended items 25 times. As the freshness increases, the conditional success rate of online-MF drops speedily while the CT method keeps a solid and stable performance. It is significant that CT outperforms online-MF by a large margin when freshness is high, in other words, it is particularly strong for recommending fresh items. Fresh items are often not popular in terms of the total number of views at the time point of recommendation. So identifying fresh items accurately implies a strong adaptation power to new and evolving forum visiting patterns. The analysis above validates our hypothesis that the CT recommender can adapt well to drifting user preferences. Another conclusion drawn from Figure 6 is that the performance of CT is as good as online-MF for items with low freshness. This is because that the context organization and context matching mechanism help old items to be identifiable though old contexts. To conclude, CT is flexible at combining old knowledge and new knowledge so that it performances well for items with various freshness, especially for fresh ones with drifting preferences.

5.3 Partial Context Matching (PCT)

At last, we introduce another technique, built on top of the standard CT, to generalize to new sequence patterns and further boost the recommendation performance. The standard CT recommender adopts a complete context matching mechanism to identify active experts for a sequence $\mathbf{s}$. That is, active experts of $\mathbf{s}$ come exactly from the set of suffixes of $\mathbf{s}$. We design a partial context matching (PCT) mechanism where active experts of a sequence are not constrained by exact suffixes, yet they can be those very similar ones. Two reasons bring us to design the PCT mechanism for context tree learning. First, PCT mechanism is a way of adding regularization. Sequential item consumption process does not have to follow exactly the same order, and slightly different sequences are also relevant for both model learning and recommendation generation. Second, the data sparsity issue we discussed before for sequential recommendation setting can be solved to some extent by considering similar contexts for learning model experts. The way PCT does aims to activate more experts to train the model, and to generate recommendations from a mixture of similar contexts.

We will focus on a *skip* operation that we add on top of the standard CT recommender. Some complex operations, like swapping item orders, are also tested, but they do not generate better performance. For a sequence $\langle s_p, \ldots, s_1 \rangle$ with length p, the skip operation generates p candidate partially matched contexts that skip one s_k for $k \in [1 \ldots p]$. All the contexts on the paths from root to partially matched contexts are activated. For example, the path to context $\langle n_2, n_1 \rangle$ can be activated from the context $\langle n_2, n_3, n_1 \rangle$ by the skipping n_3. However, for each partially matched context, there may not exist a fully matched path in the current context tree. In this case, for each partially matched context, we identify the longest path that corresponds it with length q. If q/p is larger than some threshold t, we update experts on this paths and use them to generate recommendations for the current observation. Predictions from multiple paths are combined by averaging the probabilities.

	Success@5	Success@5Ahead	Ratio
PCT-0.5	[+0.4, +0.6, +0.2]%	[+0.8, +0.9, +0.4]%	[4.9, 4.5, 3.3]
PCT-0.6	[+0.5, +0.8, +0.3]%	[+1.1, +1.3, +0.5]%	[4.4, 4.1, 2.9]
PCT-0.7	[+0.7, +0.9, +0.5]%	[+1.6, +1.9, +0.7]%	[3.7, 3.2, 2.5]
PCT-0.8	[+0.8, +1.1, +0.6]%	[+1.9, +2.4, +1.0]%	[3.2, 2.9, 2.1]
PCT-0.9	[+1.0, +1.4, +0.7]%	[+2.0, +2.7, +1.3]%	[2.4, 2.2, 1.4]

Table 4: Performance comparison of PCT against CT for three courses

Table 4 shows the performance of applying PCT for both model update and recommendation with threshold t (PCT-t). Results are compared with the full CT recommender with separated sequences and non-personalized evaluation. For cases where the threshold is smaller than 0.5, we sometimes obtain negative results since partially matched contexts are too short to be relevant. The "Ratio" column is the ratio of the number of updated paths in PCT compared with standard CT. We can see that PCT updates more paths and it offers us consistent performance boosts at the cost of computation.

6. CONCLUSION AND FUTURE WORK

In this paper, we formulate the MOOC forum recommendation problem as a sequential decision problem. Through experimental analysis, both performance boost and adaptation to drifting preferences are achieved using a new method called context tree. Furthermore, a partial context matching mechanism is studied to allow a mixture of different but similar paths. As a future work, exploratory algorithms are interesting to be tried. As exploring all options for all contexts are not feasible, we consider to explore only those top options from similar contexts. Deploying the CT recommender in some MOOCs for online evaluation would be precious to obtain more realistic evaluation.

7. REFERENCES

[1] J. Abernethy, K. Canini, J. Langford, and A. Simma. Online collaborative filtering. *University of California at Berkeley, Tech. Rep*, 2007.

[2] R. Begleiter, R. El-Yaniv, and G. Yona. On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, pages 385–421, 2004.

[3] R. M. Bell, Y. Koren, and C. Volinsky. The Bellkor 2008 solution to the Netflix prize. *Statistics Research Department at AT&T Research*, 2008.

[4] G. Bonnin, A. Brun, and A. Boyer. A low-order Markov model integrating long-distance histories for collaborative recommender systems. In *International Conference on Intelligent User Interfaces*, pages 57–66. ACM, 2009.

[5] C. Dimitrakakis. Bayesian variable order Markov models. In *International Conference on Artificial Intelligence and Statistics*, pages 161–168, 2010.

[6] Y. Ding and X. Li. Time weight collaborative filtering. In *ACM International Conference on Information and Knowledge Management*, pages 485–492. ACM, 2005.

[7] Y. Ding, X. Li, and M. E. Orlowska. Recency-based collaborative filtering. In *Australasian Database Conference*, pages 99–107. Australian Computer Society, Inc., 2006.

[8] F. Garcin, C. Dimitrakakis, and B. Faltings. Personalized news recommendation with context trees. In *ACM Conference on Recommender Systems*, pages 105–112. ACM, 2013.

[9] F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber. Offline and online evaluation of news recommender systems at swissinfo.ch. In *ACM Conference on Recommender Systems*, pages 169–176. ACM, 2014.

[10] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. Fast matrix factorization for online recommendation with implicit feedback. In *International ACM Conference on Research and Development in Information Retrieval*, volume 16, 2016.

[11] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[12] S. S. Kozat, A. C. Singer, and G. C. Zeitler. Universal piecewise linear prediction via context trees. *IEEE Transactions on Signal Processing*, 55(7):3730–3745, 2007.

[13] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *International Conference on World Wide Web*, pages 661–670. ACM, 2010.

[14] E. Liebman, M. Saar-Tsechansky, and P. Stone. DJ-MC: A reinforcement-learning agent for music playlist recommendation. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 591–599. IFAAMAS, 2015.

[15] N. N. Liu, M. Zhao, E. Xiang, and Q. Yang. Online evolutionary collaborative filtering. In *ACM Conference on Recommender Systems*, pages 95–102. ACM, 2010.

[16] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(Jan):19–60, 2010.

[17] G. Shani, R. I. Brafman, and D. Heckerman. An MDP-based recommender system. In *Conference on Uncertainty in Artificial Intelligence*, pages 453–460. Morgan Kaufmann Publishers Inc., 2002.

[18] A. Srivihok and P. Sukonmanee. E-commerce intelligent agent: personalization travel support agent using Q-Learning. In *International Conference on Electronic Commerce*, pages 287–292. ACM, 2005.

[19] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:4, 2009.

[20] X. Wang, Y. Wang, D. Hsu, and Y. Wang. Exploration in interactive personalized music recommendation: a reinforcement learning approach. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 11(1):7, 2014.

[21] F. M. Willems, Y. M. Shtarkov, and T. J. Tjalkens. The context-tree weighting method: Basic properties. *IEEE Transactions on Information Theory*, 41(3):653–664, 1995.

[22] D. Yang, D. Adamson, and C. P. Rosé. Question recommendation with constraints for massive open online courses. In *ACM Conference on Recommender Systems*, pages 49–56. ACM, 2014.

[23] D. Yang, M. Piergallini, I. Howley, and C. Rose. Forum thread recommendation for massive open online courses. In *Educational Data Mining*, 2014.

[24] A. Zimdars, D. M. Chickering, and C. Meek. Using temporal data for making recommendations. In *Conference on Uncertainty in Artificial Intelligence*, pages 580–588. Morgan Kaufmann Publishers Inc., 2001.

Analysis of problem-solving behavior in open-ended scientific-discovery game challenges

Aaron Bauer
awb@cs.washington.edu

Jeff Flatten
jflat06@cs.washington.edu

Zoran Popović
zoran@cs.washington.edu

Center for Game Science, Computer Science and Engineering
University of Washington
Seattle, WA 98195, USA

ABSTRACT

Problem-solving skills in creative, open-ended domains are both important and little understood. These domains are generally ill-structured, have extremely large exploration spaces, and require high levels of specialized skill in order to produce quality solutions. We investigate problem-solving behavior in one such domain, the scientific-discovery game *Foldit*. Our goal is to discover differentiating patterns and understand what distinguishes high and low levels of problem-solving skill. To address the challenges posed by the scale, complexity, and ill-structuredness of *Foldit* solver behavior data, we devise an iterative visualization-based methodology and use this methodology to design a concise, meaning-rich visualization of the problem-solving process in *Foldit*. We use this visualization to identify key patterns in problem-solving approaches, and report how these patterns distinguish high-performing solvers in this domain.

Keywords

Problem Solving; Scientific-Discovery Games; Visualization

1. INTRODUCTION

As efforts in scalable online education expand, interest continues to increase in moving beyond small, highly constrained tasks, such as multiple choice or short answer questions, and incorporating creative, open-ended activities [7, 14]. Existing research supports this move, showing that problem-based learning can enhance students' problem-solving and metacognitive skills [11]. Scaling such activities poses significant challenges, however, in terms of both assessment and feedback. It will be vital to devise scalable techniques not only to assess students' final products, but also to understand their progress through complex and heterogeneous problem-solving spaces. These techniques will apply to a broad range of education settings, from purely online programs like Udacity's Nanodegrees to more traditional settings where new standards like the Common Core emphasize strategic problem solving.

A growing body of work has found that educational and serious games are fertile ground for assessing students' capabilities and problem-solving skills [6, 10]. Our work continues this general line of inquiry by examining creative, problem-solving behavior among players in the scientific-discovery game *Foldit*. By modeling the functions of proteins, the workhorses of living cells, *Foldit* challenges players, hereafter referred to as solvers, to resolve the shape of proteins as a 3D puzzle. These puzzles are completely open and often under-specified, making it a highly suitable setting in which to gain insight into student progress through complex solution spaces. In the *Foldit* scientific-discovery community, the focus is on developing people from novices to experts that are eventually capable of solving protein structure problems that are currently unsolved by the scientific community. In fact, solutions produced in *Foldit* have led to three results published in Nature [3, 5, 16]. *Foldit* is an attractive learning space domain because its solvers are capable of contributing to state-of-the-art biochemistry results, and the vast majority of best performing solvers had no exposure to biochemistry prior to joining *Foldit* community. Hence, solver behavior in *Foldit* represents development of highly effective problem-solving in an open-ended domain over long time horizons. In this work, we identify six strategic patterns employed by *Foldit* solvers and show how these patterns differentiate between successful and less successful solvers. These patterns cover instances where solvers investigate multiple hypotheses, explore more greedily or more inquisitively, try to escape local optima, and make structured use of the manual or automated tools available in *Foldit*.

The aspects of the *Foldit* environment that make it an attractive setting in which to study problem solving also present significant challenges. Problems in *Foldit* share many of the properties Jonassen attributes to *design problems*, which they describe as "among the most complex and ill-structured kinds of problems that are encountered in practice" [13]. These properties include a *vague goal with few constraints* (in *Foldit*, the goal is often entirely open-ended: find a good configuration of the protein), *answers that are neither right or wrong, only better or worse*, and *limited feedback* (in *Foldit*, real-time feedback and solution evaluation are limited to a single numerical score corresponding to the protein's current energy state, and solvers frequently must progress through many low-scoring states to reach a good configuration; more nuanced feedback from biochemists is sometimes available, but on a timescale of weeks). The ill-structured nature of problems posed in *Foldit* necessarily deprives us of the structures, such as clear goal states and straightforward relationships between intermediate states and goal states, that typically form the basis of existing detailed and quantitative analyses of problem-solving behavior.

The size and complexity of *Foldit*'s problem space presents another major challenge. Even though the logs of solver interactions consist only of regular snapshots of a solver's current solution (along with attendant metadata), the record of a single solver's performance on a given problem frequently consists of thousands of such snapshots (which in turn are just a sparse sampling of the actual solving process). Furthermore, the nature of the solution state, the configuration of hundreds of components in continuous three-dimensional space, renders collapsing the state space by directly comparing solution states impractical. Compounding the size of the problem space is the complexity of the actions available to *Foldit* solvers. In addition to manual manipulation of the protein configuration, solvers can invoke various low-level automated optimization routines (some

of which run until the solver terminates them) and place different kinds of constraints on the protein configuration (*rubber bands* in *Foldit* parlance) that restrict its modification in a variety of ways. Solvers can also deploy many of these tools programmatically via Lua scripts called *recipes*. Taken together these challenges of ill-structuredness, size, and complexity threaten to make analysis of high-level problem-solving behavior in *Foldit* intractable.

To overcome these obstacles, we devise a visualization-based methodology capable of producing tractable representations of *Foldit* solvers' problem-solving behavior while maintaining the key encodings necessary for analysis of high-level strategic behavior. A process of iterative summarization forms the core of this methodology, and ensures that the transformations applied to the raw data do not elide structures potentially relevant to understanding solvers' unique strategic behavior. Using this methodology, we examine solver activity logs from 11 *Foldit* puzzles, representing 970 distinct solvers and nearly 3 million solution snapshots. Leveraging metadata present in the solution snapshots, we represent solving behavior as a tree, and apply our methodology to visualize a summarized tree showing where they branched off to investigate multiple hypotheses, how they employed some of the automated tools available to them, and other salient problem-solving behavior. We use these depictions to determine key distinguishing features of this exploration process. We subsequently use these features to better understand the patterns of expert-level problem solving.

Our work focuses on the following research questions: (1) how can we visually represent an open-ended exploration towards a high-quality solution in a large, ill-structured problem space? (2) what are the key patterns of problem-solving behavior exhibited by individuals?, and (3) what are the key differences along these patterns between high-performing and lower-performing solvers in an open-ended domain like *Foldit*? In addressing these questions we find that high-performing solvers explore the solution space more broadly. In particular, they pursue more hypotheses and actively avoid getting stuck in local minima. We also found that both high- and lower-performing solvers have similar proportion of manual and automated tool actions, indicating that better performance on open-ended challenges stems from the quality of the action intermixing rather than aggregate quantity.

2. RELATED WORK

While automated grading has mostly been explored for well-specified tasks where the correct answer has a straightforward and concise description, some previous work has developed techniques for more complex activities. Some achieve scalability through a crowd-sourcing framework such as Udacity's system for hiring external experts as project reviewers [14]. Other work has demonstrated automated approaches that leverage machine learning to enable scalable grading of more complex assignments. For example, Geigle et al. describe an application of online active learning to minimize the training set a human grader must produce [7] when automatically grading an assignment where students must analyze medical cases. Our work does not focus on grading problem-solving behavior, but instead approaches the issue of scalability at a more fundamental level: understanding fine-grained problem-solving strategies and how they contribute to success in an open-ended domain.

A robust body of prior work has addressed the challenge of both visualizing and gleaning insight from player activity in educational and serious games. Andersen et al. developed Playtracer, a general method for visualizing players' progress through a game's state space when a spatial relationship between the player and the virtual environment is not available [1]. Wallner and Kriglstein provide a thorough review of visualization-based analysis of gameplay data [21]. Prior work has analyzed gameplay data without visualization as well. Falakmasir et al. propose a data analysis pipeline for modeling player behavior in educational games. This system can produce a simple, interpretable model of in-game actions that can predict learning outcomes [6]. Our work differs in its aims from this prior work. We do not seek to develop a general visualization technique, but instead to design and leverage a domain-specific visualization to analyze problem-solving behavior. We are also not predicting player behavior, nor modeling players in terms of low-level actions, but rather identifying higher-level strategy use.

The work most similar to ours is that which focuses on problem-solving behavior, including both the long-running efforts in educational psychology to develop general theories and more recent work data-driven on understanding the problem-solving process. Our formulation of solving behavior in *Foldit* as a search through a problem space follows from classic information-processing theories of problem solving (e.g., [9, 19]). Gick reviews research on both problem-solving strategies and the differences in strategy use between experts and novices [8]. Our work complements the existing literature by focusing on understanding problem solving in the little-studied domain of scientific-discovery games, and on the ill-structured problems present in *Foldit*. Our findings on the differences in strategy use between high- and lower-performing solvers in *Foldit* are consistent with the consensus in the literature that expert's knowledge allows them to effectively use strategies that are poorly or infrequently used by less-skilled solvers. We also contribute a granular understanding of the specific strategies and differences at work in the *Foldit* domain.

Significant recent work has investigated problem-solving behavior in educational games and intelligent tutoring systems using a variety of techniques. Tóth et al. used clustering to characterize problem-solving behavior on tasks related to understanding a system of linear structural equations. The clusters distinguished between students that used a *vary-one-thing-at-a-time* strategy (both more and less efficiently) and those that used other strategies [20]. Through a combination of automated detectors, path analysis, and classroom studies, Rowe et al. investigated the relationship between a set of six strategic moves in a Newtonian physics simulation game and performance on pre- and post-assessments. They found that the use of some moves mediated the relationship between prior achievement and post scores [18]. Eagle et al. discuss several applications of using *interaction networks* to visualize and categorize problem-solving behavior in education games and intelligent tutoring systems. These networks offer insight for hint generation and a flexible method for visualizing student work in rule-using problem solving environments [4] . Using decision trees to build separate models for optimal and non-optimal student performance, Malkiewich et al. gained insight into how learning environments can encourage elegant problem solving [17]. Our primary contribution is to extend analysis of problem-solving behavior to a more complex and open-ended domain that those studied in similar previous work. The size and complexity of *Foldit*'s problem space, the volume of data necessary to capture exploration in this space, and the ill-structured nature of the *Foldit* problems all pose unique challenges. We devise a visualization-based methodology focused on iterative summarization, and successfully apply it to identify key problem-solving patterns exhibited by *Foldit* solvers.

3. FOLDIT

Foldit is a scientific-discovery game that crowdsources protein folding. It presents solvers with a 3D representation of a protein and tasks them with manipulating it into the lowest energy configuration. Each protein posed to the solvers is called a puzzle. Solvers' solutions to each puzzle are scored according to their energy configuration, and solvers compete to produce the highest scoring results.

Figure 1: The *Foldit* interface. *Foldit* solvers use a variety of tools to interactively reshape proteins. In this figure, a solver uses rubber bands to pull together two sheets, long flat regions of the protein.

Solvers have many tools at their disposal when solving *Foldit* puzzles. They can manipulate and constrain the structure in various ways, employ low-level automated optimization (e.g., a *wiggle* tool makes small, rapid, local adjustments to try and improve the score), and trigger solver-created automated scripts called *recipes* that can programmatically use the other tools. There is, however, a subset of the basic actions that cannot be used by recipes. We will call these *manual-only actions*. Previous work analyzing solver behavior in *Foldit* has focused primarily on recipe use and dissemination [2] and recipe authoring [15].

Foldit has several different types of puzzles for solvers to solve. In this work, we focus on the most common type of puzzle, *prediction* puzzles. These are puzzles in which biochemists know the amino acids that compose the protein in question, but do not know how the particular protein folds up in 3D space. This is in contrast to *design* puzzles in which solvers insert and delete which amino acids compose the protein to satisfy a variety of scientific goals, including designing new materials and targeting problematic molecules in diseases. We focus on prediction puzzles in this work to simplify our analysis by having a consistent objective (i.e., maximize score) across the problem-solving behavior we analyze.

4. METHODOLOGY

Prior work has demonstrated the power of visualization to support understanding of problem-solving behavior (e.g., [12]). Hence, we devise a methodology capable of producing concise, meaning-rich visualizations of the problem-solving process in *Foldit*, and then leverage these visualizations to identify key patterns of solver behavior. We are specifically interested in how solvers navigate from a puzzle's start state to a high-quality solution, what states they pass through in between, and what other avenues they explored. Since solving a *Foldit* puzzle can be represented as a directed search through a problem space, the clear encoding of parent-child relationships between nodes offered by a tree make it well-suited for visualizing these aspects of the solving process.

The scale of the *Foldit* data necessitates significant transformation of the raw data in order to render concise visualizations. Without any transformation, meaningful patterns are overwhelmed by sparse, repetitive data and would be far more challenging to identify. While there are many existing techniques for large-scale tree visualization, we find clear benefits to developing a visualization tailored to the *Foldit* domain. Specifically, preserving the semantics of our visual encoding is crucial for allowing us to connect patterns in the visualization to concrete strategic behavior in *Foldit*. To accomplish this, the process by which concise visualization are constructed must be carefully designed to maintain these links. Hence, we devise a design methodology focused on *iterative summarization*.

This process begins by visualizing the raw data. This is followed by iteratively building and refining a set of transformations to summarize the raw data while preserving meaning. The design of these transformations should be guided by frequently occurring structures. That is, those structures that the transformations can condense without eliding structures corresponding to unique strategic behavior. In parallel to this iterative design, a set of visual encodings are developed to represent the solving process as richly as possible. Key to this entire process is frequent consultation with domain experts, in our case experts on *Foldit* and its community. By applying this iterative methodology for several cycles, we designed a domain-specific visualization that we use to identify patterns of strategic behavior among *Foldit* solvers. We follow up on these patterns with computational investigation, and quantify their application by high- and lower-performing solvers.

4.1 Data

For our analysis, we selected 11 prediction puzzles spanning the range of time for which the necessary data is available. Though *Foldit* has been in continuous use since 2010, the data necessary to track a solver's progress through the problem space has only been collected since mid-2015. Our chosen dataset represents 970 unique solvers and nearly 3 million solution snapshots. These 11 puzzles are just a small subset of the available *Foldit* data. We chose a subset of similar puzzles (i.e., a subtype of relatively less complex prediction puzzles) in order to make common solving-behavior patterns easier to identify. The size of the subset was also guided by practical constraints, as each puzzle constitutes a large amount of data (20-60 GB for the data from all players on a single puzzle).

The data logged by *Foldit* primarily consists of snapshots of solver solutions as they play, stored as text files using the Protein Data Bank (pdb) format. These snapshots include the current protein pose, a timestamp, the solution's score, the number of times the solver has invoked each action and recipe, and a record of the intermediate states that led up to the solution at the time of the snapshot. This record, or *solution history*, is a list of unique identifiers each corresponding to a previous solution state. This list is extended every time the solver undoes an action or reloads a previous solution. Hence, by comparing the histories of two snapshots from the same solver, we can answer questions about their relationship (e.g., does one snapshot represent the predecessor of another; where did two related snapshots diverge). The key relationship for the purposes of this analysis is the direct parent-child relationship, which we use to generate trees that represent a solver's solving process.

4.2 Visualizing Solution Trees

We applied our methodology to our chosen subset of *Foldit* data to design a visualization of an individual's problem-solving process as a *solution tree*. Several key principles guided this design. First, since our goal is to discover key patterns, the visualization needs to highlight distinctly different strategies and approaches. These differences cannot be buried amidst enormous structures, nor destroyed by graph transformations. Second, the visualization must depict the closeness of each step to the ultimate solution in both time and quality to give a sense of the solver's progression. Third, the solver's use of automation in the form of recipes should be apparent since the use of automation is an important part of *Foldit*.

The fundamental organization of the visualization is that each node corresponds to a solution state encountered while solving. Using the solution history present in the logged snapshots of solver solutions, we establish parent-child relationships between solutions. If solution β is a child of solution α, it indicates that β was generated when the solver performed actions on α. One crucial limitation, however, is that a snapshot of the solver's current solution is captured far less often (only once every two minutes) than the solver takes actions. This means that our data is sparsely distributed along a solution's history going back to the puzzle's starting state. Hence, when naively constructing the tree from the logged solution histories, it ends up dominated by vast quantities of nodes with no associated data.

We address this issue by performing summarization on the solution trees, condensing them into concise representations amenable to analysis for important features. This summarization takes place in two stages. The first stage trims out nodes that (1) do not have corresponding data and (2) have zero children. This eliminates large numbers of leaf nodes that we are unable to reason about given that we lack the corresponding data. This stage also combines sequences of nodes each with only one child into a single node. For the median tree, this stage reduced the number of nodes by an order of magnitude from over 12,000 nodes to about 1,600.

The second stage consists of four phases, each informed by our observations of common patterns in trees produced by the first stage that would benefit from summarization. The first phase, called *prune*, focuses on simplifying uninteresting branches. We observed many of the branches preserved by the first stage were small, with at most three children, and only continued the tree from one of those children. Prune removes the leaf children of these branches from the tree. *Collapse*, the second phase, transforms each of the sequences of single-child nodes left behind after prune into single nodes. The third phase, *condense*, targets another common pattern where a sequence of branches feed into each other, with a child of each branch the parent of the next branch. These sequences are summarized into a single node labeled CASCADE along with the depth (number of branches) and width (average branching factor) of the summarized branches. See Figure 2 for an example of the features summarized by these three phases. The final phase, *clean*, targets the ubiquitous empty nodes (i.e., nodes for which we lack associated data) shown in black in Figure 2. We eliminate them by merging them with their parent node, doing so repeatedly until they all have been merged into nodes that contain data. In addition to making the trees more concise, this step allows us to reason more fully over the trees since all nodes are guaranteed to contain data. This second stage of summarization further reduced the number of nodes in the median tree by another order of magnitude to about 300 nodes. Summarization similarly reduces the space required to store the data by two orders of magnitude.

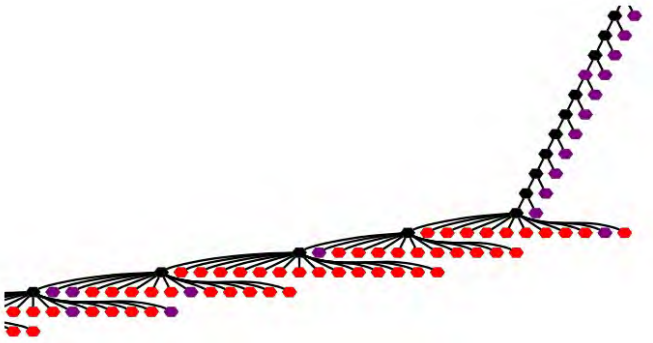


Figure 2: A solution tree after only the first stage of summarization. The non-black node color represents the score of the solution at that node (red is worse). The black nodes are empty in that we do not have solution data corresponding to that node. This figure also shows examples of the features targeted by the second summarization stage: *prune* and *collapse* eliminate long chains like the one on the right, and *condense* combines sequences of branches like those going down to left in single CASCADE nodes.

Child-parent relationships are not the only part of the data we visually encoded in the solution trees. Nodes are colored on a continuous gradient from red to blue according to the score of the solution represented by that node (red is low-scoring, blue is high-scoring). The best-scoring node is highlighted as a yellow star. Edges are colored on a continuous gradient from light to dark green according to the time the corresponding transition took place, and the children of each node are arranged left to right in chronological order. Finally, use of automation via recipes is an important aspect of problem-solving in *Foldit*. Since the logged solution snapshots contain a record of which recipes have been used at that point, we can use this to annotate nodes where a recipe was triggered. The annotations consist of the id of that recipe (a 4 to 6 digit number) and the number of times it was started.

One major weakness in the data available to us is the lack of a consistent way to determine when the execution of a recipe ended (some recipes save and restore, possibly being responsible for multiple nodes in the graph beyond where they were triggered). We partially address this by further annotating a node with the label MANUAL whenever the solver took a manual-only action at that node. This indicates that no previously triggered recipe continued past that node because no recipe could have performed the manual-only action. Since nodes in the summarized trees can represent many individual steps, it is possible for them to have several of these recipe and manual action annotations.

5. RESULTS

Using visualized solution trees for a large set of solvers across our sample of 11 puzzles, we identify a set of six prominent patterns in solvers' problem-solving behavior. These patterns do not encompass all solving behavior in *Foldit*, but instead capture key instances of strategic behavior in three categories: exploration, optimization, and human-computer collaboration. Future work is needed to generate a comprehensive survey of the strategic patterns in these and other categories. In this analysis, our focus is on identifying a small, diverse set of commonly occurring patterns to both provide initial

insight into problem-solving behavior, and to demonstrate the potential of our approach. In addition to identification, we also perform a quantitative comparison of how these patterns are employed by high-performing and lower-performing solvers to gain an understanding of how these patterns contribute to success in an open-end environment like *Foldit*.

5.1 Problem-Solving Patterns

Exploration. *Foldit* solvers are confronted with a highly discontinuous solution space with many local optima, creating a trade-off between narrowly focusing their efforts or taking the time to explore a broader range of possibilities. In our first two patterns, we examine the broader exploration side of this trade-off at two different scales. Taking the macro-scale first, we identify a pattern where solvers make significant progress on distinct branches of the tree (see Figure 3 for an example). We interpret this pattern as the solver investigating multiple hypotheses about the puzzle solution, using multiple instances of the game client or *Foldit*'s save and restore features to deeply explore them all. We call this the *multiple hypotheses* pattern.

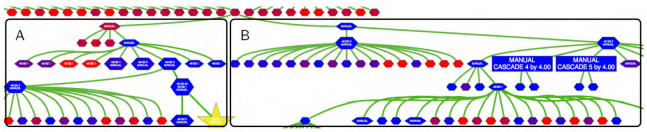


Figure 3: An example of the *multiple hypotheses* pattern. The two hypotheses branch out one of the nodes at the top and continue to the left (A) and right (B).

At the micro-scale, solvers very frequently generate a large number of possible next steps (i.e., a branch with a large number of children), but most often proceed to explore only one of them further. This is natural given the iterative refinement needed to successfully participate in *Foldit*. Hence, solvers that exhibit a pattern of much more frequently exploring multiple local possibilities demonstrate an unusual effort to explore more broadly. We call this the *inquisitive* pattern. Figure 4 shows an example of this behavior.

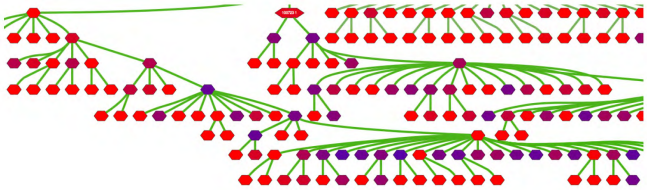


Figure 4: An example of the *inquisitive* pattern. Note how frequently multiple children of the same node are explored when compared to the tree in Figure 3.

Optimization. Navigating the extremely heterogeneous solution space is the primary challenge in *Foldit*, so we look closely at how solvers attempt to optimize their solutions, digging deeper into solvers' approach to exploration than the previous two patterns. We identify two related patterns describing solvers' fine-grained approach to optimization. The solution spaces of *Foldit* puzzles contain numerous local optima that solvers must escape, and we identify an *optima escape* pattern highly suggestive of a deliberate attempt to escape a local optima. This pattern occurs when a solver has a high-scoring node with a low-scoring child, and then chooses to explore from the low-scoring child. The solver was willing to ignore the short-term drop in score to try and reach a more beneficial state in the long-term. Figure 5 gives an example of this pattern.

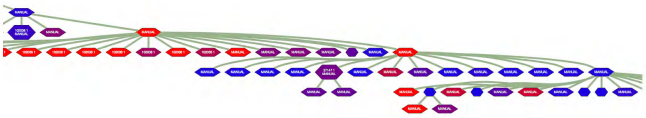


Figure 5: An example of the *optima escape* pattern. The solver transitions from a relatively high-scoring (i.e., blue) state in the upper left to a low-scoring (i.e., red) state. What makes this an example of the pattern is that exploration from the low-scoring state. In this case, the perseverance paid off as the solver reaches even higher-scoring states in the lower right.

In the other direction, we identify the *greedy* pattern in which solvers exclusively explore from the best-scoring of the available options. Obviously, some amount of greedy exploration is necessary in order to refine solutions, but in its extreme form deserves recognition as a pattern with significant potential impact on problem-solving success. Naturally, these two patterns do not cover all the ways solvers explore the problem space, but they do characterize specific strategic behavior of interest in this analysis.

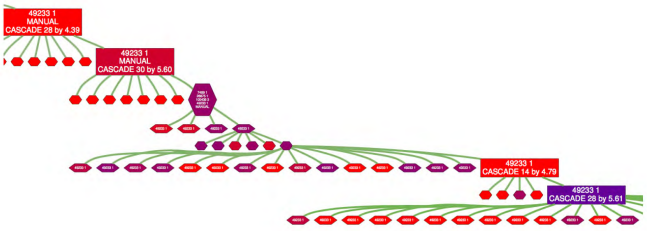


Figure 6: An example of the *repeated recipe* pattern. At three points in this solution tree snippet, the solver applies recipe 49233 to every child of a node.

Human-computer collaboration. Human-computer collaboration is a vital part of *Foldit*, and managing the trade-off between automation and manual intervention is a key feature of solving *Foldit* puzzles. We identify two patterns that each focus on one side of this trade-off. The first, the *manual* pattern, corresponds to extended sections of exclusively manual exploration. Since recipe use is very common, extended manual exploration represents a significant investment in the manual intervention side of the trade-off. Limitations with *Foldit* logging data prevent us from capturing all the manual exploration (i.e., it is not always possible to determine whether an action was performed by a solver manually or triggered as part of an automated recipe), but what can be captured is still an important dimension of variance among problem-solving behavior.

Our final pattern concerns recipe use. Some solvers apply a recipe to every child of a node periodically throughout their solution tree, using it as a clean-up or refinement step before continuing on (see Figure 6). We call this the *repeated recipe* pattern. Recipe use is very diverse and frequently doesn't display any specific structure, making this pattern interesting for its regimented way of managing some of the automation while solving.

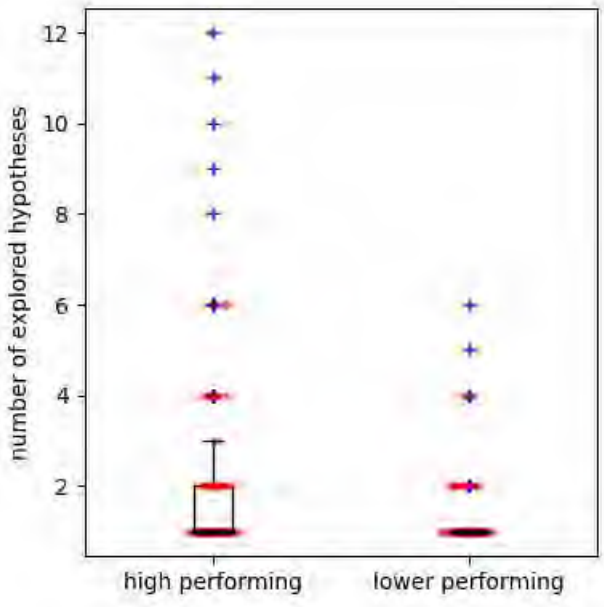


Figure 7: The number of hypotheses pursued in each solution tree for high- and lower-performing solvers. High-performing solvers frequently pursue two or more hypotheses, whereas lower-performing solvers most often pursue just one. Red circles show the distribution of individual solvers.

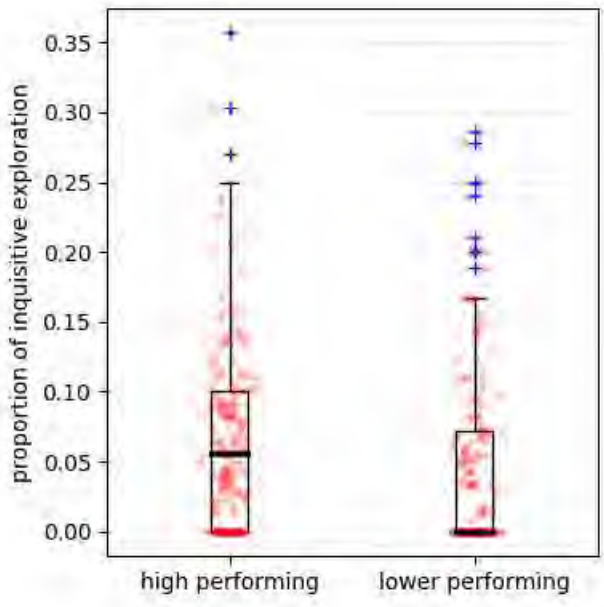


Figure 8: The proportion of all the branches in a solver's solution tree in which the solver explored more than one child for high- and lower-performing solvers. Red circles show the distribution of individual solvers.

5.2 Problem-Solving Patterns and Solver Performance

To understand how the patterns we identify relate to skillful problem-solving in an open-ended domain like *Foldit*, we compare their use among high-performing solvers to that among lower-performing solvers. Specifically, we analyze the occurrence of these patterns in the 15 best-scoring solutions from each puzzle and compare that to the occurrence in solutions from each puzzle ranked from 36th to 50th. Though it varies somewhat between puzzles, in general the solutions ranked 36th to 50th represent a *middle ground* in terms of quality. They fall outside the puzzle's state-of-the-art solutions, but remain well above the least successful efforts. Throughout these comparisons we use non-parametric Mann-Whitney U tests with $\alpha = 0.008$ confidence (Bonferroni correction for six comparisons, $\alpha = 0.05/6$), as our data is not normally distributed. For each test, we report the test statistic U, the two-tailed significance p, and the rank-biserial correlation measure of effect size r. In addition, since some of the metrics we compute may not apply to all solution trees (e.g., the tree contains no branches where the inquisitive pattern can be evaluated), we report the number of solvers involved in the comparison n for each test (the full sample is $n = 330$).

We find high-performing solvers explore more broadly than lower-performing solvers. For the *multiple hypotheses* pattern, high-performing solvers pursued significantly more hypotheses than lower-performing solvers ($U = 10569$, $p = 0.000014$, $r = 0.217$, $n = 330$) (see Figure 7). For the *inquisitive* pattern, we compute the proportion of each solver's exploration that matches the pattern (i.e., of all the branches in a solver's solution tree, in what fraction of them did the solver explore more than one child) and find high-performing solvers explore inquisitively more often than lower-performing solvers ($U = 9343$, $p = 0.000295$, $r = 0.231$, $n = 313$) (see Figure 8).

We also find high-performing solvers work harder to avoid local optima. For the *optima escape* pattern, we compute the number of times this behavior occurs in each solution and find that high-performing solvers engage in this behavior more than lower-performing solvers ($U = 11183.5$, $p = 0.00185$, $r = 0.173$, $n = 330$) (see Figure 9). For the *greedy* pattern, we compute the proportion of each solver's exploration that matches the pattern (i.e., of all the branches in a solver's solution tree, in what fraction of them did the solver only explore the best-scoring child). While high-performing solvers engaged in greedy optimization less often than lower-performing solvers, the difference was not significant ($U = 9079$, $p = 0.0158$, $r = -0.163$, $n = 295$) (see Figure 10).

Finally, we find no significant difference between high- and lower-performing solvers in the frequency they manually explore and employ recipes. For the *manual* pattern, we compute the number of manual exploration sections in each solution and find no significant difference between high- and lower-performing solvers ($U = 13334$, $p = 0.789$, $r = 0.014$, $n = 330$). For the *repeated recipe* pattern, we computed the median frequency of recipe use along all paths in the solution (i.e., for each path from the root to a leaf, in what fraction of the nodes did the solver trigger at least one recipe) and though lower-performing solvers used recipes more frequently, the difference between high- and lower-performing solvers was not significant ($U = 11342$, $p = 0.0140$, $r = -0.157$, $n = 329$).

6. DISCUSSION

The results from our analysis of our solution tree visualizations illuminate some key problem-solving patterns exhibited by individual *Foldit* solvers. Namely, how broadly an individual explores, both on a macro- and micro-scale, how actively an individual avoids

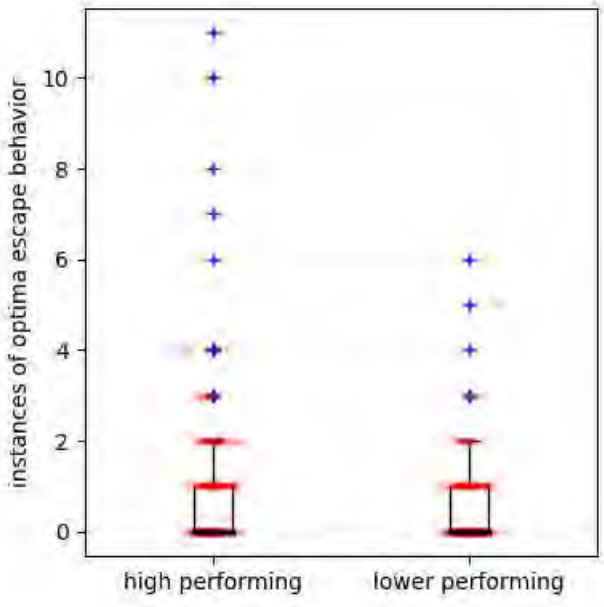


Figure 9: The number of times in each solution a solver engages in *optima escape* behavior for high- and lower-performing solvers. Red circles show the distribution of individual solvers.

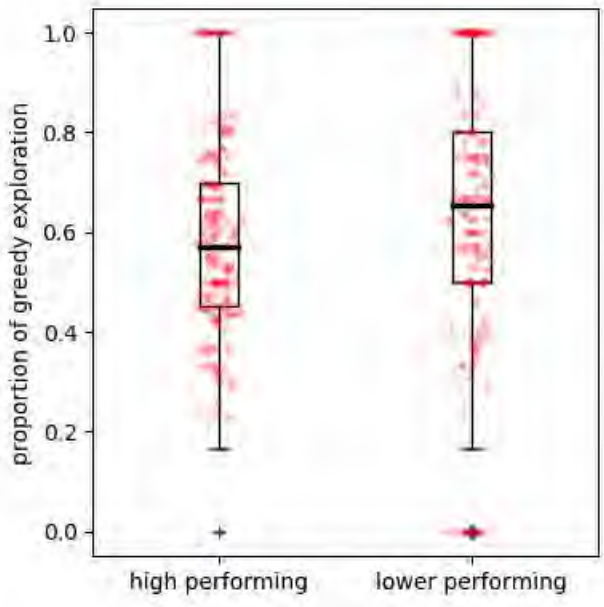


Figure 10: The proportion of all the branches in a solver's solution tree in which the solver explored only the best-scoring child for high- and lower-performing solvers. The fact that the median for both categories of solver is above 0.5 indicates that this pattern in an important part of refining solutions in *Foldit*. Red circles show the distribution of individual solvers.

local optima by engaging in less greedy optimization and actively pursuing locally suboptimal lines of inquiry, and how an individual manages the interplay between automation and manual intervention.

Comparing high- and lower-performing solvers in their application of these patterns suggests that skillful problem-solving in an open-end domain like *Foldit* involves broader exploration and more conscious avoidance of local minima. This finding that a key feature of high-skill solving behaviors is not being enamored by the current best solution and possessing strategies for avoiding myopic thinking had implications for the strategies that should be taught to develop successful problem solvers. Further work is required on other large open-ended domains to confirm this trend.

The finding that solvers of different skill use greedy exploration, manual exploration, and automation in similar amounts suggests skillful deployment of non-greedy exploration, automation, and manual intervention takes place at a more fine-grained level than overall quantity. Though this work focuses on the presence or absence of specific solving behavior, the timing and sequencing of strategic moves are likely to be critical to success. Further work is needed to investigate what differentiates effective and ineffective use of specific solving strategies.

The *Foldit* dataset itself presented significant challenges for our analysis, and we addressed these through an iterative visualization-based methodology. This process served as a design method for generating a visual grammar to describe a complex problem-solving process. We do not study the generalization of this approach to other datasets and domains in this work, but the prerequisites for its application to other open-ended problem-solving domains can be concisely enumerated: (1) the logs of solver activity establish clear temporal relationships between solution states such that those states can be visualized as a progression through the solution space, (2) the solution state or associated metadata is amenable to visual encoding, so that the visualized progressions can represent fine-grained details of the solving process, and (3) deep problem-solving domain expertise is available to provide the necessary context for interpreting and summarizing the visualized structures.

Our chosen subset of *Foldit* data represents only a small fraction of the total available data. In particular, we limited our analysis to a sample of similar prediction puzzles, and compared specific ranges of high- and lower-performing solvers. Though these choices are well-motivated, it is an important question for future work as to whether our results hold across different datasets and groups of comparison. More broadly, *Foldit* supports numerous variations on the prediction and design puzzle archetypes, which offers an exciting opportunity to study problem solving across a number of related contexts with varying goals, constraints, inputs, and tools.

7. CONCLUSION

Gaining a better understanding of key patterns in problem-solving behavior in complex, open-ended environments is important for deploying this kind of activity in an educational setting at scale. In this work, we identified six key patterns in problem-solving behavior among solvers of *Foldit*. The protein folding challenges in *Foldit* present rich, completely open, heterogeneous solution spaces, making them a compelling domain in which to analyze these patterns. To facilitate the identification of these patterns, we used an iterative methodology to design visualizations of solvers' problem-solving activity as solution trees. The size and complexity of the *Foldit* data required us to develop domain-specific techniques to summarize the solution trees and render them tractable for analysis while preserving the salient problem-solving behaviors. Finally, we compared the

occurrence of the patterns we identified between high- and lower-performing solvers. We found that high-performing solvers explore more broadly and more aggressively avoid local optima. We also found that both categories of solvers employ automation and manual intervention in similar quantities, inviting future work to study how these tools are used at a more fine-grained level.

We have only scratched the surface in our analysis of a subset of *Foldit* data. Two integral aspects of the *Foldit* environment are not within the scope of this work: collaboration and expert feedback. We only considered solutions produced by individual solvers, but *Foldit* solver can also take solutions produced by others and try and improve them. This collaborative framework may involve specialization and unique solving strategies, and deserves careful study. Expert feedback comes into play for design puzzles, where biochemists will select a small number of the solutions to try and synthesize in the lab. Experts will also impose additional constraints on future design puzzles to try and guide solutions toward more promising designs. The interaction of these channels for expert feedback and problem-solving behavior is an important topic for future research. Also outside the scope of this work is how individual solvers change their problem-solving behavior over time. Many solvers have been participating in the *Foldit* community for many years, and studying how their behavior evolves could yield insights into the acquisition of high-level problem-solving skills.

Looking more broadly at the impact of this work, our methodology and analysis can serve as a first step toward discovering the scaffolding necessary to develop high-level problem-solving skills. These results could contribute to a hint generation system, where solvers could be guided toward known effective strategies, or a meta-planner component in *Foldit* that could tailor the parameters of particular puzzles to optimize the quality of the scientific results. In all of these cases, this work contributes to the necessary foundational understanding of the problem-solving behavior involved.

8. ACKNOWLEDGEMENTS

This work was supported by the National Institutes of Health grant 1UH2CA203780, RosettaCommons, and Amazon. This material is based upon work supported by the National Science Foundation under Grant No. 1629879.

9. REFERENCES

[1] E. Andersen, Y.-E. Liu, E. Apter, F. Boucher-Genesse, and Z. Popović. Gameplay analysis through state projection. In *Proceedings of the fifth international conference on the foundations of digital games*, pages 1–8. ACM, 2010.

[2] S. Cooper, F. Khatib, I. Makedon, H. Lu, J. Barbero, D. Baker, J. Fogarty, Z. Popović, et al. Analysis of social gameplay macros in the foldit cookbook. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, pages 9–14. ACM, 2011.

[3] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, Z. Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.

[4] M. Eagle, D. Hicks, B. Peddycord III, and T. Barnes. Exploring networks of problem-solving interactions. In *Proceedings of the 5th Conference on Learning Analytics And Knowledge*. ACM, 2015.

[5] C. B. Eiben, J. B. Siegel, J. B. Bale, S. Cooper, F. Khatib, B. W. Shen, B. L. Stoddard, Z. Popovic, and D. Baker. Increased diels-alderase activity through backbone remodeling guided by foldit players. *Nature biotechnology*, 30(2):190–192, 2012.

[6] M. H. Falakmasir, J. P. Gonzalez-Brenes, G. J. Gordon, and K. E. DiCerbo. A data-driven approach for inferring student proficiency from game activity logs. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 341–349. ACM, 2016.

[7] C. Geigle, C. Zhai, and D. C. Ferguson. An exploration of automated grading of complex assignments. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 351–360. ACM, 2016.

[8] M. L. Gick. Problem-solving strategies. *Educational psychologist*, 21(1-2):99–120, 1986.

[9] J. G. Greeno. Natures of problem-solving abilities. *Handbook of learning and cognitive processes*, 5:239–270, 1978.

[10] E. Harpstead, C. J. MacLellan, K. R. Koedinger, V. Aleven, S. P. Dow, and B. Myers. Investigating the solution space of an open-ended educational game using conceptual feature extraction. In *Proceedings of The 6th Conference on Educational Data Mining*, 2013.

[11] W. Hung, D. H. Jonassen, R. Liu, et al. Problem-based learning. *Handbook of research on educational communications and technology*, 3:485–506, 2008.

[12] M. Johnson, M. Eagle, and T. Barnes. Invis: An interactive visualization tool for exploring interaction networks. In *Proceedings of the 6th Conference on Educational Data Mining*, 2013.

[13] D. H. Jonassen. Toward a design theory of problem solving. *Educational Technology Research and Development*, 48(4):63–85, dec 2000.

[14] D. A. Joyner. Expert evaluation of 300 projects per day. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 121–124. ACM, 2016.

[15] F. Khatib, S. Cooper, M. D. Tyka, K. Xu, I. Makedon, Z. Popović, and D. Baker. Algorithm discovery by protein folding game players. *Proceedings of the National Academy of Sciences*, 108(47):18949–18953, 2011.

[16] F. Khatib, F. DiMaio, S. Cooper, M. Kazmierczyk, M. Gilski, S. Krzywda, H. Zabranska, I. Pichova, J. Thompson, Z. Popović, et al. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature structural & molecular biology*, 18(10):1175–1177, 2011.

[17] L. Malkiewich, R. S. Baker, V. Shute, S. Kai, and L. Paquette. Classifying behavior to elucidate elegant problem solving in an educational game. In *Proceedings of the 9th Conference on Educational Data Mining*, 2016.

[18] E. Rowe, R. S. Baker, and J. Asbell-Clarke. Strategic game moves mediate implicit science learning. In *Proceedings of the 8th Conference on Educational Data Mining*, 2015.

[19] H. A. Simon. Information-processing theory of human problem solving. *Handbook of learning and cognitive processes*, 5:271–295, 1978.

[20] K. Tóth, H. Rölke, S. Greiff, and S. Wüstenberg. Discovering students' complex problem solving strategies in educational assessment. In *Proceedings of the 7th Conference on Educational Data Mining*, 2014.

[21] G. Wallner and S. Kriglstein. Visualization-based analysis of gameplay data–a review of literature. *Entertainment Computing*, 4(3):143–155, 2013.

The Antecedents of and Associations with Elective Replay in an Educational Game: Is Replay Worth It?

Zhongxiu Liu
North Carolina State University
zliu24@ncsu.edu

Christa Cody
North Carolina State University
cncody@ncsu.edu

Tiffany Barnes
North Carolina State University
tmbarnes@ncsu.edu

Collin Lynch
North Carolina State University
cflynch@ncsu.edu

Teomara Rutherford
North Carolina State University
taruther@ncsu.edu

ABSTRACT

Replayability has long been touted as a benefit of educational games. However, little research has measured its impact on learning, or investigated when students choose to replay prior content. In this study, we analyzed data on a sample of 4,827 3rd-5th graders from ST Math, a game-based educational platform integrated into classroom instruction in over 3,000 classrooms across the U.S. We identified features that describe elective replays relative to prior gameplay performance, and associated elective replays with in-game accuracy, confidence, and general math ability assessments outside of the games. We found some elective replay patterns were associated with learning, whereas others indicated that students were struggling in the current educational content. We suggest, therefore, that educational games should use elective replay behaviors to target interventions according to when and whether replay is helpful for learning.

Keywords

Educational Games, Serious Game Analytics, Replayability

1. INTRODUCTION

"Replayability is an important component of successful games." [15] In most games, there are two types of plays: play and replay to pass a level (*pass attempts*) and replay after passing a level (*elective replay*). In this paper, we investigate the latter. Elective replay (*ER*) is particularly interesting because the motivations behind a student's decision to replay and the impact of those replays are relatively unknown. This paper explores potential associations between elective replay and student characteristics and performance in the domain of educational games.

Replayability has been touted as a benefit of educational games [9]. Replayability encourages players to engage in repeated judgement-behavior-feedback loops, where users make decisions based on the situation and/or feedback, act on those decisions, and receive feedback based on their actions [18]. In the RETAIN model designed by Gunter et al. [10] to evaluate educational games, replayability is a criteria for naturalization – an important component in helping students make their knowledge automatic, reducing the cognitive load of low-level details to allow for higher order thinking. In the RETAIN model, "replay is encouraged to assist in retention and to remediate shortcomings." [10] Meaningful elective replay is often encouraged by game features such as score leaderboards, which inspire students to replay for higher scores [4]. Because higher scores typically require a deeper understanding of the educational content in a well-designed game, encouraging elective replay may promote mastery. Games with replay also allow the student to be exposed to more material and give them more freedom to control their learning. Studies have shown that giving students control over their learning process can increase motivation, engagement, and performance [6, 8].

However, few studies have investigated when students choose to replay, why they do so, or have measured the outcomes associated with elective replay. One reason is that educational game studies are often comparatively brief, so replayability is often minimally assessed with post-game questionnaires asking about students' intention for future play [14, 5]. Consequently, there is a need to investigate elective replay with actual logged actions in a game setting where students have sufficient time and freedom to replay.

This work analyzed gameplay logs from a series of math games within the year-long supplemental digital mathematics curriculum Spatial Temporal (ST) Math. We analyzed gameplay data from 4,827 3rd-5th graders throughout the 2012-2013 school year. Our data contained 37,452 logged elective replays, accounting for 1.48% of the logged play. We analyzed gameplay and elective replay features in association with students' demographic information, in-game math objective tests, and the state standardized math test. We sought to answer three research questions: Q1: What are the characteristics of students who engage in elective replay, Q2: What gets replayed, and under what circumstances? And Q3: Is elective replay associated with improvements in students' accuracy on math objectives, confidence, and

general math ability?

2. RELATED WORK

2.1 Factors Influencing Elective Replay

Few empirical studies have investigated the motivations behind elective replay in educational games. Burger et al. [5] studied the effect of verbal feedback from a virtual agent on replay in the context of a brain-training game. They found that elaborated feedback increases, whereas comparative feedback decreases, the students' interest in future replay. They also found that negative feedback generated an immediate interest in replay, whereas positive feedback created long term interest in the educational content. In another study, Plass et al. [14] compared three conditions in a math game: working individually, competing with another player, or collaborating with a peer. The study showed that both competition and collaboration modes heightened students' intention to replay when compared with the individual mode, with the latter result being statistically significant. However, both studies measured replay via questionnaires asking the students' desire to play the entire game again instead of observed replay behavior. Moreover, these studies sought to understand replay only from the angle of game design, and did not address the connections, if any, between student characteristics and interest in replay.

Other studies suggest elective replay is a habitual behavior that arises from individual need, although these studies did not directly investigate replay. Bartle [3] found one type of player who is primarily motivated by concrete measurements of success. In ST Math, these *achiever-type* players may largely use replay to get better 'scores' (losing fewer lives when passing a level). Mostow et al. [12] observed a student in a reading tutor who used the learner-control features to spend the majority of time replaying stories or writing "junk" stories instead of progressing to new material. Thus, some students may also use replay as a form of work avoidance – playing already passed levels instead of solving the current problem or moving on. Sabourin et al. [17] found that students in an educational game used off-task behaviors to cope with frustration, implying that off-task behavior can be a productive self-regulation of negative emotions. In ST Math, when students get frustrated with the current educational content but still have to play the game in the classroom, they may replay already learned content as a mental break from the current task. These studies showed that the circumstances of replay and students' characteristics influence their decisions to replay and its outcomes.

2.2 The Outcomes of Replay

Despite the believed benefits of replayability [9, 18, 10, 4], few studies have investigated the educational impact of elective replay. Boyce et al. [4] evaluated the effects of game elements that were designed to motivate gameplay and elective replay. These included a leaderboard that shows each student's rank based upon their score, a tool for creating custom puzzles, and a social system for messaging among players. The experimental design required students to play the game in one session, and to replay the game as more features were added in the subsequent sessions. The study found a sharp increase in test scores as these features were added to the game. The authors concluded that features designed to increase replayability can increase learning gains. However, this result may be due to increased time on task as the same group replaying the base game with new features. In another study, Clark et al. [7] analyzed logged student-initiated elective replay in a digital game. They found that frequency of elective replay did not correlate with learning gains, prior gaming habits/experience, or how much students liked the game. They also found that, while there was no statistically significant difference between the male and female students, males replayed more than the females. This may have been responsible for their slightly higher, although not statistically significant, "best level scores" – the highest score received on each level. These studies showed that elective replay may lead to increased learning or higher in-game performance. However, more research is needed to understand the potential educational impact of replay in educational games, particularly elective replays initiated solely by the players.

3. GAME, DATA AND FEATURES

3.1 ST Math Game

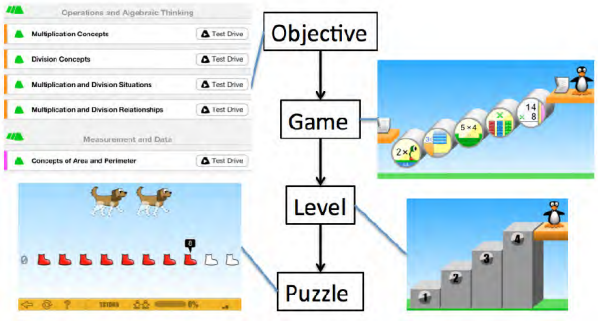


Figure 1: ST Math Content and Examples

ST Math is designed to act as a supplemental program to a school's existing mathematics curriculum. ST Math is mostly played during classroom sessions, but students have the option to play it at home. In ST Math [16], mathematics concepts are taught through spatial puzzles within various game-like arenas. ST Math games are structured at the top level by objectives, which are broad learning topics. Within each objective, individual games teach more targeted concepts through presentation of puzzles, which are grouped into levels for students to play. Students start by completing a series of training games on the use of the ST Math platform and features. They are then guided to complete the first available objective in their grade-level curriculum, such as "Multiplication Concepts." Students can only see this objective and must complete a pre-test before beginning the content. Games represent scenarios for problem-solving using a particular mathematical concept, such as "finding the right number of boots for X animals of Y legs." Each game contains between one and ten levels, which follow the same general structure of the game, but increase in difficulty. Figure 1 illustrates the hierarchy of ST Math content and examples.

As with many games, the student is given a set number of 'lives' at the start of each level. Every time they fail to

complete a puzzle correctly they lose one life. If all of their lives for a given level are exhausted, they will fail the level and be required to restart the level with a new set of lives. Once a student has passed a level, they can elect to replay it at any time. After a student has passed every level in an objective, they can take the objective post-test. Students cannot progress to the next objective until they have completed the last objective post-test. Both the objective pre- and post-tests consist of 5-10 multiple choice questions related to the objective. The post-tests parallel the pre-tests in both the question format and difficulty of the content. While answering each question in both tests, students indicate their relative confidence in their answer (low/high).

3.2 Data

MIND Research Institute (MIND), the developers of ST-Math, collected and provided to the researchers gameplay data from 4,827 3rd-5th graders during the school year 2012-2013. These students came from 17 schools and 221 classrooms. Table 1 summarizes students' demographic information. These demographic data, together with students' state standardized test scores in 2012 and 2013, were matched to gameplay data through anonymized IDs.

Table 1: Populations' Demographics Information

	Grade3	Grade4	Grade5
#Students	1567	1528	1732
Male	50.6% na:2.9%	50.1% na:2.0%	52.2% na:3.5%
Eligible for Reduced Lunch	80.7% na:2.9%	77.8% na:2.1%	81.4% na:3.2%
Hispanic or Latino	84.7% na:2.8%	82.3% na:1.9%	83.5% na:3.1%
English Language Learner	66.2% na:2.9%	56.1% na:2.1%	53.0% na:3.2%
with Listed Disability	10.9% na:2.1%	11.5% na:1.7%	11.9% na:2.8%

This gameplay data includes pre- and post-tests for each objective and the number of level attempts. For each pre- and post-test, ST Math logged students' accuracy and self-reported confidence level (1 for 'high' and 0 for 'low') for each question. For each play at a level, ST Math logged the student's ID, timestamp, and the number of puzzles completed. From these data, we identified ER as plays made after a student initially passed the level. We found ERs in 89.6% of all objectives in ST Math, accounting for 1.48% of all level attempts. Among 4,827 students, 59.85% ERed at least one level, with an average of 7.84 levels (SD=12.99, 95% CI [7.37, 8.32]) across 3.06 average objectives replayed per student. In the next section, we describe the features we created to analyze ER.

3.3 Features

We created features at three different levels of granularity (from finest to largest): level, objective, and student. For the level granularity, we treated each unique student-level combination as an observation. We calculated the features by averaging all gameplay for a specific student at a specific level. For objective granularity, each unique student-objective combination was treated as a single observation. Features were created by averaging across all levels played by a specific student within a single objective. The objective granularity also included the objective pre- and post-test accuracy and confidence. For the student granularity, we treated each student as a single observation. We calculated the features by averaging across all objectives played by a student over the entire year. The student granularity also included student demographic data and state standardized math test scores. These granularities ensured that our analysis did not favor units with the majority of data logs. Each student was considered equally in our analysis, regardless of how many objectives they played. Our data contained 4,827 students and 2,524,681 plays, which yielded 1,462,660 student-level observations, and 74,985 student-objective observations.

Table 2 shows five example plays of "Division-Level3," including four pass attempts and one ER of this level, interspersed with ERs from other levels. We consider consecutive ERs as an ER Session, as these ERs are circumstanced on the same pass attempts.

Table 2: Example of ER and Pass Attempts

Play	Objective-Level	Passed?	Play Type
1	Division- Level3	No	Pass Attempt
2	Division- Level3	No	Pass Attempt
3	Division-Level1	Yes	ER (ER Session1)
4	Division- Level3	No	Pass Attempt
5	Division-Level1	Yes	ER (ER Session2)
6	Division- Level3	Yes	Pass Attempt
7	Division- Level3	Yes	ER (ER Session3)
8	Subtraction-Level1	No	ER (ER Session3)

3.3.1 *Pass Attempt Features*

We defined performance to be the percentage of puzzles a student completed before losing all lives on the level. Pass attempts are plays prior to ER, where we assumed students play with the intention of passing the level. Pass attempt features included: performance when a student first attempted a level (*1st pass attempt performance*), number of attempts taken to pass a level (*# pass attempts*), and average performance of all pass attempts (*average pass attempt performance*). At the student granularity, students took an average of 1.91 (sd=0.89) attempts to pass each level, with average performance of 0.80 (sd=0.10) on the first pass attempt, and 0.87 (sd=0.07) on all pass attempts (indicating overall improved performance on later attempts).

3.3.2 *Elective Replay Features*

Table 3 shows ER features that describe ER from three angles: (I) the frequencies of ER, (II) the performance of ER, and (III) the circumstances of ER in terms of the ER's prior plays. To summarize, the majority of ERs had higher performance than their levels' first attempt, and resulted in another pass of their levels. Levels that were ERed had similar performance compared to levels that weren't ERed, but levels that were followed(54.65%) or interrupted (54.35%) by ER had much lower performance than those that weren't followed or interrupted by ER. Most ERs' immediately prior pass attempts were from different levels or objectives. There were few instances (9.80%) where students passed a level and immediately ERed it following the pass.

Table 3: Elective replay (ER) Features and their Descriptive Statistics among Students who Electively Replayed, Collapsed to the Student Granularity.

ER Features	Descriptive Stats
I. Frequencies of ER	
% ER out of all plays	M=2.40%, SD=4.26%
% Objectives that have been electively replayed	M=22.94%, SD=20.89%
% Objectives whose pass attempts were interrupted/followed by ER	M=19.48%, SD=17.57%
II. Performance of ER	
Performance of ER	M=0.71, SD=0.28
% ERs performed better than the level's first attempt	M=71.96%, SD=31.44%
% ERs that result in another pass of the level	M=60.36%, SD=35.51%
III. Circumstances of ER	
The Replayed Level E.g. "Division-lvl1," "Division-lvl3," and "Subtraction-lvl1" in Table 2	
Pass Attempts Features	M=0.79, 1.98, 0.87 for 1st performance, #pass attempts, and avg performance
The Immediately-Prior play of the ER E.g. Play 2 is the immediately-prior play of play 3 in Table2	
Performance on the immediately-prior play	M=0.63, SD=0.29
% ERs whose immediately-prior plays is also an ER	M=0.31, SD=0.28
% ER whose immediately prior pass attempt is on the same level	M=9.80%, SD=23.84%
% ..... on a different level in the same objective	M=40.75%, SD=39.09%
% ..... on a different objective	M=49.44%, SD=40.76%
The Immediate Prior Pass Attempts followed or interrupted by ER and ER Session E.g. "Division-lvl3" for all ER Sessions in Table 2	
Pass Attempts Features	M=0.51, 3.62, 0.55 for 1st performance, #pass attempts, and avg performance
% ER sessions whose prior pass attempt passed the level	M=45.65%, SD=40.69%

Note. statistics are reported at the student granularity, which are calculated through averaging across all objectives played by a student, and then averaged across all students who electively replayed. This means each student contributes equally to the average, regardless of how many objectives s/he played.

3.3.3 Student Grouping From ER Features

We created student groups to encapsulate the circumstances under which ER occurred, based on students' majority ER and ER sessions. Based on prior literature, we hypothesized that ER is a habitual behavior that arises from individual needs, such as gaining higher scores [3], avoiding progress on the current task [12], or taking a mental break from negative emotions [17]. Thus, grouping students based upon the circumstances of replay based on their majority behaviors provides high level profiles to investigate characteristics of students who engaged in ER and benefited from ER.

We characterized ER by the timing relative to the student's current learning objectives and gameplay. The first grouping describes whether the majority ER sessions started before (Group B) or after (Group A) passing the previous attempted level (current learning objective). If there is a tie between the two types of replay session, the student belongs to neither group. For example, Table 2 describes a group B student, who has two replay sessions before passing "Division-level3," and one replay session after passing this level but before moving on to the next level.

The second grouping describes whether an ER followed plays on the same level (SL), a different level under the same objective (DLSO), or a different objective (DO). For our example in Table 2, the student's pass attempts on "Division-Level3" was interrupted twice on the third and fifth plays, by replays on "Division-level1"(DLSO). After passing "Division-level3", the student replayed the same level(SL) once during the seventh play, and a different objective " Subtraction-level1" (DO) once during the eighth play. This Group B student had two DLSO replays, one SL, and one DO replays. Thus, this student also belongs to Group DLSO, because the two groupings are independent of each other.

4. METHODS & RESULTS

4.1 Who Engaged in Elective Replay?

We first investigated the demographic characteristics of students who engaged in elective replay. We found that males did so more often than females (male: 63.2%, female: 57.0%, $c2(1, N=4827) = 17.99$, $p<.001$). We also found that English Language Learners (ELL) did so more often than their non-ELL peers (ELL: 62.3%, non-ELL: 57.1%, $c2(1, N=4827) = 12.69$, $p<.001$), as did students with reported disabilities (disability: 68.7%, non disability: 59.1%, $c2(1, N = 4827) = 18.17$, $p<.001$). There were no statistically significant differences in the frequencies of ER based on race when operationalized as Hispanic/non Hispanic, or based on free/reduced lunch eligibility. The frequency of ER was not found to be correlated with other out-of-game student factors, such as state standardized math test scores.

The frequency of ER was also not correlated with in-game pre-test accuracy and confidence at the objective granularity. Next, we investigated the gameplay characteristics of students who electively replayed. We first separated students into groups based on their replay patterns. The first

Table 4: Mann-Whitney U Tests Comparing Gameplay Characteristics between ER Pattern Student Groups

Group (# students)	Pre-test Accuracy	Pre-test Confidence	Avg Pass Attempts' Performance	Avg 1st Attempt Performance	#Pass Attempts	ER Performance
Base:No ER (N=1938)	M=0.61 SD=0.17	M=0.75 SD=0.23	M=0.88 SD=0.08	M=0.81 SD=0.11	M=1.82 SD=0.84	NA
ER (N=2889)	*M=0.57 SD=0.17	M=0.74 SD=0.24	*M=0.87 SD=0.07	*M=0.80 SD=0.10	*M=1.92 SD=0.78	M=0.72 SD=0.29
Group A (N=1114)	M=0.62 SD=0.16	M=0.77 SD=0.22	*M=0.90 SD=0.05	*M=0.84 SD=0.08	*M=1.62 SD=0.52	*M=0.77 SD=0.27
Group B (N=1464)	*M=0.52 SD=0.17	*M=0.72 SD=0.25	*M=0.84 SD=0.07	*M=0.75 SD=0.09	*M=2.28 SD=1.09	*M=0.67 SD=0.29
Group SL (N=173)	M=0.61 SD=0.17	M=0.75 SD=0.23	M=0.88 SD=0.07	M=0.81 SD=0.09	M=1.82 SD=0.81	*M=0.84 SD=0.29
Group DLSO (N=983)	*M=0.54 SD=0.18	M=0.73 SD=0.24	*M=0.84 SD=0.08	*M=0.76 SD=0.10	*M=2.27 SD=1.16	*M=0.67 SD=0.32
Group DO (N=1399)	*M=0.58 SD=0.16	M=0.75 SD=0.23	M=0.88 SD=0.06	M=0.81 SD=0.08	M=1.80 SD=0.71	M=0.73 SD=0.26

Note. 1) Green and red indicate statistically significances higher and lower than the base class, with $*p < .001$, $+p < .01$ 2) Group A, B: most ER sessions happened before (B), after (A) passing the prior non-replay level. Group SL, DLSO, DO: most ER followed pass attempts on the same level(SL), different level in same objective(DLSO), or different objective (DO)

5 columns of Table 4 shows the results of Mann-Whitney U tests with Benjamini-Hochberg correction to compare each group in-game performance to the students who never electively replayed any levels (the Base group). The last column compares the averaged ER performance of each group to the rest of students who electively replayed.

Compared to the base group, students for whom most replays happened before passing the prior non-replay level (Group B) and students for whom most replays followed a different level on the same objective (Group DLSO) started with significantly lower pre-test scores and did worse in gameplay, as measured by the three pass attempt features described in section 3.3.2. For example, students in Group B started with lower accuracy and confidence at pre-test, took an average 0.5 more attempts to pass a level, and had lower performance on the 1st pass attempt and all pass attempts (including the 1st). It seems that Group B students who replayed earlier levels before passing the current one had less prior knowledge, and struggled more in the game. By contrast, students in Group A, for whom most replay happened after passing the current level, did slightly better in gameplay compared to students who never electively replayed (the Base group). Because these students started with pre-test scores that were not statistically significantly different from the base group, their replay patterns are associated with higher gameplay performance.

4.2 What Gets Replayed, and When?

Next, we studied what levels get replayed, and under what circumstances. We used a decision tree classifier which allowed us to identify which factors are most important in relative to ER. Our goal was not to find precise predictive models, but to augment our understanding of performance and its relationship to ER. We used R's *rpart* package with parameters minsplit=5% and cp=0.02 to build trees to classify levels that were replayed from levels that were not replayed, and levels whose pass attempts were interrupted or followed by replay from levels that were not interrupted or followed by replay. We randomly undersampled the majority class (levels without replay, levels were not interrupted or followed by replay), so that each class represented half of the observations. We used pass attempt features at the level granularity together with pre-test results, objective, and demographic information to build our tree. We used 10-fold cross validation to access the trees' accuracies.

Table 5 reports the trees and the importance of the features. We found that a student's performance on a particular level influenced whether replay happened during/after the level's pass attempts. For example, a student was more likely to replay a different level under the same objective (DLSO) if they took more than two attempts to pass the current level. This result is related to the previous result in Table 4, showing that, at the student level, those with lower gameplay performance were more likely to replay another level under the same objective.

On the other hand, the objective to which a level belongs influences whether or not a level would be ERed. We built trees to predict if a level is replayed following the same level (same condition of the last row in Table 5, N=1,776), the same objective but a different level (N=12,616), or a different objective (N=31,852). For all three conditions, the trees only contains a single node – objective, with accuracy of 55.2%, 62.0%, and 66.9% respectively. This ER decision could have been influenced by either the content or timing of the objectives. In our tree node, we noticed that many objectives with a higher chance of ER occurred earlier in the curriculum, this could be because students had more time in which these objectives were available for ER. Our tree model also had only 55.2% accuracy when predicting whether a level would be ERed following the pass attempts of itself. One explanation is that we do not have puzzle granularity data on how many lives a student actually lost. From prior literature [4] [7], students may replay the same

Table 5: Decision Trees to Predict Levels whose Pass Attempts were Interrupted or Followed by ER

Condition: interrupted/followed by	**Trees**
ER from a different level in the same objective (N=8,094)	77.8% accuracy #pass attempts $<$ 2.5, No #pass attempts $\geq$ 2.5, Yes
ER from a different objective (N=12,506)	78.7% accuracy 1st attempt performance $\geq$ 0.94 -objective group A, No -objective group B, Yes 1st attempt performance $<$ 0.94 -objective group A —# pass attempts $<$ 6.5, No —# pass attempts $\geq$ 6.5, Yes -objective group B, Yes
ER on the same level (N=1,766)	55.2% accuracy objective group A, No objective group B, Yes

Note. Trees are presented in text format. For example, the first tree shows that if a student passed a level with less than 2.5 pass attempts, the tree predicts this student will not replay another level during/after this level.

level following it pass attempts to get a better score, which means losing fewer lives (making fewer errors) at a level. As shown in Table 4, Group SL students who performed most of their ERs after the same level also achieved the highest ER performance.

4.3 Is Elective Replay Associated with Gains?

In this section we will address our second research question. As part of our analysis we considered three gain scores: accuracy gain, confidence gain, and math gain. The first two were measured by in-game pre- and post-tests. Recall that both before and after a student attempts an objective, ST Math logs the students' correctness and confidence scores on each question on the pre- and post-tests. We averaged these scores across the pre- and post-test questions to compute the first two gain scores. These were assessed at the objective granularity. Math gain was calculated based upon the difference between the students' state standardized math test scores in years 2012 and 2013. This was assessed at the student granularity.

11.8% of the students were excluded from the math gain analysis due to missing state math test records. These excluded students performed statistically significantly worse in the game as measured by the three pass attempt features; this implies that we excluded weaker students. 8.5% of the objective observations were excluded from the accuracy and confidence gain analysis due to missing pre- or post-tests. These excluded observations were not statistically significantly different from the rest as measured by pass attempt features. The accuracy and confidence gains were significantly correlated (r=0.37, p<0.001), but these two gains were not strongly correlated with math gain scores at the student granularity (r<0.1, p<0.001). Table 6 reports the percentage of data points that gained, dropped (mainly for avoiding ceiling effect in this data), and did not gain for each

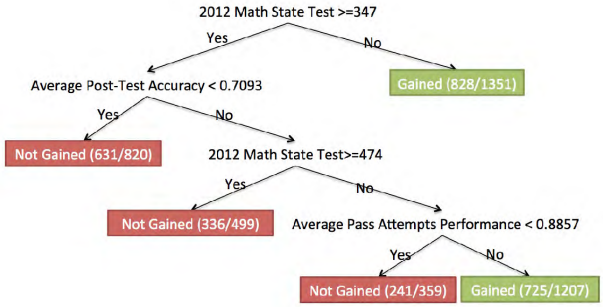


Figure 2: Decision Tree to Predict Whether a Student will Gain in State Standardized Math Test

type of gain based on the Marx and Cummings Normalization method [11].

Table 6: %Observations with Gains, No Gains, and Percentage Dropped for the Three Gains

Gain Types	**ER?**	**Gained**	**Dropped**	**No Gain**
Accuracy	ER	48.10%	8.60%	37.90%
(N=75,083)	No ER	43.70%	6.10%	36.60%
Confidence	ER	28.30%	42.60%	23.70%
(N=75,083)	No ER	26.40%	37.40%	22.70%
Math Test	ER	41.60%	0.40%	46.90%
(N=4,827)	No ER	40.80%	0.50%	45.70%

Note. 1)Observations in the 'Dropped' column (pre- and post-tests were both 0 or 1) were excluded from analysis. 2)Accuracy and Confidence Gains were measured at objective granularity, Math gain was measured at student granularity. 3)ER and no ER were collapsed across level.

We first constructed decision trees to partition our data to see which factors influence gains, using the method described in the prior section. No sampling was necessary because the groups had similar sizes. We used pass attempt features, ER features, pre-test results, and demographics. For student granularity, we also added the percentage of required objectives attempted by the student.

At the objective granularity, we found that pre-test accuracy and confidence were the only selected nodes that predicted accuracy (70.0% accuracy) and confidence gain (74.1% accuracy). Students with a pre-test accuracy of $<$ 0.71 (at least 2 questions wrong out of 5-10) had a 64.7% chance of positive accuracy gain in the same objective, while the remainder of the students had only a 25.9% chance. Students with high pre-test confidence ($\leq$0.95, indicated confidence on almost all questions) had a 62.5% chance of positive confidence gain in the same objective. It could be that these in-game tests were too easy, as 18.9% of pretests achieved full scores in accuracy and 54.5% achieved full scores in confidence.

Our decision tree for the student granularity is shown in Figure 2, with a cross-validated accuracy of 57.8%. Students who started with medium level of math abilities (2012 state test math scores $<$474, and $\geq$ 347) improved their scores when they performed well in ST Math (average pass attempts performance $>$ 0.8857). This shows that the game-

play data in ST Math has predictive power for assessment outside of the game. However, for all three gain scores, the ER features were not selected for inclusion in the decision tree nor was any correlation found with the students gains.

Table 7: Mann-Whitney U Tests Comparing Gains between ER Pattern Student Groups.

Group (# students)	Math (max=600)	Accuracy (max=1)	Confidence (max=1)
Base:No ER (N=1938)	M=31.5 SD=146.6	M=0.31 SD=0.25	M=0.33 SD=0.38
ER (N=2889)	M=27.3 SD=139.7	M=0.30 SD=0.25	M=0.32 SD=0.37
Group A (N=1114)	M=53.4 SD=167.9	*M=0.35 SD=0.24	+M=0.38 SD=0.36
Group B (N=1464)	+M=6.7 SD=109.0	*M=0.24 SD=0.25	*M=0.26 SD=0.37
Group SL (N=173)	M=46.2 SD=161.2	M=0.31 SD=0.28	M=0.31 SD=0.37
Group DLSO (N=983)	M=21.4 SD=123.0	*M=0.25 SD=0.26	*M=0.27 SD=0.37
Group DO (N=1399)	M=32.3 SD=150.6	M=0.32 SD=0.23	M=0.34 SD=0.36

Note. green and red indicate statistically significances higher and lower than the base class, with $^*p < .001$, $+p < .01$

Finally, we investigated how ER patterns relate to gains. Table 7 reports the result from separating students into 6 groups based on ER patterns and conducting Mann-Whitney U tests with Benjamini-Hochberg correction (as in the previous section). Moreover, although decision trees constructed from the complete dataset show that low pre-test results led to more gains, some ER pattern groups showed opposite trends. For example, Group B, who primarily ERed before passing the current level, started with lower pre-test scores, did worse in the game, and had less gains, which were statistically significant, in all three gain measures. The same applies to Group DLSO. These two groups of students also had the lowest ER performance.

On the other hand, the Base group and Group A (who mostly ERed after passing the current level) started with pre-test accuracy and confidence scores that are not significantly different (Table 4), but Group A did significantly better in game, and had higher gains in accuracy and confidence, which were statistically significant. Because the mean pre-test score for the Base and A groups is approximately 0.6, these students were reasonably familiar with the objective before they began playing it. The difference in accuracy and confidence gains suggest that ER after students successfully pass a level helped students learn, or implied better learning in the previous gameplay.

5. DISCUSSION AND CONCLUSIONS

This work presents a significant extension on prior studies of replay which have typically taken place over a short period of time and have assessed replay via intentional questionnaires not observed behaviors [14, 5]. This work analyzed logged student-initiated elective replay from a sample of 4,827 3rd-5th graders during school year 2012-2013 in ST Math in a natural educational setting. We sought to answer three research questions: Q1: What are the characteristics of students who electively replay? Q2: What gets replayed, and under what circumstances? And Q3: Is elective replay associated with improvements in students' accuracy on math objectives, confidence, and general math ability?

We concluded that, with over half of students who electively replayed at least one level, ER is a common behavior in ST Math. Moreover, examining elective replay can enhance our understanding about how students play and the characteristics of successful play. For example, we found that students who did poorly on the current level were more likely to electively replay a different level during/after the level's pass attempts. We also found that students who generally engaged in elective replay before passing the current level (Group B) started with lower pre-test scores, did worse during gameplay, and had the lowest objective-level accuracy and confidence gain and math gains. One explanation for this result is that weaker students used ER as a work avoidance tactic, as found in Mostow et al. [12], and that instances of ER stand in for lower motivation or engagement for the objective topic, ST Math, or mathematics overall.

On the other hand, compared to students who didn't ER, students who mostly electively replayed after passing the current level (Group A) started with pre-test scores that were not significantly different, did better in the game, and had higher learning and confidence gains. One reason could be that these students electively replayed for a better score, as we also found that students who mostly replayed the same level immediately after passing it (Group SL) had the highest ER performance. This association is especially true among achiever-type players [3] that prefer to gain concrete measurements of success. Because losing fewer lives in ST Math requires better mastery of the math content, ER may have helped these students learn. Another explanation is that these students' ERs could imply better learning during prior gameplay, as Table 4 also shows that Group A students had better pass attempt performance. Possibly, successful prior performance motivated these students to electively replay more of the game. Moreover, because successful prior performance feeds self-efficacy [2, 13], confidence gains in Group A students, who chose more ER, may be linked to electively replaying levels they have already mastered.

From the application perspective, as expected from this complex environment, our effect-sizes are too small to claim ER itself as a powerful intervention for learning. Instead, our findings suggest the potential of using ER patterns to identify weaker students and their struggling moments for intervention. For example, students with Group B ER patterns started weaker, did poorly in the game, and had lower gains in learning, confidence, and math state test scores. It may be the case that Group B ER (before passing a level) is a signal that students are struggling in current content and are in need of a mental break [17] or help. If this is the case, it would be beneficial upon detecting these ER patterns for ST Math to alert teachers or to provide interventions, such as suggesting the student to take a break or providing supplemental resources to further explain the math concepts from the pass attempts interrupted by ER. Our results also suggest avenues for experimental studies that designs a more effective ER experience, such as preventing work-avoidance

in ER. For example, changing the number of lives students have at each replay, or constraining the problems offered each time they are replayed to be isomorphic but not identical.

This work has several limitations. First, the in-game pre-post- tests may be too easy for students, as 18.9% of pretests achieved a full score in accuracy, and 54.5% achieved a full score in confidence. The high percentage of students with non-positive learning and accuracy gain could also be caused by students' slipping or guessing in multiple-choice questions (e.g., 1 incorrect answer reduces accuracy by 14%-20%). The accuracy of the pre- and post-test questions for assessing knowledge might be improved by using short answer questions. The second limitation is that we did not have puzzle granularity data on how many lives a student actually lost or the types of errors they made. Third, the grouping of students based on the majority of elective replay assumes that elective replay is a habitual and consistent behavior. Future research should investigate other groupings, as well as examining whether there were changes in how students used replay, and what caused the changes. Fourth, future work may also include creating quantified features to compare the content and game features across objectives so we may better understand how the game's content influence students' decision to engage in elective replay.

In summary, this work adds new insights to our understanding of elective replay in educational games. Our work reveals differential associations between elective replay and performance when replay is categorized by the timing in relation to the student's current learning objectives and gameplay. Our work suggests that low-performing students did not benefit from ER; high-performing students both chose ER at better times and their ERs were associated with benefits from either ER or previous gameplay, which supports the results of prior self-regulation research by Aleven et al [1]. This work presents prospects for both examining more detailed characteristics of replay and utilizing experimental manipulations.

6. ACKNOWLEDGEMENTS

This work was supported by NSF grant IUSE #1544273 "Evaluation for Actionable Change: A Data-Driven Approach" Teomara Rutherford PI, Tiffany Barnes & Collin F. Lynch Co-PIs.

7. REFERENCES

[1] V. Aleven, E. Stahl, S. Schworm, F. Fischer, and R. Wallace. Help seeking and help design in interactive learning environments. *Review of Educational Research*, 73(3):277–320, 2003.

[2] A. Bandura. Perceived self-efficacy in cognitive development and functioning. *Educational Psychologist*, 28:117–148.

[3] R. Bartle. Hearts, clubs, diamonds, spades: Players who suit muds. *Journal of MUD research*, 1(1):19, 1996.

[4] A. Boyce, K. Doran, A. Campbell, S. Pickford, D. Culler, and T. Barnes. Beadloom game: Adding competitive, user generated, and social features to increase motivation. In *the 6th International Conference on Foundations of Digital Games*, pages 139–146. ACM, 2011.

[5] C. Burgers, A. Eden, M. D. van Engelenburg, and S. Buningh. How feedback boosts motivation and play in a brain-training game. *Computers in Human Behavior*, 48:94–103, 2015.

[6] S. L. Calvert, B. L. Strong, and L. Gallagher. Control as an engagement feature for young children's attention to and learning of computer content. *American Behavioral Scientist*, 48(5):578–589, 2005.

[7] D. B. Clark, B. C. Nelson, H. Y. Chang, M. Martinez-Garza, K. Slack, and C. M. D'Angelo. Exploring newtonian mechanics in a conceptually-integrated digital game: Comparison of learning and affective outcomes for students in taiwan and the united states. *Computers Education*, 57(3):2178–2195, 2011.

[8] D. I. Cordova and M. R. Lepper. Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice. *Journal of Educational Psychology*, 88(4):715, 1996.

[9] J. P. Gee. *What video games have to teach us about learning and literacy.* St. Martin's Griffin - Macmillan, New York, USA, 2007.

[10] G. A. Gunter, R. F. Kenny, and E. H. Vick. Taking educational games seriously: Using the retain model to design endogenous fantasy into standalone educational games. *Educational Technology Research and Development*, 56(5-6):511–537, 2008.

[11] J. D. Marx and K. Cummings. Normalized change. *American Journal of Physics*, 75(1):87–91, 2007.

[12] J. Mostow, J. Beck, R. Chalasani, A. Cuneo, P. Jia, and K. Kadaru. A la recherche du temps perdu, or as time goes by: Where does the time go in a reading tutor that listens? In *In International Conference on Intelligent Tutoring Systems*, pages 320–329, 2002.

[13] F. Pajares. Self-efficacy beliefs in academic setting. *Review of Educational Research*, 66:543–578.

[14] J. L. Plass, P. A. O'keefe, B. D. Homer, J. Case, E. O. Hayward, M. Stein, and K. Perlin. The impact of individual, competitive, and collaborative mathematics game play on learning, performance, and motivation. *Journal of Educational Psychology*, 105(4):1050, 2013.

[15] M. Prensky. Computer games and learning: Digital game-based learning. In *Handbook of computer games studies*. The MIT Press, Cambridge, MA, USA, 2005.

[16] T. Rutherford, G. Farkas, G. Duncan, M. Burchinal, M. Kibrick, J. Graham, L. Richland, N. Tran, S. Schneider, L. Duran, and M. Martinez. A randomized trial of an elementary school mathematics software intervention: spatial-temporal math. *Journal of Research on Educational Effectiveness*, 7(4):358–383, 2014.

[17] J. L. Sabourin, J. P. Rowe, B. W. Mott, and J. C. Lester. Considering alternate futures to classify off-task behavior as emotion self-regulation: A supervised learning approach. *JEDM-Journal of Educational Data Mining*, 5(1):9–38, 2013.

[18] S. Thomas, G. Schott, and M. Kambouri. Designing for learning or designing for fun? setting usability guidelines for mobile educational games. *Learning with mobile devices: A book of papers*, pages 173–181, 2004.

Grade Prediction with Temporal Course-wise Influence

Zhiyun Ren
Computer Science
George Mason University
4400 University Drive,
Fairfax, VA 22030
zren4@gmu.edu

Xia Ning
Computer & Information Science
Indiana University - Purdue University Indianapolis
420 University Blvd,
Indianapolis, IN 46202
xning@cs.iupui.edu

Huzefa Rangwala
Computer Science
George Mason University
4400 University Drive,
Fairfax, VA 22030
rangwala@cs.gmu.edu

ABSTRACT

There is a critical need to develop new educational technology applications that analyze the data collected by universities to ensure that students graduate in a timely fashion (4 to 6 years); and they are well prepared for jobs in their respective fields of study. In this paper, we present a novel approach for analyzing historical educational records from a large, public university to perform next-term grade prediction; i.e., to estimate the grades that a student will get in a course that he/she will enroll in the next term. Accurate next-term grade prediction holds the promise for better student degree planning, personalized advising and automated interventions to ensure that students stay on track in their chosen degree program and graduate on time. We present a factorization-based approach called Matrix Factorization with Temporal Course-wise Influence that incorporates course-wise influence effects and temporal effects for grade prediction. In this model, students and courses are represented in a latent "knowledge" space. The grade of a student on a course is modeled as the similarity of their latent representation in the "knowledge" space. Course-wise influence is considered as an additional factor in the grade prediction. Our experimental results show that the proposed method outperforms several baseline approaches and infer meaningful patterns between pairs of courses within academic programs.

Keywords

next-term grade prediction, course-wise influence, temporal effect, latent factor

1. INTRODUCTION

Data analytics is at the forefront of innovation in several of today's popular Educational Technologies (EdTech) [17]. Currently, one of the grand challenges facing higher education is the problem of student retention and graduation [19]. There is a critical need to develop new EdTech applications that analyze the data collected by universities to ensure that students graduate in a timely fashion (4 to 6 years), and they are well prepared for jobs in their respective fields of study. To this end, several universities deploy a suite of software and tools. For example, *degree planners* [1] assist students in deciding their majors or fields of study, choosing the sequence of courses within their chosen major and providing advice for achieving career and learning objectives. *Early warning systems* [27] inform advisors/students of progress, and additionally provide cues for intervention when students are at the risk of failing one or more courses and dropping out of their program of study. In this work, we focus on the problem of next-term grade prediction where the goal is to predict the grade that a student is expected to obtain in a course that he/she may enroll in the next term (future).

In the past few years, several algorithms have been developed to analyze educational data, including Matrix Factorization (MF) algorithms inspired from recommender system research. MF methods decompose the student-course (or student-task) grade matrix into two low-rank matrices, and then the prediction of the grade for a student on an untaken course is calculated as the product of the corresponding vectors in the two decomposed matrices [22, 11]. Traditional MF algorithms have shown a strong ability to deal with sparse datasets [14] and their extensions have incorporated temporal and dynamic information [12]. In our setting, we consider that a student's knowledge is continuously being enriched while taking a sequence of courses; and it is important to incorporate this dynamic influence of sequential courses within our models. Therefore, we present a novel approach referred as Matrix Factorization with Temporal Course-wise Influence (MFTCI) model to predict next term student grades. MFTCI considers that a student's grade on a certain course is determined by two components: (i) the student's competence with respect to each course's topics, content and requirement, etc., and (ii) student's previous performance over other courses. We performed a comprehensive set of experiments on various datasets. The experimental results show that the proposed method outperforms several state-of-the-art methods. The main contributions of our work in this paper are as follows:

1. We model and incorporate temporal course-wise influence in addition to matrix factorization for grade

[1] http://www.blackboard.com/mobile-learning/planner.aspx

prediction. Our experimental results demonstrate significant improvement from course-wise influence.

2. Our model successfully captures meaningful course-wise influences which correlate to the course content.

3. The learned influences between pairs of courses help in understanding pre-requisite structures within programs and tuning academic program chains.

2. RELATED WORK

Over the past few years, several methods have been developed to model student behavior and academic performance [2, 9], and they gain improvement of learning outcomes [21]. Methods influenced by Recommender System (RS) research [1], including Collaborative Filtering (CF) [18] and Matrix Factorization [13], have attracted increasing attention in educational mining applications which relate to student grade prediction [32] and in-class assessment prediction [8]. Sweeney *et. al.* [31, 30] performed an extensive study of several recommender system approaches including SVD, SVD-kNN and Factorization Machine (FM) to predict next-term grade performance. Inspired by content-based recommendation [20] approaches, Polyzou *et. al.* [23] addressed the future course grade prediction problem with three approaches: course-specific regression, student-specific regression and course-specific matrix factorization. Moreover, neighborhood-based CF approaches [25, 4, 6] predict grades based on the student similarities, i.e., they first identify similar students and use their grades to estimate the grades of the students with similar profiles.

In order to capture the changing of user dynamics over time in RS, various dynamic models have been developed. Many of such models are based on Matrix Factorization and state space models. Sun *et. al.* [28, 29] model user preference change using a state space model on latent user factors, and estimate user factors over time using noncausal Kalman filters. Similarly, Chua *et.al.* [5] apply Linear Dynamical Systems (LDS) on Non-negative Matrix Factorization (NMF) to model user dynamics. Ju *et. al.* [12] encapsulate the temporal relationships within a Non-negative matrix formulation. Zhang *et. al.* [34] learn an explicit transition matrix over the latent factors for each user, and estimate the user and item latent factors and the transition matrices within a Bayesian framework. Other popular methods for dynamic modeling include time-weighting similarity decaying [7], tensor factorization [33] and point processes [16]. The method proposed in this paper tackle the challenges of next-term grade prediction which relates to the evolvement of student knowledge over taking a sequence of courses. Our key contribution involves how we incorporate the temporal course-wise relationships within a MF approach. Additionally, the proposed approach learns pairwise relationships between courses that can help in understanding pre-requisite structures within programs and tuning academic program chains.

3. PRELIMINARIES

3.1 Problem Statement and Notations

Formally, student-course grades will be represented by a series of matrices $\{\mathrm{G}_1, \mathrm{G}_2, ..., \mathrm{G}_T\}$ for T terms. Each row of G_t represents a student, each column of G_t represents a course, and each value in G_t, denoted as $g^t_{s,c}$, represents a grade that student s got on course c in term t ($g^t_{s,c} \in (0, 4]$, $g^t_{s,c} = 0$ indicates that student s did not take the course c in term t. We add a small value to failing grade to distinguish 0 score from such situation.). Student-course grades up to the t_{th} term will be represented by $\mathrm{G}^t = \sum_{i=1}^{t} \mathrm{G}_i$ with size of $n \times m$, where n is the number of students and m is the number of courses. Given the database of (student, course, grade) up to term $(T-1)$ (i.e., G^{T-1}), the next-term grade prediction problem is to predict grades for each student on courses they might enroll in the next term T. To simplify the notations, if not specifically stated in this paper, we will use $g_{s,c}$ to denote $g^t_{s,c}$. Our testing set is then (student, course, grade) triples in the T_{th} term, represented by matrix G_T. Rows from the grade matrices representing a student s will simply be represented as $G(s,:)$ and the specific courses that student has a grade for in this row can be given by $c' \in G(s,:)$.

In this paper, all vectors (e.g., $\mathbf{u}_s^\mathsf{T}$ and $\mathbf{v}_c$) are represented by bold lower-case letters and all matrices (e.g., A) are represented by upper-case letters. Column vectors are represented by having the transpose supscript$^\mathsf{T}$, otherwise by default they are row vectors. A predicted/approximated value is denoted by having a $\tilde{}$ head.

4. METHODS

4.1 MF with Temporal Course-wise Influence

We consider the student s' grade on a certain course c, denoted as $g_{s,c}$, as determined by two factors. The first factor is the student s' competence with respect to the course c's topics, content and requirement. This is modeled through a latent factor model, in which s' competence is captured using a size-k latent factor $\mathbf{u}_s$, c's topics and contents are captured using a size-k latent factor $\mathbf{v}_c$ in the same latent space as $\mathbf{u}_s$. Then the competence of s over c is modeled by the "similarity" between $\mathbf{u}_s$ and $\mathbf{v}_c$ via their dot product (i.e., $\mathbf{u}_s^\mathsf{T}\mathbf{v}_c$).

The second factor is the previous performance of student s over other courses. We hypothesize that if course c' has a positive influence on course c, and student s achieved a high grade on c', then s tends to have a high grade on c. Under this hypothesis, we model this second factor as a product between the performance of student on a previous "related" course where the pairwise course relationships are learned in our formulation. Note that we consider this pairwise course influence as time independent, i.e., the influence of one course over another does not change over time. However, the impact from previous performance/grades can be modeled using a decay function over time. Taking these two factors, the estimated grade is given as follows:

$$\begin{aligned}\tilde{g}_{s,c} = \mathbf{u}_s^\mathsf{T}\mathbf{v}_c \\ + \underbrace{e^{-\alpha}\frac{\sum_{c' \in G_{T-1}(s,:)} A(c',c) g_{s,c'}}{|G_{T-1}(s,:)|}}_{\Delta(T-1)} \\ + \underbrace{e^{-2\alpha}\frac{\sum_{c'' \in G_{T-2}(s,:)} A(c'',c) g_{s,c''}}{|G_{T-2}(s,:)|}}_{\Delta(T-2)},\end{aligned} \tag{1}$$

in which $A(c', c)$ is the influence of c' on c, $G_{T-1}(s,:)/G_{T-2}(s,:)$ is the subset of courses out of all courses that s has taken in the first/second previous terms, $|G_{T-1}(s,:)|/|G_{T-2}(s,:)|$ is the number of such taken courses. $e^{-\alpha}/e^{-2\alpha}$ denote the time-decay factors. In Equation 1, we consider previous two terms. More previous terms can be included with even stronger time-decay factors. Given the grade estimation as in Equation 1, we formulate the grade prediction problem for term T as the following optimization problem,

$$\begin{aligned}\min_{U,V,A} \ & \frac{1}{2}\sum_{s,c}(g_{s,c} - \tilde{g}_{s,c})^2 + \frac{\gamma}{2}(\|U\|_F^2 + \|V\|_F^2) \\ & + \tau\|A\|_* + \lambda\|A\|_{\ell_1} \\ & \text{s.t., } A \geq 0\end{aligned}$$

where U and V are the latent non-negative student factors and course factors, respectively; $\|A\|_*$ is the nuclear norm of A, which will induce an A of low rank; and $\|A\|_{\ell_1}$ is the ℓ_1 norm of A, which will introduce sparsity in A. In addition, the non-negativity constraint on A is to enforce only positive influence across courses.

4.1.1 Optimization Algorithm of MFTCI

We apply the ADMM [3] technique for Equation 2 by reformulating the optimization problem as follows,

$$\begin{aligned}\min_{U,V,A,U_1,U_2,Z_1,Z_2} \quad & \frac{1}{2}\sum_{s,c}(g_{s,c} - \tilde{g}_{s,c})^2 + \frac{\gamma}{2}(\|U\|_F^2 + \|V\|_F^2) \\ & +\tau\|Z_1\|_* + \lambda\|Z_2\|_{\ell_1} \\ & +\frac{\rho}{2}(\|A - Z_1\|_F^2 + \|A - Z_2\|_F^2) \\ & +\rho(tr(U_1^\mathsf{T}(A - Z_1))) \\ & +\rho(tr(U_2^\mathsf{T}(A - Z_2))) \\ \text{s.t.,} \quad & A \geq 0\end{aligned}$$

where Z_1 and Z_2 are two auxiliary variables, and U_1 and U_2 are two dual variables. All the variables are solved via an alternating approach as follows.

Step 1: Update U and V. Fixing all the other variables and solving for U and V, the problem becomes a classical matrix factorization problem:

$$\min_{U,V} \frac{1}{2}\sum_{s,c}(f_{s,c} - \mathbf{u}_s^\mathsf{T}\mathbf{v}_c)^2 + \frac{\gamma}{2}(\sum_s \|u_s\|_2^2 + \sum_c \|v_c\|_2^2) \quad (2)$$

where $f_{s,c} = g_{s,c} - \Delta(T-1) - \Delta(T-2)$ (See Eq 1). The matrix factorization problem can be solved using alternating minimization.

Step 2: Update A. Fixing all the other variables and solving for A, the problem becomes

$$\begin{aligned}\min_{A} \quad & \frac{1}{2}\sum_{s,c}(g_{s,c} - \tilde{g}_{s,c})^2 + \frac{\rho}{2}(\|A - Z_1\|_F^2 + \|A - Z_2\|_F^2) \\ & +\rho(tr(U_1^\mathsf{T}(A - Z_1))) + \rho(tr(U_2^\mathsf{T}(A - Z_2))) \\ \text{s.t.,} \quad & A \geq 0\end{aligned}$$

Using the gradient descent, the elements in A can be updated as follows.

$$\begin{aligned}A(c_i, c_j) = A(c_i, c_j) - lr \times & [\rho(A(c_i, c_j) - Z_1(c_i, c_j)) \\ & + \rho(A(c_i, c_j) - Z_2(c_i, c_j)) + \rho U_1(c_i, c_j) + \rho U_2(c_i, c_j) \\ & - \sum_{s,c_j}(g_{s,c_j} - \tilde{g}_{s,c_j}) \\ & \times \begin{cases} \frac{e^{-\alpha}}{|G_{T-1}(s,:)|} g_{s,c_i} & \text{(if } c_i \text{ is taken in term } T-1\text{)} \\ \frac{e^{-2\alpha}}{|G_{T-2}(s,:)|} g_{s,c_i} & \text{(if } c_i \text{ is taken in term } T-2\text{)]} \end{cases}\end{aligned} \quad (3)$$

with projection into $[0, +\infty)$, where lr is a learning rate.

Step 3: Update Z_1 and Z_2. For Z_1, the problem becomes

$$\min_{Z_1} \tau\|Z_1\|_* + \frac{\rho}{2}\|A - Z_1\|_F^2 + \rho(tr(U_1^\mathsf{T}(A - Z_1))) \quad (4)$$

The closed-form solution of this problem is

$$Z_1 = S_{\frac{\tau}{\rho}}(A + U_1) \quad (5)$$

where $S_\alpha(X)$ is a soft-thresholding function that shrinks the singular values of X with a threshold α, that is,

$$S_\alpha(X) = U\text{diag}((\Sigma - \alpha)_+)V^\mathsf{T} \quad (6)$$

where $X = U\Sigma V^\mathsf{T}$ is the singular value decomposition of X, and

$$(x)_+ = \max(x, 0). \quad (7)$$

For Z_2, the problem becomes

$$\min_{Z_2} \lambda\|Z_2\|_{\ell_1} + \frac{\rho}{2}\|A - Z_2\|_F^2 + \rho(tr(U_2^\mathsf{T})(A - Z_2)) \quad (8)$$

The closed-form solution is

$$Z_2 = E_{\frac{\lambda}{\rho}}(A + U_2) \quad (9)$$

where $E_\alpha(X)$ is a soft-thresholding function that shrinks the values in X with a threshold α, that is,

$$E_\alpha(X) = (X - \alpha, 0)_+ \quad (10)$$

where $()_+$ is defined as in Equation 7.

Step 4: Update U_1 and U_2. U_1 and U_2 are updated based on standard ADMM updates:

$$U_1 = U_1 + (A - Z_1); \qquad U_2 = U_2 + (A - Z_2) \quad (11)$$

In addition, we conduct computational complexity analysis of MFTCI and put it in Appendix.

5. EXPERIMENTS

5.1 Dataset Description

We evaluated our method on student grade records obtained from George Mason University (GMU) from Fall 2009 to Spring 2016. This period included data for 23,013 transfer students and 20,086 first-time freshmen (non-transfer i.e., students who begin their study at GMU) across 151 majors enrolled in 4,654 courses.

Specifically, we extracted data for six large and diverse majors for both non-transfer and transfer students. These majors include: (i) Applied Information Technology (AIT), (ii)

Table 1: Dataset Descriptions

Major	Non-Transfer Students			Transfer Students		
	#S	#C	#(S,C)	#S	#C	#(S,C)
AIT	239	453	5,739	982	465	14,396
BIOL	1,448	990	33,527	1,330	833	22,691
CEIE	393	642	9,812	227	305	4,538
CPE	340	649	7,710	91	219	1,614
CS	908	818	18,376	480	464	7,967
PSYC	911	874	22,598	1504	788	24,661
Total	4,239	1,115	97,762	4,614	1,019	75,867

#S, #C and #S-C are number of students, courses and student-course pairs in educational records across the 6 majors from Fall 2009 to Spring 2016, respectively.

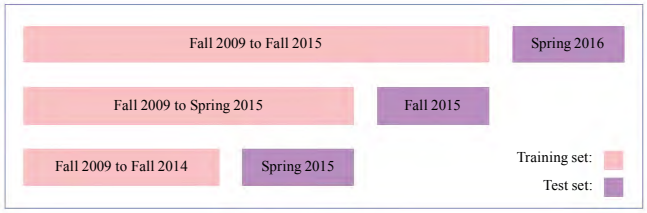


Figure 1: Different Experimental Protocols

Biology (BIOL), (iii) Civil, Environmental and Infrastructure Engineering (CEIE), (iv) Computer Engineering (CPE) (v) Computer Science (CS) and (vi) Psychology (PSYC). Table 1 provides more information about these datasets.

5.2 Experimental Protocol

To assess the performance of our next-term grade prediction models, we trained our models on data up to term $T-1$ and make predictions for term T. We evaluate our method for three test terms, i.e., Spring 2016, Fall 2015 and Spring 2015. As an example, for evaluating predictions for term Fall 2015, data from Fall 2009 to Spring 2015 is considered as training data and data from Fall 2015 is testing data. datasets. Figure 1 shows the three different train-test splits.

5.3 Evaluation Metrics

We use **Root Mean Squared Error (RMSE)** and **Mean Absolute Error (MAE)** as metrics for evaluation, and are defined as follows:

$$RMSE = \sqrt{\frac{\sum_{s,c \in G_T} (g_{s,c} - \tilde{g}_{s,c})^2}{|G_T|}},$$

$$MAE = \frac{\sum_{s,c \in G_T} |g_{s,c} - \tilde{g}_{s,c}|}{|G_T|}$$

where $g_{s,c}$ and $\tilde{g}_{s,c}$ are the ground truth and predicted grade for student s on course c, and G_T is the testing set of (student, course, grade) triples in the T_{th} term. Normally, in next-term grade prediction problem, MAE is more intuitive than RMSE since MAE is a straightforward method which calculates the deviation of errors directly while RMSE has implications such as penalizing large errors more.

For our dataset, a student's grade can be a letter grade (i.e. A, A-, ..., F). As done previously by Polyzou et. al. [24] we define a tick to denote the difference between two consecutive letter grades (e.g., C+ vs C or C vs C-). To assess the performance of our grade prediction method, we convert the predicted grades into their closest letter grades and compute the percentage of predicted grades with no error (or 0-ticks), within 1-tick and within 2-ticks denoted by Pct_0, Pct_1 and Pct_2, respectively. For the problem of course selection and degree planning, courses predicted within 2 ticks can be considered sufficiently correct. We name these metrics as **Percentage of Tick Accuracy (PTA)**.

5.4 Baseline Methods

We compare the performance of our proposed method to the following baseline approaches.

5.4.1 *Matrix Factorization*

Matrix factorization is known to be successful in predicting ratings accurately in recommender systems [26]. This approach can be applied directly on next-term grade prediction problem by considering student-course grade matrix as a user-item rating matrix in recommender systems. Based on the assumption that each course and student can be represented in the same low-dimensional space, corresponding to the knowledge space, two low-rank matrices containing latent factors are learned to represent courses and students [30]. Specifically, the grade a student s will achieve on a course c is predicted as follows:

$$\tilde{g}_{s,c} = \mu + \mathbf{p}_s + \mathbf{q}_c + \mathbf{u}_s^\top \mathbf{v}_c \quad (12)$$

where μ is a global bias term, $\mathbf{p}_s$ ($\mathbf{p} \in \mathbb{R}^n$) and $\mathbf{q}_c$ ($\mathbf{q} \in \mathbb{R}^m$) are the student and course bias terms (in this case, for student s and course c), respectively, and $\mathbf{u}_s$ ($\mathbf{U} \in \mathbb{R}^{k \times n}$) and $\mathbf{v}_c$ ($\mathbf{V} \in \mathbb{R}^{k \times m}$) are the latent factors for student s and course c, respectively.

5.4.2 *Matrix Factorization without Bias (MF_0)*

We only considered the student and course latent factors to predict the next-term grades. Therefore, the grade a student s will achieve on a course c is calculated as follows:

$$\tilde{g}_{s,c} = \mathbf{u}_s^\top \mathbf{v}_c \quad (13)$$

5.4.3 *Non-negative Matrix Factorization (NMF) [15]*

We add non-negative constraints on matrix $\mathbf{U}$ and matrix $\mathbf{V}$ in Equation 13. The non-negativity constraints allows MF approaches to have better interpretability and accuracy for non-negative data [10].

6. RESULTS AND DISCUSSION

6.1 Overall Performance

Table 2 presents the comparison of Pct_0, Pct_1 and Pct_2 for non-transfer students for the three terms considered as test: Spring 2016, Fall 2015 and Spring 2015. We observe that the MFTCI model outperforms the baselines across the different test sets. On average, MFTCI outperforms the MF, MF_0 and NMF methods by 34.18%, 11.59% and 4.08% in terms of Pct_0, 16.64%, 7.96% and 4.03% in terms of Pct_1, and 2.10%, 3.00% and 1.98% in terms of Pct_2, respectively. We observe similar results for transfer students as well (not included here for brevity).

Table 2: Comparison Performance with PTA (%)

Methods	Spring 2016			Fall 2015			Spring 2015		
	$Pct_0(\uparrow)$	$Pct_1(\uparrow)$	$Pct_2(\uparrow)$	Pct_0	Pct_1	Pct_2	Pct_0	Pct_1	Pct_2
MF	13.25	27.71	58.02	12.05	26.63	58.89	13.03	26.09	54.83
MF_0	16.52	31.65	57.46	15.51	30.03	55.64	15.53	29.53	54.94
NMF	13.21	27.04	57.18	15.33	30.12	56.15	15.56	29.23	54.93
MFTCI	**19.78**	**35.52**	**61.44**	**19.71**	**35.16**	**60.12**	**18.56**	**32.78**	**58.80**

i) "↑" indicates the higher the better. ii) Reported values of Pct_0, Pct_1 and Pct_2 are percentages. iii) Best performing methods are highlighted with bold.

Table 3 presents the performance of the baselines and MFTCI model for the three different terms of both non-transfer and transfer students using RMSE and MAE as evaluation metrics. The MFTCI model consistently outperforms the baselines across the different datasets in terms of MAE. In addition, the results shows that MF_0, NMF and MFTCI tend to have better performance for Spring 2016 term than Fall 2015 term. Similar trend is observed between Fall 2015 term and Spring 2015 term. This suggests that MFTCI is likely to have better performance with more information in the training set.

6.2 Analysis on Individual Majors

We divide non-transfer students based on their majors and test the baselines and MFTCI model on each major, separately. Table 4 shows the comparison of Pct_0, Pct_1 and Pct_2 on different majors. The results show that MFTCI has the best performance for almost all the majors. Among all the results, MFTCI has the highest accuracy when predicting grades for PSYC and BIOL students for which we have more student-course pairs in the training set.

6.3 Effects from Previous Terms on MFTCI

In order to see the influence of number of previous terms considered in MFTCI, we run our model with only $\Delta(T-1)$ in Equation 1. This method is represented as $MFTCI_{p1}$. Figure 2 shows the comparison results of MAE for six subsets of data which are reported in Table 3, where "NTR" stands for non-transfer students and "TR" stands for transfer students. The results show that MFTCI consistently outperforms $MFTCI_{p1}$ on all datasets. This suggests that considering two previous terms is necessary for achieving good prediciton results. Moreover, since we consider that the student's knowledge is modeled using an exponential decaying function over time, we do not include the influence from the third previous term in our model as its influence for the grade prediction is negligible in comparison to the previous two terms.

6.4 Visualization of Course Influence

To interpret what is captured in the course influence matrix A (See Eq 1), we extract the top 20 values with the corresponding course names (and topics) for analysis. Figure 3 and 4 show the captured pairwise course influences for CS and AIT majors, respectively. Each node corresponds to one course which is represented by the shortened course's name. We can notice from the figures that most influences reflect content dependency between courses. For example, in the CS major, "Object Oriented Programming" course has significant influence on performance of "Low-Level Programming" course (the former one is also the latter one's prerequisite course); "Linear Algebra" and "Discrete Mathematics" have influence on each other; "Formal Methods & Models" course has influence on "Analysis of Algorithms" course. In case of the AIT major, both "Introductory IT" course and "Introductory Computing" course have influence on "IT Problem & Programming" course; "Multimedia & Web Design" course has influence on both "Applied IT Programming" course and "IT in the Global Economy" course. GMU has a sample schedule of eight-term courses for each major in order to guide undergraduate students to finish their study step by step based on the level, content and difficulty of courses [2]. Among the identified relationships shown in Figures 3 and 4 we found 17 and 13 of the CS and AIT courses influences in the guide map, respectively. The rest of the identified influences are among other general electives but required courses (e.g., "Public Speaking" course), or specific electives pertaining to the major (e.g., "Research Methods" course). This shows that our model learns meaningful course-wise influences and successfully uses it to improve MF model.

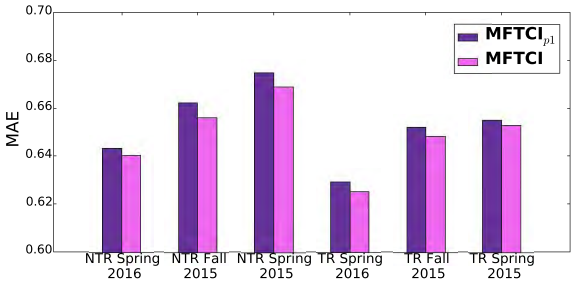


Figure 2: Comparison performance for $MFTCI_{p1}$ and MFTCI

Figure 5 shows the identified course influences for the BIOL, CEIE, CPE and PSYC majors. These identified course-wise influences seem to capture similarity of course content.

7. CONCLUSION AND FUTURE WORK

We presented a Matrix Factorization with Temporal Course-wise Influence (MFTCI) model that integrates factorization models and the influence of courses taken in the preceding terms to predict student grades for the next term.

We evaluate our model on the student educational records from Fall 2009 to Spring 2016 collected from George Ma-

[2]http://catalog.gmu.edu

Table 3: Comparison Performance with RMSE and MAE.

Methods	Non-Transfer Students						Transfer Students					
	Spring 2016		Fall 2015		Spring 2015		Spring 2016		Fall 2015		Spring 2015	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
MF	0.999	0.754	1.037	0.786	1.023	0.784	0.925	0.688	**0.921**	0.686	0.985	0.732
MF_0	0.929	0.714	0.977	0.752	1.014	0.778	0.893	0.668	0.944	0.705	1.011	0.765
NMF	1.020	0.769	**0.967**	0.746	**1.000**	0.771	0.906	0.683	0.932	0.701	**0.979**	0.746
MFTCI	**0.928**	**0.685**	0.982	**0.717**	1.012	**0.750**	**0.887**	**0.636**	0.927	**0.662**	1.000	**0.721**

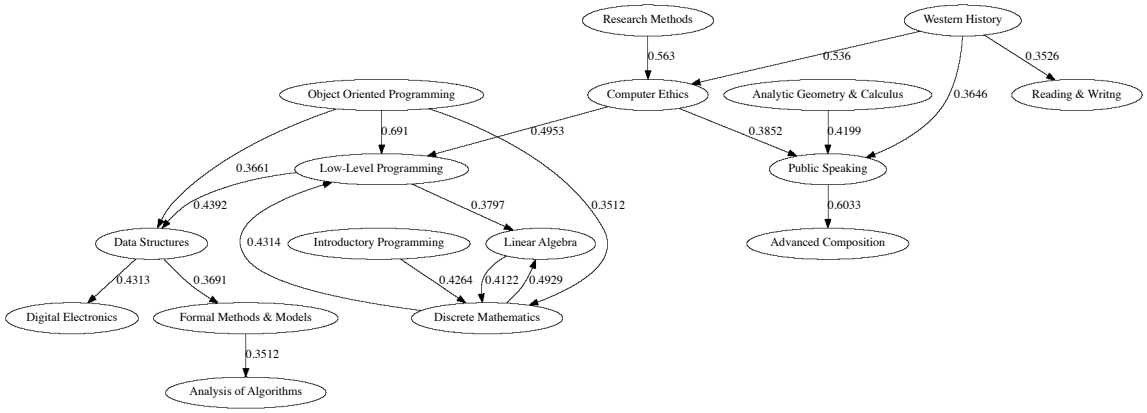


Figure 3: Identified course influences for CS major

Table 4: Comparison Performance for Different Majors

	Methods	AIT	BIOL	CEIE	CPE	CS	PSYC
Pct_0	MF	18.71	18.00	15.99	12.99	15.98	20.18
	MF_0	19.45	22.10	16.70	14.21	16.47	22.12
	NMF	19.77	22.16	**17.01**	14.32	16.61	22.17
	MFTCI	**22.30**	**24.24**	16.80	**14.32**	**17.32**	**25.83**
Pct_1	MF	37.95	35.43	31.47	27.86	31.53	39.41
	MF_0	37.21	39.68	31.87	**27.97**	30.51	39.63
	NMF	36.79	39.74	31.67	27.19	30.43	39.36
	MFTCI	**39.64**	**40.87**	**32.38**	27.53	**31.78**	**42.29**
Pct_2	MF	**67.02**	67.78	58.66	52.28	56.91	**71.01**
	MF_0	66.17	67.54	58.35	50.72	56.24	67.74
	NMF	66.70	67.54	58.55	51.17	56.17	67.79
	MFTCI	66.70	**68.25**	**58.76**	**52.94**	**58.18**	68.29

son University. The dataset in this study contains both non-transfer and transfer students from six different majors. Our experimental evaluation shows that MFTCI consistently outperforms the different state-of-the-art methods. Moreover, we analyze the effects from previous terms on MFTCI, and we make the conclusion that it is necessary to consider two previous terms. In addition, we visualize the patterns learned between pairs of courses. The results strongly demonstrate that the learned course influences correlate with the course content within academic programs.

In the future, we will explore incorporation of additional constraints over the the pairwise course influence matrix, such as prerequisite information, compulsory and elective provision of a course. We will explore using the course influence information to build a degree planner for future students.

8. ACKNOWLEDGMENTS

Funding was provided by NSF Grant, 1447489.

APPENDIX

A. COMPUTATIONAL COMPLEXITY ANALYSIS

The computational complexity of MFTCI is determined by the four steps in the alternating approach as described above. To update U and V as in Equation 2 using gradient descent method via alternating minimization, the computational complexity is $O(\text{niter}_{uv}(k \times n_{s,c} + k \times m + k \times n)) = O(\text{niter}_{uv}(k \times n_{s,c}))$ (typically $n_{s,c} \geq \max(m, n)$), where $n_{s,c}$ is the total number of student-course dyads, n is the number of students, m is the number of courses, k is the latent dimensions of U and V, and niter_{uv} is the number of iterations. To update A as in Equation 3 using gradient descent method, the computational complexity is upper-bounded by $O(\text{niter}_a(n_{cc} \times \frac{n_{s,c}}{m}))$, where n_{cc} is the number of course pairs that have been taken by at least one student, $\frac{n_{s,c}}{m}$ is the average number of students for a course, which upper bounds the average number of students who co-take two courses, and niter_a is the number of iteractions. Essentially, to update A, we only need to update $A(c_i, c_j)$ where c_i and c_j have been co-taken by some students. For $A(c_i, c_j)$ where c_i and c_j have never been taken together, they will remain 0. To update Z_1 as in Equation 4, a singular value decomposition is involved and thus its computational complexity is upper bounded by $O(m^3)$. To update Z_2 as in Equation 8, the computational complexity is $O(m^2)$. To update

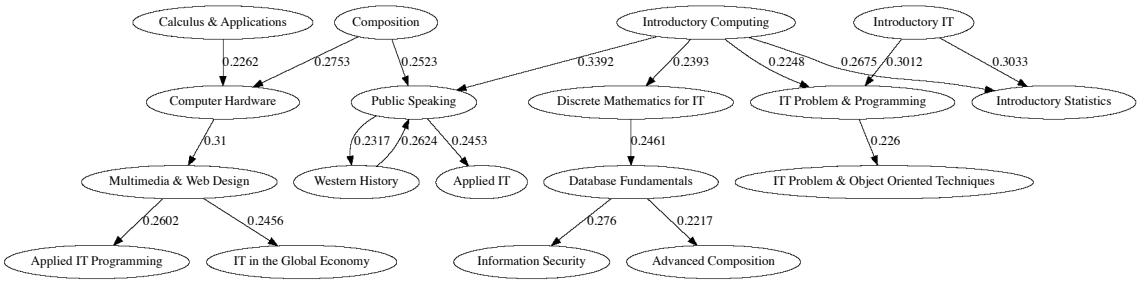
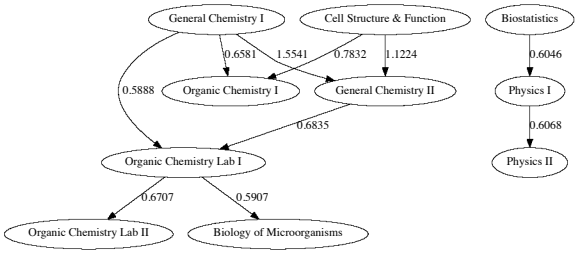
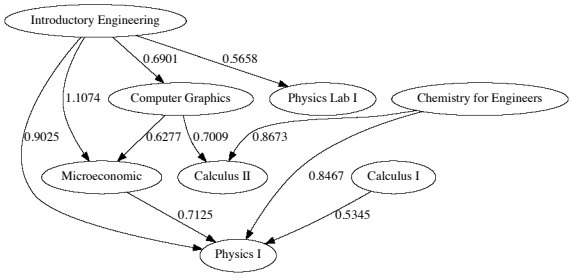


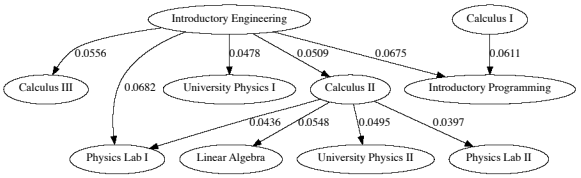
Figure 4: Identified course influences for AIT major



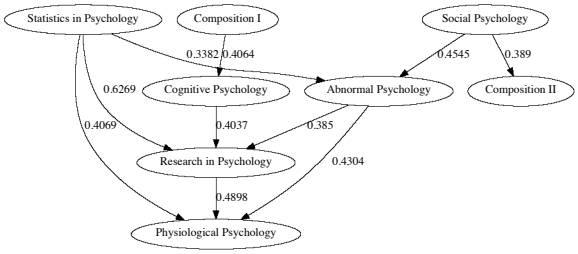
(a) Identified course influences for BIOL major



(b) Identified course influences for CEIE major



(c) Identified course influences for CPE major



(d) Identified course influences for PSYC major

Figure 5: Identified course influences for different majors

U_1 and U_2 as in Equation 11, the computational complexity is $O(m^2)$. Thus, the computational complexity for MTFCI is $O(\text{niter}(\text{niter}_{uv}(k \times n_{s,c}) + \text{niter}_a(n_{cc} \times \frac{n_{s,c}}{m}) + m^3 + m^2))$ $= O(\text{niter}(\text{niter}_{uv}(k \times n_{s,c}) + \text{niter}_a(n_{cc} \times \frac{n_{s,c}}{m}) + m^3))$, where niter is the number of iterations for the four steps. Although the complexity is dominated by m^3 due to the SVD on $A + U_1$, since n (i.e., the number of courses) is typically not large, the run time will be more dominated by $n_{s,c}$ (i.e., the number of student-course dyads).

B. REFERENCES

[1] Charu C. Aggarwal. *Recommender Systems: The Textbook*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[2] RSJD Baker et al. Data mining for education. *International encyclopedia of education*, 7:112–118, 2010.

[3] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[4] Hana Bydžovská. Are collaborative filtering methods suitable for student performance prediction? In *Portuguese Conference on Artificial Intelligence*, pages 425–430. Springer, 2015.

[5] Freddy Chong Tat Chua, Richard J Oentaryo, and Ee-Peng Lim. Modeling temporal adoptions using dynamic matrix factorization. In *2013 IEEE 13th International Conference on Data Mining*, pages 91–100. IEEE, 2013.

[6] Tristan Denley. Course recommendation system and method, January 10 2013. US Patent App. 13/441,063.

[7] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 485–492, New York, NY, USA, 2005. ACM.

[8] Asmaa Elbadrawy, Scott Studham, and George Karypis. Personalized multi-regression models for predicting students performance in course activities. *UMN CS*, pages 14–011, 2014.

[9] Wu He. Examining studentsâĂŹ online interaction in a live video streaming environment using data mining and text mining. *Computers in Human Behavior*, 29(1):90–102, 2013.

[10] Ngoc-Diep Ho. *Nonnegative matrix factorization algorithms and applications*. PhD thesis, ÉCOLE POLYTECHNIQUE, 2008.

[11] Chein-Shung Hwang and Yi-Ching Su. Unified clustering locality preserving matrix factorization for student performance prediction. *IAENG Int. J. Comput. Sci*, 42(3):245–253, 2015.

[12] Bin Ju, Yuntao Qian, Minchao Ye, Rong Ni, and Chenxi Zhu. Using dynamic multi-task non-negative matrix factorization to detect the evolution of user preferences in collaborative filtering. *PloS one*, 10(8):e0135090, 2015.

[13] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[14] Yehuda Koren, Robert Bell, Chris Volinsky, et al. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[15] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[16] Dixin Luo, Hongteng Xu, Yi Zhen, Xia Ning, Hongyuan Zha, Xiaokang Yang, and Wenjun Zhang. Multi-task multi-dimensional hawkes processes for modeling event sequences. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 3685–3691. AAAI Press, 2015.

[17] Rabab Naqvi. Data mining in educational settings. *Pakistan Journal of Engineering, Technology & Science*, 4(2), 2015.

[18] Xia Ning, Christian Desrosiers, and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 37–76. Springer, 2015.

[19] Michelle Parker. Advising for retention and graduation. 2015.

[20] Michael J. Pazzani and Daniel Billsus. The adaptive web. chapter Content-based Recommendation Systems, pages 325–341. Springer-Verlag, Berlin, Heidelberg, 2007.

[21] 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.

[22] Štefan Pero and Tomáš Horváth. Comparison of collaborative-filtering techniques for small-scale student performance prediction task. In *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering*, pages 111–116. Springer, 2015.

[23] Agoritsa Polyzou and George Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, pages 1–13, 2016.

[24] Agoritsa Polyzou and George Karypis. Grade prediction with models specific to students and courses. *International Journal of Data Science and Analytics*, pages 1–13, 2016.

[25] Sanjog Ray and Anuj Sharma. A collaborative filtering based approach for recommending elective courses. In *International Conference on Information Intelligence, Systems, Technology and Management*, pages 330–339. Springer, 2011.

[26] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. Recommender systems handbook., 2011.

[27] Jill M Simons. *A National Study of Student Early Alert Models at Four-Year Institutions of Higher Education.* ERIC, 2011.

[28] John Z Sun, Dhruv Parthasarathy, and Kush R Varshney. Collaborative kalman filtering for dynamic matrix factorization. *IEEE Transactions on Signal Processing*, 62(14):3499–3509, 2014.

[29] John Z Sun, Kush R Varshney, and Karthik Subbian. Dynamic matrix factorization: A state space approach. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1897–1900. IEEE, 2012.

[30] Mack Sweeney, Jaime Lester, and Huzefa Rangwala. Next-term student grade prediction. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 970–975. IEEE, 2015.

[31] Mack Sweeney, Huzefa Rangwala, Jaime Lester, and Aditya Johri. Next-term student performance prediction: A recommender systems approach. *arXiv preprint arXiv:1604.01840*, 2016.

[32] Nguyen Thai-Nghe, Lucas Drumond, Artus Krohn-Grimberghe, and Lars Schmidt-Thieme. Recommender system for predicting student performance. *Procedia Computer Science*, 1(2):2811–2819, 2010.

[33] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G. Carbonell. *Temporal Collaborative Filtering with Bayesian Probabilistic Tensor Factorization*, pages 211–222. 2010.

[34] Chenyi Zhang, Ke Wang, Hongkun Yu, Jianling Sun, and Ee-Peng Lim. Latent factor transition for dynamic collaborative filtering. In *SDM*, pages 452–460. SIAM, 2014.

Toward the Automatic Labeling of Course Questions for Ensuring their Alignment with Learning Outcomes

S. Supraja
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
ssupraja001@e.ntu.edu.sg

Kevin Hartman
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
khartman@ntu.edu.sg

Sivanagaraja Tatinati
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
tatinati@ntu.edu.sg

Andy W. H. Khong
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
andykhong@ntu.edu.sg

ABSTRACT

Expertise in a domain of knowledge is characterized by a greater fluency for solving problems within that domain and a greater facility for transferring the structure of that knowledge to other domains. Deliberate practice and the feedback that takes place during practice activities serve as gateways for developing domain expertise. However, there is a difficulty in consistently aligning feedback about a learner's practice performance with the intended learning outcomes of those activities – especially in situations where the person providing feedback is unfamiliar with the intention of those activities. To address this problem, we propose an intelligent model to automatically label opportunities for practice (assessment questions) according to the learning outcomes intended by the course designers. As a proof of concept, we used a reduced version of Bloom's Taxonomy to define the intended learning outcomes. Using a factorial design, we employed term frequency-inverse document frequency (TF-IDF) and latent Dirichlet allocation (LDA) to transform questions from text to word weightages with support vector machine (SVM) and extreme learning machine (ELM) to train and automatically label the questions. We trained our models with 120 questions labeled by the subject matter expert of an undergraduate engineering course. Compared to existing works which create models based on a self-generated dataset, our proposed approach uses 30 untrained questions from online/textbook sources to validate the performance of our models. Exhaustive comparison analysis of the testing set showed that TF-IDF with ELM outperformed the other combinations by yielding 0.86 reliability (F1 measure) with the subject matter expert.

Keywords

Learning outcomes, Term frequency-inverse document frequency, Latent Dirichlet allocation, Extreme learning machine, Support vector machine

1. INTRODUCTION

Increasingly, modern curriculum design in tertiary and adult learning settings has become a collaborative endeavor between subject matter experts, learning designers, and learning technologists. While these teams employ a variety of process models for the planning, execution, and revision of their curriculum and activity designs, often greater attention is paid to the construction of a course design and the course content rather than the assessment practices that measure learning and their ongoing maintenance.

The algorithms and use case described in this paper exist in a particular context of outcome-based education. In this context, learning is defined by observable changes in a learner's behavior. These changes commensurate with Krathwohl's model of learning objectives [1] but learning outcomes go beyond objectives. Learning outcomes are predicated on having learners observably demonstrate their growing understanding of a topic or proficiency within a field [2]. When learning activities become more open-ended and exploratory, and when learners are offered choices for how to proceed, learners often look to how they will ultimately be assessed to gauge which learning strategies they should employ [3].

When a course's learning activities support its assessment practices and the assessment practices support the types of outcomes that are relevant to learners in the future, the course's activities and intended learning outcomes exhibit constructive alignment with each other [2]. Adhering to constructive alignment creates a seamless path from learning, to applying, to transferring concepts and relationships when solving novel problems.

However, the promise of constructive alignment is not easily delivered upon. Oftentimes, a course's learning outcomes cannot be measured by its assessment practices, or its assessment practices are decontextualized from the types of activities and practices learners are actually preparing for [4]. Whether in the context of higher learning or professional development, when thinking about developing flexible, life-long learners it is paramount to have mechanisms in place to support learners as they work to gain domain expertise. These processes should reliably measure learning and link assessment practices to authentic activities.

1.1 Learning design for domain expertise

Prior work in designing for adaptive domain expertise, the kind of expertise necessary for learners to function in changing environments and flexible job scopes, has shown that learning design teams need to be cognizant of three elements which will be discussed in turn.

1.1.1 Levels of learning outcomes

Learning outcomes range in sophistication and vary by field. In medicine, Miller's Pyramid [5] lists learning outcomes beginning with knowing about a subject, progressing to knowing how to do something, to being able to actually demonstrate it in a contrived setting like a role-play with actors, and to being able to demonstrate it in a real environment like a surgical theater [6]. The idea is based on the belief that the development of expertise is a progression from

the recall of facts to the execution of skills. However, as research on problem based learning has shown, demonstration of skill and the recall of facts can proceed independently of each other depending on the learning environment [7].

In [8], a field agnostic method of classifying learning outcomes based on their quality is presented. Essentially, the Structure of Observed Learning Outcomes (SOLO) taxonomy identifies the level of cognitive sophistication a learning outcome requires. Lower level learning outcomes indicate a learner is capable of remembering facts in isolation. More sophisticated levels require learners to assimilate information from various sources to make connections and transform that understanding into something new.

Perhaps the most popular listing of learning outcomes is Bloom's Taxonomy. Similar to Miller's Pyramid, Bloom's Revised Taxonomy also begins with the retrieval of facts and information as its foundation and builds up to application of knowledge and further to analyzing, evaluating, and creating. Because of its simplicity and familiarity with learning designers and subject matter experts alike, Bloom's Taxonomy can easily be used to identify the levels of learning outcomes in a course [9].

1.1.2 Opportunities for deliberate practice

Along with identifying a learning activity's intended outcomes, expertise development requires opportunities for deliberate practice. In contrast to repetitive practice intended for learners to develop automaticity in either the recall of information or the application of a skill, often during time-limited tasks, deliberate practice focuses on mastering the nuances of the domain itself to fine-tune performance [10]. In fact, a learner's level of grit, a combination of perseverance and passion, predicts how close to expert performance a learner will eventually show [11].

The key difference in processes between repetitive practice and deliberate practice leads to different forms of expertise: adaptive and routine [12]. Routine forms of expertise allow a learner to conduct a task at an optimal level. Adaptive expertise allows learners to learn new tasks or solve novel problems at an accelerated rate. In an industrial setting, routine expertise helps a worker complete a particular job function. Adaptive expertise enables that same worker to retrain to fill new job functions. Typically, the amount of time necessary to achieve expert performance in a domain is in the order of years to decades [13]. However, incremental improvement can be seen in a few practice cycles when activities align to the intended learning outcomes.

1.1.3 Formative assessments and actionable feedback

Hand in hand with creating opportunities for deliberate practice is providing formative feedback to the learner about how to improve that practice while that improvement is still relevant. Imagine students who diligently answer every question in an engineering textbook but never receive feedback on the quality of their solutions. In this case, the learners would be unable to gauge their performance in relation to the course learning outcomes or have an idea about how to improve their performance in the future. Now imagine if those same students do receive feedback, but that feedback arrives after the course's final examination. If the content of the course is mostly self-contained and will not be revisited, the feedback is mostly irrelevant.

Formative feedback consists of two parts: 1) an interpretable indication of a learner's performance on an assessment of learning with respect to a standard of performance (learning outcome) and 2) the opportunity to improve performance before the final evaluation [14].

Cognitive tutors provide a clear example of the power of coupling formative assessment and actionable feedback together in the domain of mathematics learning [15]. By presenting learners with a series of structured problems, cognitive tutors are capable of intervening at any point during the problem-solving process to provide students with feedback about their performance. This feedback may be the identification of an error, the presentation of a hint, or the request for more information about the learner's reasoning. After the feedback, learners have the opportunity to adjust their problem-solving heuristics to improve their performance going forward.

Such an interaction sequence works with highly structured tasks with application-oriented learning outcomes. However, the feedback cycle is more difficult to manage when the learning outcomes are aligned to higher-order reasoning like evaluation, analyzing and creating. These outcomes have multiple paths for reaching a satisfactory answer.

With this difficulty in mind, we looked at techniques to automate the process of identifying the reasoning level of text-based assessment items (questions) with the intention of better aligning questions to learning outcomes as a first step toward being able to provide opportunities for deliberate practice. Subsequently, the outcome of our proposed work is to link actionable feedback to a learner's performance on assessment items.

1.2 Automated question classification techniques

Prior work has shown the viability of automatically labeling questions in accordance with a course's learning outcomes. However, our work goes beyond labeling existing content to helping course instructors promote deliberate practice and expertise development by providing a method of finding new questions that align to the course designer's original intended learning outcomes. We highlight the drawbacks of prior work and how our proposed approach addresses those limitations.

1.2.1 Labeling questions based on difficulty level

Early attempts at automatically labeling questions relied on subject matter experts to pre-define the difficulty levels of questions. Artificial neural network trained by backpropagation then used the question features and assigned difficulty levels in the training set to classify new questions. A five-dimensional feature vector that consisted of query-text relevance, mean term frequency, length of questions and answers, term frequency distribution (variance), distribution of questions and answers in a text were used. The method yielded an F1 measure, a classification reliability metric that measures a test's accuracy, of 0.78 [16]. However, a major pitfall this method is its lack of semantic analysis.

Entropy-Based Decision Tree has also been used to label questions [17]. The weakness in this strategy is that there is high possibility of overfitting the model during the training phase that then negatively affects the subsequent prediction performance.

1.2.2 Labeling questions based on Bloom's Taxonomy using Natural Language Processing

Natural Language Processing (NLP) has been used for the generation of assessments, answering questions, supporting users in Learning Management Systems and preparing course materials. The Wordnet package has been used to detect semantic similarity. By performing a rule-based approach, the accuracy of labeling a

question based on Bloom's Taxonomy reaches 82% [18]. To improve the rule-based approach, a hybrid technique of using an N-gram classifier with a rule-based approach has also been explored. Rules were based on combining parts-of-speech tagging, and the N-gram classifier found the probabilities of predicting certain words. Such a hybrid method yielded an F1 measure of 0.86 [19].

1.2.3 Labeling questions based on Bloom's Taxonomy using machine learning techniques

Machine learning algorithms can be broadly split into either supervised or unsupervised training implementations. Generally, supervised training is adopted when, during training, labels have been pre-determined and questions are labeled by an expert. The most commonly used method in such cases is the term frequency-inverse document frequency (TF-IDF). The algorithm assigns weightages to individual words in a question statement to define a custom vector space to each question.

Machine learning techniques such k-nearest neighbors, Naïve Bayes and support vector machine (SVM) have been implemented for labeling questions. When doing a performance comparison among these three techniques, an F1 measure of 0.71 was achieved using SVM [20]. To increase the accuracy level, additional features were incorporated in future versions of the work. Three different feature selection processes, namely: Odd Ratio, Chi-square statistic and Mutual Information were used with the three machine learning techniques. The F1 measure result reached 0.9 [21].

Furthermore, an integrated approach of feature extraction has been proposed by using headword, semantic, keyword and syntactic extractions, which are fed into SVM [22]. However, this work has not yet been completed by using a testing dataset to quantify the reliability of prediction.

A major downside in existing works is that both the training as well as testing questions are part of the same course curriculum; the questions are generated by the same author/instructor. Even when a high F1 measure is achieved, it does not enable the algorithm to label questions written by another subject matter expert. Our work increases the flexibility of labeling methods by testing our models with a new set of questions compiled from textbook and online resources.

In addition, our work introduces extreme learning machine (ELM), which has been shown to outperform SVM during similar labeling tasks [23]. Moreover, we introduce LDA as an alternative technique to TF-IDF for transforming question statements into numerical word weightages.

By comparing combinations of these new techniques with more traditional techniques, we aim to gauge which combination attains the highest labeling reliability with the subject matter expert when automatically labeling untrained questions. For our purposes, using the combination with the highest F1 measure (fewest false negatives and false positives) becomes paramount. In our use case, a mislabeling by the algorithm will lead to the wrong set of practice questions to be given to students and diminish the impact of deliberate practice on reaching the intended learning outcomes.

2. METHODS

2.1 Materials

2.1.1 Labeling scheme

The core of this study centers on a labeling scheme for identifying the sophistication of learning outcomes based on a simplified version of Bloom's Taxonomy. In this labeling scheme, the first two levels of Bloom's Taxonomy (Remembering and Understanding) were collapsed into Remember. Applying remained its own category. All of the higher-order reasoning categories (Analyzing, Evaluating, and Creating) were collapsed into Transfer. Figure 1 shows how our labeling scheme categories map onto the original categories from Bloom's Revised Taxonomy.

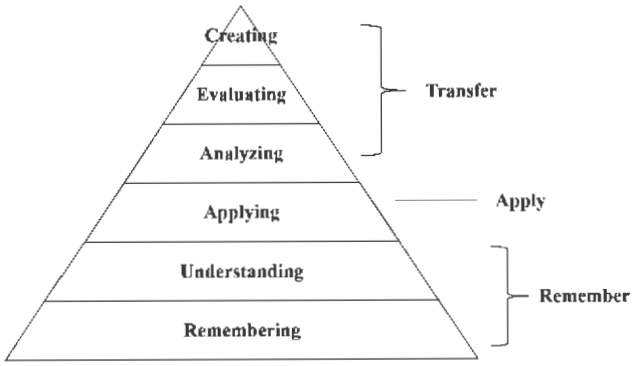


Figure 1: Mapping of Bloom's Revised Taxonomy [24]

We collapsed the taxonomy into three categories for two reasons. First, the subject matter expert tasked with labeling the questions was unsure about how reliably the questions could be labeled by someone without a background in learning design, educational psychology, or curriculum development. Collapsing the categories to Remember, Apply, and Transfer made manually labeling hundreds of questions to train the machine learning algorithms more tractable. Second, collapsing the categories had the effect of making Bloom's Taxonomy more analogous to the successful use cases of Miller's Pyramid by subject matter experts in both higher education and professional development settings [5].

2.1.2 Question dataset

The dataset consists of a total of 150 questions used for training and testing the machine learning algorithms based on the content of an undergraduate electrical and electronic engineering course.

For this study, we formed a training set of 120 questions by randomly selecting 40 Remember, Apply, and Transfer items from the larger question pool of more than 200 questions used in that course. The pool came from a repository of four years' worth of assignment, homework, quiz and exam questions presented to students. These questions prompt students for a range of answer types (i.e., open-ended, multiple-choice, short-structured, essay).

We then created a testing set of 30 new questions compiled from external sources such as textbooks and online question banks. This set was also balanced with equal representation of Remember, Apply, and Transfer questions.

2.2 Data pre-processing procedures

We pre-processed the raw questions in two phases. First, the subject matter expert labeled every question according to the labeling scheme described above. Second, we transformed the text of every question into a machine-readable format before passing them through the machine learning algorithms.

2.2.1 Subject matter expert pre-processing

The subject matter expert manually labeled each question in the training set based on its intended learning outcome (Remember, Apply or Transfer). The subject matter expert then labeled the 30 new questions in the testing set in the same manner. These new questions are labeled for the purpose of knowing the ground truth for performance evaluation. Table 1 below shows some examples of the labeled questions.

Table 1 - Examples of labeled questions

Remember
Consider a signal described by y[n] = 2n +4. What would be the amplitude of the signal at sample index n=3?
Apply
Consider the following input and output signals: find the transfer function and state the poles and zeros of this transfer function.
Transfer
Describe how the bandpass filter can be utilized for radar applications.

2.2.2 Text pre-processing

The text transformation began by excising all equations, mathematical symbols and diagrams from the questions. We only kept the core of the question prompts by removing the descriptive and explanatory text from scenario and hypothetical questions. For example, if a question began by setting the stage with "Peter has been asked to perform…", followed by the question prompt "How much voltage should Peter expect in the circuit?", all of the descriptive text prior to the question prompt was removed to improve the consistency of word length and usage between items.

For the remaining words in the questions, we changed all of the characters to lower case, removed all punctuation marks, numbers, and non-unicode characters. We then stemmed the remaining words to obtain a list of root words. From this list of root words, we removed all words with fewer than three letters. Because we were unsure of the relationship between the words and the labels, we did not create a list of stopwords for removal.

3. TECHNIQUES

We tested four combinations (in no particular order) of word weighting and question labeling algorithms, as shown in Figure 2, to identify the techniques with the highest reliability for our automated learning outcome labeler.

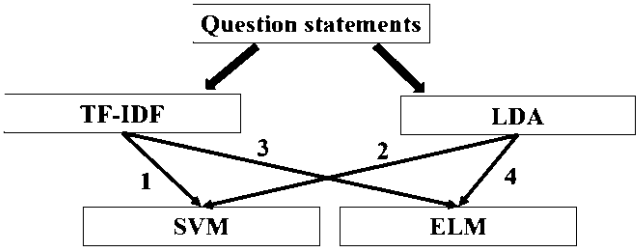


Figure 2: Four combinations of algorithms

Every word in each question prompt was assigned a weightage value based on either term frequency-inverse document frequency (TF-IDF) or latent Dirichlet allocation (LDA). Subsequently, the vector values for each question were passed through either support vector machine (SVM) or extreme learning machine (ELM) to assign a label. All algorithms were implemented in R Studio.

3.1 Term frequency-inverse document frequency

Term frequency-inverse document frequency (TF-IDF) is a technique for finding the relative frequency of words in a given document, and comparing those frequencies with the inverse of how often each of those words appear in the complete document corpus. The resulting ratio can be used to signify the relevance of each unique word within a single document.

We implemented a modified version of TF-IDF that used individual questions as the source of the analysis instead of complete documents. This focused the model on finding the relevance of each word within each single question. By converting each question into a vector of weightages based on word frequencies, the machine learning algorithms were then used to label the questions. The modified TF-IDF model can be described by

$$TF - IDF(w_i, q_k) = \#(w_i, q_k) \times \log \frac{TR}{\#TR(w_i)} \quad (1)$$

where w_i refers to a particular word i, q_k refers to a particular question k, $\#(w_i,q_k)$ refers to number of times w_i occurs in q_k, TR refers to total number of questions and #TR(w_i) refers to question frequency, or the number of questions in which w_i occurs [20].

In the case where the term frequency (TF) count is biased towards longer questions, the TF count is normalized as

$$TF_{i,k} = \frac{n_{i,k}}{\sum_j n_{j,k}} \quad (2)$$

where $n_{i,k}$ refers to the number of times w_i occurs in q_k, the denominator term (size of each question) refers to the sum of the number of times each word appears in q_k [25].

For our work, the pre-processing procedures registered a total of 465 unique stemmed words in our compilation of 120 training questions and 30 testing questions. This led to each question being represented as a vector of 1 row and 465 columns arranged in alphabetical order by stemmed word. When a word is present in a question, the normalized weight of that word is assigned to that question's vector element. If a word is not present in the question, the weight is zero.

After determining the unique word weightage vectors for all 150 questions, the entire matrix is sorted such that for each question, the weightages are arranged in ascending order. The top ten weightages are chosen for each question. The 10 weightages may correspond to different words in each question, but their combinations remain question-specific and give a numerical representation of each question statement. This new vector of 10 columns per question serves as the input to the machine learning algorithms.

As an example, we will use the pre-processed question prompt:

for signal which begin when the one side unilateral ztransform given

Table 2 below shows the weightages assigned to the above example after the application of the TF-IDF technique. The weightages are then arranged in ascending order and the top 10 values are taken.

Table 2 - TF-IDF weightage arrangement

Word (alphabetical order)	Weightage
begin	0.392
for	0.140
given	0.140
one	0.222
side	0.356
signal	0.116
the	0.007
unilateral	0.392
when	0.279
which	0.230
ztransform	0.216

3.2 Latent Dirichlet allocation

Latent Dirichlet allocation (LDA) is a probabilistic technique for topic modeling based on the Bayesian model. The essential idea of LDA is that each document consists of a mixture of topics, with the continuous-valued mixture properties distributed in a Dirichlet random variable, a continuous multivariate probability distribution.

Again, in the context of our work, we applied LDA to questions in the dataset by substituting the original notion of documents in the LDA algorithm with questions in our modified model. Therefore, the modified model attempted to find *k* number of topics (*k* is a user-defined parameter to determine the desired number of topics, or dimensionality of the Dirichlet distribution) for a given set of question statements based on the choice and usage of words in each question. The joint distribution of a topic mixture, a set of topics and a set of words can be represented by

$$p(\theta, t, w|\alpha, \beta) = p(\theta|\alpha) \prod_{i=1}^{M} p(t_i|\theta) p(w_i|t_i, \beta) \quad (3)$$

where parameter α is a *k*-vector with components more than zero, parameter β refers to the matrix of word probabilities, θ refers to a *k*-dimensional Dirichlet random variable, t_i refers to a topic, w_i refers to a word [26].

Figure 3 shows a graphical model representation of LDA. The bigger circle refers to questions while the smaller circle refers to the repeated choice of topics and words within each question.

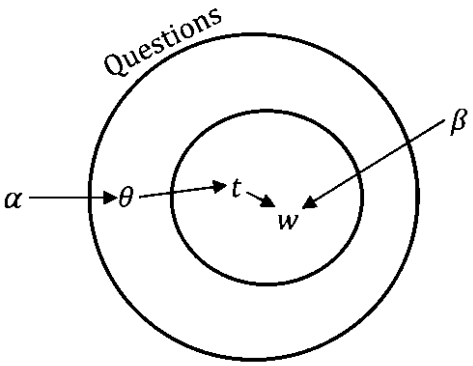


Figure 3: Graphical model representation of LDA

Since LDA involves topic modeling, an appropriate *k* value chosen for our work was ten. This allowed a standard comparison between LDA and the top ten weightages from the TF-IDF method. The generated unique topics (based on the stemmed words) are shown in Table 3.

Table 3 - Topic names generated by LDA

Topic number	Stemmed topic name
1	differ
2	discrete
3	impulse
4	signal
5	filter
6	apply
7	dft
8	output
9	sample
10	system

Out of the entire set of stemmed words detected, ten words have been identified as topic names. Hence, LDA automatically associates the remaining words the above-mentioned ten topics. Based on the words that appear in each question, LDA displays the number of topics per question. Based on the topic assignments, the topic weightages for each question is generated. For topics not present in a question, a minimal weightage is given to those topics in lieu of a zero value. The value ensures that the topic weightages for a question sum to one. Similar to the TF-IDF output, the new vector of 10 columns per question becomes the input for the machine learning algorithms.

3.3 Extreme learning machine

Extreme learning machine (ELM) is a learning algorithm for single-hidden layer feedforward neural networks (SLFNs). ELM can be used for classification, regression, clustering, compression and feature learning. ELM randomly chooses the hidden nodes and determines the output weights of the neural networks.

The following three-step learning model explains ELM. Given a training set that is labeled (information about the target nodes), hidden node activation function and number of hidden nodes,

Step 1: Randomly assign hidden node parameters

Step 2: Calculate the hidden layer output matrix, **H**

Step 3: Calculate the output weight γ

Given a set of inputs with unknown labels, the objective is to find the target outputs [27]. Once the inter-layer weights have been found, the same weights are used during the testing phase. For a given set of input samples x_k, the target/output is given by t_k. For number of hidden nodes *L* and with a certain activation function *f(x)*, the SLFN is modeled as

$$\sum_{j=1}^{L} \gamma_j f_j(x_k) = \sum_{j=1}^{L} \gamma_j f\left(w_j \cdot x_k + b_j\right) = o_k, k = 1, \dots, L \quad (4)$$

where w_j refers to the weight vector that stores the weights between input and hidden nodes, γ_j refers to the weight vector that stores the weights between the hidden and output nodes, b_j refers to the threshold of the jth hidden nodes. The objective is that o_k and t_k (original target) should have zero difference [23] using possible activation functions that include sigmoid, sine, radial basis and hard-limit.

In our case, the output of the ELM are three continuous values that represent the values assigned to the three learning outcome categories (Remember, Apply and Transfer). To convert the three values into a binary value for comparing the predicted labels with the actual labels, we set the learning outcome category with the highest value to one and the remaining two to zero.

3.4 Support vector machine

Support vector machine (SVM) is a mapping of data samples such that these samples can be distinctly labeled. The concept of SVM is derived from margins and subsequently separating data into groups with large gaps between them. Deriving an optimal hyperplane for identifying linearly separable patterns is the key to SVM. This idea is extended to cases where the patterns are non-linearly separable, by using a kernel function to transform the original data samples to map onto a new space [28]. Possible kernels are: linear, polynomial, radial basis and sigmoid.

For our work, we used the C-support vector classification type. Given a set of inputs and targets, the cost function is given by [29]

$$\min_{p,m,\xi} \frac{1}{2} p^T p + C \sum_{j=1}^{k} \xi_j \quad (5)$$

subject to $y_j\left(p^T \phi\left(v_j\right) + m\right) \geq 1 - \xi_j, \xi_j \geq 0, j = 1, \dots, k$

where C>0 is the regularization parameter, m is a constant, p is the vector of coefficients, ξ_j refers to parameters that handle the inputs, index j refers to labeling the k training cases, v refers to the independent variables, y refers to the class labels, ϕ refers to the kernel used that transforms data from the input to the chosen feature space.

Fundamentally, support vectors are data points that lie close to the decision boundary, which are the hardest to classify. SVM maximizes the margin around the hyperplane that separates these points. The cost function is determined based on the training samples (support vectors). These support vectors are the basic elements of a training set that would change the position of the hyperplane dividing the dataset. SVM becomes an optimization problem for determining the optimal hyperplane.

3.5 Performance metrics

To evaluate the reliability of our four technique combinations with the subject matter expert's labels, we looked at using the F1 measure. Accuracy is the number of correct labels divided by the size of testing data. The F1 measure is a harmonic mean of two other metrics: precision and recall. Precision refers to the correctness of questions that have been selected as a particular category. Recall refers to the correctness of selection of the correct category given all the questions that were correctly classified.

Because minimizing the number of false positives and false negatives was important for accurately assigning new questions to the correct practice sets, we used the F1 measure as the basis for our algorithm comparisons. To explain the F1 measure, we will step through the confusion matrix used to describe the performance of a labeling model on a set of testing data. There are four concepts used to construct the confusion matrix:

True positive (TP) refers to the number of questions that the algorithm correctly identifies as presenting a label.

False positive (FP) refers to the number of questions that the algorithm identifies as presenting a label while the subject matter expert indicates the label was absent.

True negative (TN) refers to the number of questions that the algorithm correctly identifies as having a label absent.

False negative (FN) refers to the number of questions that the algorithm identifies as having a label absent while the subject matter expert indicates the label was present.

The F1 measure is calculated as follows [30]

$$Precision = \frac{TP}{(TP+FP)} \quad (6)$$

$$Recall = \frac{TP}{(TP+FN)} \quad (7)$$

$$F1\ measure = \frac{2 \times precision \times recall}{precision+recall} \quad (8)$$

4. RESULTS AND ANALYSIS

4.1 Insights by subject matter expert

When looking at every question presented to students over the course of a semester, the subject matter expert identified the number of questions corresponding to Remember, Apply and Transfer as shown in Table 4. Just by labeling the course questions, the subject matter expert realized how misaligned the course's learning outcomes were with its assessment practices. A large emphasis on Apply questions was expected, but the dearth of Transfer questions was surprising. Of those 23 Transfer items, most were presented during the final exam.

Table 4 - Frequency of questions aligned to learning outcomes

Learning outcome	Frequency (number of questions)
Remember	62
Apply	131
Transfer	23

One of the stated learning outcomes of the course was to prepare students to flexibly transfer course content to novel problems and new situations. However, waiting until the final exam to present students with such opportunities denied them actionable feedback during the semester. In response to the pre-processing labeling efforts, the subject matter expert then added 42 new transfer questions throughout the course for the next semester.

4.2 Model reliability with subject matter expert

The objective of this implementation is to evaluate whether the trained model is able to predict the type of question (Remember, Apply or Transfer). Based on the trained model using questions from the undergraduate course, the testing questions from textbooks and online sources were passed through our model to determine the level of reliability of labeling new questions that were not generated by the subject matter expert. In our intended use case, the testing dataset would not need to be manually labeled. However, to determine the level of reliability of our labeling algorithms, the subject matter expert's manual labels served as a ground truth for the F1 measure calculations.

4.2.1 Parameter selection

We first determined the best set of parameters based on 10-fold cross validation of the training dataset. As there were 120 questions, 90% of the questions (108 questions) were used for training and 10% of the questions (12 questions) were used as a validation set. This process was done 10 times using 10 different bundles of the 120 questions. The best set of parameters were chosen based on a grid search for both ELM and SVM.

The parameters that were varied for ELM were:

1. Number of hidden nodes
2. Activation function (sigmoid / radial basis / hard-limit)

The parameters yielding the best results corresponded to 72 hidden nodes using hard-limit activation function.

The parameters that were varied for SVM were:

1. Kernel (sigmoid / radial basis)
2. Cost value
3. Gamma value

The parameters yielding the best results corresponded to sigmoid kernel, cost value = 1, gamma value = 0.26

4.2.2 Comparing four combinations

With respect to the F1 measure, calculations were done separately for the three labels. The mean of those calculations was then used as the algorithm's overall performance measure. With respect to ELM, the calculation was repeated 10 times because the initialization weights are randomly assigned in each iteration. The mean value of the F1 measure was taken.

Table 5 below shows the F1 measure values (for each individual class and overall F1 mean) for the four combinations. "R" refers to Remember, "A" refers to Apply, "T" refers to Transfer and "s.d." refers to standard deviation.

Table 5 - F1 measure values for four combinations

Combination	R	A	T	Mean	s.d.
1. TF-IDF with SVM	0.870	0.737	0.667	0.758	0.084
2. LDA with SVM	0.400	0.593	0.556	0.516	0.084
3. TF-IDF with ELM	0.926	0.815	0.840	0.860	0.048
4. LDA with ELM	0.467	0.520	0.647	0.545	0.076

TF-IDF with ELM achieved the highest mean F1 measure value and the lowest standard deviation – indicating that it was the most reliable combination. It can be seen that the Remember label yields the highest F1 values out of the three labels in Combination 3. In general, Remember-labeled questions are short, resulting in about four to five zero values in the TF-IDF vector of 10 columns that is passed as an input into the ELM. Hence, the algorithm identifies Remember-labeled questions very accurately due to their size.

The result of high reliability in using ELM is as expected because it has already been demonstrated that ELM outperforms SVM when comparing in terms of standard deviation of training and testing root-mean-square values, time taken, network complexity, as well as performance comparison in real medical diagnosis application [23]. On the other hand, although LDA has been shown to achieve higher performance as it groups words together in terms of topics instead of looking at combinations of individual words which may not link together, in the context of our work, TF-IDF outperforms LDA instead. This is because for LDA, the goal is to correctly assign each document (or question) to a class label in a reduced dimensional space [31]. However, in our corpus of questions, there are several technical terms involved, without any prior labeling of topics. Hence, LDA is not appropriate for our analysis.

5. CONCLUSIONS

Based on the comparison of our four algorithms, our most reliable model (TF-IDF with ELM) is able to accurately label new course questions for the undergraduate electrical and electronic engineering course with 0.86 reliability in terms of F1 measure. Any novice instructor who takes over this course in the future or teaching assistants tasked with refreshing the course assignments would be able to extract new questions from any external source and pass them to the algorithm to automatically label the questions as the original course coordinator would. This allows members of the course design team without a strong background in learning to make curriculum decisions regarding the alignment of the course's learning outcomes.

As discussed earlier, outcome-based learning environments facilitate transforming the model of instruction from instructor-centric and lecture-based to being more learner focused filled with a variety of activities and learning pathways. However, in learner-centered environments, assessment is still the key driver, and often the key inhibitor of learning [3]. If the assessments require shallow understanding, then learners calibrate their efforts to achieve this low bar. When assessments require deep understanding or great proficiency, learners are likely to put in more effortful practice.

In line with this assessment philosophy, our TF-IDF with ELM model is theoretically capable of matching any learning activity to any set of learning outcomes as long as the course designers or subject matter experts provide enough examples that are explicitly aligned to the intended learning outcomes when training the model. For the convenience of the subject matter expert in our context, we used a reduced version of Bloom's Taxonomy in this study. However, the final algorithm is capable of using the full Bloom's model, a different model, or a custom set of learning outcomes as its labeling framework.

Hence, with the high reliability of the prediction algorithm presented in our work, our process for calibrating the algorithm can be used in any academic or industrial setting to provide the right set of formative assessment opportunities to students (enhancing subject knowledge) or employees (professional development). Once the learning outcomes of activities are labeled reliably, it is then easier to think about how to engage learners in deliberate practice to reach those outcomes and develop their expertise. Once opportunities for deliberate practice that align to the course learning outcomes are implemented into a course, it becomes easier to think about how to align the feedback regarding those opportunities to support the development of domain expertise.

This work provides a first step at being able to regularly introduce learning activities that promote the development of adaptive expertise into a course by matching external sources of activities with the course's learning outcomes. Deliberate practice requires repetition that varies in ways that highlight the structural elements of a domain. Having a way to incorporate new sources of questions and problems into a course that align with the course's goals provides learners more opportunities for internalizing when to apply their domain specific skills and knowledge. Finally, our algorithm is potentially useful for designing courses to reach non-content-based learning outcomes, making policies that support constructive alignment, and evaluating course assessment of learning plans.

6. FUTURE WORK

Building off of our machine learning labeling work, we would like to explore constructing a new version of LDA that can be tailor-made to label questions. There are situations in which weightages given to words are the same, with different words representing those weightages. Similarly, the same words can have different weightages. We are keen to continue working on features based on word arrangement, word context and word order that affect weightage assignments. In addition, ELM can be enhanced by using kernels.

From the learning aspect, we would like to extend our question label categories to all six outcomes described in Bloom's Taxonomy and expand the model to label outcomes based on the types of sentences used in forum conversations and other collaborative learning activities. Eventually, we aim to determine the proficiency level of learners so we can put learning supports in place to guide their learning journeys. Ultimately, we wish to provide learners with learning activities and opportunities for deliberate practice embedded with actionable feedback to develop their adaptive expertise.

7. ACKNOWLEDGMENTS

This work was conducted within the Delta-NTU Corporate Lab for Cyber-Physical Systems with funding support from Delta Electronics Inc and the National Research Foundation (NRF) Singapore under the Corp Lab@University Scheme.

8. REFERENCES

[1] Krathwohl, D.R. 2002. A Revision of Bloom's Taxonomy: An Overview. *Theory into Practice*. 41, 4 (2002), 212-218. DOI= http://dx.doi.org/10.1207/s15430421tip4104_2

[2] Biggs, J. 1996. Enhancing teaching through constructive alignment. *Higher Education*. 32, 3 (1996), 347-364. DOI= http://dx.doi.org/10.1007/BF00138871

[3] Boud, D. 2010. Sustainable Assessment: Rethinking assessment for the learning society. *Studies in Continuing Education*. 22, 2 (2010), 151-167. DOI= http://dx.doi.org/10.1080/713695728

[4] Boud, D. and Falchikov, N. 2006. Aligning assessment with long-term learning. *Assessment & Evaluation in Higher Education*. 31, 4 (2006), 399-413. DOI= http://dx.doi.org/10.1080/02602930600679050

[5] Miller, G. E. 1990. The Assessment of Clinical Skills/Competence/Performance. *Academic Medicine*. 65, 9 (1990), S63-S67. DOI= http://dx.doi.org/10.1097/00001888-199009000-00045

[6] Wass, V. et al. 2001. Assessment of clinical competence. *The Lancet*. 357, 9260 (2001), 945-949. DOI= http://dx.doi.org/10.1016/S0140-6736(00)04221-5

[7] Hmelo-Silver, C.E. 2004. Problem-based learning: What and how do students learn? *Educational Psychology Review*. 16, 3 (2004). 235-266. DOI= http://dx.doi.org/10.1023/B:EDPR.0000034022.16470.f3

[8] Biggs, J. B. and Collis, K.F. 2014. *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcomes).* Academic Press.

[9] Crowe, A. et al. 2008. Biology in bloom: implementing Bloom's taxonomy to enhance student learning in biology. *CBE-Life Sciences Education.* 7, 4 (2008), 368-381. DOI= http://dx.doi.org/10.1187/cbe.08-05-0024

[10] Ericsson, K.A. et al. 1993. The Role of Deliberate Practice in the Acquisition of Expert Performance. *Psychological Review.* 100, 3 (1993), 363-406. DOI= http://dx.doi.org/10.1037/0033-295X.100.3.363

[11] Duckworth, A. L. et al. 2007. Grit: perseverance and passion for long-term goals. *Journal of personality and social psychology*. 92, 6 (2007), 1087. DOI= http://dx.doi.org/10.1037/0022-3514.92.6.1087

[12] Schwartz D. L. et al. 2005. Efficiency and innovation in transfer. *Transfer of learning from a Modern Multidisciplinary Perspective*. Information Age Publishing. 1-51.

[13] Chi, M. T. 2006. Two approaches to the study of experts' characteristics. *The Cambridge Handbook of expertise and expert* performance. Cambridge University Press. 21-30.

[14] Black, P. and William, D. 1998. Assessment and Classroom Learning. *Assessment in Education Principles Policy and Practice.* 5, 1 (1998), 7-74. DOI= http://dx.doi.org/10.1080/0969595980050102

[15] Ritter, S. et al. 2007. Cognitive Tutor: Applied research in mathematics education. *Psychonomic bulletin & review.* 14, 2 (2007), 249-255. DOI= http://dx.doi.org/10.3758/BF03194060

[16] Fei, T. et al. 2003. Question Classification for E-learning by Artificial Neural Network. In *Proceedings of the 2003 Joint Fourth International Conference on Information, Communications and Signal Processing and the Fourth Pacific Rim Conference on Multimedia* (Singapore, 2003), 1-5. DOI= http://dx.doi.org/10.1109/ICICS.2003.1292768

[17] Cheng, S. C. et al. 2005. Automatic Leveling System for E-Learning Examination Pool Using Entropy-Based Decision Tree. In *Advances in Web-Based Learning – ICWL 2005* (Hong Kong, 2005), 273-278. DOI= http://dx.doi.org/10.1007/11528043_27

[18] Jayakodi, K. et al. 2015. An Automatic Classifier for Exam Questions in Engineering: A Process for Bloom's Taxonomy. In *2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)* (Zhuhai, China, 2015). DOI= https://dx.doi.org/10.1109/TALE.2015.7386043

[19] Haris, S. S. and Omar, N. 2015. Bloom's taxonomy question categorization using rules and N-gram approach. *Journal of Theoretical and Applied Information Technology.* 76, 3 (2015), 401-407.

[20] Yahya, A. A. et al. 2013. Analyzing the cognitive level of classroom questions using machine learning techniques. In *The 9th International Conference on Cognitive Science* (Kuching, Sarawak, Malaysia, 2013). 587-595. DOI= http://dx.doi.org/10.1016/j.sbspro.2013.10.277

[21] Abduljabbar, D. A. and Omar, N. 2015. Exam questions classification based on Bloom's taxonomy cognitive level using classifiers combination. *Journal of Theoretical and Applied Information Technology.* 78, 3 (2015), 447-455.

[22] Sangodiah, A. et al. 2014. A Review in Feature Extraction Approach in Question Classification Using Support Vector Machine. In *2014 IEEE International Conference on Control System, Computing and Engineering* (Penang, Malaysia, 2014), 536-541. DOI= http://dx.doi.org/10.1109/ICCSCE.2014.7072776

[23] Huang, G. B. et al. 2006. Extreme learning machine: Theory and applications. *Neurocomputing.* 70, 1-3 (2006), 489-501. DOI= http://dx.doi.org/10.1016/j.neucom.2005.12.126

[24] Trinity University Course Assessment and Outcomes: 2016 *https://inside.trinity.edu/collaborative/collaborative-grants/course-redesign-stipends/course-assessment-and-outcomes*. Accessed: 2017-02-24.

[25] Bernardi, R. *Term Frequency and Inverted Document Frequency.* University of Trento, Trentino.

[26] Blei, D. M. et al. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research.* 3 (2003), 993-1022.

[27] Huang, G. B. 2015. What are Extreme Learning Machines? Filling the Gap Between Frank Rosenblatt's Dream and John von Neumann's Puzzle. *Cognitive Computation.* 7, 3 (2015), 263-278. DOI= http://dx.doi.org/10.1007/s12559-015-9333-0

[28] Weston, J. *Support Vector Machine (and Statistical Learning Theory).* NEC Labs America, Princeton.

[29] Chang, C. C. and Lin, C. J. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST).* 2, 3 (2011), 1-39. DOI= http://dx.doi.org/10.1145/1961189.1961199

[30] Santra, A. K. and Christy, C. J. 2012. Genetic Algorithm and Confusion Matrix for Document Clustering. *IJCSI International Journal of Computer Science Issues.* 9, 1 (2012), 322-328.

[31] Hu, D. J. 2009. *Latent Dirichlet Allocation for Text, Images, and Music*.

Behavior-Based Latent Variable Model for Learner Engagement

Andrew S. Lan[1], Christopher G. Brinton[2], Tsung-Yen Yang[3], Mung Chiang[1]
[1]Princeton University, [2]Zoomi Inc., [3]National Chiao Tung University
andrew.lan@princeton.edu, christopher.brinton@zoomiinc.com, tsungyenyang.eecs02@nctu.edu.tw, chiangm@princeton.edu

ABSTRACT

We propose a new model for learning that relates video-watching behavior and engagement to quiz performance. In our model, a learner's knowledge gain from watching a lecture video is treated as proportional to their latent engagement level, and the learner's engagement is in turn dictated by a set of behavioral features we propose that quantify the learner's interaction with the lecture video. A learner's latent concept knowledge is assumed to dictate their observed performance on in-video quiz questions. One of the advantages of our method for determining engagement is that it can be done entirely within standard online learning platforms, serving as a more universal and less invasive alternative to existing measures of engagement that require the use of external devices. We evaluate our method on a real-world massive open online course (MOOC) dataset, from which we find that it achieves high quality in terms of predicting unobserved first-attempt quiz responses, outperforming two state-of-the-art baseline algorithms on all metrics and dataset partitions tested. We also find that our model enables the identification of key behavioral features (e.g., larger numbers of pauses and rewinds, and smaller numbers of fast forwards) that are correlated with higher learner engagement.

Keywords

Behavioral data, engagement, latent variable model, learning analytics, MOOC, performance prediction

1. INTRODUCTION

The recent and rapid development of online learning platforms, coupled with advancements in machine learning, has created an opportunity to revamp the traditional "one-size-fits-all" approach to education. This opportunity is facilitated by the ability of many learning platforms, such as massive open online course (MOOC) platforms, to collect several different types of data on learners, including their assessment responses as well as their learning behavior [9]. The focus of this work is on using different forms of data to model the learning process, which can lead to effective learning analytics and potentially improve learning efficacy.

1.1 Behavior-based learning analytics

Current approaches to learning analytics are focused mainly on providing feedback to learners about their knowledge states – or the level to which they have mastered given concepts/topics/knowledge components – through analysis of their responses to assessment questions [10, 24]. There are other cognitive (e.g., engagement [17, 31], confusion [37], and emotion [11]) as well as non-cognitive (e.g., fatigue, motivation, and level of financial support [14]) factors beyond assessment performance that are crucial to the learning process as well. Accounting for them thus has the potential to yield more effective learning analytics and feedback.

To date, it has been difficult to measure these factors of the learning process. Contemporary online learning platforms, however, have the capability to collect *behavioral data* that can provide some indicators of them. This data commonly includes learners' usage patterns of different types of learning resources [12, 15], their interactions with others via social learning networks [7, 28], their clickstream and keystroke activity logs [2, 8, 30], and sometimes other metadata including facial expressions [35] and gaze location [6].

Recent research has attempted to use behavioral data to augment learning analytics. [5] proposed a latent response model to classify whether a learner is gaming an intelligent tutoring system, for example. Several of these works have sought to demonstrate the relationship between behavior and performance of learners in different scenarios. In the context of MOOCs, [22] concluded that working on more assignments lead to better knowledge transfer than only watching videos, [12] extracted probabilistic use cases of different types of learning resources and showed they are predictive of certification, [32] used discussion forum activity and topic analysis to predict test performance, and [26] discovered that submission activities can be used to predict final exam scores. In other educational domains, [2] discovered that learner keystroke activity in essay-writing sessions is indicative of essay quality, [29] identified behavior as one of the factors predicting math test achievement, and [25] found that behavior is predictive of whether learners can provide elegant solutions to mathematical questions.

In this work, we are interested in how behavioral data can be used to model a learner's *engagement*.

1.2 Learner engagement

Monitoring and fostering engagement is crucial to education, yet defining it concretely remains elusive. Research has sought to identify factors in online learning that may drive engagement; for example, [17] showed that certain production styles of lecture videos promote it. [20] defined disengagement as dropping out in the middle of a video and studied the relationship between disengagement and video content, while [31] considered the relationship between engagement and the

semantic features of mathematical questions that learners respond to. [33] studied the relationship between learners' self-reported engagement levels in a learning session and their facial expressions immediately following in-session quizzes, and [34] considered how engagement is related to linguistic features of discussion forum posts.

There are many types of engagement [3], with the type of interest depending on the specific learning scenario. Several approaches have been proposed for measuring and quantifying different types. These approaches can be roughly divided into two categories: device-based and activity-based. Device-based approaches measure learner engagement using devices external to the learning platform, such as cameras to record facial expressions [35], eye-tracking devices to detect mind wandering while reading text documents [6], and pupil dilation measurements, which are claimed to be highly correlated with engagement [16]. Activity-based approaches, on the other hand, measure engagement using heuristic features constructed from learners' activity logs; prior work includes using replies/upvote counts and topic analysis of discussions [28], and manually defining different engagement levels based on activity types found in MOOCs [4, 21].

Both of these types have their drawbacks. Device-based approaches are far from universal in standard learning platforms because they require integration with external devices. They are also naturally invasive and carry potential privacy risks. Activity-based approaches, on the other hand, are not built on the same granularity of data, and tend to be defined from heuristics that have no guarantee of correlating with learning outcomes. It is therefore desirable to develop a statistically principled, activity-based approach to inferring a learner's engagement.

1.3 Our approach and contributions

In this paper, we propose a probabilistic model for inferring a learner's engagement level by treating it as a latent variable that drives the learner's performance and is in turn driven by the learner's behavior. We apply our framework to a real-world MOOC dataset consisting of clickstream actions generated as learners watch lecture videos, and question responses from learners answering in-video quiz questions.

We first formalize a method for quantifying a learner's behavior while watching a video as a set of nine *behavioral features* that summarize the clickstream data generated (Section 2). These features are intuitive quantities such as the fraction of video played, the number of pauses made, and the average playback rate, some of which have been associated with performance previously [8]. Then, we present our statistical model of learning (Section 3) as two main components: a *learning model* and a *response model*. The learning model treats a learner's gain in concept knowledge as proportional to their latent engagement level while watching a lecture video. Concept knowledge is treated as multidimensional, on a set of latent concepts underlying the course, and videos are associated with varying levels to different concepts. The response model treats a learner's performance on in-video quiz questions, in turn, as proportional to their knowledge on the concepts that this particular question relates to.

By defining engagement to correlate directly with performance, we are able to learn which behavioral features lead to high engagement through a single model. This differs from prior works that first define heuristic notions of engagement and subsequently correlate engagement with performance, in separate procedures. Moreover, our formulation of latent engagement can be made from entirely within standard learning platforms, serving as a more universally applicable and less invasive alternative to device-based approaches.

Finally, we evaluate two different aspects of our model (Section 4): its ability to predict unobserved, first-attempt quiz question responses, and its ability to provide meaningful analytics on engagement. We find that our model predicts with high quality, achieving AUCs of up to 0.76, and outperforming two state-of-the-art baselines on all metrics and dataset partitions tested. One of the partitions tested corresponds to the beginning of the course, underscoring the ability of our model to provide early detection of struggling or advanced students. In terms of analytics, we find that our model enables us to identify behavioral features (e.g., large numbers of pauses and rewinds, and small numbers of fast forwards) that indicate high learner engagement, and to track learners' engagement patterns throughout the course. More generally, these findings can enable an online learning platform to detect learner disengagement and perform appropriate interventions in a fully automated manner.

2. BEHAVIORAL DATA

In this section, we start by detailing the setup of lecture videos and quizzes in MOOCs. We then specify video-watching clickstream data and our method for summarizing it into behavioral features.

2.1 Course setup and data capture

We are interested in modeling learner engagement while watching lecture videos to predict their performance on in-video quiz questions. For this purpose, we can view an instructor's course delivery as the sequence of videos that learners will watch interspersed with the quiz questions they will answer. Let $Q = (q_1, q_2, \ldots)$ be the sequence of questions asked through the course. A video could have any number of questions generally, including none; to enforce a 1:1 correspondence between video content and questions, we will consider the "video" for question q_n to be all video content that appears between q_{n-1} and q_n. Based on this, we will explain the formats of video-watching and quiz response data we work with in this section.

Our dataset. The dataset we will use is from the fall 2012 offering of the course *Networks: Friends, Money, and Bytes* (FMB) on Coursera [1]. This course has 92 videos distributed among 20 lectures, and exactly one question per video.

2.1.1 Video-watching clickstreams

When a learner watches a video on a MOOC, their behavior is typically recorded as a sequence of clickstream actions. In particular, each time a learner makes an action – `play`, `pause`, `seek`, `ratechange`, `open`, or `close` – on the video player, a clickstream event is generated. Formally, the ith event created for the course will be in the format

$$E_i =< u_i, v_i, e_i, p_i', p_i, x_i, s_i, r_i >$$

Here, u_i and v_i are the IDs of the specific learner (user) and video, respectively, and e_i is the type of action that u_i made on v_i. p_i is the position of the video player (in seconds) immediately after e_i is made, p'_i is the position immediately before,[1] x_i is the UNIX timestamp (in seconds) at which e_i was fired, s_i is the binary state of the video player – either `playing` or `paused` – once this action is made, and r_i is the playback rate of the video player once this action is made. Our FMB dataset has 314,632 learner-generated clickstreams from 3,976 learners.[2]

The set $E_{u,v} = \{E_i | u_i = u, v_i = v\}$ of clickstreams for learner u recorded on video v can be used to reconstruct the behavior u exhibits on v. In Section 2.2 we will explain the features computed from $E_{u,v}$ to summarize this behavior.

2.1.2 *Quiz responses*

When a learner submits a response to an in-video quiz question, an event is generated in the format

$$A_m =< u_m, v_m, x_m, a_m, y_m >$$

Again, u_m and v_m are the learner and video IDs (*i.e.,* the quiz corresponding to the video). x_m is the UNIX timestamp of the submission, a_m is the specific response, and y_m is the number of points awarded for the response. The questions in our dataset are multiple choice with a single response, so y_m is binary-valued.

In this work, we are interested in whether quiz responses were correct on first attempt (CFA) or not. As a result, with $A_{u,v} = \{A_m | u_m = u, v_m = v\}$, we consider the event $A'_{u,v}$ in this set with the earliest timestamp $x'_{u,v}$. We also only consider the set of clickstreams $E'_{u,v} \subseteq E_{u,v}$ that occur before $x'_{u,v}$, as the ones after would be anti-causal to CFA.

2.2 Behavioral features and CFA score

With the data $E'_{u,v}$ and $A'_{u,v}$, we construct two sets of information for each learner u on each video v, *i.e.,* each learner-video pair. First is a set of nine behavioral features that summarize u's video-watching behavior on v [8]:

(1) Fraction spent. The fraction of time the learner spent on the video, relative to the playback length of the video. Formally, this quantity is $e_{u,v}/l_v$, where

$$e_{u,v} = \sum_{i \in \mathcal{S}} \min(x_{i+1} - x_i, l_v)$$

is the elapsed time on v obtained by finding the total UNIX time for u on v, and l_v is the length of the video (in seconds). Here, $\mathcal{S} = \{i \in A'_{u,v} : a_{i+1} \neq \texttt{open}\}$. l_v is included as an upper bound for excessively long intervals of time.

(2) Fraction completed. The fraction of the video that the learner completed, between 0 (none) and 1 (all). Formally, it is $c_{u,v}/l_v$, where $c_{u,v}$ is the number of unique 1 second segments of the video that the learner visited.

[1] p_i and p'_i will only differ when i is a skip event.

[2] This number excludes invalid `stall`, `null`, and `error` events, as well as `open` and `close` events which are generated automatically.

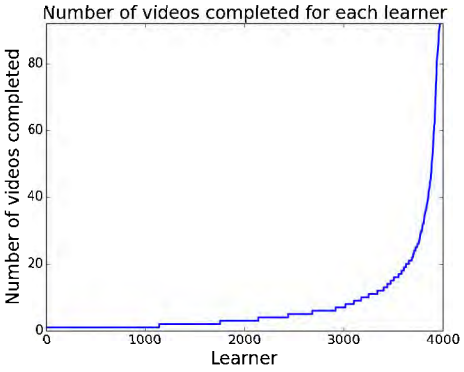


Figure 1: Distribution of the number of videos that each each learner completed in FMB. More than 85% of learners completed less than 20 videos.

(3) Fraction played. The fraction of the video that the learner played relative to the length. Formally, it is calculated as $g_{u,v}/l_v$, where

$$g_{u,v} = \sum_{i \in \mathcal{S}} \min(p'_{i+1} - p_i, l_v)$$

is the total length of video that was played (while in the playing state). Here, $\mathcal{S} = \{i \in A'_{u,v} : a_{i+1} \neq \texttt{open} \ \wedge \ s_i = \texttt{playing}\}$.

(4) Fraction paused. The fraction of time the learner stayed paused on the video relative to the length. It is calculated as $h_{u,v}/l_v$, where

$$h_{u,v} = \sum_{i \in \mathcal{S}} \min(t_{i+1} - t_i, l_v)$$

is the total time the learner stayed in the `paused` state on this video. Here, $\mathcal{S} = \{i \in A'_{u,v} : a_{i+1} \neq \texttt{open} \ \wedge \ s_i = \texttt{paused}\}$.

(5) Number of pauses. The number of times the learner paused the video, or

$$\sum_{i \in A'_{u,v}} \mathbb{1}\{a_i = \texttt{pause}\}$$

where $\mathbb{1}\{\}$ is the indicator function.

(6) Number of rewinds. The number of times the learner skipped backwards in the video, or

$$\sum_{i \in A'_{u,v}} \mathbb{1}\{a_i = \texttt{skip} \ \wedge \ p'_i < p_i\}$$

(7) Number of fast forwards. The number of times the learner skipped forward in the video, *i.e.,* with $p'_i > p_i$ in the previous equation.

(8) Average playback rate. The time-average of the learner's playback rate on the video. Formally, it is calculated as

$$\bar{r}_{u,v} = \frac{\sum_{i \in \mathcal{S}} r_i \cdot \min(x_{i+1} - x_i, l_v)}{\sum_{i \in \mathcal{S}} \min(x_{i+1} - x_i, l_v)}$$

where $\mathcal{S} = \{i \in A'_{u,v} : a_{i+1} \neq \texttt{open} \ \wedge \ s_i = \texttt{playing}\}$.

(9) Standard deviation of playback rate. The standard deviation of the learner's playback rate. It is calculated as

$$\sqrt{\frac{\sum_{i\in\mathcal{S}}(r_i-\bar{r}_{u,v})^2\cdot\min(x_{i+1}-x_i,l_v)}{\sum_{i\in\mathcal{S}}\min(x_{i+1}-x_i,l_v)}}$$

with the same $\mathcal{S}$ as the average playback rate.

The second piece of information for each learner-video pair is u's CFA score $y_{u,v}\in\{0,1\}$ on the quiz question for v.

2.3 Dataset subsets

We will consider different groups of learner-video pairs when evaluating our model in Section 4. Our motivation for doing so is the heterogeneity of learner motivation and high dropoff rates in MOOCs [9]: many will quit the course after watching just a few lectures. Modeling in a small subset of data, particularly those at the beginning of the course, is desirable because it can lead to "early detection" of those who may drop out [8].

Figure 1 shows the dropoff for our dataset in terms of the number of videos each learner completed: more than 85% of learners completed just 20% of the course. "Completed" is defined here as having watched some of the video and responded to the corresponding question. Let T_u be the number of videos learner u completed and $\gamma(v)$ be the index of video v in the course, we define $\Omega^{u_0,v_0}=\{(u,v):T_u\geq u_0\ \wedge\ \gamma(v)\leq v_0\}$ to be the subset of learner-video pairs such that u completed at least u_0 videos and v is within the first v_0 videos. The full dataset is $\Omega^{1,92}$, and we will also consider $\Omega^{20,92}$ as the subset of 346 active learners over the full course and $\Omega^{1,20}$ as the subset of all learners over the first two weeks[3] in our evaluation.

3. STATISTICAL MODEL OF LEARNING WITH LATENT ENGAGEMENT

In this section, we propose our statistical model. Let U denote the number of learners (indexed by u) and V the number of videos (indexed by v). Further, we use T_u to denote the number of time instances registered by learner u (indexed by t); we take a time instance to be a learner completing a video, i.e., watching a video and answering the corresponding quiz question. For simplicity, we use a discrete notion of time, i.e., each learner-video pair will correspond to one time instance for one learner.

Our model considers learners' responses to quiz questions as measurements of their underlying knowledge on a set of concepts; let K denote the number of such concepts. Further, our model considers the action of watching lecture videos as part of learning that changes learners' latent knowledge states over time. These different aspects of the model are visualized in Figure 2: there are two main components, a response model and a learning model.

3.1 Response Model

Our statistical model of learner responses is given by

$$p(y_u^{(t)}=1|\mathbf{c}_u^{(t)})=\sigma(\mathbf{w}_{v(u,t)}^T\mathbf{c}_u^{(t)}-\mu_{v(u,t)}+a_u),\qquad(1)$$

[3]In FMB, the first two weeks of lectures is the first 20 videos.

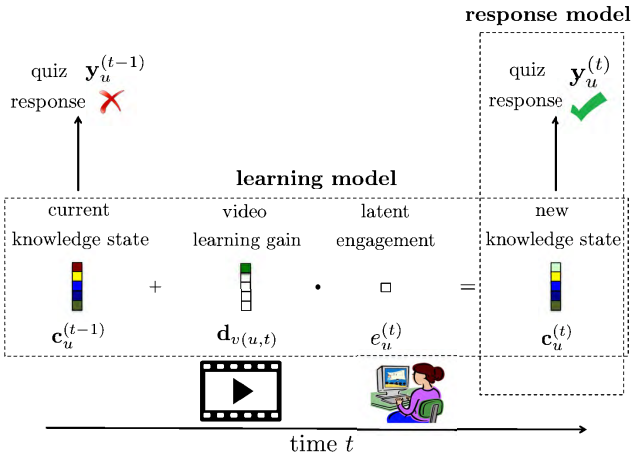


Figure 2: Our proposed statistical model of learning consists of two main parts, a response model and a learning model.

where $v(u,t):\Omega\subseteq\{1,\ldots,U\}\times\{1,\ldots,\max_u T_u\}\rightarrow\{1,\ldots,V\}$ denotes a mapping from a learner index-time index pair to the index of the video v that u was watching at t. $y_u^{(t)}\in\{0,1\}$ is the binary-valued CFA score of learner u on the quiz question corresponding to the video they watch at time t, with 1 denoting a correct response (CFA) and 0 denoting an incorrect response (non-CFA).

The variable $\mathbf{w}_v\in\mathbb{R}_+^K$ denotes the non-negative, K-dimensional quiz question–concept association vector that characterizes how the quiz question corresponding to video v tests learners' knowledge on each concept, and the variable μ_v is a scalar characterizing the intrinsic difficulty of the quiz question. $\mathbf{c}_u^{(t)}$ is the K-dimensional concept knowledge vector of learner u at time t, characterizing the knowledge level of the learner on each concept at the time, and a_u denotes the static, intrinsic ability of learner u. Finally, $\sigma(x)=\frac{1}{1+e^{-x}}$ is the sigmoid function.

We restrict the question–concept association vector $\mathbf{w}_v$ to be non-negative in order to make the parameters interpretable [24]. Under this restriction, the values of concept knowledge vector $\mathbf{c}_u^{(t)}$ can be understood as follows: large, positive values lead to higher chances of answering a question correctly, thus corresponding to high knowledge, while small, negative values lead to lower chances of answering a question correctly, thus corresponding to low knowledge.

3.2 Learning Model

Our model of learning considers transitions in learners' knowledge states as induced by watching lecture videos. It is given by

$$\mathbf{c}_u^{(t)}=\mathbf{c}_u^{(t-1)}+e_u^{(t)}\mathbf{d}_{v(u,t)},\quad t=1,\ldots,T_u,\qquad(2)$$

where the variable $\mathbf{d}_v\in\mathbb{R}_+^K$ denotes the non-negative, K-dimensional learning gain vector for video v; each entry characterizes the degree to which the video improves learners' knowledge level on each concept. The assumption of non-negativity on $\mathbf{d}_v$ implies that videos will not negatively affect learners' knowledge, as in [23]. $\mathbf{c}_u^{(0)}$ is the initial knowledge state of learner u at time $t=0$, i.e., before starting the

	$\Omega^{20,92}$		$\Omega^{1,20}$		$\Omega^{1,92}$	
	ACC	AUC	ACC	AUC	ACC	AUC
Proposed model	**0.7293±0.0070**	**0.7608±0.0094**	**0.7096±0.0057**	**0.7045±0.0066**	**0.7058±0.0054**	**0.7216±0.0054**
SPARFA	0.7209±0.0070	0.7532±0.0098	0.7061±0.0069	0.7020±0.0070	0.6975±0.0048	0.7124±0.0050
BKT	0.7038±0.0084	0.7218±0.0126	0.6825±0.0058	0.6662±0.0065	0.6803±0.0055	0.6830±0.0059

Table 1: Quality comparison of the different algorithms on predicting unobserved quiz question responses. The obtained ACC and AUC metrics on different subsets of the FMB dataset are given. Our proposed model obtains higher quality than the SPARFA and BKT baselines in each case.

course and watching any video.

The scalar latent variable $e_u^{(t)} \in [0, 1]$ in (2) characterizes the *engagement level* that learner u exhibits when watching video $v(u, t)$ at time t. This is in turn modeled as

$$e_u^{(t)} = \sigma(\boldsymbol{\beta}^T \mathbf{f}_u^{(t)}), \tag{3}$$

where $\mathbf{f}_u^{(t)}$ is a 9-dimensional vector of the behavioral features defined in Section 2.2, summarizing learner u's behavior while the video at time t. $\boldsymbol{\beta}$ is the unknown, 9-dimensional parameter vector that characterizes how engagement associates with each behavioral feature.

Taken together, (2) and (3) state that the knowledge gain a learner will experience on a particular concept while watching a particular video is given by

(i) the video's intrinsic association with the concept, modulated by

(ii) the learner's engagement while watching the video, as manifested by their clickstream behavior.

From (2), a learner's (latent) engagement level dictates the fraction of the video's available learning gain they acquire to improve their knowledge on each concept. The response model (1) in turn holds that performance is dictated by a learner's concept knowledge states. In this way, engagement is directly correlated with performance through the concept knowledge states. Note that in this paper, we treat the engagement variable $e_u^{(t)}$ as a scalar; the extension of modeling it as a vector and thus separating engagement by concept is part of our ongoing work.

It is worth mentioning the similarity between our characterization of engagement as a latent variable in the learning model and the input gate variables in long-short term memory (LSTM) neural networks [18]. In LSTM, the change in the latent memory state (loosely corresponding to the latent concept knowledge state vector $\mathbf{c}_u^{(t)}$) is given by the input vector (loosely corresponding to the video learning gain vector $\mathbf{d}_v$) modulated by a set of input gate variables (corresponding to the engagement variable $e_u^{(t)}$).

Parameter inference. Our statistical model of learning and response can be seen as a particular type of recurrent neural network (RNN). Therefore, for parameter inference, we implement a stochastic gradient descent algorithm with standard backpropagation. Given the graded learner responses $y_u^{(t)}$ and behavioral features $\mathbf{f}_u^{(t)}$, our parameter inference algorithm estimates the quiz question–concept association vectors $\mathbf{w}_v$, the quiz question intrinsic difficulties μ_v, the the video learning gain vectors $\mathbf{d}_v$, the learner initial knowledge vectors $\mathbf{c}_u^{(0)}$, the learner abilities a_u, and the engagement–behavioral feature association vector $\boldsymbol{\beta}$. We omit the details of the algorithm for simplicity of exposition.

4. EXPERIMENTS

In this section, we evaluate the proposed latent engagement model on the FMB dataset. We first demonstrate the gain in predictive quality of the proposed model over two baseline algorithms (Section 4.1), and then show how our model can be used to study engagement (Section 4.2).

4.1 Predicting unobserved responses

We evaluate our proposed model's quality by testing its ability to predict unobserved quiz question responses.

Baselines. We compare our model against two well-known, state-of-the-art response prediction algorithms that do not use behavioral data. First is the sparse factor analysis (SPARFA) algorithm [24], which factors the learner-question matrix to extract latent concept knowledge, but does not use a time-varying model of learners' knowledge states. Second is a version of the Bayesian knowledge tracing (BKT) algorithm that tracks learners' time-varying knowledge states, which incorporates a set of guessing and slipping probability parameters for each question, a learning probability parameter for each video, and an initial knowledge level parameter for each learner [13, 27].

4.1.1 Experimental setup and metrics

Regularization. In order to prevent overfitting, we add ℓ_2-norm regularization terms to the overall optimization objective function for every set of variables in both the proposed model and in SPARFA. We use a parameter λ to control the amount of regularization on each variable.

Cross validation. We perform 5-fold cross validation on the full dataset ($\Omega^{1,92}$), and on each subset of the dataset introduced in Section 2.3 ($\Omega^{20,92}$ and $\Omega^{1,20}$). To do so, we randomly partition each learner's quiz question responses into 5 data folds. Leaving out one fold as the test set, we use the remaining four folds as training and validation sets to select the values of the tuning parameters for each algorithm, i.e., by training on three of the folds and validating on the other. We then train every algorithm on all four observed folds using the tuned values of the parameters, and evaluate them on the holdout set. All experiments are repeated for 20 random partitions of the training and test sets.

For the proposed model and for SPARFA, we tune both the

Feature	Coefficient
Fraction spent	0.1941
Fraction completed	0.1443
Fraction played	0.2024
Fraction paused	0.0955
Number of pauses	0.2233
Number of rewinds	0.4338
Number of fast forwards	−0.1551
Average playback rate	0.2797
Standard deviation of playback rate	0.0314

Table 2: Regression coefficient vector $\boldsymbol{\beta}$ learned over the full dataset, associating each clickstream feature to engagement. All but one of the features (number of fast forwards) is positively correlated with engagement.

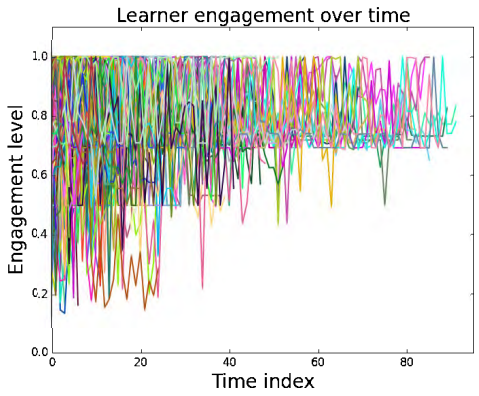


Figure 3: Plot of the latent engagement level $e_j^{(t)}$ over time for one third of the learners in FMB, showing a diverse set of behaviors across learners.

number of concepts $K \in \{2, 4, 6, 8, 10\}$ and the regularization parameter $\lambda \in \{0.5, 1.0, \ldots, 10.0\}$. Note that for the proposed model, when a question response is left out as part of the test set, only the response is left out of the training set: the algorithm still uses the clickstream data for the corresponding learner-video pair to model engagement.

Metrics. To evaluate the quality of the algorithms, we employ two commonly used binary classification metrics: prediction accuracy (ACC) and area under the receiver operating characteristic curve (AUC) [19]. The ACC metric is simply the fraction of predictions that are made correctly, while the AUC measures the tradeoff between the true and false positive rates of the classifier. Both metrics take values in $[0, 1]$, with larger values indicating higher quality.

4.1.2 *Results and discussion*

Table 1 gives the evaluation results for the three algorithms. The average and standard deviation over the 20 random data partitions are reported for each dataset group and metric.

First of all, the results show that our proposed model consistently achieves higher quality than both baseline algorithms on both metrics. It significantly outperforms BKT in particular (SPARFA also outperforms BKT). This shows the potential of our model to push the envelope on achievable quality in performance prediction research.

Notice that our model achieves its biggest quality improvement on the full dataset, with a 1.3% gain in AUC over SPARFA and a 5.7% gain over BKT. This observation suggests that as more clickstream data is captured and available for modeling – especially as we observe more video-watching behavioral data from learners over a longer period of time (the full dataset $\Omega^{1,92}$ contains clickstream data for up to 12 weeks, while the $\Omega^{1,20}$ subset only contains data for the first 2 weeks) – the proposed model achieves more significant quality enhancements over the baseline algorithms. This is somewhat surprising, since prior work on behavior-based performance prediction [8] has found the largest gains in the presence of fewer learner-video pairs, i.e., before there are many question responses for other algorithms to model on. But our algorithm also benefits from additional question responses, to update its learned relationship between behavior and concept knowledge.

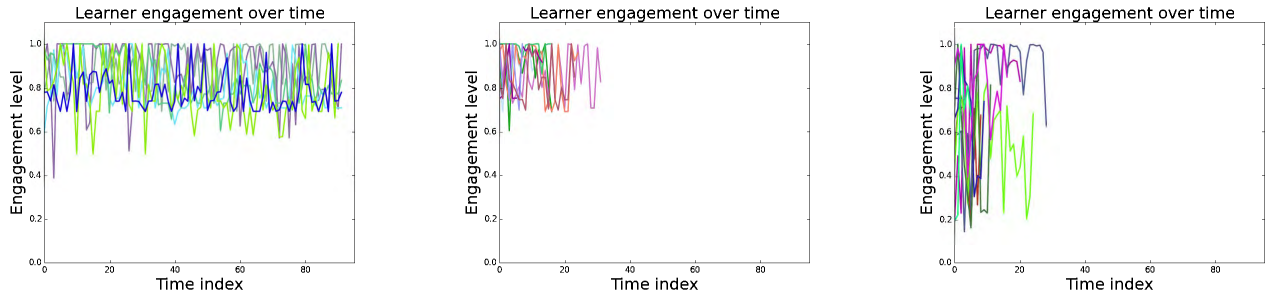
The first two weeks of data ($\Omega^{1,20}$) is sparse in that the majority of learners answer at most a few questions during this time, many of whom will drop out (see Figure 1). In this case, our model obtains a modest improvement over SPARFA, which is static and uses fewer parameters. The gain over BKT is particularly pronounced, at 5.7%. This, combined with the findings for active learners over the full course ($\Omega^{20,92}$), shows that observing video-watching behavior of learners who drop out of the course in its early states (these learners are excluded from $\Omega^{20,92}$) leads to a slight increase in the performance gain of the proposed model over the baseline algorithms. Importantly, this shows that our algorithm provides benefit for *early detection*, with the ability to predict performance of learners who will end up dropping out [8].

4.2 Analyzing engagement

Given predictive quality, one benefit of our model is that it can be used to analyze engagement. The two parameters to consider for this are the regression coefficient vector $\boldsymbol{\beta}$ and the engagement scalar $e_u^{(t)}$ itself.

Behavior and engagement. Table 2 gives each of the estimated feature coefficients in $\boldsymbol{\beta}$ for the full dataset $\Omega^{1,92}$, with regularization parameters chosen via cross validation. All of the features except for the number of fast forwards are positively correlated with the latent engagement level. This is to be expected since many of the features are associated with processing more video content, e.g., spending more time, playing more, or pausing longer to reflect, while fast forwarding involves skipping over the content.

The features that contribute most to high latent engagement levels are the number of pauses, the number of rewinds, and the average playback rate. The first two of these are likely indicators of actual engagement as well, since they indicate whether the learner was thinking while pausing the video or re-visiting earlier content which contains knowledge that they need to recall or revise. The strong, positive correlation of average playback rate is somewhat surprising though: we may expect that a higher playback rate would have a



(a) Learners that consistently exhibit high engagement and finish the course.

(b) Learners that exhibit high engagement but drop out early.

(c) Learners that exhibit inconsistent engagement and drop out.

Figure 4: Plot of the latent engagement level $e_j^{(t)}$ over time for selected learners in three different groups.

negative impact on engagement, like fast forwarding does, as it involves speeding through content. On the other hand, it may be an indication that learners are more focused on the material and trying to keep their interest higher.

Engagement over time. Figure 3 visualizes the evolution of $e_u^{(t)}$ over time for 1/3 of the learners (randomly selected). Patterns in engagement differs substantially across learners; those who finish the course mostly exhibit high engagement levels throughout, while those who drop out early vary greatly in their engagement, some high and others low.

Figure 4 breaks down the learners into three different types according to their engagement patterns, and plots their engagement levels over time separately. The first type of learner (a) finishes the course and consistently exhibits high engagement levels throughout the duration. The second type (b) also consistently exhibits high engagement levels, but drops out of the course after up to three weeks. The third type of learner (c) exhibits inconsistent engagement levels before an early dropout. Equipped with temporal plots like these, an instructor could determine which learners may be in need of intervention, and could design different interventions for different engagement clusters [8, 36].

5. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a new statistical model for learning, based on learner behavior while watching lecture videos and their performance on in-video quiz questions. Our model has two main parts: (i) a response model, which relates a learner's performance to latent concept knowledge, and (ii) a learning model, which relates the learner's concept knowledge in turn to their latent engagement level while watching videos. Through evaluation on a real-world MOOC dataset, we showed that our model can predict unobserved question responses with superior quality to two state-of-the-art baselines, and also that it can lead to engagement analytics: it identifies key behavioral features driving high engagement, and shows how each learner's engagement evolves over time.

Our proposed model enables the measurement of engagement solely from data that is logged within online learning platforms: clickstream data and quiz responses. In this way, it serves as a less invasive alternative to current approaches for measuring engagement that require external devices, e.g., cameras and eye-trackers [6, 16, 35]. One avenue of future work is to conduct an experiment that will correlate our definition of latent engagement with these methods.

Additionally, one could test other, more sophisticated characterizations of the latent engagement variable. One such approach could seek to characterize engagement as a function of learners' previous knowledge level. An alternative or addition to this would be a generative modeling approach of engagement to enable the prediction of future engagement given each learner's learning history.

One of the long-term, end-all goals of this work is the design of a method for useful, real-time analytics to instructors. The true test of this ability comes from incorporating the method into a learning system, providing its outputs – namely, performance prediction forecasts and engagement evolution – to an instructor through the user interface, and measuring the resulting improvement in learning outcomes.

Acknowledgments

Thanks to Debshila Basu Mallick for discussions on the different types of engagement.

6. REFERENCES

[1] Networks: Friends, Money, and Bytes. `https://www.coursera.org/course/friendsmoneybytes`.

[2] L. Allen, M. Jacovina, M. Dascalu, R. Roscoe, K. Kent, A. Likens, and D. McNamara. {ENTER}ing the time series {SPACE}: Uncovering the writing process through keystroke analyses. In *Proc. Intl. Conf. Educ. Data Min.*, pages 22–29, June 2016.

[3] A. Anderson, S. Christenson, M. Sinclair, and C. Lehr. Check & connect: The importance of relationships for promoting engagement with school. *J. School Psychol.*, 42(2):95–113, Mar. 2004.

[4] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Engaging with massive online courses. In *Proc. Intl. Conf. World Wide Web*, pages 687–698, Apr. 2014.

[5] R. Baker, A. Corbett, and K. Koedinger. Detecting student misuse of intelligent tutoring systems. In *Proc. Intl. Conf. Intell. Tutoring Syst.*, pages 531–540, Aug. 2004.

[6] R. Bixler and S. D'Mello. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Model. User-adapt. Interact.*, 26(1):33–68, Mar. 2016.

[7] C. Brinton, S. Buccapatnam, F. Wong, M. Chiang, and H. Poor. Social learning networks: Efficiency optimization for MOOC forums. In *Proc. IEEE Conf.*

Comput. Commun., pages 1–9, Apr. 2016.

[8] C. Brinton and M. Chiang. MOOC performance prediction via clickstream data and social learning networks. In *Proc. IEEE Conf. Comput. Commun.*, pages 2299–2307, April 2015.

[9] C. Brinton, R. Rill, S. Ha, M. Chiang, R. Smith, and W. Ju. Individualization for education at scale: MIIC design and preliminary evaluation. *IEEE Trans. Learn. Technol.*, 8(1):136–148, Jan. 2015.

[10] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis – A general method for cognitive model evaluation and improvement. In *Proc. Intl. Conf. Intell. Tutoring Syst.*, pages 164–175, June 2006.

[11] L. Chen, X. Li, Z. Xia, Z. Song, L. Morency, and A. Dubrawski. Riding an emotional roller-coaster: A multimodal study of young child's math problem solving activities. In *Proc. Intl. Conf. Educ. Data Min.*, pages 38–45, June 2016.

[12] C. Coleman, D. Seaton, and I. Chuang. Probabilistic use cases: Discovering behavioral patterns for predicting certification. In *Proc. ACM Conf. Learn at Scale*, pages 141–148, Mar. 2015.

[13] A. Corbett and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Model. User-adapt. Interact.*, 4(4):253–278, Dec. 1994.

[14] C. Farrington, M. Roderick, E. Allensworth, J. Nagaoka, T. Keyes, D. Johnson, and N. Beechum. *Teaching Adolescents to Become Learners: The Role of Noncognitive Factors in Shaping School Performance–A Critical Literature Review*. Consortium on Chicago School Research, 2012.

[15] B. Gelman, M. Revelle, C. Domeniconi, A. Johri, and K. Veeramachaneni. Acting the same differently: A cross-course comparison of user behavior in MOOCs. In *Proc. Intl. Conf. Educ. Data Min.*, pages 376–381, June 2016.

[16] M. Gilzenrat, J. Cohen, J. Rajkowski, and G. Aston-Jones. Pupil dynamics predict changes in task engagement mediated by locus coeruleus. In *Proc. Soc. Neurosci. Abs.*, page 19, Nov. 2003.

[17] P. Guo, J. Kim, and R. Rubin. How video production affects student engagement: An empirical study of MOOC videos. In *Proc. ACM Conf. Learn at Scale*, pages 41–50, Mar. 2014.

[18] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[19] H. Jin and C. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Trans. Knowl. Data Eng.*, 17(3):299–310, Mar. 2005.

[20] J. Kim, P. Guo, D. Seaton, P. Mitros, K. Gajos, and R. Miller. Understanding in-video dropouts and interaction peaks in online lecture videos. In *Proc. ACM Conf. Learn at Scale*, pages 31–40, Mar. 2014.

[21] R. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: Analyzing learner subpopulations in massive open online courses. In *Proc. Intl. Conf. Learn. Analyt. Knowl.*, pages 170–179, Apr. 2013.

[22] K. Koedinger, J. Kim, J. Jia, E. McLaughlin, and N. Bier. Learning is not a spectator sport: Doing is better than watching for learning from a MOOC. In *Proc. ACM Conf. Learn at Scale*, pages 111–120, Mar. 2015.

[23] A. Lan, C. Studer, and R. Baraniuk. Time-varying learning and content analytics via sparse factor analysis. In *Proc. ACM SIGKDD Intl. Conf. Knowl. Discov. Data Min.*, pages 452–461, Aug. 2014.

[24] A. Lan, A. Waters, C. Studer, and R. Baraniuk. Sparse factor analysis for learning and content analytics. *J. Mach. Learn. Res.*, 15:1959–2008, June 2014.

[25] L. Malkiewich, R. Baker, V. Shute, S. Kai, and L. Paquette. Classifying behavior to elucidate elegant problem solving in an educational game. In *Proc. Intl. Conf. Educ. Data Min.*, pages 448–453, June 2016.

[26] J. McBroom, B. Jeffries, I. Koprinska, and K. Yacef. Mining behaviours of students in autograding submission system logs. In *Proc. Intl. Conf. Educ. Data Min.*, pages 159–166, June 2016.

[27] Z. Pardos and N. Heffernan. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In *Proc. Intl. Conf. User Model. Adapt. Personalization*, pages 255–266, June 2010.

[28] J. Reich, B. Stewart, K. Mavon, and D. Tingley. The civic mission of MOOCs: Measuring engagement across political differences in forums. In *Proc. ACM Conf. Learn at Scale*, pages 1–10, Apr. 2016.

[29] M. San Pedro, E. Snow, R. Baker, D. McNamara, and N. Heffernan. Exploring dynamical assessments of affect, behavior, and cognition and Math state test achievement. In *Proc. Intl. Conf. Educ. Data Min.*, pages 85–92, June 2015.

[30] C. Shi, S. Fu, Q. Chen, and H. Qu. VisMOOC: Visualizing video clickstream data from massive open online courses. In *IEEE Pacific Visual. Symp.*, pages 159–166, Apr. 2015.

[31] S. Slater, R. Baker, J. Ocumpaugh, P. Inventado, P. Scupelli, and N. Heffernan. Semantic features of Math problems: Relationships to student learning and engagement. In *Proc. Intl. Conf. Educ. Data Min.*, pages 223–230, June 2016.

[32] S. Tomkins, A. Ramesh, and L. Getoor. Predicting post-test performance from online student behavior: A high school MOOC case study. In *Proc. Intl. Conf. Educ. Data Min.*, pages 239–246, June 2016.

[33] A. Vail, J. Wiggins, J. Grafsgaard, K. Boyer, E. Wiebe, and J. Lester. The affective impact of tutor questions: Predicting frustration and engagement. In *Proc. Intl. Conf. Educ. Data Min.*, pages 247–254, June 2016.

[34] X. Wang, D. Yang, M. Wen, K. Koedinger, and C. Rosé. Investigating how student's cognitive behavior in MOOC discussion forums affect learning gains. In *Proc. Intl. Conf. Educ. Data Min.*, pages 226–233, June 2015.

[35] J. Whitehill, Z. Serpell, Y. Lin, A. Foster, and J. Movellan. The faces of engagement: Automatic recognition of student engagement from facial expressions. *IEEE Trans. Affect. Comput.*, 5(1):86–98, Jan. 2014.

[36] J. Whitehill, J. Williams, G. Lopez, C. Coleman, and J. Reich. Beyond prediction: Towards automatic intervention in MOOC student stop-out. In *Proc. Intl. Conf. Educ. Data Min.*, pages 171–178, June 2015.

[37] D. Yang, R. Kraut, and C. Rosé. Exploring the effect of student confusion in massive open online courses. *J. Educ. Data Min.*, 8(1):52–83, 2016.

Efficient Feature Embeddings for Student Classification with Variational Auto-encoders

Severin Klingler
Dept. of Computer Science
ETH Zurich, Switzerland
kseverin@inf.ethz.ch

Rafael Wampfler
Dept. of Computer Science
ETH Zurich, Switzerland
wrafael@inf.ethz.ch

Tanja Käser
Graduate School of Education
Stanford University, USA
tkaeser@stanford.edu

Barbara Solenthaler
Dept. of Computer Science
ETH Zurich, Switzerland
sobarbar@inf.ethz.ch

Markus Gross
Dept. of Computer Science
ETH Zurich, Switzerland
grossm@inf.ethz.ch

ABSTRACT

Gathering labeled data in educational data mining (EDM) is a time and cost intensive task. However, the amount of available training data directly influences the quality of predictive models. Unlabeled data, on the other hand, is readily available in high volumes from intelligent tutoring systems and massive open online courses. In this paper, we present a semi-supervised classification pipeline that makes effective use of this unlabeled data to significantly improve model quality. We employ deep variational auto-encoders to learn efficient feature embeddings that improve the performance for standard classifiers by up to 28% compared to completely supervised training. Further, we demonstrate on two independent data sets that our method outperforms previous methods for finding efficient feature embeddings and generalizes better to imbalanced data sets compared to expert features. Our method is data independent and classifier-agnostic, and hence provides the ability to improve performance on a variety of classification tasks in EDM.

Keywords

semi-supervised classification, variational auto-encoder, deep neural networks, dimensionality reduction

1. INTRODUCTION

Building predictive models of student characteristics such as knowledge level, learning disabilities, personality traits or engagement is one of the big challenges in educational data mining (EDM). Such detailed student profiles allow for a better adaptation of the curriculum to the individual needs and is crucial for fostering optimal learning progress. In order to build such predictive models, smaller-scale and controlled user studies are typically conducted where detailed information about student characteristics are at hand (labeled data). The quality of the predictive models, however, inherently depends on the number of study participants, which is typically on the lower side due to time and budget constraints. In contrast to such controlled user studies, digital learning environments such as intelligent tutoring systems (ITS), educational games, learning simulations, and massive open online courses (MOOCs) produce high volumes of data. These data sets provide rich information about student interactions with the system, but come with no or only little additional information about the user (unlabeled data).

Semi-supervised learning bridges this gap by making use of patterns in bigger unlabeled data sets to improve predictions on smaller labeled data sets. This is also the focus of this paper. These techniques are well explored in a variety of domains and it has been shown that classifier performance can be improved for, e.g., image classification [15], natural language processing [28] or acoustic modeling [21]. In the education community, semi-supervised classification has been used employing self-training, multi-view training and problem-specific algorithms. Self-training has e.g. been applied for problem-solving performance [22]. In self-training, a classifier is first trained on labeled data and is then iteratively retrained using its most confident predictions on unlabeled data. Self-training has the disadvantage that incorrect predictions decrease the quality of the classifier. Multi-view training uses different data views and has been explored with co-training [27] and tri-training [18] for predicting prerequisite rules and student performance, respectively. The performance of these methods, however, largely depends on the properties of the different data views, which are not yet fully understood [34]. Problem-specific semi-supervised algorithms have been used to organize learning resources in the web [19], with the disadvantage that they cannot be directly applied for other classification tasks.

Recently, it has been shown (outside of the education context) that variational auto-encoders (VAE) have the potential to outperform the commonly used semi-supervised classification techniques. VAE is a neural network that includes an encoder that transforms a given input into a typically lower-dimensional representation, and a decoder that reconstructs the input based on the latent representation. Hence, VAEs learn an efficient feature embedding (feature representation) using unlabeled data that can be used to improve the performance of any standard supervised learning algorithm [15]. This property greatly reduces the need for problem-specific algorithms. Moreover, VAEs feature the advantage that the trained deep generative models are able to produce realistic samples that allow for accurate data imputation and simulations [23], which makes them an appealing choice for EDM. Inspired by these advantages, and the demonstrated superior classifier performance in other domains as in computer vision [16, 23], this paper explores VAE for student classification in the educational context.

We present a complete semi-supervised classification pipeline that employs deep VAEs to extract efficient feature embeddings from unlabeled student data. We have optimized the architecture of two different networks for educational data - a simple variational auto-encoder and a convolutional variational auto-encoder. While our method is generic and hence widely applicable, we apply the pipeline to the problem of detecting students suffering from developmental dyscalculia (DD), which is a learning disability in arithmetics. The large and unlabeled data set at hand consists of student data of more than 7K students and we evaluate the performance of our pipeline on two independent small and labeled data sets with 83 and 155 students. Our evaluation first compares the performance of the two networks, where our results indicate superiority of the convolutional VAE. We then apply different classifiers to both labeled data sets, and demonstrate not only improvements in classification performance of up to 28% compared to other feature extraction algorithms, but also improved robustness to class imbalance when using our pipeline compared to other feature embeddings. The improved robustness of our VAE is especially important for predicting relatively rare student conditions - a challenge that is often met in EDM applications.

2. BACKGROUND

In the semi-supervised classification setting we have access to a large data set $\mathcal{X}_B$ without labels and a much smaller labeled data set $\mathcal{X}_S$ with labels $\mathcal{Y}_S$. The idea behind semi-supervised classification is to make use of patterns in the unlabeled data set to improve the quality of the classifier beyond what would be possible with the small data set $\mathcal{X}_S$ alone. There are many different approaches to semi-supervised classification including transductive SVMs, graph-based methods, self-training or representation learning [35]. In this work we focus on learning an efficient encoding $\mathbf{z} = E(\mathbf{x})$ for $\mathbf{x} \in \mathcal{X}_B$ of the data domain using the unlabeled data $\mathcal{X}_B$ only. This learnt data transformation $E(\cdot)$ - the encoding - is then applied to the labeled data set $\mathcal{X}_S$. Well-known encoders include principle component analysis (PCA) or Kernel PCA (KPCA). PCA is a dimensionality reduction method that finds the optimal linear transformation from an N-dimensional to a K-dimensional space (given a mean-squared error loss). Kernel PCA [24] extends PCA allowing non-linear transformations into a K-dimensional space and has, among others, been successfully used for novelty detection in non-linear domains [11]. Recently, variational auto-encoders (VAE) have outperformed other semi-supervised classification techniques on several data sets [15]. VAE combine variational inference networks with generative models parametrized by deep neural networks that exploit information in the data density to find efficient lower dimensional representations (feature embeddings) of the data.

Auto-encoder. An auto-encoder or autoassociator [2] is a neural network that encodes a given input into a (typically lower dimensional) representation such that the original input can be reconstructed approximately. The auto-encoder consists of two parts. The encoder part of the network takes the N-dimensional input $\mathbf{x} \in \mathbb{R}^N$ and computes an encoding $\mathbf{z} = E(\mathbf{x})$ while the decoder D reconstructs the input based on the latent representation $\hat{\mathbf{x}} = D(\mathbf{z})$. If we train a network using the mean squared error loss and the network consists of a single linear hidden layer of size K, e.g. $E(\mathbf{x}) = \mathbf{W_1}\mathbf{x} + \mathbf{b_1}$ and $D(\mathbf{z}) = \mathbf{W_2}\mathbf{z} + \mathbf{b_2}$ for weights $\mathbf{W}_1 \in \mathbb{R}^{K \times N}$ and $\mathbf{W}_2 \in \mathbb{R}^{N \times K}$ and offsets $\mathbf{b}_1 \in \mathbb{R}^K$ and $\mathbf{b}_2 \in \mathbb{R}^N$, the autoencoder behaves similar to PCA in that the network learns to project the input into the span of the K first principle components [2]. For more complex networks with non-linear layers multi-modal aspects of the data can be learnt. Auto-encoders can be used in semi-supervised classification tasks because the encoder can compute a feature representation $\mathbf{z}$ of the original data $\mathbf{x}$. These features can then be used to train a classifier. The learnt feature embedding facilitates classification by clustering related observations in the computed latent space.

Variational auto-encoder. Variational auto-encoders [15] are generative models that combine Bayesian inference with deep neural networks. They model the input data $\mathbf{x}$ as

$$p_\theta(\mathbf{x}|\mathbf{z}) = f(\mathbf{x}; \mathbf{z}, \theta) \qquad p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, I) \tag{1}$$

where f is a likelihood function that performs a non-linear transformation with parameters θ of $\mathbf{z}$ by employing a deep neural network. In this model the exact computation of the posterior $p_\theta(\mathbf{z}|\mathbf{x})$ is not computationally tractable. Instead, the true posterior is approximated by a distribution $q_\phi(\mathbf{z}|\mathbf{x})$ [16]. This inference network $q_\phi(\mathbf{z}|\mathbf{x})$ is parametrized as a multivariate normal distribution as

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \mathrm{diag}(\sigma^2_\phi(\mathbf{x}))), \tag{2}$$

where $\mu_\phi(\mathbf{x})$ and $\sigma^2_\phi(\mathbf{x})$ denote vectors of means and variance respectively. Both functions $\mu_\phi(\cdot)$ and $\sigma^2_\phi(\cdot)$ are represented as deep neural networks. Hence, variational autoencoders essentially replace the deterministic encoder $E(\mathbf{x})$ and decoder $D(\mathbf{z})$ by a probabilistic encoder $q_\phi(\mathbf{z}|\mathbf{x})$ and decoder $p_\theta(\mathbf{x}|\mathbf{z})$. Direct maximization of the likelihood is computationally not tractable, therefore a lower bound on the likelihood has been derived [16]. The learning task then amounts to maximizing this variational lower bound

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \mathrm{KL}\left[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right], \tag{3}$$

where KL denotes the Kullback-Leibler divergence. The lower bound consists of two intuitive terms. The first term is the reconstruction quality while the second one regularizes the latent space towards the prior $p(\mathbf{z})$. We perform optimization of this lower bound by applying a stochastic optimization method using gradient back-propagation [14].

3. METHOD

In the following we introduce two networks. First, a simple variational auto-encoder consisting of fully connected layers to learn feature embeddings of student data. These encoders have shown to be powerful for semi-supervised classification [15], and are often applied due to their simplicity. Second, an advanced auto-encoder that combines the advantages of VAE with the superiority of asymmetric encoders. This is motivated by the fact that asymmetric auto-encoders have shown superior performance and more meaningful feature representations compared to simple VAE in other domains such as image synthesis [29].

Student snapshots. There are many applications where we want to predict a label y_n for each student n within an ITS based on behavioral data X_n. These labels typically relate to external variables or properties of a student, such

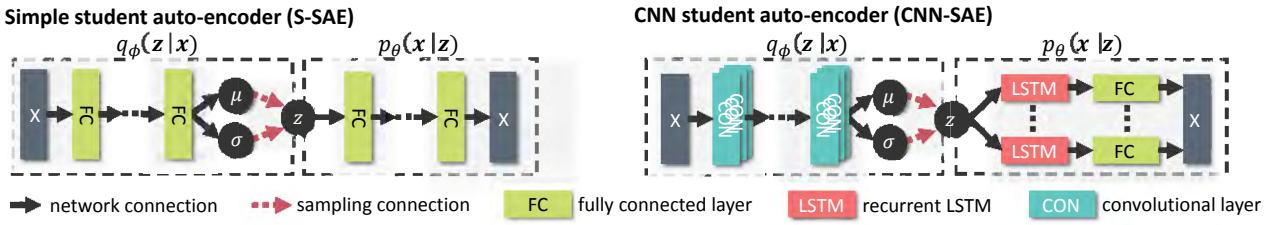


Figure 1: Network layouts for our simple student auto-encoder (left) using only fully connected layers and our improved CNN student auto-encoder (right) using convolutions for the encoder and recurrent LSTM layers for the decoder. In contrast to standard auto-encoders, the connections to the latent space z are sampled (red dashed arrows) from a Gaussian distribution.

as age, learning disabilities, personality traits, learner types, learning outcome etc. Similar to Knowledge Tracing (KT) we propose to model the data $X_n = \{\mathbf{x}_{n1}, \ldots, \mathbf{x}_{nT}\}$ as a sequence of T observations. In contrast to KT we store F different feature values $\mathbf{x}_{nt} \in \mathbb{R}^F$ for each element in the sequence, where t denotes the t^{th} opportunity within a task. This allows us to simultaneously store data from multiple tasks in $\mathbf{x}_{nt}$, e.g. $\mathbf{x}_{n1}$ stores all features for student n that were observed during the first task opportunities. For every task in an ITS we can extract various different features that characterize how a student n was approaching the task. These features include performance, answer times, problem solving strategies, etc. We combine this information into a student snapshot $\mathbf{X}_n \in \mathbb{R}^{T \times F}$, where T is the number of task opportunities and F is the number of extracted features.

Simple student auto-encoder (S-SAE). Our simple variational autoencoder is following the general design outlined in Section 2 and is based on the student snapshot representation. For ease of notation we use $\mathbf{x} := \text{vec}(\mathbf{X_n})$, where $\text{vec}(\cdot)$ is the matrix vectorization function to represent the student snapshot of student n. The complete network layout is depicted in Figure 1, left. The encoder and decoder networks consist of L fully connected layers that are implemented as an affine transformation of the input followed by a non-linear activation function $\beta(\cdot)$ as $\mathbf{x}_l = \beta(\mathbf{W}_l \mathbf{x}_{l-1} + \mathbf{b}_l)$, where l is the layer index and $\mathbf{W}_l$ and $\mathbf{b}_l$ are a weight matrix and offset vector of suitable dimensions. Typical choices for $\beta(\cdot)$ include tanh, rectified linear units or sigmoid functions [6]. To produce latent samples $\mathbf{z}$ we sample from the normal distribution (see Equation (2)) using re-parametrization [16]

$$\mathbf{z} = \mu_\phi(\mathbf{x}) + \sigma_\phi(\mathbf{x})\epsilon, \tag{4}$$

where $\epsilon \sim \mathcal{N}(0, 1)$, to allow for back-propagation of gradients. For $p_\theta(\mathbf{x}|\mathbf{z})$ (see (1)) any suitable likelihood function can be used. We used a Gaussian distribution for all presented examples. Note that the likelihood function is parametrized by the entire (non-linear) decoder network.

The training of variational auto-encoders can be challenging as stochastic optimization was found to set $q_\phi(\mathbf{z}|\mathbf{x}) = p(\mathbf{z})$ in all but vanishingly rare cases [3], which corresponds to a local maximum that does not use any information from $\mathbf{x}$. We therefore add a warm-up phase that gradually gives the regularization term in the target function more weight:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - \alpha \,\text{KL}\left[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right], \tag{5}$$

where $\alpha \in [0, 1]$ is linearly increased with the number of epochs. The warm-up phase has been successfully used for training deep variational auto-encoders [25]. Furthermore, we initialize the weights of the dense layer computing $\log(\sigma^2_\phi(\mathbf{x}))$ to 0 (yielding a variance of 1 at the beginning of the training). This was motivated by our observations that if we employ standard random weight initialization techniques (glorot-norm, he-norm [9]) we can get relatively high initial estimates for the variance $\sigma^2_\phi(\mathbf{x})$, which due to the sampling leads to very unreliable samples $\mathbf{z}$ in the latent space. The large variance in sampled points in the latent space leads to bad convergence properties of the network.

CNN student auto-encoder (CNN-SAE). Following the recent findings in computer vision we present a second, more advanced network that typically outperforms simpler VAE. In [29], for example, these asymmetric auto-encoders resulted in superior reconstruction of images as well as more meaningful feature embeddings. A specific kind of convolutional neural network was combined with an auto-encoder, being able to directly capture low level pixel statistics and hence to extract more high-level feature embeddings.

Inspired by this previous work, we combine an asymmetric auto-encoder (and a decoder that is able to capture low level statistics) with the advantages of variational auto-encoders. Figure 1, right, shows our combined network. We employ multiple layers of one-dimensional convolutions to parametrize the encoder $q_\phi(\mathbf{z}|\mathbf{x})$ (again we assume a Gaussian distribution, see (2)). The distribution is parametrized as follows:

$$\begin{aligned} \mu_\phi(\mathbf{x}) &= \mathbf{W}_\mu \mathbf{h} + \mathbf{b}_\mu \\ \log(\sigma^2_\phi(\mathbf{x})) &= \mathbf{W}_\sigma \mathbf{h} + \mathbf{b}_\sigma \\ \mathbf{h} &= \text{conv}_l(\mathbf{x}) = \beta(\mathbf{W}_l * \text{conv}_{l-1}(\mathbf{x})), \end{aligned}$$

where $*$ is the convolution operator, $\mathbf{W}_l, \mathbf{W}_\mu, \mathbf{W}_\sigma$ are weights of suitable dimensions, $\beta(\cdot)$ is a non-linear activation function and l denotes the layer depth. Further, $\text{conv}_0(\mathbf{x}) = \mathbf{x}$. We keep the standard variational layer (see (4)) while changing the output layer to a recurrent layer using long term short term units (LSTM). Recurrent layers have successfully been used in auto-encoders before, e.g. in [5]. LSTM were very successful for modeling temporal sequences because they can model long and short term dependencies between time steps. Every LSTM unit receives a copy of the sampled points in latent-space, which allows the LSTM network to combine context information (point in the latent

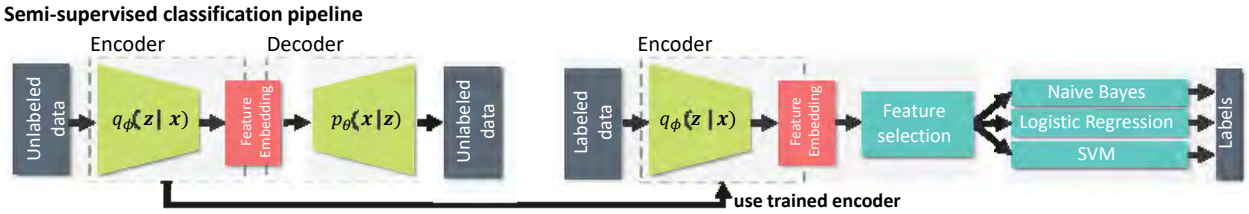


Figure 2: Pipeline overview. We train the variational auto-encoder on a large unlabeled data set. The trained encoder of the auto-encoder can be used to transform other data sets into an expressive feature embedding. Based on this feature embedding we train different classifiers to predict the student labels.

space) with the sequence information (memory unit in the LSTM cell). Using LSTM cells the decoder $p_\theta(\mathbf{x}|\mathbf{z})$ assumes a Gaussian distribution and is parametrized as follows:

$$\mu_{\theta t}(\mathbf{z}) = \mathbf{W}_{\mu z} \cdot \text{lstm}_t(\mathbf{z}) + \mathbf{b}_{\mu z}$$
$$\log(\sigma^2_{\theta t}(\mathbf{z})) = \mathbf{W}_{\sigma z} \cdot \text{lstm}_t(\mathbf{z}) + \mathbf{b}_{\sigma z},$$

where $\mu_{\theta t}(\mathbf{z})$ and $\sigma^2_{\theta t}(\mathbf{z})$ are the t^{th} components of $\mu_\theta(\mathbf{z})$ and $\sigma^2_\theta(\mathbf{z})$, respectively, $\text{lstm}_t(\cdot)$ denotes the t^{th} LSTM cell and $\mathbf{W}_*$ and $\mathbf{b}_*$ denote suitable weight and offset parameters.

Feature selection. VAE provide a natural way for performing feature selection. The inference network $q_\phi(\mathbf{z}|\mathbf{x})$ infers the mean and variance for every dimension z_i. Therefore, the most informative dimension z_i has the highest KL divergence from the prior distribution $p(z_i) = \mathcal{N}(0, 1)$ while uninformative dimensions will have a KL divergence close to 0 [10]. The KL divergence of z_i to $p(z_i)$ is given by

$$KL\left[q_\phi(z_i|\mathbf{x})||p(z_i)\right] = -\log(\sigma_i) + \frac{\sigma_i^2 \mu_i^2}{2} - \frac{1}{2}, \qquad (6)$$

where μ_i and σ_i are the inferred parameter for the Gaussian distribution $q_\phi(z_i|\mathbf{x})$. Feature selection proceeds by keeping the K dimensions z_i with the largest KL divergence.

Semi-supervised classification pipeline. The encoder and the decoder of the variational auto-encoder can be used independently of each other. This independence allows us to take the trained encoder and map new data to the learnt feature embedding. Figure 2 provides an overview of the entire pipeline for semi-supervised classification. In a first unsupervised step we train a VAE on unlabeled data. The learnt encoder $q_\phi(\mathbf{z}|\mathbf{x})$ is then used to transform labeled data sets to the feature embedding. We finally apply our feature selection step that considers the relative importance of the latent dimensions as previously described. We then train standard classifiers (Logistic Regression, Naive Bayes and Support Vector Machine) on the feature embeddings.

4. RESULTS

We evaluated our approach for the specific example of detecting developmental dyscalculia (DD), which is a learning disability affecting the acquisition of arithmetic skills [33]. Based on the learnt feature embedding on a large unlabeled data set the classifier performance was measured on two independent, small and labeled data sets from controlled user studies. We refer to them as *balanced* and *imbalanced* data sets since their distribution of DD and non-DD children differs: the first study has approximately 50% DD, while the second one includes 5% DD (typical prevalence of DD).

4.1 Experimental Setup

All three data sets were collected from *Calcularis*, which is an intelligent tutoring system (ITS) targeted at elementary school children suffering from DD or exhibiting difficulties in learning mathematics [13]. *Calcularis* consists of different games for training number representations and calculation. Previous work identified a set of games that are predictive of DD within *Calcularis* [17]. Since timing features were found to be one of the most relevant indicators for detecting DD [4] and to facilitate comparison to other feature embedding techniques we limited our analysis to log-normalized timing features, for which we can assume normal distribution [30]. Therefore, we evaluated our pipeline on the subset of games from [17] for which meaningful timing features could be extracted and sufficient samples were available in all data sets (we used >7000 samples for training the VAEs). Since our pipeline currently does not handle missing data only students with complete data were included.

Timing features were extracted for the first 5 tasks in 5 different games. The selected games involve addition tasks (adding a 2-digit number to a 1-digit number with ten-crossing; adding two 2-digit numbers with ten-crossing), number conversion (spoken to written numbers in the ranges 0-10 and 0-100) and subtraction tasks (subtracting a 1-digit number from a 2-digit number with ten-crossing). For every task we extracted the total answer time (time between the task prompt until the answer was entered) and the response time (time between the task prompt and the first input by the student). Hence, each student is represented by a 50-dimensional snapshot $\mathbf{x}$ (see Section 3).

Unlabeled data set. The unlabeled data set was extracted using live interaction logs from the ITS *Calcularis*. In total, we collected data from 7229 children. Note that we have no additional information about the children such as DD or grade. We excluded all teacher accounts as well as log files that were < 20KB. Since every new game in *Calcularis* is introduced by a short video during the very first task, we excluded this particular task for all games.

Balanced data set. The first labeled data set is based on log files from 83 participants of a multi-center user study

conducted in Germany and Switzerland, where approximately half of the participants were diagnosed with DD (47 DD, 36 control) [31]. During the study, children trained with *Calcularis* at home for five times per week during six weeks and solved on average 1551 tasks. There were 28 participants in 2^{nd} grade (9 DD, 19 control), 40 children in 3^{rd} grade (23 DD, 17 control), 12 children in 4^{th} grade (12 DD) and 3 children in 5^{th} grade (3 DD). The diagnosis of DD was based on standardized neuropsychological tests [31].

Imbalanced data set. The second labeled data set is from a user study conducted in the classroom of ten Swiss elementary school classes. In total, 155 children participated, and a prevalence of DD of 5% could be detected (8 DD, 147 control). There were 97 children in 2^{nd} grade (3 DD, 94 control) and 58 children in 3^{rd} grade (5 DD, 53 control). The DD diagnosis was computed based on standardized tests assessing the mathematical abilities of the children [32, 7]. During the study, children solved 85 tasks directly in the classroom. On average, children needed 26 minutes to complete the tasks.

Implementation. The unlabeled data set was used to train the unsupervised VAE for extracting compact feature embeddings of the data. Based on the learnt data transformations we evaluated two standard classifiers: Logistic Regression (LR) and Naive Bayes (NB). We restricted our evaluation to simple classification models because we wanted to assess the quality of the feature embedding and not the quality of the classifier. More advanced classifiers typically perform a (sometimes implicit) feature transformation as part of their data fitting procedure. To represent at least one model that performs such an embedding we included Support Vector Machine (SVM) in all our results. All classifier parameters were chosen according to the default values in *scikit-learn*. Note that we have additionally performed randomized cross-validated hyper-parameter search for all classifiers, which, however, resulted in marginal improvements only. Because of that, and to keep the model simple and especially easily reproducible, we use the default parameter set in this work. For Logistic Regression we used L2 regularization with $C = 1$, for Naive Bayes we used Gaussian distributions and for the SVM RBF kernels and data point weights have been set inversely proportional to label frequencies. All results are cross-validated using 30 randomized training-test splits on the unlabeled data (test size 5%). The classification part of the pipeline is additionally cross-validated using 300 label-stratified random training-test splits (test size 20%) to ensure highly reproducible classification results.

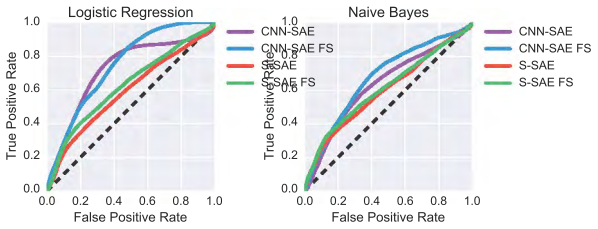
Network hyper-parameters were defined using the approach described in [1]. We increased the number of nodes per layer, the number of layers and the number of epochs until a good fit of the data was achieved. We then regularized the network using dropout [26] with increasing dropout rate until the network was no longer overfitting the data. Activation and weight initialization have been chosen according to common standards: We employ the most common activation function, namely rectified linear activation units (RELU) [20], for all activations. Weight initialization was performed using the method by He et al. [9]. Following this procedure, the following parameters were used for the S-SAE model: encoder and decoders used 3 layers of size 320. The CNN-SAE model was parametrized as follows: 3 convolution layers with 64 convolution kernels and a filter length of 3. We used a single layer of LSTM cells with 80 nodes. We used a batch size of 500 samples and batch normalization and dropout ($r = 0.25$) at every layer. The warm-up phase (see Section 3) was set to 300 epochs. Training was stopped after 1000 (S-SAE) and 500 (CNN-SAE) epochs. The number of latent units was set to 8 in accordance to previous work on detecting students with DD that used 17 features but found that about half of the features were sufficient to detect DD with high accuracy [17]. When feature selection was applied we set the number of features to $K = 4$ and thus we kept exactly half of the latent space features. All networks were implemented using the Keras framework with TensorFlow™ and optimized using Adam stochastic optimization with standard parameters according to [14].

4.2 Performance comparison

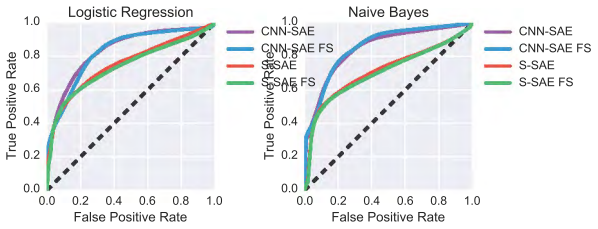
Our VAE models are trained to extract efficient feature embeddings of the data. To assess the quality of these computed feature representations, we compare the classification performance of our method to previous techniques for finding efficient feature embeddings, as well as to feature sets optimized specifically for the task of predicting DD.

Network comparison. In a first experiment we compared the feature embeddings generated by our simple S-SAE and our asymmetric CNN-SAE with and without feature selection. Figure 3 illustrates the average ROC curves of our complete semi-supervised classification pipeline. Our feature embeddings based on asymmetric CNN-SAE clearly outperform the ones from the simple S-SAE on both the imbalanced and the balanced data set for Naive Bayes (NB) and Logistic Regression (LR). For both models, feature selection improves the area under the ROC curve (AUC) for the imbalanced data set (CNN-SAE: LR 4.2%, NB 6.3%; S-SAE: LR 6.8%, NB: 1.6%), but has no effect for the balanced data set. We believe that this is due to the ability of the classifiers to distinguish useful features from noisy ones given enough samples. Since the performance of the classifiers with feature selection (FS) is better or equal to no feature selection in each experiment, we used the CNN-SAE FS model for all further evaluations.

Classification performance. In Figure 4 we compare the classifier performance for different feature embeddings. We compare our method based on VAE to two well-known methods for finding optimal feature embeddings, namely principle component analysis (PCA, green) and Kernel PCA (KPCA, red) [24]. For comparison and as a baseline for the performance of the different methods, we include direct classification results (gray), for which no feature embedding was computed. We used $K = 8$ (dimensionality of feature embedding) for all methods. The features extracted by our pipeline compare favorably to PCA and Kernel PCA showing improvements in terms of AUC of 28% for Logistic Regression and 23% for Naive Bayes on the imbalanced data set and an improvement of 3.75% for Logistic Regression and 7.5% for Naive Bayes on the balanced data set. By using simple classifiers, we demonstrated that our encoder learns an effective feature embedding. More sophisticated classifiers (such as SVM with non-linear kernels) typically proceed by first embedding the input into a specific feature space that is different from the original space.



(a) Imbalanced data set



(b) Balanced data set

Figure 3: ROC curves for the two proposed models with and without feature selection (FS). Our asymmetric CNN-SAE outperforms the simple S-SAE consistently with (blue) and without (purple) feature selection. Feature selection improves performance only on the imbalanced data set.

For the imbalanced data set the overall performance for SVM is significantly lower for all embeddings. This is in line with previous work [12] showing that for imbalanced data sets, the decision boundaries of SVMs are heavily skewed towards the minority class resulting in a preference for the majority class and thus a high miss-classification rate for the minority class. Indeed, we found that SVM predicted only majority labels on the imbalanced data set. For the balanced data set our feature embedding shows improvements of 2.5% over alternative embeddings when using SVM.

Further, Table 1 shows the performance of all feature embeddings using three additional common classification metrics: root mean squared error (RMSE), classification accuracy (Acc.) and area under the precision recall curve (AUPR). We statistically compared the classification metrics of our feature embedding to the best alternative feature embedding using an independent t-test and Bonferroni correction for multiple tests ($\alpha = 0.05$). Our feature embedding significantly outperformed alternative embeddings for all classifiers on both the balanced and imbalanced data sets on most metrics. The main exception was the performance of SVM on the imbalanced data set, which exhibited large variance for all feature embeddings and the worst overall classification performance (compared to the other classifiers).

When comparing classification performance on the imbalanced and the balanced data sets we observed that our pipeline using VAEs showed significant performance improvements compared to other methods for finding feature embeddings. While the unlabeled and the balanced data sets stem from an adaptive version of *Calcularis* the imbalanced data was collected using a fixed task sequence. As our method shows larger improvements on the imbalanced data, we believe CNN-SAE learned an embedding that is robust beyond adaptive ITS. The relative improvements of our feature embeddings is smallest for SVM on the balanced data set. We believe that this is due to ability of the SVM to learn complex decision boundaries given sufficient data. However, the ability for complex decision boundaries renders SVMs more vulnerable to class imbalance, yielding performance at random level on the imbalanced data set.

Comparison to specialized models. Recently, a specialized Naive Bayes classifier (S-NB) for the detection of developmental dyscalculia (DD) was introduced presenting a set of features optimized for the detection of DD [17]. The development of S-NB including the set of features was based on the balanced data set used in this work. In comparison to S-NB, our approach relies on timing data only and the extracted features are independent of the classification task. We compared the performance of S-NB to our CNN-SAE model on both data sets. For the balanced data set we found an AUC of 0.94 for the specialized model (S-NB) compared to an AUC of 0.86 for Naive Bayes using our feature embedding. On the imbalanced data set we found an AUC of 0.67 for S-NB compared to an AUC of 0.77 using Logistic Regression with our feature embedding. These findings demonstrate that while our feature embedding performs slightly worse on the balanced data set (for which the S-NB was developed), we significantly outperform S-NB by 15% on the imbalanced data set, which suggests that our VAE model automatically extracts feature embeddings that are more robust than expert features.

Robustness on sample size. Ideally, a classifier's performance should gracefully decrease as fewer data is provided. A good feature embedding allows a classifier to generalize well based on few labeled examples because similar samples are clustered together in the feature embedding. We therefore investigated the robustness of the different feature representations with respect to the training set size. For this we used the balanced data set where we varied the training set size between 7 (10% of the data) and 62 (90% of the data) by random label-stratified sub-sampling. Figure 5 compares the AUC of the different feature embeddings over different sizes of the training set. In case of Naive Bayes and Logistic Regression our embedding provides superior performance for all training set sizes. For large enough data sets SVM using the raw feature data (Direct, grey) is performing as well as using our embedding (CNN-SAE, blue). However, for smaller data sets starting at 30 samples the performance of SVM based on the raw features declines more rapidly compared to the SVM based on our feature embedding.

5. CONCLUSION

We adapted the recently developed variational auto-encoders to educational data for the task of semi-supervised classification of student characteristics. We presented a complete pipeline for semi-supervised classification that can be used with any standard classifier. We demonstrated that extracted structures from large scale unlabeled data sets can significantly improve classification performance for different labeled data sets. Our findings show that the improvements are especially pronounced for small or imbalanced data sets. Imbalanced data sets typically arise in EDM when detecting relatively rare conditions such as learning disabilities. Im-

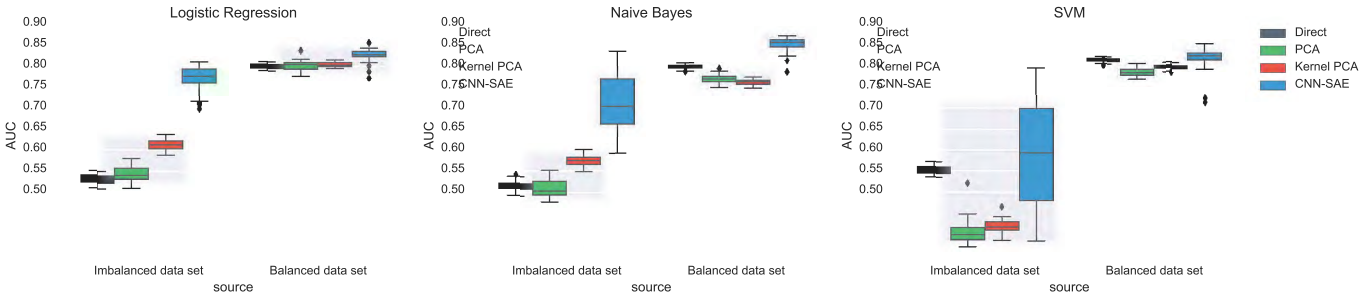


Figure 4: Classification performance for different feature embeddings. Our variational auto-encoder (blue) outperforms other embeddings by up to 28% (imbalanced data set) and by up to 7.5% (balanced data set).

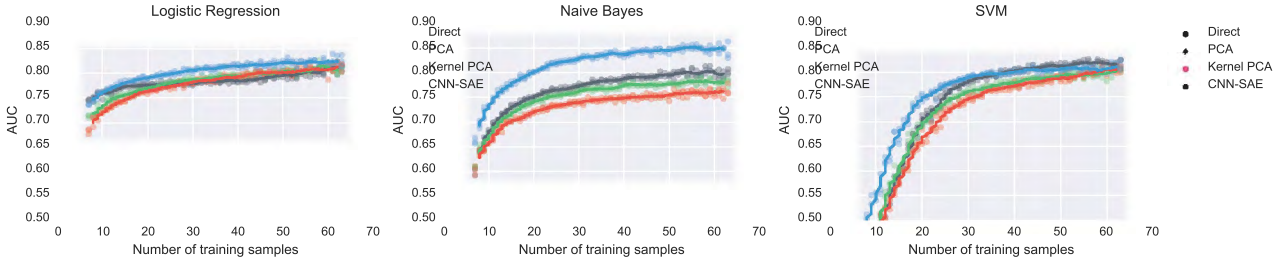


Figure 5: Comparison of classifier performance on the balanced data for different training set sizes (moving average fitted to data points). The features automatically extracted by our variational auto-encoder (blue) maintain a performance advantage even if the training size shrinks to 7 samples (10% of the original size).

Table 1: Comparison of our method to alternative embeddings. Our approach using a variational auto-encoder (CNN-SAE) significantly outperforms other approaches for most cases. The best score for each metric and classifier is shown in bold. *= statistically significant difference (t-test with Bonferroni correction, $\alpha = 0.05$).

	Direct				**PCA**				**Kernel PCA**				**CNN-SAE**			
	AUC	RMSE	AUPR	Acc.	AUC	RMSE	AUPR	Acc.	AUC	RMSE	AUPR	Acc.	AUC	RMSE	AUPR	Acc.
Imbalanced data set																
Logistic Regression	0.53	0.27	0.18	0.91	0.54	0.25	0.17	0.93	0.61	0.25	0.16	0.93	**0.78***	**0.24***	**0.28***	**0.94***
Naive Bayes	0.51	0.29	0.23	0.91	0.50	0.29	0.10	0.90	0.57	0.28	0.20	0.91	**0.70***	**0.25***	**0.24**	**0.93***
SVM	0.55	0.25	**0.22***	0.94	0.40	0.25	0.08	0.94	0.42	0.25	0.09	0.93	**0.59**	0.25	0.16	0.94
Balanced data set																
Logistic Regression	0.80	0.44	0.82	0.73	0.80	0.42	0.84	0.73	0.80	0.42	0.83	0.75	**0.83***	**0.40***	0.84	**0.77**
Naive Bayes	0.80	0.49	0.80	0.73	0.77	0.46	0.77	0.71	0.76	0.46	0.76	0.70	**0.86***	**0.38***	**0.86***	**0.80***
SVM	0.81	0.42	**0.84***	0.75	0.79	0.43	0.81	0.73	0.80	0.43	0.83	0.73	**0.83**	**0.40***	0.81	**0.79***

proved classification results with simple classifiers such as Logistic Regression might indicate that VAEs learn feature embeddings that are interpretable by human experts. In the future we want to explore the learnt representations and compare it to traditional categorizations of students (skills, performance, etc.). Additionally, we want to extend our results to include additional feature types and data reliability indicators to handle missing data. Although we trained our networks on comparatively small sample sizes, the presented method scales (due to mini-batch learning) to much larger data sets (>100K users) allowing the training of more complex VAE. Moreover, the generative model $p_\theta(\mathbf{x}|\mathbf{z})$ that is part of any VAE can be used to produce realistic data samples [29]. Up-sampling of the minority class provides a potential way to improve the decision boundaries for classifiers. In contrast to common up-sampling methods such as ADASYN [8], VAE-based sampling does not require nearest neighbor computations which makes them better applicable to small data sets. Preliminary results for random subsets of the balanced data set showed improvements in AUC by up-sampling based on VAE of 2-3% compared to ADASYN. While we applied our method to the specific case of detecting developmental dyscalculia, the presented pipeline is generic and thus can be applied to any educational data set and used for the detection of any student characteristic.

Acknowledgments. This work was supported by ETH Research Grant ETH-23 13-2.

6. REFERENCES

[1] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. 2012.

[2] Y. Bengio et al. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2009.

[3] S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. Generating Sentences from a Continuous Space. In *Proc. CONLL*, pages 10–21, 2016.

[4] B. Butterworth. *Dyscalculia screener*. Nelson Publishing Company Ltd., 2003.

[5] O. Fabius and J. R. van Amersfoort. Variational recurrent auto-encoders. In *Proc. ICLR*, 2015.

[6] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[7] J. Haffner, K. Baro, P. Parzer, and F. Resch. Heidelberger Rechentest: Erfassung mathematischer Basiskomptenzen im Grundschulalter, 2005.

[8] H. He, Y. Bai, E. A. Garcia, and S. Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Proc. IJCNN*, pages 1322–1328, 2008.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proc. ICCV*, pages 1026–1034, 2015.

[10] I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint arXiv:1606.05579*, 2016.

[11] H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, pages 863–874, 2007.

[12] T. Imam, K. Ting, and J. Kamruzzaman. z-svm: an svm for improved classification of imbalanced data. *AI 2006: Advances in Artificial Intelligence*, pages 264–273, 2006.

[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*, 2013.

[14] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *Proc. ICLR*, 2015.

[15] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Proc. NIPS*, pages 3581–3589, 2014.

[16] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *Proc. ICLR*, 2014.

[17] S. Klingler, T. Käser, A. Busetto, B. Solenthaler, J. Kohn, M. von Aster, and M. Gross. Stealth Assessment in ITS - A Study for Developmental Dyscalculia. In *Proc. ITS*, pages 79–89, 2016.

[18] G. Kostopoulos, S. B. Kotsiantis, and P. B. Pintelas. Predicting Student Performance in Distance Higher Education Using Semi-supervised Techniques. In *Proc. MEDI*, pages 259–270, 2015.

[19] I. Labutov and H. Lipson. Web as a textbook: Curating Targeted Learning Paths through the Heterogeneous Learning Resources on the Web. In *Proc. EDM*, pages 110–118, 2016.

[20] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, pages 436–444, 2015.

[21] H. Liao, E. McDermott, and A. Senior. Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription. In *Proc. ASRU*, pages 368–373, 2013.

[22] W. Min, B. W. Mott, J. P. Rowe, and J. C. Lester. Leveraging semi-supervised learning to predict student problem-solving performance in narrative-centered learning environments. In *Proc. ITS*, pages 664–665, 2014.

[23] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proc. ICML*, pages 1278–1286, 2014.

[24] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In *Proc. ICANN*, pages 583–588, 1997.

[25] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Proc. NIPS*, pages 3738–3746, 2016.

[26] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, pages 1929–1958, 2014.

[27] V. Tam, E. Y. Lam, S. Fung, W. Fok, and A. H. Yuen. Enhancing educational data mining techniques on online educational resources with a semi-supervised learning approach. In *Proc. TALE*, pages 203–206, 2015.

[28] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proc. ACL*, pages 384–394, 2010.

[29] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional Image Generation with PixelCNN Decoders. In *Proc. NIPS*, pages 4790–4798, 2016.

[30] W. J. van der Linden. A lognormal model for response times on test items. *Journal of Educational and Behavioral Statistics*, 31(2):181–204, 2006.

[31] M. Von Aster, L. Rauscher, K. Kucian, T. Käser, U. McCaskey, and J. Kohn. Calcularis - Evaluation of a computer-based learning program for enhancing numerical cognition for children with developmental dyscalculia, 2015. 62nd Annual Meeting of the American Academy of Child and Adolescent Psychiatry.

[32] M. von Aster, M. W. Zulauf, and R. Horn. *Neuropsychologische Testbatterie für Zahlenverarbeitung und Rechnen bei Kindern: ZAREKI-R*. Pearson, 2006.

[33] M. G. Von Aster and R. S. Shalev. Number development and developmental dyscalculia. *Developmental Medicine & Child Neurology*, pages 868–873, 2007.

[34] C. Xu, D. Tao, and C. Xu. A survey on multi-view learning. *Neural Comput. Appl.*, pages 2031–2038, 2013.

[35] X. Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison, 2006.

Predicting Short- and Long-Term Vocabulary Learning via Semantic Features of Partial Word Knowledge

SungJin Nam
School of Information
University of Michigan
Ann Arbor, MI 48109
sjnam@umich.edu

Gwen Frishkoff
Department of Psychology
University of Oregon
Eugene, OR 97403
gfrishkoff@gmail.com

Kevyn Collins-Thompson
School of Information
University of Michigan
Ann Arbor, MI 48109
kevynct@umich.edu

ABSTRACT

We show how the novel use of a semantic representation based on Osgood's semantic differential scales can lead to effective features in predicting short- and long-term learning in students using a vocabulary learning system. Previous studies in students' intermediate knowledge states during vocabulary acquisition did not provide much information on which semantic knowledge students gained during word learning practice. Moreover, these studies relied on human ratings to evaluate the students' responses. To solve this problem, we propose a semantic representation for words based on Osgood's semantic decomposition of vocabulary [16]. To demonstrate our method can effectively represent students' knowledge in vocabulary acquisition, we build models for predicting the student's short-term vocabulary acquisition and long-term retention. We compare the effectiveness of our Osgood-based semantic representation to that provided by Word2Vec neural word embedding [13], and find that prediction models using features based on Osgood scale-based scores (OSG) perform better than the baseline and are comparable in accuracy to those using Word2Vec score-based models (W2V). By using more interpretable Osgood-based scales, our study results can help with better understanding of students' ongoing learning states and designing personalized learning systems that can address an individual's weak points in vocabulary acquisition.

Keywords

Vocabulary learning, semantic similarity, prediction model, intelligent tutoring system

1. INTRODUCTION

Studies of word learning have shown that knowledge of individual words is typically not all-or-nothing. Rather, people acquire varying degrees of knowledge of many words incrementally over time, by exposure to them in context [9]. This is especially true for so-called "academic" words that are less common and more abstract — e.g., *pontificate*, *probity*, or *assiduous* [7]. Binary representations and measures model word knowledge simply as correct or incorrect on a particular item (word), but in reality, a student's knowledge level may reside between these two extremes. Thus, previous studies of vocabulary acquisition have suggested that students' partial knowledge be modeled using a representation that adding an additional label corresponding to an intermediate knowledge state [6] or further, in terms of continuous metrics for semantic similarity [3].

In addition, there are multiple dimensions to a word's meaning [16]. Measuring a student's partial knowledge on a single scale may only provide abstract information about the student's general answer quality and not give enough information to specify *which* dimensions of word knowledge a student already has learned or needs to improve. In order to achieve detailed understanding of a student's learning state, online learning systems should be able to capture a student's "learning trajectory" that tracks their partial knowledge on a particular item over time, over multiple dimensions of meaning in a multidimensional semantic representation.

Hence, multidimensional representations of word knowledge can be an important element for building an effective intelligent tutoring system (ITS) for reading and language. Maintaining a fine-grained semantic representation of a student's degree of word knowledge can be helpful for the ITS to design more engaging instructional content, more helpful personalized feedback, and more sensitive assessments [17, 19]. Selecting semantic representations to model, understand, and predict learning outcomes is important to designing a more effective and efficient ITS.

In this paper, we explore the use of multidimensional semantic word representations for modeling and predicting short- and long-term learning outcomes in a vocabulary tutoring system. Our approach derives predictive features using a novel application of existing methods in cognitive psychology combined with methods from natural language processing (NLP). First, we introduce a new multidimensional representation of a word based on the Osgood semantic differential [16], an empirically based, cognitive framework that uses a small number of scales to represent latent components of word meaning. We compare the effectiveness of model features based on this Osgood-based representation to features based on a different representation, the widely-used Word2Vec word embedding [13]. Second, we evaluate our prediction models using data from a meaning-generation task that was conducted during a computer-based intervention. Our study results demonstrate how similarity-based metrics based on rich

semantic representation can be used to automatically evaluate specific components of word knowledge, track changes in the student's knowledge toward the correct meaning, and compute a rich set of features for use in predicting short- and long-term learning outcomes. Our methods could support advances in real-time, adaptive support for word semantic learning, resulting in more effective personalized learning systems.

2. RELATED WORK

The present study is informed by three areas of research: (1) studies of partial word knowledge; (2) the Osgood framework for multiple dimensions of word meaning, and (3) computational methods for estimating semantic similarity.

Partial Word Knowledge. The concept of partial word knowledge has interested vocabulary researchers for several decades, particularly in the learning and instruction of "Tier 2" words [20]. Tier 2 words are low-frequency and typically have complex (multiple, nuanced) meanings. By nature, they are rarely learned through "one-shot" learning or direct definition. Instead, they are learned partially and gaps are filled in over time.

Words in this intermediate state, neither novel nor fully known, are sometimes called "frontier words" [5]. Durso and Shore operationalized the frontier word as a word the student had seen previously but was not actively using it [6]. Based on this definition, the student may have had implicit memory of frontier words, such as general information like whether the word indicates a good or bad situation or refers a person or an action. They discovered that students are more familiar with frontier words than other types of words in terms of their sounds and orthographic characteristics [6]. This previous work suggested that the concept of frontier words can be used to represent a student's partial knowledge states in a vocabulary acquisition task [5, 6].

In some studies, partial word knowledge has been represented using simple, categorical labels, e.g., multiple-choice tests that include "partially correct" response options, as well as a single "best" (correct) response. In other studies, the student is presented with a word and is asked to say what it means [1]. The definition is given partial credit if it reflects knowledge that is partial or incomplete. For example, a student may recognize that the word *probity* has a positive connotation, even if she cannot give a complete definition. However, single categorical or score-based indicators may not explain which specific aspects of vocabulary knowledge the student is missing. Moreover, these studies relied on human ratings to evaluate students' responses for unknown words [6]. Although widely used in psychometric and psycholinguistic studies [4, 16], hiring human raters is expensive and may not be done in real time during students' interaction with the tutoring system.

To address these problems, we propose a data-driven method that can automatically extract semantic characteristics of a word based on a set of relatively simple, interpretable scales. The method benefits from existing findings in cognitive psychology and natural language processing. In the following sections, we illustrate more details of related findings and how they can be used in an intelligent tutoring system setting.

Semantic Representation & the Osgood Framework. To quantify the semantic characteristics of a student's intermediate knowledge of vocabulary, this paper uses a "spatial analogue" for capturing semantic characteristics of words. In [16], Osgood investigated how the meaning of a word can be represented by a series of general semantic scales. By using these scales, Osgood suggested that the meanings of any word can be projected and explored in a continuous semantic space.

Osgood asked human raters to evaluate a set of words using a large number of scales (e.g., tall-short, fat-thin, heavy-light) and captured the semantic representation of a word [16]. Respondents gave Likert ratings, which indicated whether they thought that a word meaning was closer to one extreme (-3) or the other (+3), or basically irrelevant (0). A principal components analysis (PCA) was used to represent the latent semantic features that can explain the patterns of response to individual words within this task.

In our study, we suggest a method that can automatically extract similar semantic information that can project a word into a multidimensional semantic space. By using semantic scales selected from [16], we verify if such representation of semantic attributes of words is useful for predicting students' short- and long-term learning.

Semantic Similarity Measures. Studies in NLP have suggested methods to automatically evaluate the semantic association between two words. For example, Markov Estimation of Semantic Association (MESA) [3, 9] can estimate the similarity between words from a random walk model over a synonym network such as WordNet [14]. Other methods like latent semantic analysis (LSA) are based on co-occurrence of the word in a document corpus. In LSA, semantic similarity between words is determined by using a cosine similarity measure, derived from a sparse matrix constructed from unique words and paragraphs containing the words [10].

For this paper, we use Word2Vec [13], a widely used word embedding method, to calculate the semantic similarity between words. Word2Vec's technique [11] transforms the semantic context, such as proximity between words, into a numeric vector space. In this way, linguistic regularities and patterns are encoded into linear translations. For example, using outputs from Word2Vec, relationships between words can be estimated by simple operations on their corresponding vectors, e.g., *Madrid - Spain + France = Paris*, or *Germany + capital = Berlin* [13].

Measures from these computational semantic similarity tools are powerful because they can provide an automated method for evaluation of partial word knowledge. However, they typically produce a single measure (e.g., cosine similarity or Euclidean distance), representing semantic similarity as a one-dimensional construct. With such a measure, it is not possible to determine represent partial semantic knowledge and changes in knowledge of latent semantic features as word knowledge progresses from unknown to frontier to fully known. In following sections, we describe how we address this problem, using novel methods to to estimate the contribution of Osgood semantic features to individual word meanings.

2.1 Overview of the Study

Based on findings from existing studies, this study will suggest an automatized method for evaluating students' partial knowledge of vocabulary that can be used to predict students' short-term vocabulary acquisition and long-term retention. To investigate this problem, we will answer the following research questions with this paper.

The first research question (RQ1): Can semantic similarity scores from Word2Vec be used to predict students' short-term learning and long-term retention? Previous studies in vocabulary tutoring systems tend to focus on how different experimental conditions, such as different spacing between question items [18], difficulty levels [17], and systematic feedback [7], affect students' short-term learning. This study will answer how computationally estimated trial-by-trial scores in a vocabulary tutoring system can be used to predict students' short-term learning and long-term retention.

RQ2: Compared to using regular Word2Vec scores, how does the model using Osgood's semantic scales [16] as features perform for immediate and delayed learning prediction tasks? As described in the previous section, the initial outcome from Word2Vec returns hundreds of semantic dimensions to represent the semantic characteristics of a word. Summary statistics for comparing such high-dimensional vectors, such as cosine similarity or Euclidean distance, only provide the overall similarity between words. If measures from Osgood scales work in a similar level to models using regular Word2Vec scores for predicting students' learning outcomes, we can argue that it can be an effective method for representing students' partial knowledge of vocabulary.

3. METHOD

3.1 Word Learning Study

This study used a vocabulary tutoring system called Dynamic Support of Contextual Vocabulary Acquisition for Reading (DSCoVAR) [8]). DSCoVAR aims to support efficient and effective learning vocabulary in context. All participants accessed DSCoVAR in a classroom-setting environment by using Chromebook devices or the school's computer lab in the presence of other students.

3.1.1 Study Participants

Participants included 280 middle school students (6th to 8th grade) from multiple schools, including children from diverse socio-economic and educational backgrounds. Table 1 provides a summary of student demographics, including location (P1 or P2), age and grade level, sex. Location P1 is a laboratory school affiliated with a large urban university in the northeastern United States. Students from location P1 were generally of high socio-economic status (e.g., children of University faculty and staff). Location P2 includes three public middle schools in a southern metropolitan area of the United States. All students from location P2 qualified for free or reduced lunch. The study included a broad range of students so that the results of this analysis were more likely to generalize to future samples.

3.1.2 Study Materials

DSCoVAR presented students with 60 SAT-level English words (also known as Tier 2 words). These "target words," lesser-known words that the students are going to learn,

Table 1: The number of participants by grade and gender

	6th grade		7th grade		8th grade	
Group	Girl	Boy	Girl	Boy	Girl	Boy
P1	16	28	19	23	18	13
P2	53	51	12	6	21	20

were balanced between different parts of speech, including 20 adjectives, 20 nouns, and 20 verbs. Based on previous works, we expected that students would have varying degrees of familiarity with the words at pre-test, but that most words would be either completely novel ("unknown") or somewhat familiar ("partially known") [8, 15]. This selection of materials ensured that there would be variability in word knowledge across students for each word and across words for each student.

In DSCoVAR, students learned how to infer the meaning of an unknown word in a sentence by using surrounding contextual information. Having more information in a sentence (i.e., a sentence with a high degree of contextual constraint) can decrease the uncertainty of inference. In this study, the degree of sentence constraint was determined using standard cloze testing methods: quantifying the diversity of responses from 30 human judges when the target word is left as a fill-in-the-blank question.

3.1.3 Study Protocol

The word learning study comprised four parts: (1) a pre-test, which was used to estimate baseline knowledge of words, (2) a training session, where learners were exposed to words in meaningful contexts, (3) an immediate post-test, and (4) a delayed post-test, which occurred approximately one week after training.

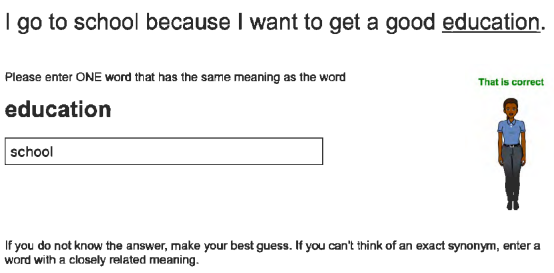
Pre-test. The pre-test session was designed to measure the students' prior knowledge of the target words. For each target word, students were asked to answer two types of questions: familiarity-rating questions and synonym selection questions. In familiarity rating questions, students provided their self-rated familiarity levels (unknown, known, and familiar) for presented target words. In synonym-selection questions, students were asked to select a synonym word for the given target word from five multiple choice options. The outcome from synonym-selection questions provided more objective measures for students' prior domain knowledge of target words.

Training. Approximately one week after the pre-test session, students participated in the training. During training, students learned strategies to infer the meaning of an unknown word in a sentence by using surrounding contextual information.

A training session consisted of two parts: an instruction video and practice questions. In the instruction video, students saw an animated movie clip about how to identify and use contextual information from the sentence to infer the meaning of an unknown word. In the practice question part, students could exercise the skill that they learned from the video. DSCoVAR provided sentences that included a target word with different levels of surrounding contextual information. The amount of contextual information for each sentence was determined by external crowd workers (details described in Section 3.1.2). In the practice question part, each target word was presented four times within

different sentences. Students were asked to type a synonym of the target word, which was presented in the sentence as underlined and bold. Over two weeks, students participated in two training sessions with a week's gap between them. Each training session contained the instruction video and practice questions for 30 target words. An immediate post-test session followed right after each training session.

Figure 1: An example of a training session question. In this example, the target word is "education" with a feedback message for a high-accuracy response.



Students were randomly selected to experience different instruction video conditions (full instruction video vs. restricted instruction video). Additionally, various difficulty level conditions and feedback conditions (e.g., DSCoVAR provides a feedback message to the student based on answer accuracy vs. no feedback) were tested within the same student. However, in this study, we focused on data from students who experienced a full instruction video and repeating difficulty conditions. Repeating difficulty conditions included questions with all high or medium contextual constraint levels. By doing so, we wanted to minimize the impact from various experimental conditions for analyzing post-test outcomes. Moreover, we filtered out response sequences that did not include all four responses for the target word. As a result, we analyzed 818 response sequences from 7,425 items in total.

Immediate and Delayed Post-test. The immediate post-test occurred right after the students finished the training; the delayed post-test was conducted one week later. Data collected during the immediate and delayed post-tests were used to estimate short-and long-term learning, respectively. Test items were identical to those in the pretest session, except for item order, which varied across tests. For analysis of the delayed post-test data, we only used the data from target words for which the student provided a correct answer in the earlier, immediate post-test session. As a result, 449 response sequences were analyzed for predicting the long-term retention.

3.2 Semantic Score-Based Features

We now describe the semantic features tested in our prediction models.

3.2.1 Semantic Scales

For this study, we used semantic scales from Osgood's study [16]. Ten scales were selected by a cognitive psychologist as being considered semantic attributes that can be detected during word learning (Figure 2). Each semantic scale consists of pairs of semantic attributes. For example, the *bad–good* scale can show how the meaning of a word can be projected on a scale with *bad* and *good* located at either

Figure 2: Ten semantic scales used for projecting target words and responses [16].

- bad – good
- complex – simple
- passive – active
- fast – slow
- powerful – helpless
- noisy – quiet
- big – small
- new – old
- helpful – harmful
- healthy – sick

end. The word's relationship with each semantic anchor can be automatically measured from its semantic similarity with these exemplar semantic elements.

3.2.2 Basic Semantic Distance Scores

To extract meaningful semantic information, we have applied the following measures that can be used to explain various characteristics of student responses for different target words. In this study, we used a pre-trained model for Word2Vec,[1] built based on the Google News corpus (100 billion tokens with 3 million unique vocabularies, using a negative sampling algorithm), to measure semantic similarity between words. The output of the pre-trained Word2Vec model contained a numeric vector with 300 hundred dimensions.

First, we calculated the relationship between word pairs (i.e., a single student response and the target word, or a pair of responses) in both the regular Word2Vec (W2V) score and the Osgood semantic scale (OSG) score. In the W2V score, the semantic relationship between words was represented with a cosine similarity between word vectors:

$$D_{w2v}(w_1, w_2) = 1 - |sim(V(w_1), V(w_2))|. \quad (1)$$

In this equation, the function V returned the vectorized representation of the word (w_1 or w_2) from the pre-trained Word2Vec model. By calculating the cosine similarity between two vectors (a cosine similarity function is noted as sim), we could extract a single numeric similarity score between two words. This score was converted into a distance-like score by taking the absolute value of the cosine similarity score and subtracting from one.

For the OSG score, we extracted two different types of scores: a non-normalized score and a normalized score. A non-normalized score showed how a word is similar to a single anchor word (e.g., *bad* or *good*) from the Osgood scale.

$$S^{non}_{osg}(w, a_{i,j}) = sim(V(w), V(a_{i,j})) \quad (2)$$

$$D^{non}_{osg}(w_1, w_2; a_{i,j}) = |S^{non}_{osg}(w_1, a_{i,j})| - |S^{non}_{osg}(w_2, a_{i,j})| \quad (3)$$

In equation 2, $a_{i,j}$ represents a single anchor word (j) in the i-th Osgood scale. The similarity between the anchor word and the evaluating word w was calculated with cosine similarity of Word2Vec outcomes for both words. In a non-normalized setting, the distance between two words given by a particular anchor word was calculated by the difference of absolute cosine similarity scores (equation 3).

The second type of OSG score is a normalized score. By using Word2Vec's ability to do arithmetical calculation of

[1]API and pre-trained model for Word2Vec was downloaded from this URL: https://github.com/3Top/word2vec-api

multiple word vectors, the normalized OSG score provided a relative location of the word from two anchor ends of the Osgood scale.

$$S_{osg}^{nrm}(w, a_i) = sim(V(w), V(a_{i,1}) - V(a_{i,2})) \quad (4)$$

$$D_{osg}^{nrm}(w_1, w_2; a_i) = |S_{osg}^{nrm}(w_1, a_i) - S_{osg}^{nrm}(w_2, a_i)| \quad (5)$$

In equation 4, the output represents the cosine similarity score between the word w and two anchor words ($a_{i,1}$ and $a_{i,2}$). For example, if the cosine similarity score of $S_{osg}^{nrm}(w, a_i)$ is close to -1, it means the word w is close to the first anchor word $a_{i,1}$. If the score is close to 1, it is vice versa. In equation 5, the distance between two words was calculated as the absolute value of the difference between two cosine similarity measures.

3.2.3 Deriving Predictive Features

Based on semantic distance equations explained in the previous section, this section explains examples of predictive features that we used to predict students' short-term learning and long-term retention.

Distance Between the Target Word and the Response. For regular Word2Vec score models and Osgood scale score models, distance measures between the target word and the response (by using equations 1 and 5) were used to estimate the accuracy of the response to a question. This feature represents the trial-by-trial answer accuracy of a student response. Each response sequence for the target word contained four distance scores.

Difference Between Responses. Another feature that we used in both types of models was the difference between responses. This feature could capture how a student's current answer is semantically different from the previous response. From each response sequence, we could extract three derivative scores from four responses.

Convex Hull Area of Responses. Alternative to the difference between responses feature, Osgood scale models were also tested with the area size of convex hull that can be generated by responses calculated with non-normalized Osgood scale scores (equation 3). For example, for each Osgood scale, a non-normalized score provided two-dimensional scores that can be used for geometric representation. By putting the target word in an origin position, a sequence of responses can create a polygon that can represent the semantic area that the student explored with responses. Since some response sequences were unable to generate the polygon by including less than three unique responses, we added a small, random noise that uniformly distributed (between -10^{-4} and 10^{-4}) to all response points. Additionally, a value of 10^{-20} was added to all convex hull area output to create a visible lower-bound value.

Unlike the measure of difference between responses, this feature also considers angles that can be created between responses and the target word. This representation can provide more information than just using difference between responses.

3.3 Modeling

To predict students' short-term learning and long-term retention, we used a mixed-effect logistic regression model (MLR). MLR is a general form of logistic regression model that includes random effect factors to capture variations from repeated measures.

3.3.1 Off-line Variables

Off-line variables capture item- or subject-level variances that can be observed repeatedly from the data. In this study, we used multiple off-line variables as random effect factors.

First, results from familiarity-rating and synonym-selection questions from the pre-test session were used to include item- and subject-level variances. Both variables include information on the student's prior domain knowledge level for target words. Second, the question difficulty condition was considered as an item group level factor. In the analysis, sentences for the target word that were presented to the student contained the same difficulty level, either high or medium contextual constraint levels, over four trials. Third, a different experiment group was used as a subject group factor. As described in Section 3.1.1, this study contains data from students in different institutions in separate geographic locations. The inclusion of these participant groups in the model can be used to explain different short-term learning outcomes and long-term retention by demographic groups.

3.3.2 Model Building

In this study, we compared the performance of MLR models with four different feature types. First, the baseline model was set to indicate the MLR model's performance without any fixed effect variables but only with random intercepts. Second, the response time model was built to be compared with semantic score-based models. Many previous studies have used response time as an important predictor of student engagement and learning [2, 12]. In this study, we used two types of response time variables, the latency for initiating the response and finishing typing the response, as predictive features. Both variables were measured in milliseconds over four trials and natural log transformed for the analysis. Third, semantic features from regular Word2Vec scores were used as predictors. This model was built to show how semantic scores from Word2Vec can be useful for predicting students' short- and long-term performance in DSCoVAR. Lastly, Osgood scale-based features were used as predictors. This model was compared with the regular Word2Vec score model to examine the effectiveness of using Osgood scales for evaluating students' performance in DSCoVAR. For these semantic-score based models, we tested out different types of predictive features that were described in Section 3.2.3. All models shared the same random intercept structure that treated each off-line variable as an individual random intercept.

For Osgood scale models, we also derived reduced-scale models. Reduced-scale models were compared with the full-scale model, which uses all ten Osgood scales. In this case, using fewer Osgood scales can provide easier interpretation of semantic analysis for intelligent tutoring system users.

3.3.3 Model Evaluation

To compare performance between different models, this study used various evaluation metrics, including AUC (an area under the curve score from a response operating characteristic (ROC) curve), F_1 (a harmonic mean of precision and recall), and error rate (a ratio of the number of

misclassified items over total items). 95% confidence interval of each evaluation metric was calculated from the outcome of a ten-fold cross-validation process repeated over ten times.

To select the semantic score-based features for models based on regular Word2Vec scores and Osgood scale scores, we used rankings from each evaluation metric. The model with the highest overall rank (i.e., sum the ranks from AUC, F_1, and error rate, and select the model with the lowest rank-sum value) was considered the best-performing model for the score type (i.e., models based on the regular Word2Vec score or Osgood scale score). More details on this process will be illustrated in the next section.

4. RESULTS

4.1 Selecting Models

In this section, we selected the best-performing model based on the models' overall ranks in each evaluation metric. All model parameters were trained in each fold of repeated cross-validation. We calculated 95% confidence intervals for comparison. To calculate the confidence interval of F_1 and error rate measures, the maximum (F_1) and minimum (error rate) scores of each fold were extracted. These maximum and minimum values were derived from applying multiple cutoff points to the mixed-effect regression model.

4.1.1 Predicting Immediate Learning

First, we built models that predict the students' immediate learning from the immediate post-test session. From models based on regular Word2Vec scores (W2V), the model with the distance between the target and responses and the difference between responses (*Dist+Resp*) provided the highest rank from various evaluation metrics (Table 2). From models based on Osgood scales (OSG), the model with the difference between responses (*Resp*) provided the highest rank.

The selected W2V model provided significantly better performance than the baseline model. The selected OSG model also showed significantly better performance than the baseline model, except for the AUC score. The selected W2V model was significantly better than the model using response time features in the AUC score and error rates.

The selected W2V model showed significantly better performance than the OSG model only with the AUC score. Figure 3 shows that the W2V model has a slightly larger area under the ROC curve than the OSG model. In the precision and recall curve, the selected W2V model provides more balanced trade-offs between precision and recall measures. The selected OSG model outperforms the W2V model in precision only in a very low recall measure range.

4.1.2 Predicting Long-Term Retention

We also built prediction models to predict the students' long-term retention in the delayed post-test session. In this analysis, a student response was included only when the student provided correct answers to the immediate post-test session questions. Among W2V score-based models, the best-performing model contained the same feature types as the immediate post-test results (Table 3). By using the distance between the target and responses and difference between responses (*Dist+Resp*), the model achieved significantly better performance than the baseline model, except for the AUC score.

For OSG models, the model with a convex hull area of responses (*Chull*) provided the highest overall rank from evaluation metrics (Table 3). The results were significantly better than the baseline model, and marginally better than the W2V model. Both selected W2V and OSG models were marginally better than the response time model, except the error rate of the OSG model was significantly better.

In Figure 3, the selected W2V model slightly outperforms the OSG model in mid-range true positive rates, while the OSG model performed slightly better in a higher true positive area. Precision and recall curves show similar patterns to those we observed from the immediate post-test prediction models. The OSG model only outperforms the W2V model in a very low recall value area.

4.1.3 Comparing Models

Compared to the selected W2V model in the immediate post-test condition, the selected W2V model in the delayed post-test retention condition showed a significantly lower AUC score, marginally higher F_1 score, and marginally higher error rate. In terms of OSG models, the selected OSG model for delayed post-test retention showed a significantly better F_1 score and error rates than the selected OSG model in the immediate post-test condition. Based on these results, we can argue that Osgood scale scores can be more useful for predicting student retention in the delayed post-test session than predicting the outcome from the immediate post-test.

In terms of selected feature types, the best-performing OSG models used features based on the difference between responses (*Resp*) or the convex hull area (*Chull*) that was created from the relative location of the responses. On the other hand, selected W2V models used both the distance between the target word and responses and difference between responses (*Dist+Resp*). When we compared both W2V and OSG models using the difference between responses feature, we found that performance is similar in the immediate post-test data. However, the OSG model was significantly better in the delayed post-test data. These results show that Osgood scale scores can be more useful for representing the relationship among response sequences.

4.2 Comparing the Osgood Scales

To identify which Osgood scales are more helpful than others for predicting students' performance, we conducted a scale-wise importance analysis. The results from this section reveal which Osgood scales are more important than others, and how the performance of prediction models with a reduced number of scales is comparable with the full-scale model.

4.2.1 Identifying More Important Osgood Scales

In this section, based on the selected Osgood score model from Section 4.1, we identified the level of contribution for features based on each Osgood scale. For example, the selected OSG model for predicting the immediate post-test data uses the difference between responses in ten Osgood scales as features. In order to diagnose the importance level of the first scale (*bad–good*), we can retrain the model with features based on the nine other scales and compare the

Table 2: Ranks of predictive feature sets for regular Word2Vec models (W2V) and Osgood score models (OSG) in the immediate post-test data. All models are significantly better than the baseline model. (Bold: the selected model with highest overall rank.)

	W2V models			OSG models		
Features	AUC	F_1	Err	AUC	F_1	Err
baseline	0.68 [0.67, 0.69] (5)	0.74 [0.73, 0.74] (5)	0.33 [0.33, 0.34] (5)	0.68 [0.67, 0.69] (5)	0.74 [0.73, 0.74] (5)	0.33 [0.33, 0.34] (7)
RT	0.69 [0.68, 0.70] (4)	0.75 [0.75, 0.76] (3)	0.31 [0.31, 0.32] (4)	0.69 [0.68, 0.70] (2)	0.75 [0.74, 0.76] (2)	0.31 [0.31, 0.32] (2)
Dist	0.72 [0.71, 0.74] (1)	0.76 [0.75, 0.76] (2)	0.29 [0.28, 0.30] (2)	0.67 [0.66, 0.68] (7)	0.73 [0.73, 0.74] (7)	0.33 [0.32, 0.34] (6)
Resp	0.70 [0.69, 0.71] (3)	0.75 [0.74, 0.76] (4)	0.31 [0.30, 0.32] (3)	**0.69 [0.68, 0.70] (1)**	**0.75 [0.75, 0.76] (1)**	**0.31 [0.30, 0.32] (1)**
Chull	NA	NA	NA	0.69 [0.68, 0.70] (3)	0.74 [0.73, 0.75] (4)	0.32 [0.31, 0.33] (4)
Dist+Resp	**0.72 [0.71, 0.73] (2)**	**0.76 [0.75, 0.77] (1)**	**0.29 [0.28, 0.30] (1)**	0.68 [0.67, 0.69] (4)	0.74 [0.73, 0.75] (3)	0.31 [0.31, 0.32] (3)
Dist+Chull	NA	NA	NA	0.67 [0.66, 0.68] (6)	0.74 [0.73, 0.74] (6)	0.33 [0.32, 0.34] (5)

Table 3: Ranks of predictive feature sets for W2V and OSG models in the delayed post-test data. All models are significantly better than the baseline model. (Bold: the selected model with highest overall rank.)

	W2V models			OSG models		
Features	AUC	F_1	Err	AUC	F_1	Err
baseline	0.65 [0.64, 0.67] (5)	0.75 [0.74, 0.76] (5)	0.33 [0.32, 0.34] (5)	0.65 [0.64, 0.67] (5)	0.75 [0.74, 0.76] (7)	0.33 [0.32, 0.34] (7)
RT	0.67 [0.65, 0.68] (3)	0.76 [0.76, 0.77] (4)	0.31 [0.30, 0.32] (3)	0.67 [0.65, 0.68] (3)	0.76 [0.76, 0.77] (5)	0.31 [0.30, 0.32] (5)
Dist	0.66 [0.64, 0.68] (4)	0.77 [0.76, 0.78] (3)	0.31 [0.30, 0.32] (4)	0.66 [0.64, 0.68] (4)	0.78 [0.77, 0.79] (3)	0.30 [0.29, 0.31] (3)
Resp	0.69 [0.67, 0.71] (1)	0.77 [0.76, 0.78] (2)	0.30 [0.29, 0.31] (2)	0.63 [0.61, 0.65] (7)	0.76 [0.75, 0.77] (6)	0.32 [0.31, 0.33] (6)
Chull	NA	NA	NA	**0.69 [0.68, 0.71] (1)**	**0.78 [0.77, 0.79] (2)**	**0.28 [0.27, 0.29] (1)**
Dist+Resp	**0.68 [0.66, 0.70] (2)**	**0.78 [0.77, 0.79] (1)**	**0.30 [0.29, 0.31] (1)**	0.64 [0.62, 0.66] (6)	0.77 [0.76, 0.78] (4)	0.31 [0.29, 0.32] (4)
Dist+Chull	NA	NA	NA	0.69 [0.67, 0.71] (2)	0.78 [0.78, 0.79] (1)	0.29 [0.27, 0.30] (2)

performance of the newly trained model with the existing full-scale model.

In Table 4, we picked the top five scales that were important in individual prediction tasks. We found that *big-small*, *helpful-harmful*, *complex-simple*, and *fast-slow* were commonly important Osgood scales for predicting students' performance in immediate post-test and delayed post-test sessions. Scales like *bad-good* and *passive-active* were only important scales in the immediate post-test prediction. Likewise, *new-old* was an important scale only in the delayed post-test prediction.

Table 4: Scale-wise importance of each Osgood scale. Scales were selected based on the sum of each evaluation metric's rank. (Bold: Osgood scales that were commonly important in both prediction tasks; *: top five scales in each prediction task including tied ranks)

	Imm. post-test				Del. post-test			
Scales	AUC	F_1	Err	All	AUC	F_1	Err	All
bad-good	1	1	1	1*	4	10	4	6
passive-active	2	4	3	2*	8	6	6	7
powerful-helpless	7	9	6	7.5	10	8	10	10
big-small	3	3	4	3*	1	3	2	2*
helpful-harmful	4	6	5	5.5*	2	1	1	1*
complex-simple	8	5	2	5.5*	3	5	7	4.5*
fast-slow	5	2	7	4*	6	4	3	3*
noisy-quiet	6	8	8	7.5	7	9	9	9
new-old	9	7	9	9	5	2	8	4.5*
healthy-sick	10	10	10	10	9	7	5	8

4.2.2 Performance of Reduced Models

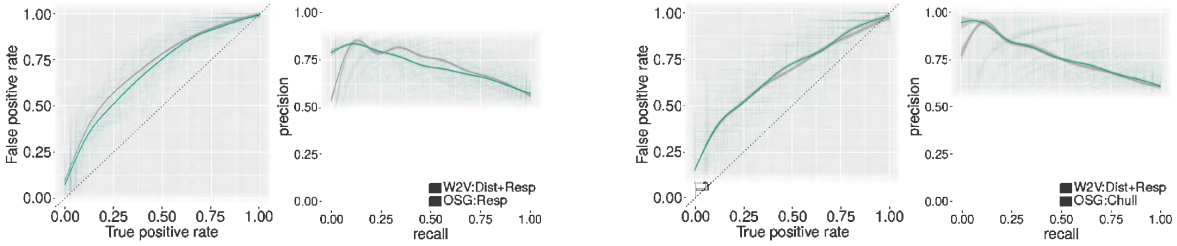
Based on the scale-wise importance analysis results, we built reduced-scale models that only contain features with more important Osgood scales. The prediction performance of reduced-scale models was similar or marginally better than full-scale OSG models. For example, the OSG model for predicting the immediate post-test outcome with the top two scales (*bad–good* and *passive–active*) were marginally better than the full-scale model (AUC: 0.71 [0.70, 0.72], F_1: 0.76 [0.75, 0.77], error rate: 0.30 [0.29, 0.30]). Similar results were observed for predicting retention in the delayed post-test (selected scales: *helpful–harmful*, *big–small*) (AUC: 0.71 [0.69, 0.72], F_1: 0.79 [0.78, 0.80], error rate: 0.28 [0.27, 0.29]). Although differences were small, the results indicate that using a small number of Osgood scales can be similarly effective to the full-scale model.

5. DISCUSSION AND CONCLUSIONS

In this paper, we introduced a novel semantic similarity scoring method that uses predefined semantic scales to represent the relationship between words. By combining Osgood's semantic scales [16] and Word2Vec [13], we could automatically extract the semantic relationship between two words in a more interpretable manner. To show this method can effectively represent students' knowledge in vocabulary acquisition, we built prediction models that can be used to predict the student's immediate learning and long-term retention. We found that our models performed significantly better than the baseline and the response-time-based models. In the future, we believe results from using an Osgood scale-based student model could be used to provide a more personalized learning experience, such as generating questions that can improve an individual student's understanding for specific semantic attributes.

Based on our findings, we have identified the following points for further discussion. First, in Section 4.1, we found that models using Osgood scale scores perform similarly with models using regular Word2Vec scores for predicting students' long-term retention of acquired vocabulary. However, we think our models can be further improved by incorporating additional features. For example, non-semantic score-based features like response time and orthographic similarity among responses can be useful features for capturing different patterns of false predictions of current models. Moreover, some general measures to capture a student's meta-cognitive or linguistic skills could be helpful to explain different retention results found even if students provided the same response sequences. Similarly, in Section 4.1.3, we found that Osgood scores can be a better metric to characterize the relationship between responses in terms of predicting students' retention. A composite model that uses both regular Word2Vec score-based feature (target-response distance) and Osgood scale score-based feature (response-response distance) may also provide better

Figure 3: ROC curves and precision and recall curves for selected immediate post-test prediction models (left) and delayed post-test prediction models (right). Curves are smoothed out with a local polynomial regression method based on repeated cross-validation results.



prediction performance.

Second, we found that models with a reduced number of Osgood scales performed marginally better than the full-scale model. However, differences were very small. Since this study only used some of the semantic scales from Osgood's study [16], further investigation would be required to examine the validity of these scales, including other scales not used for this study, for capturing the semantic attributes of student responses during vocabulary learning.

Also, there were some limitations in the current study and areas for future work. First, expanding the scope of analysis to the full set of experimental conditions used in the study may reveal more complex interactions between these conditions and students' short- and long-term learning. Second, this study used a fixed threshold of 0.5 for investigating false prediction results. However, an optimal threshold for each participant group or prediction model could be selected, especially if there are different false positive or negative patterns observed for different groups of students. Lastly, this study collected data from a single vocabulary tutoring system that was used in a classroom setting. Applying the proposed method to data that was collected from a non-classroom setting or other vocabulary learning system would be useful to show the generalization of our suggested method.

6. ACKNOWLEDGMENTS

The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A140647 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. We thank Dr. Charles Perfetti and his lab team at the University of Pittsburgh, particularly Adeetee Bhide and Kim Muth, and the helpful personnel at all of our partner schools.

7. REFERENCES

[1] S. Adlof, G. Frishkoff, J. Dandy, and C. Perfetti. Effects of induced orthographic and semantic knowledge on subsequent learning: A test of the partial knowledge hypothesis. *Reading and Writing*, 29(3):475–500, 2016.

[2] 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.

[3] K. Collins-Thompson and J. Callan. Automatic and human scoring of word definition responses. In *HLT-NAACL*, pages 476–483, 2007.

[4] M. Coltheart. The MRC psycholinguistic database. *The Quarterly Journal of Experimental Psychology*, 33(4):497–505, 1981.

[5] E. Dale. Vocabulary measurement: Techniques and major findings. *Elementary English*, 42(8):895–948, 1965.

[6] F. T. Durso and W. J. Shore. Partial knowledge of word meanings. *Journal of Experimental Psychology: General*, 120(2):190, 1991.

[7] G. A. Frishkoff, K. Collins-Thompson, L. Hodges, and S. Crossley. Accuracy feedback improves word learning from context: Evidence from a meaning-generation task. *Reading and Writing*, 29(4):609–632, 2016.

[8] G. A. Frishkoff, K. Collins-Thompson, S. Nam, L. Hodges, and S. A. Crossley. Dynamic support of contextual vocabulary acquisition for reading (DSCoVAR): An intelligent tutoring system for contextual word learning. *Handbook on Educational Technologies for Literacy*, 2016.

[9] G. A. Frishkoff, C. A. Perfetti, and K. Collins-Thompson. Predicting robust vocabulary growth from measures of incremental learning. *Scientific Studies of Reading*, 15(1):71–91, 2011.

[10] T. K. Landauer. *Latent Semantic Analysis*. Wiley Online Library, 2006.

[11] Y. Li, L. Xu, F. Tian, L. Jiang, X. Zhong, and E. Chen. Word embedding revisited: A new representation learning and explicit matrix factorization perspective. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina*, pages 3650–3656, 2015.

[12] Y. Ma, L. Agnihotri, M. H. Education, R. Baker, and S. Mojarad. Effect of student ability and question difficulty on duration. In *Educational Data Mining*, 2016.

[13] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[14] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[15] S. Nam. Predicting off-task behaviors in an adaptive vocabulary learning system. In *Educational Data Mining*, 2016.

[16] C. E. Osgood, G. J. Suci, and P. H. Tannenbaum. *The Measurement of Meaning*. University of Illinois Press, 1957.

[17] K. Ostrow, C. Donnelly, S. Adjei, and N. Heffernan. Improving student modeling through partial credit and problem difficulty. In *Proc. of the Second ACM Conference on Learning@Scale*, pages 11–20. ACM, 2015.

[18] P. I. Pavlik and J. R. Anderson. Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cog. Science*, 29(4):559–586, 2005.

[19] E. G. Van Inwegen, S. A. Adjei, Y. Wang, and N. T. Heffernan. Using partial credit and response history to model user knowledge. *International Educational Data Mining Society*, 2015.

[20] L. M. Yonek. *The Effects of Rich Vocabulary Instruction on Students' Expository Writing*. PhD thesis, University of Pittsburgh, 2008.

Generalizability of Face-Based Mind Wandering Detection Across Task Contexts

Angela Stewart
University of Notre Dame
384 Fitzpatrick Hall
Notre Dame, IN, 46556, USA
astewa12@nd.edu

Nigel Bosch
University of Illinois at Urbana-Champaign
1205 West Clark Street
Urbana, IL, 61801, USA
pnb@illinois.edu

Sidney K. D'Mello
University of Notre Dame
118 Haggar Hall
Notre Dame, IN, 46556
sdmello@nd.edu

ABSTRACT

We investigate generalizability of face-based detectors of mind wandering across task contexts. We leveraged data from two lab studies: one where 152 college students read a scientific text and another where 109 college students watched a narrative film. We automatically extracted facial expressions and body motion features, which were used to train supervised machine learning models on each dataset, as well as a concatenated dataset. We applied models from each task context (scientific text or narrative film) to the alternate context to study generalizability. We found that models trained on the narrative film dataset generalized to the scientific text dataset with no modifications, but the predicted mind wandering rate needed to be adjusted before models trained on the scientific text dataset would generalize to the narrative film dataset. Additionally, we analyzed generalizability of individual features and found that the lip tightener and jaw drop action units had the greatest potential to generalize across task contexts. We discuss findings and applications of our work to attention-aware learning technologies.

Keywords

Mind Wandering, Mental States, Attention Aware Interfaces, Cross-Corpus training.

1. INTRODUCTION

Consider a typical day when you were an undergraduate college student. Your first class is your favorite, so you are engaged in the lecture content and processing new information. In your next class, you watch a documentary about a subject that does not interest you, causing your attention to focus on unrelated thoughts of your social life, rather than processing the information in the video. Later, you work on a homework assignment that you find frustrating, leading to waning motivation. Towards the end of your day, you attend a chemistry lab, where you interact with a new educational game that teaches you the basics of chemical bonds. At some points you are enjoying the game, and thus engaged in deeply learning the content. However, you later become bored during a long period of repetitive gameplay, causing you to become distracted and miss important information. Throughout the day, your mental states (engagement, frustration, boredom) influenced your learning. Your learning experience could have been augmented with technology that responded to your changing mental state, thus assisting you in achieving the most effective learning experience.

Educational interfaces that detect and respond to student mental states are driven by work on cognitive and affective state modeling, which has been investigated for many years. For example, attention and affect has been modeled in educational tasks such as reading comprehension [6, 16, 28] and computerized tutoring [3, 19], among others. In general, there has been a plethora of work that has modeled a variety of mental states within specific educational tasks (e.g., [2, 15, 19]) to better understand these states and use that knowledge to facilitate student learning.

However, prior research has overwhelmingly investigated single task contexts, and has overlooked generalizability to different contexts. For example, models that track attention during reading might not generalize to lecture viewing, educational gaming, and so on. This makes it difficult to decouple task-specific effects from more fundamental patterns. In contrast, models that successfully generalize across multiple contexts should reveal observable signals (i.e. eye gaze, facial features, and physiology data) that are general, rather than task-specific. Models using such indicators will be key to developing adaptive technologies that are sensitive to student mental states and that can operate across a range of educational activities.

We report results on modeling mental states in a generalized way using mind wandering (MW) as a case study. MW is a ubiquitous phenomenon where thoughts shift from task-related processing to task-unrelated thoughts [15]. MW is estimated to occur anywhere from 20% - 50% of the time, depending on the person, task, and environmental context [23]. It is has also been associated with lower performance on a variety of educational tasks, such as reading comprehension [16] and retention of lecture content [29], thus impacting student learning.

As with work on other mental states, research on MW has largely failed to address models that generalize across contexts [6, 15]. MW detection has been investigated in reading comprehension [6, 16], narrative and instructional film comprehension [25, 26], and student interaction with an intelligent tutoring system (ITS) [19]. To our knowledge, no work has investigated MW detection with the goal of generalizability across task contexts.

We specifically investigate the generalizability of MW models across two task contexts - reading a scientific text and viewing a narrative film. These contexts were chosen because of their broad applicability to education in the classroom and online. For example, a documentary film could be shown in a sociology course or distance learning students could read instructional texts prior to engaging in an online discussion.

1.1 Related Work

Cross corpus training has been researched in a variety of classification problems, such as sentiment analysis [31] and acoustic-based emotion recognition [35]. Cross corpus training seeks to improve robustness of machine-learned models by leveraging multiple datasets in classifier training and testing. For example, Webb and Ferguson [32] applied cross corpus training techniques to characterize the function of segments of dialogue using automatically extracted lexical and syntactic features called cue phrases. Each extracted cue phrase was used to classify a segment of dialogue. They trained separate classifiers on two different datasets, and applied the classifier to the dataset on which it was not trained. They found the cross-training results were comparable to the results of training and testing on the same dataset (e.g. the best cross-trained classifier achieved and accuracy of 71%, compared to an accuracy of 81% when trained and tested on the same dataset). Additionally, they examined generalizability of the cue phrases across datasets by reducing the feature set to contain only cues present in both datasets. They found that reducing the feature set yielded slight improvements, and demonstrated the discriminative nature of a small number of features.

Zhang et. al. [35] similarly explored the use of multiple datasets for creating context-generalizable models. They built classifiers for valence and arousal on highly varied emotional speech datasets using a leave-one-corpora-out cross-validation technique. Additionally, they explored methods for data normalization (within each dataset and between datasets) and agglomeration of both labeled and unlabeled data. They found that, of their six emotional speech corpora, training on some subsets yielded higher accuracy than others. Their work suggested that careful selection of corpora best suited for training might yield better emotional speech recognition performance than an all-or-nothing approach to cross-corpus training.

Our work approaches cross-corpus modeling through detection of MW. A variety of studies have investigated MW detection during educational tasks, such a reading [15], interacting with an intelligent tutoring system (ITS) [19], or watching an educational video [26]. No work has focused on MW from a cross-corpus modeling perspective, to our knowledge, so we review the individual studies below.

Detection of MW from eye gaze features while reading has been amply investigated. For example, Bixler and D'Mello [4] built models to detect MW while students read texts about scientific research methods. This work made use of probe-caught reports (students respond yes or no to auditory thought probes of whether they were MW), instead of self-caught reports (students report whenever they catch themselves MW). Their analysis of eye gaze features showed that certain types of fixations were longer during MW. Specifically, they found that longer gaze fixations (consecutive fixations on a single word), first-pass fixations (fixations on a word during the first pass through a text), and single fixations (fixations on a word only fixated on once) were predictive of MW. In other work, Bixler and D'Mello [5] similarly used eye gaze features, but used self-caught reports of MW. They found that a greater number of fixations, longer saccade length, and line cross saccades were indicative of MW. Across studies on MW detection during reading, longer fixations were found to be indicative of MW [4, 15, 28], suggesting these features might generalize well.

Pham and Wang [26] similarly used consumer-grade equipment to detect MW while students watched videos from massively open online courses (MOOCs). They made use of heart rate, detected by monitoring fingertip blood flow, using the back camera of a smartphone (i.e., photoplethysmography). Their models achieved a 22% improvement over chance. Although their method for detecting MW could be implemented across a variety of tasks, the question of whether heart rate is indicative of MW across task contexts has not yet been investigated.

Hutt et. al. provided limited evidence of generalizability of MW detection across different learning tasks during student interaction with an ITS [19]. They employed a genetic algorithm to train a neural network using context-independent eye-gaze features and context-dependent interaction features (e.g., current progress within the ITS). They achieved an F_1 value of .490 (chance = .190). This work provided some evidence of generalizability because the visual stimuli and interaction patterns varied throughout. For example, students interacted with an animated pedagogical agent in a scaffolded dialogue phase and completed concept maps without the tutoring agent in another interaction phase. However, it is still unclear if their model would generalize to a broader range of tasks, particularly less interactive ones like reading or film viewing. Furthermore, their best-performing models used context-dependent features, which could prevent the detector from generalizing to a task where those features could not be used.

1.2 Novelty

Our contribution is novel in a variety of ways. First, we demonstrate the feasibility of building cross-context detectors of mental states, specifically MW. Further, previous work on MW detection has sometimes made use of context-specific features (e.g., reading times) that are not expected to generalize to other contexts [19, 25]. In contrast, our work detects MW using only facial features and upper body movement, recorded using commercial-off-the-shelf (COTS) webcams that are expected to generalize more broadly. Additionally, the use of COTS webcams support a broader implementation of MW detectors as webcams are ubiquitous in modern technology. This is in contrast to prior research that has used specialized equipment, like eye trackers [15, 19, 25] or physiology sensors [7], which students would likely not have access to.

2. DATASETS

This study makes use of narrative film [23] and scientific reading comprehension [22] datasets collected as part of a larger project. Here, we include details pertaining to video-based detection of MW.

2.1 Narrative Film Comprehension

Participants were 68 undergraduate students from a medium-sized private Midwestern university and 41 undergraduate students from a large public university in the Southern United States. Of the 109 students, 66% were female and their average age was 20.1 years. Students were compensated with course credit. Data from four students were discarded due to equipment failure.

Students viewed the narrative film *The Red Balloon* (1956), a 32.5-minute French-language film with English subtitles (Figure 1). The film has a musical score but only sparse dialogue. This short fantasy film depicts the story of a young Parisian boy who finds a red helium balloon and quickly discovers it has a mind of its own as it follows him wherever he goes. This film was selected because of the low likelihood that participants have previously seen it and because it has been used in other film comprehension studies [34].

measurement necessitate, but which in more enlightened countries are wholly unnecessary. This book is not prepared to meet the requirements and artificial restrictions of any syllabus, and it is not prepared to help students through any examination. I cannot help thinking, however, that it the type of student who puts more faith in learning formulae

Figure 1. A screenshot of the narrative film (left) and scientific text (right) are shown.

Students' faces and upper bodies were recorded with a low-cost ($30) consumer-grade webcam (Logitech C270).

Students were instructed to report MW throughout the film by pressing labeled keys on the keyboard. Specifically, students were asked to report a task-unrelated thought if they were "thinking about anything else besides the movie" and a task-related interference if they were "thinking about the task itself but not the actual content of the movie." A small beep sounded to register their report, but film play was not paused. After viewing the film, students took a short test about the content and completed additional measures not discussed further.

We recorded a total of 1,368 MW reports from the 105 participants with valid video recordings. In this work, we do not distinguish between the two types of MW, instead merging the task-unrelated thoughts and the task-related interferences, both of which represent thoughts independent of the content of the film.

2.2 Scientific Reading Comprehension

Participants were 104 undergraduate students from a medium-sized private Midwestern university and 48 undergraduate students from a large public university in the Southern United States. Of the 152 participants, 61% were female and their average age was 20.1 years. Participants were compensated with course credit. Data from eight participants were discarded due to equipment failure.

Students read an excerpt from *Soap-Bubbles and the Forces which Mould Them* [8]. Like *The Red Balloon* (Figure 1), we chose this text because its content would likely be unfamiliar to a majority of readers. The text contained around 6,500 words from the first chapter of the book. In all, 57 pages (screens of text) with an average of 115 words each were displayed on a computer screen in 36-pt Courier New typeface. The only modification to the text was the removal of images and references to them after verifying that these were not needed for comprehension.

Students who read the scientific text were instructed to report MW in the same way as those who watched the narrative film. They were instructed to report a task-unrelated thought if they were "thinking about anything else besides the task" and a task-related interference if they were "thinking about the task itself but not the actual content of the text." Participants completed a comprehension assessment after reading the text. We recorded a total of 3,168 MW reports from the 144 students with valid video recordings.

2.3 Self Reports of MW

MW was measured via self-reports in both studies, so it is prudent to discuss the validity of self-reports. We used self-reports because this is currently the most common approach to measure an inherently internal (but conscious) phenomenon [5, 15]. Self-reported MW has been linked to predictable patterns in physiology [30], pupillometry [17], eye-gaze [28] and task performance [27], providing evidence for the convergent and predictive validity for this approach. To improve the quality of self-reports, we encouraged students to report honestly and assured them that reporting MW would not in any way effect the credit they received for participation.

The alternative to using self-caught reports is using probe-caught reports, which require a student response to a thought-probe (e.g., a beep). We chose self-caught reports over the probe-caught because the probe-caught method can potentially interrupt the comprehension process (i.e., when participants report "no" to the probes). Interruptions are particularly problematic in the film comprehension task, as participants did not have control over the media presentation (i.e., no pausing or rewinding of the film). Furthermore, it is also unclear if a probe-caught report takes place at the beginning or end of MW, or somewhere in between. Conversely, self-caught reports are likely to occur at the end of a MW episode when the student became aware that they were not attending to the task at hand.

3. MACHINE LEARNING

We explored a variety of machine learning techniques for cross-context MW detection using the same approach to segmenting instances and constructing features for both datasets.

3.1 Segmenting Instances

Reports of MW were distributed throughout the course of the film viewing or text reading session. We created instances that corresponded to reports of MW by first adding a 4-second offset prior to the report. This was done to ensure that we captured participants' faces while MW vs. in the act of reporting MW itself (i.e., the preparation and execution of the key press). This 4-second offset was chosen based on four raters judgements of whether or not movement related to the key-press could be seen within offsets ranging from 0 to 6 seconds. Data was then extracted from the 20 seconds prior to the MW report. A window size of 20 seconds was chosen based on prior experimentation that sought to balance creating as many instances as possible (shorter window sizes) and having sufficient data in each window (longer window sizes) to detect MW.

We extracted "not MW" instances from windows of data between MW reports. The entire session (reading or video watching) was divided into 24-second segments (20 second windows of data and a 4 second offset as with the MW segments). Any segments

overlapping the 30 seconds prior to a MW report were discarded. We do not know precisely when MW starts, so we chose to discard instances overlapping the 30 seconds prior to MW reports, to separate students when they were actually MW from when they were not. We also discarded any segments overlapping a page turn (discussed in Section 3.2). All remaining segments were labeled Not MW. Our approach to segmenting instances is shown in Figure 2.

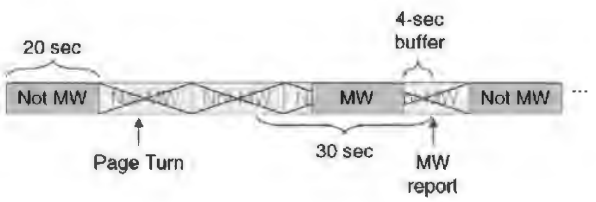


Figure 2. Illustration of the instance extraction method.

3.2 Instance Selection

A full accounting of the instance selection process is shown in Table 1. Our goal was to make the two data sets as similar as possible so that task-specific effects could be studied without additional confounds.

We first discarded any instances where there was less than one second of usable data in that time window. Data was not usable when the student's face was occluded due to extreme head pose or position, hand-to-face gestures, and rapid movements. Additionally, for the scientific reading dataset, we discarded instances that overlapped with page turn events. In prior experimentation, we trained a model to detect MW using only a binary feature of whether or not that instance overlapped a page turn boundary. MW was detected at rates above chance in this experimental model. Therefore, we concluded that including instances that overlapped page turn boundaries would inflate performance as the detector could simply be picking up on the act of pressing the key to advance to the next page.

After discarding instances using the method above, we matched the scientific reading and narrative film datasets on school (medium-sized Midwestern private university or large Southern public university), reported ethnicity, and reported gender. The scientific reading dataset was randomly downsampled to contain approximately the same number of students in each gender, race, or school category, as the film dataset. This participant-level matching on school, ethnicity, and gender was done to eliminate external sources of variance that could influence MW detection, potentially obfuscating task effects from population effects.

Finally, the datasets were downsampled to contain equal numbers of instances because the size of the training set is known to bias classifier performance [13]. We also downsampled the data to achieve a 25% MW rate in order to be consistent with research that suggests that MW occurs between 20% and 30% of the time during reading and film comprehension [6, 23]. Further, the MW rates of 30% and 14% obtained in these data are more artefacts of the instance segmentation approach rather than the objective rate, so resampling ensures a dataset that is more reflective of expected MW rates.

Table 1. An accounting of instance selection process

	Reading (% MW)	Film (% MW)
Base	7,267 (30%)	7,313 (14%)
Face Detected	7,266 (30%)	7,238 (14%)
Page Boundary	1,400 (36%)	N/A
Participant Matching	1,273 (35%)	N/A
Downsampling	1,100 (25%)	1,100 (25%)

3.3 Feature Extraction and Selection

We used commercial software, the Emotient SDK [36] to extract facial features. The Emotient SDK, a version of the CERT computer vision software [24] (Figure 3) provides likelihood estimates of the presence of 20 facial action units (AUs; specifically 1, 2, 4, 5, 6 ,7, 9, 10, 12, 14, 15, 17, 18, 20, 23, 24, 25, 26, 28, and 43 [14]) as well as head pose (orientation), face position (horizontal and vertical within the frame), and face size (a proxy for distance to camera). Additionally, we used a validated motion estimation algorithm to compute gross body movements [33]. Body movement was calculated by measuring the proportion of pixels in each video frame that differed by a threshold from a continuously updated estimate of the background image generated from the four previous frames.

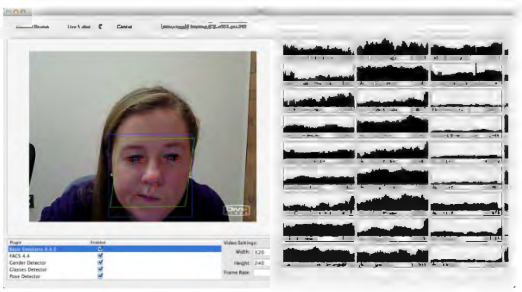


Figure 3. Interface demonstrating AU estimates detected from a face video.

Features were created by aggregating Emotient estimates in a window of time leading up to each MW or Not MW instance using minimum, maximum, median, mean, range, and standard deviation for aggregation. In all, there were 162 facial features (6 aggregation functions × [20 AUs + 3 head pose orientation axes + 2 face position coordinates + face size + Motion]). Outliers (values greater than three standard deviations from the mean) were replaced by the closest non-outlier value in a process called Winsorization [11].

We used tolerance analysis to eliminate features with high multicollinearity (variance inflation factor > 5) [1], after which, 37 features remained. This was followed by RELIEF-F [21] feature selection (on the training data only) to rank features. We retained a proportion of the highest ranked features for use in the models (proportions ranging from .05 to 1.0 were tested). Feature selection was performed using nested cross-validation on training data only. We ran 5 iterations of feature selection within each cross-validation fold (discussed below), using data from a randomly chosen 67% of students within the training set in each iteration.

3.4 Supervised Classification and Validation

Informed by preliminary experiments, we selected seven classifiers for more extensive tests (Naïve Bayes, Simple Logistic Regression, LogitBoost, Random Forest, C4.5, Stochastic Gradient Descent, and Classification via Regression) using the WEKA data mining

toolkit [18]. For each classifier, we applied SMOTE [9] to the training set only. SMOTE, a common machine learning technique for dealing with data imbalance, creates synthetic interpolated instances of the minority class to increase classification performance.

We evaluated the performance of our classifiers using leave-one-participant-out cross-validation. This process runs multiple iterations of each classifier in which, for each fold, the instances pertaining to a single participant are added to the test set and the training set is comprised of the instances for the other participants. Feature selection was performed on a subset of participants in the training set. The leave-one-out process was repeated for each participant, and the classifications of all folds were weighted equally to produce the overall result. This cross-validation approach ensured that in each fold, data from the same participant was in the training set or testing set but never both, thereby improving generalization to new participants.

Accuracy (recognition rate) is a common measure to evaluate performance in machine learning tasks. However, any classifier that defaults to predicting the majority class label of an imbalanced dataset can appear to have high accuracy despite incorrect predictions of all instances of the minority class label [20]. This is particularly detrimental in applications where detecting the minority class is of upmost importance. In our task, we prioritized the detection of MW despite the large imbalance in our dataset. Therefore, we considered the F_1 score for the MW label as our key measure of detection accuracy since F_1 attempts to strike a balance between precision and recall.

4. RESULTS

4.1 Cross-dataset Training and Testing

We trained three classifiers: one on the scientific text dataset, one on the narrative film dataset, and one on a concatenated dataset comprised of the first two. For each of the three training sets, the classifier that yielded the highest MW F_1 is shown in Table 2. We used leave-one-student-out cross validation for within-dataset evaluations. Conversely, to measure generalizability of the models across contexts we applied the classifier trained on scientific text data to the narrative film data, and vice versa. We compared our model to a chance model that classified a random 25% (MW prior proportion) of the instances as MW. This chance-level method yielded a precision and recall of .250 (equal to the MW base rate).

Table 2. Results for the models with highest MW F_1 for the within-data set validation (cross-training results in parentheses).

Training Set	Classifier	MW F_1	Precision	Recall
Scientific Text	Logitboost	.441 (.267)	.376 (.252)	.553 (.284)
Narrative Film	C4.5	.436 (.407)	.303 (.278)	.775 (.760)
Both	Logistic	.424	.314	.655

We calculated improvement over chance as (actual performance – chance)/(perfect performance – chance). All three models showed improvement over chance (25% for scientific text, 25% for narrative film, and 23% for the concatenated dataset) when trained and tested on the same dataset. When tested on the alternative dataset, the narrative film classifier generalized well to the scientific text dataset (21% improvement over chance). However, the scientific text model showed chance-level performance on the narrative film corpus (2% improvement over chance). The MW F_1 of the concatenated dataset model was simply an average of the MW F_1 score of the individual datasets when the instance predictions of the individual datasets are separated (.413 for the scientific reading dataset and .436 on the narrative film dataset). These results showed that the concatenated classifier does not skew towards predicting one dataset better than the other, but rather predicts both models with comparable accuracy.

Table 2 also shows precision and recall for each of the models. Across all models, recall was higher than precision, indicating a lot false positives. It is important to note the near chance-level recall and precision of the model trained on scientific reading data when applied to the narrative film data. The lack of improvement over chance for both recall and precision demonstrated the need to improve generalizability in both dimensions. Conversely, the cross-trained narrative film model had lower precision, but good recall, resulting in an improved MW F_1 score.

4.2 Classifier Generalizability

To address the negligible improvement over chance of the scientific text model when tested on the narrative film dataset, we repeated the training and testing using C4.5 as the classifier. The C4.5 classifier was chosen because it generalized better when trained on the narrative film dataset than the Logitboost classifier generalized when trained on the scientific text dataset. The results are shown in Table 3, where we note no notable improvement over the previous Logitboost classifier in Table 2 (change from .267 to .287 when tested on the narrative film dataset). Therefore, the lack of evidence for generalizability for the scientific text model could be due to overfitting to the training set, rather than classifier selection.

Table 3. Results (MW F_1) for the C4.5 classifier for within- and cross- validation.

Training Set	Within	Cross
Scientific Text	0.425	0.287
Narrative Film	0.436	0.407
Both	0.415	N/A

4.3 Prediction Threshold Adjustment

We further investigated the lack of generalizability of the scientific text model by considering the MW prediction rate. We compared the performance of both models on the narrative film dataset. Recall dropped considerably more than precision (Table 2; recall dropped from .775 to .284; precision decreased from .303 to .252). We hypothesized that recall decreased because of a difference in predicted MW rates (Table 4). In fact, the predicted MW rate in the narrative film data dropped from 64% to 28% when applying the scientific text model to the same data. This supported our hypothesis that the low recall was linked to lower predicted MW rates. Furthermore, 39% of the correctly classified instances (true positives and true negatives) were MW when applying the narrative film model to the narrative film data compared to 12% for the scientific text model applied to the same data. This demonstrated that the scientific text model was much more prone to missing MW instances, further supporting our hypothesis.

To address this, we adjusted the predicted MW rate of the scientific text model when applied to the narrative film dataset. The classifier outputs a likelihood of MW and we previously considered instances with likelihoods greater than .5 as MW. We adjusted that prediction threshold from .1 to 1 in increments of .1 (Figure 4) to investigate how changes in predicted MW rate (higher for lower thresholds) effected recall, and thus MW F_1.

Table 4. Predicted MW Rates.

Training Set	Within	Cross
Scientific Text	38%	**28%**
Narrative Film	**64%**	68%
Both	52%	N/A

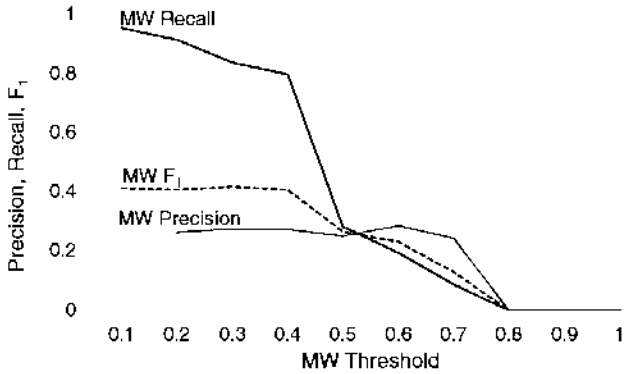


Figure 4. MW precision, recall, and F_1 as the prediction threshold varies for the scientific text model applied to the narrative film dataset.

We note that MW F_1 score degrades at a threshold of .5. We adjusted the threshold to .3 and yielded the results shown in Table 5. After adjusting the MW prediction threshold, both precision and recall of the narrative film data applied to the scientific text model showed comparable performance to the cross-trained narrative film model. It is important to note that the adjusted MW prediction threshold yielded a predicted MW rate of 76%, much higher than the MW rate of the dataset (25%). As with the generalized narrative film model, this reduced precision because the high predicted MW rate produced a large number of false positives.

Table 5. Results for models with highest MW F_1 (cross-training results in parentheses). Cross-training results for the scientific text model reflect a MW prediction threshold of .3.

Training Set	Classifier	MW F_1	Precision	Recall
Scientific Text	Logitboost	.441 (**.416**)	.376 (**.276**)	.553 (**.836**)
Narrative Film	C4.5	.436 (.407)	.303 (.278)	.775 (.760)
Both	Logistic	.424	.314	.655

4.4 Feature Analysis

We analyzed the facial features to further study generalizability by predicting MW with different subsets of the entire feature set. The C4.5 classifier was chosen for this feature analysis because of its consistency on both the scientific text model and concatenated dataset. Each subset consisted of the features (e.g., median, standard deviation) from one AU, or from face position, size, orientation, or motion. Since tolerance analysis was not used here, we only considered the minimum, maximum, median, and standard deviation aggregated features to prevent redundancy (e.g., between median and mean). For example, we used the minimum, maximum, median, and standard deviation feature values for AU5 (upper lid raiser) to predict MW. This approach was applied to the 20 AU subsets, as well as face position, size, orientation, and motion subsets. We generated the same cross-training configurations of in Section 4.1 (i.e., train on scientific text, test on narrative film, etc.).

To rank the subsets of features on generalizability, we examined MW F_1 scores when testing on the alternative dataset only. For example, using the AU9 (nose wrinkle) subset, we investigated MW F_1 value of scientific text model applied to the narrative film dataset and the narrative film model applied to the scientific text dataset. Table 4 shows these results only for features that achieved a MW F_1 of greater than .250 (chance) on all dimensions (within dataset validation and cross-training). We selected features for further analysis if their MW F_1 was greater than .300 for both cross-training results. This value of .300 was used to filter out features that performed well on the within-dataset validation, but fell short on cross training. It also ensured that a feature performed better than chance on both cross-trained results (i.e., train on narrative film and test on scientific text, and vice versa), rather than only generalizing to one dataset. Using this criterion, only AU23 and AU26 showed notable improvement over chance.

We used the C4.5 classifier to generate the same models in Table 2 (train/test scientific text, train scientific text/test narrative film, etc.) using only the features from AU23 and AU26 (Table 7). None of these models (scientific text, narrative film, or concatenated) achieved a MW F_1 as high as those in Table 2, which used a combination of tolerance analysis and RELIEF-F to select features. This suggested that, while AU23 and AU26 might individually predict MW, when used together, their prediction power might be limited, compared to other feature selection techniques.

Table 6. MW F_1 score for within-data set validation with cross-data set scores (in parentheses).

	Training Set	
Facial Feature	**Scientific Text**	**Narrative Film**
AU4 (brow lowerer)	.378 (.278)	.398 (.395)
AU6 (cheek raiser)	.369 (.259)	.361 (.321)
AU9 (nose wrinkler)	.300 (.268)	.392 (.303)
AU14 (dimpler)	.303 (.267)	.383 (.376)
AU23 (lip tightener)	**.334 (.333)**	**.363 (.317)**
AU26 (jaw drop)	**.414 (.321)**	**.365 (.357)**
Face Height (size)	.322 (.256)	.339 (.289)
Face X (position)	.404 (.316)	.382 (.282)

Table 7. Results for models when only using the C4.5 classifier on AU23 and AU26.

Training Set	Classifier	MW F_1	Precision	Recall
Scientific Text	C4.5	.383 (.272)	.255 (.206)	.764 (.404)
Narrative Film	C4.5	.397 (.257)	.333 (.235)	.491 (.284)
Both	C4.5	.368	.271	.575

5. ANALYSIS

We developed automated detectors of MW using video-based features in the contexts of narrative film viewing and scientific reading. The generalizability of these models was dependent on corpora on which the model was trained and the rate at which the model predicts MW. In this section, we discuss our main findings and applications of this work. We also discuss limitations and future work.

5.1 Main Findings

We expanded on previous MW detection work through cross-context modeling. We trained three models on three datasets

(scientific text, narrative film, and a dataset concatenated from the two). We found each of these models (trained and tested on the same corpus) performed at a notable 23% to 25% improvement over chance. This demonstrated the feasibility of detecting MW on individual corpora. However, recall was greater than precision, indicating prediction of false positives. This should be considered when implementing MW detectors in educational environments where excessive prediction of student MW could be demotivating.

We investigated generalizability of the single-dataset models (i.e. scientific text or narrative film) by applying the model to the dataset on which it was not trained. The model trained on the narrative film dataset maintained performance when applied to the scientific text dataset (Table 2), providing some evidence for generalizability, but this performance was boosted by high recall (and comparatively low precision). Precision and recall (and thus MW F_1) were near chance-level when the model trained on the scientific text dataset was applied to the narrative film dataset, suggesting that the model might overfit to the scientific text training set.

We attempted to address this problem by applying the C4.5 classifier, as it comparatively generalized well when trained on the narrative film dataset. MW F_1 score for the scientific text classifier applied to the narrative film data again negligibly increased. This suggested that the training data (only scientific text) used was not appropriate for model generalization. This idea is supported by the performance of the narrative film model on the scientific text data (although detection of false positives is a limitation) and the notable improvement over chance (22% to 23%) for the concatenated dataset. The performance of both models suggested that there were discernable similarities between MW instances across the two datasets, which can be detected using our techniques.

In addition to training data, we also found that predicted MW rate effected model generalizability. We adjusted MW predictions according to a sliding threshold for the narrative film predictions obtained from the scientific text model. We found that relaxing the criteria for classifying an instance as MW (i.e. adjusting the likelihood prediction threshold from .5 to .3) yielded results comparable to the cross-trained narrative film model. However, this approach to increasing recall should be used with caution as it leads to increased likelihood of false positives. Perhaps in a real-time MW intervention scenario, a more balanced approach could be taken where the MW likelihood prediction is used to determine if a MW intervention is triggered (e.g., if the detector determines there is a 40% likelihood the student is MW, then there is a 40% chance a MW intervention is triggered).

We detected MW using individual feature subsets to ascertain whether certain face-based features (i.e. AUs, head orientation, position, size, and motion) generalize. We found two feature subsets (AU23 – lip tightener and AU26 – jaw drop) that showed a MW F_1 of at least .300 on both cross-trained models. It is notable that when looking at the generalizability of these features, they did not individually achieve MW F_1 scores as high as the best performing models in Table 2. This demonstrated the need for multiple features to work together to detect MW, rather than relying on a single feature. Furthermore, this showed that our method of feature selection (tolerance analysis and selecting a proportion of features using RELIEFF) was important to model performance.

5.2 Applications

The present findings are applicable to educational user interfaces that involve reading or film comprehension. Monitoring and responding to MW could greatly improve student performance on these tasks. Films and instructional texts play a major role in learning (both in the classroom and online). For example, films can give historical background on a time period being discussed in literature classes and instructional texts can supplement lecture content through textbooks or technical articles. Due to the relationship between MW and low task performance, user interfaces that detect and respond to MW in contexts where attention is key (i.e. education) would help students remain focused on their learning.

These findings are particularly promising for implementation in massively open online courses (MOOCs). Our method for detecting MW exclusively uses COTS webcams. These webcams are ubiquitous in today's computers and mobile devices; thus our work would integrate into a variety of learning environments without extra cost. Such a video-based detector of MW could feasibly respond to student MW through suggesting a student revisit text or video content, asking a reengaging question, or advising the student to take a break.

5.3 Limitations and Future Work

While we demonstrated techniques for modeling generalizability across task contexts, our work has a few limitations. First, precision is moderate, even on our best models. High predicted MW rates lead to high recall, but also more false positives. In this work, we chose to accept this tradeoff, with the goal of generalizability in mind. However, raising precision, while maintaining recall is key to task-generalizable MW detectors being successful in educational environments. Since MW is the minority class (25% of all instances), investigating skew-insensitive classifiers, such as Hellinger Distance Decision Trees [10], could improve precision.

Additionally, this work focuses exclusively on generalizability from the perspective of task context (viewing a narrative film vs. reading a scientific text). Claims of generalizability could be strengthened through MW detection across environments. Both the narrative film and scientific reading datasets were collected in a controlled lab setting. MW detection in the field, such as computer-enabled classrooms or the personal workstations of MOOC users, should be considered prior to implementation in such environments. Furthermore, student generalizability should be further examined. In this work, we detect MW in a student-independent way. However, participants were all of similar age and enrolled in college. Future work could examine the generalizability of our method for detecting MW in non-college-aged students, such as elementary students in a computer-enabled classroom or non-traditional students enrolled in distance learning courses.

5.4 Concluding Remarks

In this work, we showed evidence that generalizable detectors of MW can be created using video-based features. The corpora used to train models of MW and predicted MW rates both play a role in the model's ability to generalize and should be considered as work on cross-context MW generalization advances. This work advances the field of attention-aware interfaces [12] by demonstrating the feasibility of modeling MW across the educational contexts of reading a scientific text and viewing a narrative film. Our approach to detecting MW is the first step towards building interfaces that detect MW across multiple educational activities.

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] Allison, P.D. 1999. *Multiple regression: A primer*. Pine Forge Press.

[2] Baker, R.S. et al. 2012. Towards automatically detecting whether student learning is shallow. *International Conference on Intelligent Tutoring Systems* (Chania, Crete, Greece, 2012), 444–453.

[3] Baker, R.S. et al. 2012. Towards sensor-free affect detection in a Cognitive Tutor for Algebra. *Educational Data Mining* (Chania, Crete, Greece, 2012).

[4] Bixler, R. and D'Mello, S. 2016. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction*. 26, 1 (2016), 33–68.

[5] Bixler, R. and D'Mello, S.K. 2015. Automatic Gaze-Based Detection of Mind Wandering with Metacognitive Awareness. *User Modeling, Adaptation and Personalization: 23rd International Conference* (Dublin, Ireland, 2015), 31–43.

[6] Bixler, R. and D'Mello, S.K. 2014. Toward fully automated person-independent detection of mind wandering. *Proceedings of the 22nd International Conference on User Modeling, Adaptation, and Personalization* (Switzerland, 2014), 37–48.

[7] Blanchard, N. et al. 2014. Automated physiological-based detection of mind wandering during learning. *Intelligent Tutoring Systems* (Honolulu, Hawaii, USA, 2014), 55–60.

[8] Boys, C.V. and others 1890. *Soap-bubbles, and the forces which mould them*. Cornell University Library.

[9] Chawla, N.V. et al. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*. (2002), 321–357.

[10] Cieslak, D.A. et al. 2012. Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*. 24, 1 (2012), 136–158.

[11] Dixon, W.J. and Yuen, K.K. 1974. Trimming and winsorization: A review. *Statistische Hefte*. 15, 2–3 (1974), 157–170.

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

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

[14] Ekman, P. and Friesen, W.V. 1977. *Facial action coding system*.

[15] Faber, M. et al. 2017. An automated behavioral measure of mind wandering during computerized reading. *Behavior Research Methods*. (2017), 1–17.

[16] Franklin, M.S. et al. 2011. Catching the mind in flight: Using behavioral indices to detect mindless reading in real time. *Psychonomic Bulletin & Review*. 18, 5 (2011), 992–997.

[17] Franklin, M.S. et al. 2013. Window to the wandering mind: pupillometry of spontaneous thought while reading. *The Quarterly Journal of Experimental Psychology*. 66, 12 (2013), 2289–2294.

[18] Holmes, G. et al. 1994. Weka: A machine learning workbench. *Proceedings of the 1994 Second Australian and New Zealand Conference on Intelligent Information Systems* (1994), 357–361.

[19] Hutt, S. et al. 2016. The eyes have it: gaze-based detection of mind wandering during learning with an intelligent tutoring system. *Proceedings of the 9th International Conference on Educational Data Mining, International Educational Data Mining Society* (2016), 86–93.

[20] Jeni, L.A. et al. 2013. Facing Imbalanced Data–Recommendations for the Use of Performance Metrics. *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on* (2013), 245–251.

[21] Kononenko, I. 1994. Estimating attributes: analysis and extensions of RELIEF. *Machine Learning: ECML-94* (1994), 171–182.

[22] Kopp, K. et al. 2015. Influencing the occurrence of mind wandering while reading. *Consciousness and cognition*. 34, (2015), 52–62.

[23] Kopp, K. et al. 2015. Mind wandering during film comprehension: The role of prior knowledge and situational interest. *Psychonomic Bulletin & Review*. 23, 3 (2015), 842–848.

[24] Littlewort, G. et al. 2011. The computer expression recognition toolbox (CERT). *2011 IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011)* (2011), 298–305.

[25] Mills, C. et al. 2016. Automatic Gaze-Based Detection of Mind Wandering during Film Viewing. *Proceedings of the 9th International Conference on Educational Data Mining* (Raleigh, NC, USA, Jun. 2016).

[26] Pham, P. and Wang, J. 2015. AttentiveLearner: Improving Mobile MOOC Learning via Implicit Heart Rate Tracking. *Artificial Intelligence in Education*. C. Conati et al., eds. Springer International Publishing. 367–376.

[27] Randall, J.G. et al. 2014. Mind-Wandering, cognition, and performance: A theory-driven meta-analysis of attention regulation. *Psychological bulletin*. 140, 6 (2014), 1411.

[28] Reichle, E.D. et al. 2010. Eye movements during mindless reading. *Psychological Science*. 21, 9 (2010), 1300–1310.

[29] Risko, E.F. et al. 2013. Everyday attention: Mind wandering and computer use during lectures. *Computers & Education*. 68, (2013), 275–283.

[30] Smallwood, J. et al. 2004. Subjective experience and the attentional lapse: Task engagement and disengagement during sustained attention. *Consciousness and cognition*. 13, 4 (2004), 657–690.

[31] Wan, X. 2009. Co-training for Cross-lingual Sentiment Classification. *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP* (Stroudsburg, PA, USA, 2009), 235–243.

[32] Webb, N. and Ferguson, M. 2010. Automatic Extraction of Cue Phrases for Cross-corpus Dialogue Act Classification. *Proceedings of the 23rd International Conference on Computational Linguistics: Posters* (Stroudsburg, PA, USA, 2010), 1310–1317.

[33] Westlund, J.K. et al. 2015. Motion Tracker: Camera-Based monitoring of bodily movements using motion silhouettes. *PloS one*. 10, 6 (2015).

[34] Zacks, J.M. et al. 2010. The brain's cutting-room floor: Segmentation of narrative cinema. *Frontiers in human neuroscience*. 4, 168 (2010), 1–15.

[35] Zhang, Z. et al. 2011. Unsupervised learning in cross-corpus acoustic emotion recognition. *2011 IEEE Workshop on Automatic Speech Recognition Understanding* (Dec. 2011), 523–528.

[36] 2016. *Emotient module: Facial expression emotion analysis*.

Addressing Student Behavior and Affect with Empathy and Growth Mindset

Shamya Karumbaiah
University of Massachusetts Amherst
140 Governors Drive
Amherst, MA 01003-9264
shamya@cs.umass.edu

Rafael Lizarralde
University of Massachusetts Amherst
140 Governors Drive
Amherst, MA 01003-9264
rezecib@cs.umass.edu

Danielle Allessio
University of Massachusetts Amherst
140 Governors Drive
Amherst, MA 01003-9264
allessio@umass.edu

Beverly Woolf
University of Massachusetts Amherst
140 Governors Drive
Amherst, MA 01003-9264
bev@cs.umass.edu

Ivon Arroyo
Worcester Polytechnic Institute
100 Institute Rd
Worcester, MA 01609
iarroyo@wpi.edu

Naomi Wixon
Worcester Polytechnic Institute
100 Institute Rd
Worcester, MA 01609
mwixon@wpi.edu

ABSTRACT

We present results of a randomized controlled study that compared different types of affective messages delivered by pedagogical agents. We used animated characters that were empathic and emphasized the malleability of intelligence and the importance of effort. Results showed significant correlations between students who received more *empathic messages* and those who were *more confident, more patient*, exhibited *higher* levels of *interest*, and *valued* math knowledge more. Students who received more *growth mindset* messages, *tended to get more problems correct* on their first attempt but *valued* math knowledge *less* and had lower *posttest scores*. Students who received more *success/failure* messages tended to make *more mistakes*, to be *less learning-oriented*, and stated that they were *more confused*. We conclude that these affective messages are powerful media to influence students' perceptions of themselves as learners, as well as their perceptions of the domain being taught. We have reported significant results that support the use of empathy to improve student affect and attitudes in a math tutor.

Keywords

student affect, empathy messages, growth mindset, pedagogical agents, intelligent tutor, confidence

1. INTRODUCTION

Students experience many emotions while studying and taking tests [16]. Students' emotions (such as confidence, boredom, and anxiety) can influence achievement outcomes [10, 18] and predispositions (such as low self-concept and pessimism) can diminish academic success [5, 14].

Pekrun's Control-Value Theory of emotion has been experimentally validated by classroom experiments that used student self-reports (answers to 5-point scale survey questions). These experiments provide evidence that educational interventions can reduce students' anxiety [16, 19].

Dweck's Growth Mindset Theory suggests that students who believe that intelligence can be increased through effort and persistence tend to seek out academic challenges, compared to those who view their intelligence as immutable [8, 9]. Students who are praised for their effort (as opposed to performance) are more likely to view intelligence as being malleable, and their self-esteem remains stable regardless of how hard they have to work to succeed at a task.

Hattie and Timperley [13] studied which types of feedback and conditions enable learning to flourish and which cases stifle growth. According to their study feedback is intended to help a student get from where they are to where they need to be. Graesser et al., [12] reported that there are significant relationships between the content of feedback dialogue and the emotions experienced during learning. They found significant correlations between dialog and the affective states of confusion, eureka (delight) and frustration.

Pekrun et al., [17] tested a theoretical model positing that a student's anticipated achievement feedback in a classroom setting influences his/her achievement goals and emotions. For example, *self-referential feedback*, in which a student's competence is defined in terms of self-improvement, had a positive influence on a student's mastery goal adoption. On the other hand, *normative feedback*, in which student competence is defined relative to other students' mastery goals and performance goals, had a positive influence on *performance-approach* and *performance-avoidance* goal adoption. Furthermore, feedback condition and achievement goals predicted test-related emotions (i.e., enjoyment, hope, pride, relief, anger, anxiety, hopelessness, and shame).

Teachers have limited opportunities to recognize and respond to individual student's affect in typical classrooms.

Ideally, digital learning environments can manage the delicate balance between motivation and cognition, promoting both interest and deep learning. The overwhelming majority of work on affect-aware virtual tutors has focused on modeling affect, i.e., designing computational models capable of detecting how students feel while they interact with intelligent tutoring systems [2]. While modeling affect is a critical first step, very little research exists on systematically exploring the impact of interventions on students' performance, learning, and attitudes, i.e., how an environment might respond to students emotions (e.g., frustration, anxiety, and boredom) as they arise. D'Mello and Graesser carried out close research work on empathic characters in AutoTutor, a conversational tutor that uses 3D companions to conduct dialogs in natural language with students [6, 7, 11].

1.1 MathSpring

The testbed for this research is MathSpring, an intelligent tutor that personalizes mathematics problems, provides help using multimedia, and effectively teaches students to improve in standardized test scores [4]. Learning companions (Figure 1) in MathSpring suggest to students that their effort contributes to success, and that making mistakes only means more effort is needed. Companions use about 20 different messages focused on effort and growth mindset (Table 2).

To date, MathSpring learning companions have provided positive significant effects for the overall population of students and were more effective for lower achieving students and for female students in general [2]. However, characters seemed to have been harmful to some students (e.g., high-achieving males), who had higher affective baselines at pretest time and seem to have been distracted by the characters. These results suggest that affective characters should probably be different for students who are not presently frustrated or anxious (often high achieving males). One possibility is that the behavior of the characters be adaptive to the affective state of the student.

1.2 Recognize and Respond to Affect

Previously, we evaluated the hypothesis that **tailored affective messages delivered by digital animated characters may positively impact students emotions, attitude, and learning performance**. Specifically, we identified concrete prescriptive principles about how to respond to student emotion as it occurs during online learning [1, 3]. With models of student emotion, we explored mechanisms to address negative emotions. Our models predict confidence, interest, frustration, and excitement in real-time, based on data from hundreds of students. The gold standard was students' self-reported responses to questions, such as "How confident do you feel right now?"

We found that **growth mindset messages** based on Dweck's theory [9] provide an apparent **boost in student math learning** [3], resulted in **less performance-oriented goals** (e.g., beating classmates, instead of a self-referenced focus), and **less boredom** reported on the posttest. Typically online educational systems only report correctness: "Your answer is correct/incorrect." We discovered that such **success/failure** messages are correlated to higher reported **anxiety** and **boredom**, and appear to increase **performance-oriented goals**[3]. Other results indicate that empathic characters can help decrease students' anxiety and boredom. Our results showed that: a) student anxiety and boredom can be reduced using simple 2D characters, as did D'Mello et al., (2007); b) these benefits are due primarily to empathy, and secondarily to growth mindset messages; and c) indicating only success or failure is actively **harmful** to students, in comparison to emphasizing the learning process and the importance of effort.

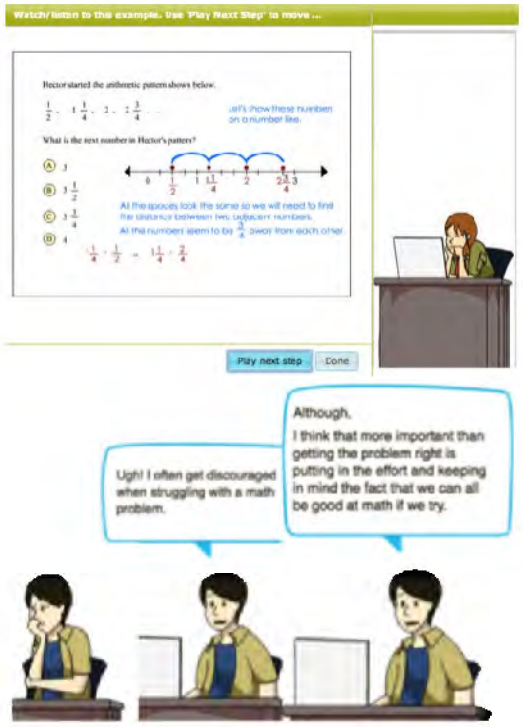


Figure 1: Learning companions respond to student actions with gestures and messages shown both as text and audio. *Above:* Companion shows high interest while the student views an example problem with solution steps shown. *Below:* Companion provides a growth mindset message, encouraging the student to put in effort to become good at math.

1.3 Research Goals

The research questions in this paper focus on identifying messages that support students' motivation to persist working on a task. Which messages (see Table 2) should a tutoring system send to students to encourage them to persist? How should agents respond to negative emotions? Should students be praised when they do well? Are the benefits to student learning and emotion due to empathic or motivational aspects of the companion? What are the results on learning and emotion of using an empathic or less empathic companion in comparison to a companion that indicates only success or failure?

Table 1: Outcomes variables measured in the experiment. The questions on the pre- and posttest were answered in a 5-point scale, going from "not at all" to "very much".

Interest - Students' interest in math. "Are you interested when solving math problems?"
Excitement - How exciting students find math. "Do you feel that solving math is exciting?"
Confusion - How confused students feel while solving math problems. "Do you feel confident that you will eventually be able to understand the Mathematics material?"
Frustration - How frustrating students find math. Average of "Do you get frustrated when solving math problems?" and "Does solving math problems make your feel frustrated?"
Learning Orientation - How much students focus on learning as opposed to performance. Average of "When you are doing math exercises, is your goal to learn as much as you can?" and "Do you prefer learning about things that make you curious even if that means you have to work harder?"
Performance Approach Goals - "Do you want to show that you are better at math than your classmates?"
Math Value - How important do students think math is. "Compared to most other activities, how important is it or you to be good at math?"
Math Liking - Measure of how much students like math. "Do you like your math class?"
Math Test Performance - Student's score on math questions that are representative of the content covered in MathSpring.

2. METHOD

We conducted a randomized controlled study to evaluate three different types of affective messages delivered by pedagogical agents (Table 2). The study took place in an urban school district in Southern California with sixty-four 6th grade students in three math classes for four class sessions, during December 2016. On part of the first and last day, students completed a pretest and posttest including questions related to various affective states, and questions about mathematics. Outcome variables measured from these questions are provided in Table 1.

Three conditions of learning companion messages were randomly assigned to students and delivered in both audio and written form in order to increase the likelihood of exposure: 1) **Empathy Condition** for 24 students, 2) **Growth Mindset Condition** for 20 students and 3) **Success/Failure Condition** for 20 students; see Table 2 for examples of the different types of messages. For all conditions, students were asked to self-report their frustration or confidence in a five-point scale every five minutes or every eight problems, which ever came first, but only after a problem was completed. The prompts were shown on a separate screen and invited students to report on their frustration or confidence.

The **Empathy** condition was set to visually reflect positive emotion with a certain probability for each math problem if the last student emotion report had a positive valence. When the most recent emotion report had a negative valence, and with a certain probability, the character first visually reflected the negative emotion; then it reported an empathy message such as "Sometimes these problems make me feel [frustrated]", and finally a connector such as "on the other hand", connected with a growth mindset message such as "I know that putting effort into problem solving and learning from hints will make our intelligence grow." Note that only students experiencing negative emotions were exposed to growth mindset messages, as opposed to the following condition.

The **Growth Mindset** condition emphasized messages that accentuate the importance of effort and perseverance in achieving success. The growth mindset condition was set to provide one of many growth mindset messages after a second incorrect attempt was made (the first incorrect attempt caused the hint button to flash), regardless of students' emotions. This condition also provided occasional growth mindset messages at the beginning of a new problem.

The **Success/Failure** condition provided both traditional success/failure messages and some more basic meta-cognitive support for when students made mistakes (e.g., acknowledging that their answer was not correct while encouraging them to use a hint). The success/failure condition provided students with a response if they answered a problem correctly and also after they made a second mistake.

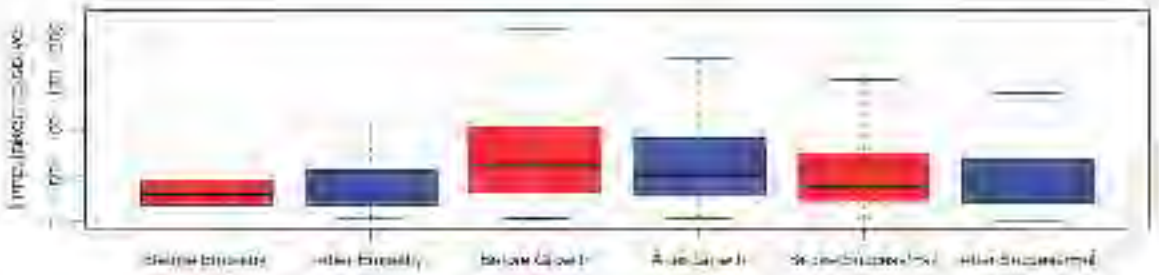
3. RESULTS

Out of the 64, three students' data were discarded due to minimal interaction with MathSpring. Across the $N = 61$ students, 21066 event log rows were recorded for three classes over four separate days, from which several behavioral features were derived and used throughout the analysis; our data and processing scripts can be found on GitHub [15]. All the students completed a pretest and posttest. Students in empathy, growth mindset and success/failure conditions received a total of 978, 763, and 882 messages respectively. Means, standard deviations and percentage shares for each type of message are given in Table 3. It is important to note that students received messages from all categories but their condition emphasized the corresponding message type. For example, a student in growth mindset condition received significantly more growth mindset messages than a student in empathy condition. This distribution of messages means that different students saw different amounts of each type of message, which allows us to perform partial correlations with respect to the counts of each message type, separating their effects.

Table 2: Examples of messages spoken by characters.

Condition	Message
Empathy	"Don't you sometimes get frustrated trying to solve math problems? I do. But guess what. Keep in mind that when you are struggling with are new idea or skill you are learning something and becoming smarter."
Growth Mindset	"Hey, congratulations! Your effort paid off, you got it right!" "Did you know that when we practice to learn new math skills our brain grows and gets stronger?" "Let's click on help, and I am sure we will learn something."
Success/ Failure	"Very good, we got another one right!" "Hmm. Wrong. Shall we work it out on paper?"

Figure 2: Time spent on a problem immediately before and after receiving the different categories of messages.



3.1 Partial Correlations

First, we attempted to replicate the results of our previous exploratory work [3]. For the three message types, partial correlations of the total number of each messages were measured for the nine posttest measures, controlling for the corresponding pretest measure, time spent in the tutor, and message frequency (total messages heard / time spent).

Table 4 shows the result of this analysis. We observe that with exposure to more **empathic** messages, students exhibited **higher levels of interest** and **valued math knowledge more** (rows 1 and 7). Increased interest can be viewed as analogous to the high negative correlation with boredom reported in our earlier work. With **growth mindset** messages, students **valued math knowledge less** and had **lower post test performance scores** (rows 7 and 9). With **success/failure** messages, students were **less learning-oriented** and claimed to be **more confused** (rows 6 and 3).

To further understand the dynamics, we derived some intutor variables and performed partial correlations shown in Table 5. The data for this analysis was derived as per student metrics based on their interaction with MathSpring. We observed that students tend to answer significantly more questions when in the **success/failure** condition and end up making more mistakes as well (rows 4 and 5). It is important to note that they also **avoid asking for hints** (row 6). It seems like these students tend to rush through the problems while being more careless. They also make **more mistakes** when they receive more growth mindset messages (row 5). This leads to simpler questions which they tend to get right in the first attempt (row 1). It appears that the students in **empathy** condition continue to **invest more time** on solving problems than rushing through the problem set. The number of problems seen by these students is significantly less (row 4).

As we see in Figure 2, students tend to spend less time on problems immediately after they receive growth mindset or success/failure messages. In contrast, the time spent on a problem increases slightly after receiving empathic messages. Students who received more empathic and growth mindset messages tend to answer fewer questions than do students who received mostly success/failure message (Figure 3). Combined with the last plot, it looks like the students in the empathy condition continue to invest more time on solving problems than rushing through the problem set.

Figure 3: Problems seen per minute across different pedagogies

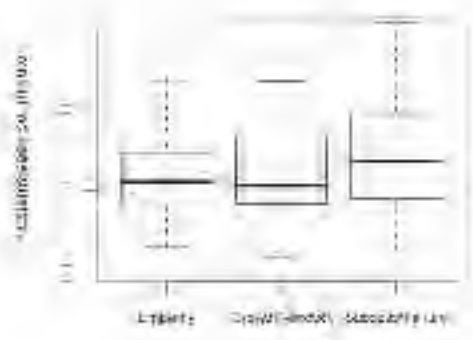


Table 3: The distribution of messages seen by students in each pedagogical conditions.

Condition	N	Empathy Messages			Growth Mindset Messages			Success/Failure Messages		
		mean	std	%	mean	std	%	mean	std	%
Empathy	21	7.48	7.0	16%	9.95	7.2	21%	29.1	22	62%
Growth Mindset	20	0.2	0.5	0.5%	10	5	26%	27.9	19.2	73%
Success/ Failure	20	1.2	1.7	2.7%	4.6	4.8	10%	38.3	26.6	86%

Table 4: Partial correlations between different types of messages seen and posttest variables (Table 1), accounting for the corresponding pretest value, time spent in tutor and message frequency.

	Variable	Empathy Messages		Growth Mindset Messages		Success/Failure Messages	
		corr	p	corr	p	corr	p
(1)	**Interest**	**0.28***	0.03	0.19	0.15	-0.20	0.14
(2)	**Excitement**	0.00	1.00	-0.07	0.60	-0.08	0.54
(3)	**Confusion**	-0.05	0.74	-0.05	0.74	**0.32***	0.02
(4)	**Frustration**	0.10	0.43	-0.08	0.54	-0.18	0.18
(5)	**Performance Approach**	-0.19	0.14	-0.05	0.70	0.20	0.12
(6)	**Learning Orientation**	0.02	0.85	0.02	0.88	**-0.24**$^{+}$	0.06
(7)	**Math Value**	**0.25***	0.05	**-0.22**$^{+}$	0.09	-0.10	0.45
(8)	**Math Liking**	0.01	0.96	0.01	0.96	0.05	0.72
(9)	**Performance**	-0.01	0.93	**-0.23**$^{+}$	0.07	-0.13	0.33

$^{+}$ $p \leq 0.10$, * $p \leq 0.05$

Table 5: Partial correlations between different types of messages seen and within-tutor variables, accounting for time spent in the tutor and message frequency.

	Variable	Empathy Messages		Growth Mindset Messages		Success/Failure Messages	
		corr	p	corr	p	corr	p
(1)	**% Problems Solved on First Attempt**	0.06	0.62	**0.34****	0.007	-0.01	0.94
(2)	**Avg Problem Difficulty**	0.07	0.61	-0.05	0.69	0.19	0.14
(3)	**Learning Gain**	-0.10	0.50	-0.07	0.63	-0.14	0.34
(4)	**Problems Seen**	**-0.23**$^{+}$	0.07	-0.04	0.78	**0.77****	4E-13
(5)	**Mistakes Made**	-0.01	0.91	**0.59****	6E-7	**0.30***	0.02
(6)	**Hints Per Problem**	0.10	0.43	0.16	0.22	**-0.22**$^{+}$	0.10

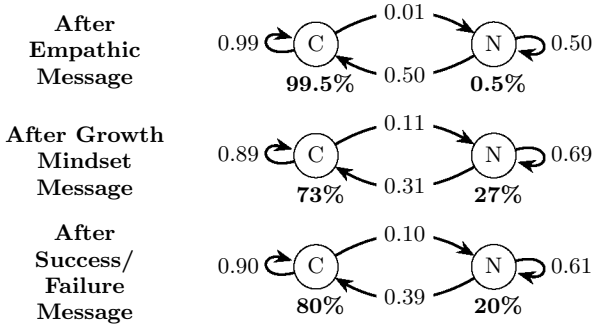
$^{+}$ $p \leq 0.10$, * $p \leq 0.05$, ** $p \leq 0.01$

3.2 Markov Chain Analysis

As students solve problems in the tutoring system, the learning companion comments on their attempts; the effect of these messages on student affect is sequential, but the partial correlations do not capture this. To analyze this effect, we built Markov Chain models using in-tutor student self-reports of confidence and frustration. Each model describes transitions in affective states, from one self-report to the next, where students received a particular type of character messages (empathy, growth mindset, and success/failure) between self-reports. To reduce the state space, the 5-point scale used in the self-reports was simplified to two values - confident (≥ 3), not confident (< 3); similarly for frustration.

The goal of the Markov models was not to predict emotional changes, but rather to examine whether different messages had significant effects on affect. Markov models can show the probability of transitioning between affective states, but also have a stationary distribution, which represents the amount of students that would be in each state after undergoing many transitions. For example, a group of students were to use the system for many hours and receive only empathic messages, our model suggests that 99.5% of them would be confident about learning math (Figure 4).

Figure 4: State transitions between the Confident (C) and Not Confident (N) affective states. The stationary distribution is shown below each state. Only the empathy model was significant in the likelihood ratio test ($p \leq 0.05$)



We used a likelihood ratio test to analyze the significance of these models: the probability of the null model (ignoring message type) generating the data divided by the probability of the alternate model (for a particular message type) generating the data gives a p-value. Figure 4 shows the state transitions for **confidence** in the null model and the model for confidence after receiving **empathic** messages, which was significant with $p = 0.0149$ (the other models were not significant). We also examined the stationary distributions for each model (Table 6).

Table 6: Stationary distributions in the Markov models of confidence and frustration.

Message Type	Confidence		Frustration	
	Conf	Not	Frust	Not
Empathy	**99.5%***	**0.05%***	35%	65%
Growth Mindset	74%	26%	30%	70%
Success/ Failure	80%	20%	25%	75%

$^*p \leq 0.05$

4. DISCUSSION

Some of our results support the hypothesis that affective messages delivered by characters can positively impact students' emotions and affective predispositions for math problem solving. This is particularly evident for empathy, as the more empathic messages a student saw the higher their interest in mathematics problem solving, as well as their beliefs that mathematics is valuable to learn (Table 4). An analysis of student behavior suggests that students who saw a high frequency of empathic messages also tended to be more patient and cautious with problem solving, suggesting that empathic messages may encourage students to persist through adversity. Exposure to empathic messages was significantly correlated to investing time in each math problem activity, leading also to fewer problems seen per session. A positive trend is exhibited between high frequency of empathic messages and hints requested, even if not significant (Table 5). Empirical temporal models generated from students' changes in self-reports of affect, within the tutor, revealed that students receiving empathic messages have a higher likelihood to become more confident and to remain confident.

The response to growth mindset messages delivered by characters yielded mixed results. As students saw more of these kinds of messages they also succeeded more often at solving problems correctly (on the first attempt); interestingly, at the same time, they also made more mistakes. This is also desirable, as growth mindset messages emphasize that making mistakes is okay and can even help learning, legitimizing a high frequency of errors. It is possible that students were using those mistakes and hints to learn and succeed later on; a (not significant) positive trend suggests that students receiving more of these kinds of messages also asked for more hints per problem. In contrast, marginally significant effects suggest that a high frequency of growth mindset messages might be detrimental to students' perception of math value, and that their posttest performance is worse when they receive more of this kind of messages. It is hard to conclude the meaning of these marginally significant effects, especially because a previous study suggested that these messages were beneficial in general [3]. Note that empathic messages used 'growth mindset' messages also, in order to resolve the negative emotion (see Table 2). One possible explanation is that the empathic condition was so positive because it was also very selective at showing growth mindset messages for only those who experienced negative emotions. It is likely that high achieving students, or those who "felt OK", rejected growth mindset messages that they might have perceived to be unnecessary.

An important comment is that we did not expect that success/failure messages could be so harmful to students. Regardless of whether messages indicated success or failure, as students received more of these messages they also exhibited lower levels of mastery/learning orientation at posttest time. They also reported higher levels of confusion at posttest time (note that the confusion can be positive for learning within the learning experience, but not after the learning experience has concluded). Regarding behavior within the tutor, the more students were exposed to success/failure messages, the more they appeared to rush through problems, make mistakes, and request fewer hints per problem.

To summarize, empathy messages were associated with variables consistent with methodical work and an increased interest/value of mathematics. However, both growth mindset and success/failure messages appeared to be associated with a greater number of mistakes. Finally, success/failure messages themselves were associated with a whole host of concerning behaviors such as confusion with the material following posttest, reduced learning orientation, hurried work, and a reduced likelihood of requesting hints. This is consistent with Dweck's findings that growth mindset messages are superior to success/failure messages [8, 9]. Whether empathic messages in fact result in improved student performance pre to posttest will likely require larger samples than this small study (N = 61). However, students in non-empathic conditions have demonstrated significantly more mistakes in their work.

5. CONCLUSIONS

This research emphasizes the importance of understanding an intervention's effect on a student's affective state, which in turn is connected to engagement, performance, and learning. Although many researchers have focused on modeling affect, very little research effort has been put into systematically measuring the impact of the intervention on the student behavior in an adaptive learning environment. Empathic messages that respond to students' recent emotions have resulted in superior results both in improving the student interaction with the system and in the overall learning experience. Growth Mindset follows next with some positive impact on in-tutor performance but its overall effect in the short-term is questionable. Success/Failure messages are generally harmful to students: reducing learning orientation, increasing confusion, and making students more careless during the learning experience.

We conclude that affective messages delivered by characters in online tutoring environments are a very important medium for building student-tutor rapport in a virtual environment, powerful signals that influence perceptions of students themselves as learners, as well as perceptions of the domain being taught. We have reported significant results that support the use of empathy to improve student affect and attitudes in a math tutor. The long-term effect of these messages needs to be studied when the novelty of this intervention wears off. In the future, we hope to study the impact of the frequency and content of these messages. To evaluate the generalizability of these results, student populations across different demographics needs to be studied as well as the applicability of the messages to domains beyond mathematics.

6. ACKNOWLEDGMENTS

This research is supported by the National Science Foundation (NSF) 1324385 IIS/Cyberlearning DIP: Collaborative Research: Impact of Adaptive Interventions on Student Affect, Performance, and Learning. Any opinions, findings, and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

7. ADDITIONAL AUTHORS

Additional authors: Winslow Burleson (New York University, 70 Washington Square South New York, New York, 10012; email: wb50@nyu.edu).

8. REFERENCES

[1] I. Arroyo, W. Burleson, M. Tai, K. Muldner, and B. P. Woolf. Gender differences in the use and benefit of advanced learning technologies for mathematics. *Journal of Educational Psychology*, 105(4):957, 2013.

[2] I. Arroyo, D. G. Cooper, W. Burleson, B. P. Woolf, K. Muldner, and R. Christopherson. Emotion sensors go to school. In *AIED*, volume 200, pages 17–24, 2009.

[3] I. Arroyo, S. Schultz, N. Wixon, K. Muldner, W. Burleson, and B. P. Woolf. Addressing affective states with empathy and growth mindset. *6th International Workshop on Personalization Approaches in Learning Environments*, 2016.

[4] I. Arroyo, B. P. Woolf, W. Burelson, K. Muldner, D. Rai, and M. Tai. A multimedia adaptive tutoring system for mathematics that addresses cognition, metacognition and affect. *International Journal of Artificial Intelligence in Education*, 24(4):387–426, 2014.

[5] L. Corno and R. E. Snow. Adapting teaching to individual differences among learners. *Handbook of research on teaching*, 3(605-629), 1986.

[6] S. D'Mello and A. Graesser. Automatic detection of learner's affect from gross body language. *Applied Artificial Intelligence*, 23(2):123–150, 2009.

[7] S. D'Mello and A. Graesser. Autotutor and affective autotutor: Learning by talking with cognitively and emotionally intelligent computers that talk back. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(4):23, 2012.

[8] C. S. Dweck. *Self-theories: Their role in motivation, personality, and development*. Psychology Press, 2000.

[9] C. S. Dweck. Beliefs that make smart people dumb. *Why smart people can be so stupid*, 24:41, 2002.

[10] D. Goleman. Emotional intelligence. why it can matter more than fq. *Learning*, 24(6):49–50, 1996.

[11] A. C. Graesser, P. Chipman, B. C. Haynes, and A. Olney. Autotutor: An intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, 48(4):612–618, 2005.

[12] A. C. Graesser, S. K. D'Mello, S. D. Craig, A. Witherspoon, J. Sullins, B. McDaniel, and B. Gholson. The relationship between affective states and dialog patterns during interactions with autotutor. *Journal of Interactive Learning Research*, 19(2):293, 2008.

[13] J. Hattie and H. Timperley. The power of feedback. *Review of educational research*, 77(1):81–112, 2007.

[14] A. N. Kluger and A. DeNisi. Feedback interventions: Toward the understanding of a double-edged sword. *Current directions in psychological science*, 7(3):67–72, 1998.

[15] R. Lizarralde and S. Karumbaiah. A collection of scripts for processing mathspring data. `https://github.com/rezecib/MathspringDataProcessing`, 2017.

[16] R. Pekrun. Emotions and learning. *International Academy of Education. Australia: International Bureau of Education*, 2014.

[17] R. Pekrun, A. Cusack, K. Murayama, A. J. Elliot, and K. Thomas. The power of anticipated feedback: Effects on students' achievement goals and achievement emotions. *Learning and Instruction*, 29:115–124, 2014.

[18] R. Pekrun, T. Goetz, L. M. Daniels, R. H. Stupnisky, and R. P. Perry. Boredom in achievement settings: Exploring control-value antecedents and performance outcomes of a neglected emotion. *Journal of Educational Psychology*, 102(3):531, 2010.

[19] R. Pekrun, E. Vogl, K. R. Muis, and G. M. Sinatra. Measuring emotions during epistemic activities: the epistemically-related emotion scales. *Cognition and Emotion*, pages 1–9, 2016.

Epistemic Network Analysis and Topic Modeling for Chat Data from Collaborative Learning Environment

Zhiqiang Cai
The University of Memphis
365 Innovation Drive, Suite 410
Memphis, TN, USA
zcai@memphis.edu

Brendan Eagan
University of Wisconsin-Madison
1025 West Johnson Street
Madison, WI, USA
eaganb@gmail.com

Nia M. Dowell
The University of Memphis
365 Innovation Drive, Suite 410
Memphis, TN, USA
niadowell@gmail.com

James W. Pennebaker
University of Texas-Austin
116 Inner Campus Dr Stop G6000
Austin, TX, USA
pennebaker@utexas.edu

David W. Shaffer
University of Wisconsin-Madison
1025 West Johnson Street
Madison, WI, USA
dws@education.wisc.edu

Arthur C. Graesser
The University of Memphis
365 Innovation Drive, Suite 403
Memphis, TN, USA
art.graesser@gmail.com

ABSTRACT

This study investigates a possible way to analyze chat data from collaborative learning environments using epistemic network analysis and topic modeling. A 300-topic general topic model built from TASA (Touchstone Applied Science Associates) corpus was used in this study. 300 topic scores for each of the 15,670 utterances in our chat data were computed. Seven relevant topics were selected based on the total document scores. While the aggregated topic scores had some power in predicting students' learning, using epistemic network analysis enables assessing the data from a different angle. The results showed that the topic score based epistemic networks between low gain students and high gain students were significantly different ($t = 2.00$). Overall, the results suggest these two analytical approaches provide complementary information and afford new insights into the processes related to successful collaborative interactions.

Keywords

chat; collaborative learning; topic modeling; epistemic network analysis

1. INTRODUCTION

Collaborative learning is a special form of learning and interaction that affords opportunities for groups of students to combine cognitive resources and synchronously or asynchronously participate in tasks to accomplish shared learning goals [15; 20]. Collaborative learning groups can range from a pair of learners (called a dyad), to small groups (3-5 learners), to classroom learning (25-35 learners), and more recently large-scale online learning environments with hundreds or even thousands of students [5; 22]. The collaborative process provides learners with a more efficient learning experience and improves learners' collaborative learning skills, which are critical competencies for students [14]. Members in a team are different in many ways. They have their own experience, knowledge, skills, and approaches to learning. A student in a collaborative learning environment can take other students' views and ideas about the information provided in the learning material. The ideas coming out of the team can then be integrated as a deeper understanding of the material, or a better solution to a problem.

Traditional collaborative learning occurred in the form of face to face group discussion or problem solving. As the internet and learning technologies develop, online collaborative learning environments come out and are playing more and more important roles. For example, MOOCs (Massive Open Online Courses) have drawn massive number of learners. Learners in MOOCs are connected by the internet and can easily interact with each other using various types of tools, such as forums, blogs and social networks [23]. These digitized environments make it possible to track the learning processes in collaborative learning environments in greater detail.

Communication is one of the main factors that differentiates collaborative learning from individual learning [4; 6; 9]. As such, chats from collaborative learning environments provide rich data that contains information about the dynamics in a learning process. Understanding massive chat data from collaborative learning environments is interesting and challenging. Many tools have been invented and used in chat data analysis, such as LIWC (linguistic inquiry and word count) [12], Coh-Metrix [10], and topic modeling, just to name a few. Epistemic network analysis (ENA) has been playing a unique role in analyzing chat data from epistemic games [18]. ENA is rooted in a specific theory of learning: the epistemic frame theory, in which the collection of skill, knowledge, identity, value and epistemology (SKIVE) forms an epistemic frame. A critical theoretical assumption of ENA is that the connections between the elements of epistemic frames are critical for learning, not their presence in isolation. The online ENA toolkit allows users to analyze chat data by comparing the connections within the epistemic networks derived from chats. ENA visualization displays the clustering of learners and groups and the network connections of individual learners and groups. ENA requires coded data which has traditionally relied on hand coded data sets or classifiers that rely on regular expression mapping. Combining topic modeling with ENA will provide a new mode of preparing data sets for analysis using ENA.

In this study, we used a combination of topic modeling and ENA to analyze chat data to see if we could detect differences between the connections made by students with high learning gains versus students with low learning gains. Incorporating topic modeling

with ENA will make the analytic tool more fully automated and of greater use to the research community.

2. RELATED WORK

Chats have two obvious features. First, they appear in the form of text. Therefore, any text analysis tool may have a role in chat analysis. Second, chats come from individuals' interaction, which reflects social dynamics between participants. Therefore, a combination of text analysis and social network analysis should be helpful in understanding underlying chat dynamics. For instance, Tuulos et al. [21] combined topic modeling with social network analysis in chat data analysis. They found that topic modeling can help identify the receiver of chats (the person who a chat is given to).

In a similar effort, Scholand et al. [16] combined LIWC and social network analysis to form a method called "social language network analysis" (SLNA). The social networks were formed by counting the number of times chat occurred between any two participants. Based on the counts, participants were clustered into a tree structure, representing the level of subgroups the participants belong to. LIWC was then used to get the text features of chats. It was found that, some LIWC features were significantly different between in group conversations and out of group conversations.

Researchers have also recently explored the advantages of combining SNA (social network analysis) with deeper level computational linguistic tools, like Coh-Metrix. Coh-Metrix computes over 100 text features. The five most important Coh-Metrix features are: narrativity, syntax simplicity, word concreteness, referential cohesion and deep cohesion. Dowell and colleagues [8] explored the extent to which characteristics of discourse diagnostically reveals learners' performance and social position in MOOCs. They found that learners who performed significantly better engaged in more expository style discourse, with surface and deep level cohesive integration, abstract language, and simple syntactic structures. However, linguistic profiles of the centrally positioned learners differed from the high performers. Learners with a more significant and central position in their social network engaged using a more narrative style discourse with less overlap between words and ideas, simpler syntactic structures and abstract words. An increasing methodological contribution of this work highlights how automated linguistic analysis of student interactions can complement social network analysis (SNA) techniques by adding rich contextual information to the structural patterns of learner interactions.

In another study, Dowell et al. [7] showed that students' linguistic characteristics, namely higher degrees of narrativity and deep cohesion, are predictive of their learning. That is, students engaged in deep cohesive interactions performed better.

In the present research, we explore collaborative interaction chat data using the combination of topic modeling and epistemic network analysis. While previous studies focused on the relationship between language features and social network connections, our study focuses on prediction learning performance by semantic network connections students make in chats.

3. METHODS

Participants. Participants were enrolled in an introductory-level psychology course taught in the Fall semester of 2011 at a large university in the USA. While 854 students participated in this course, some minor data loss occurred after removing outliers and those who failed to complete the outcome measures. The final sample consisted of 844 students. Females made up 64.3% of this final sample. Within the population, 50.5% of the sample identified as Caucasian, 22.2% as Hispanic/Latino, 15.4% as Asian American, 4.4% as African American, and less than 1% identified as either Native American or Pacific Islander.

Course Details and Procedure. Students were told that they would be participating in an assignment that involved a collaborative discussion on personality disorders and taking quizzes. Students were told that their assignment was to log into an online educational platform specific to the University at a specified time, where they would take quizzes and interact via web chat with one to four random group members. Students were also instructed that, prior to logging onto the educational platform, they would have to read material on personality disorders. After logging into the system, students took a 10 item, multiple choice pretest quiz. This quiz asked students to apply their knowledge of personality disorders to various scenarios and to draw conclusions based on the nature of the disorders. The following is an example of the types of quiz questions students were exposed to:

- *Jacob was diagnosed with narcissistic personality disorder. Why might Dr. Simon think this was the wrong diagnosis?*
- *Dr. Level has measured and described his 10 mice of varying ages in terms of their length (cm) and weight (g). How might he describe them on these characteristics using a dimensional approach?*
- *Danielle checks her facebook page every hour. Does Danielle have narcissistic personality disorder?*

After completing the quiz, they were randomly assigned to other students who were waiting to engage in the chatroom portion of the task. When there were at least 2 students and no more than 5 students (M = 4.59), individuals were directed to an instant messaging platform that was built into the educational platform. The group chat began as soon as someone typed the first message and lasted for 20 minutes. The chat window closed automatically after 20 minutes, at which time students took a second 10 multiple-choice question quiz. Each student contributed 154.0 words on average (SD = 104.9) in 19.5 sentences (SD = 12.5). As a group, discussions were about 714.8 words long (SD = 235.7) and 90.6 sentences long (SD = 33.5).

An excerpt of a collaborative interaction chat in a chat room is shown below in Table 1. (student names have been changed):

Table 1. An excerpt of a collaborative interaction chat

Student	*Chat Text*
Art	ok cool, everyone's here. sooo first question
Art	ok so the certain characteristics to be considered to have a personality disorder?
Shaffer	Alright sooo first question: Based on these criteria describe several reasons why a psychologist might not label someone with grandiose thoughts as having narcissistic personality disorder?
Shaffer	hahaha never mind
Shaffer	that was the second question.
Art	lol its all good
Shaffer	okay so certain characteristics: doesn't it have to be like a stable thing?
Carl	i think the main thing about having a disorder is that its disruptive socially and/or makes the person a danger to himself or others

Vasile	yes, stable over time
Shaffer	yeah, and it also mentioned it can't be because of drugs
Art	also they have to have like unrealistic fantasies
Nia	yeah and not normal in their culture
Carl	no drugs or physical injury
Vasile	begins in early adulthood or adolescence
Shaffer	i think that covers them? haha
Art	ok, so arrogance doesn't just define it, they have to have most of these characteristics
Art	yeah i think we got them
Shaffer	is it most or is it like 6?

From the above excerpt, we can see several obvious things. First, the lengths of the utterances varied from one single word to multiple sentences. This needs to be considered in text analysis because some methods work only for longer texts. For example, Coh-Metrix usually works well for texts with more than 200 words. Topic modeling also needs enough length to reliably infer topic scores. Second, the number of utterances each participant gave were different. From how much and what a member said, we can see each member played a different role in that chat. Third, the ordered sequence of the utterances forms a time series. Understanding and visualizing the underlying discourse dynamics are important for meaning making with this type of data.

The data set contained 15,670 utterances, pretest scores (the first quiz) and post test scores (the second quiz) for 844 students, grouped in 182 chat rooms. Each chat room had 2 to 5 students, 4.73 by average. The average speech turns each student gave was 18.2 and the average speech turns in each room was 86.1.

The average pretest score was 36.01% correct and the average post-test scores 45.73% correct. Paired sample test shows that the post-test is significantly higher ($t = 14.13, N = 844$). We computed the learning gain of each student, using the formula

$$gain = \frac{posttest\ score - pretest\ score}{1 - pretest\ score}.$$

For all students ($N = 844$), the average learning gain is 0.11, 59.5% had positive learning gains above 0.1. 16.5% had the same scores and 23% had negative learning gains. Not surprisingly, students who had lower pretest scores had higher learning gains because they had greater potential to learn. Figure 1 shows the average learning gain as function of pretest score.

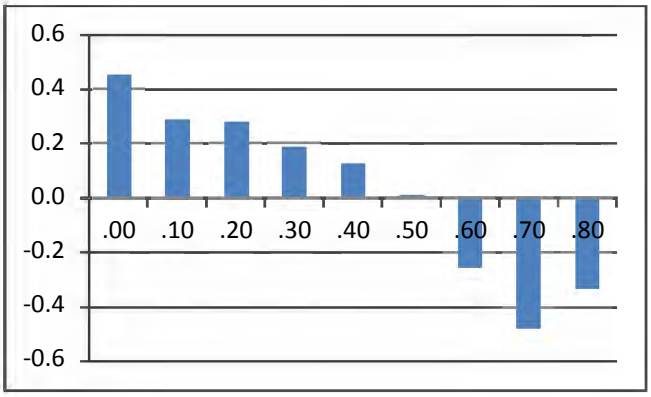


Figure 1. Average learning gain as a function of pretest score.

For students with pretest scores less than 50% correct (N=624), the average learning gain is 0.88, 69.7% had positive learning gains, 15.7% had the same scores and 14.6% had negative learning gains.

This data set has been analyzed in multiple studies. Cade et al. [3] analyzed the cohesion of the chats and found that deep cohesion of the chats predicts the students feeling of power and connectedness to the group. Dowell et al. [7] found that some Coh-Metrix measures predicts learning. Coh-Metrix measures describe common textual features that are not content specific. For example, cohesion is about how text segments are semantically linked to each other, which has nothing to do with what the text content is about. In this study, we use topic modeling to provide content dependent features and use epistemic network analysis to explore how the topics were associated in the chats.

4. TOPIC MODELING

Topic modeling has been widely used in text analysis to find what topics are in a text and what proportion/amount of each topic is contained. Latent Dirichlet Allocation (LDA) [2; 24] is one of the most popular methods for topic modeling. LDA uses a generative process to find topic representations. LDA starts from a large document set $D = \{d_1, d_2, \cdots, d_m\}$. A word list $W = \{w_1, w_2, \cdots, w_n\}$ is then extracted from the document set. LDA assumes that the document set contains a certain number of topics, say, *K* topics. Each document has a probability distribution over the *K* topics and each topic has a probability distribution over the given list of words. When a document was composed, each word that occurred in a document was assumed to be drawn based on the document-topic probability and the topic-word probability. For a given corpus (document set) and a given number of topics *K*, LDA can compute the topic assignment of each word in each document.

For a given topic, the word probability distribution can be easily computed from the number of times each word was assigned to the given topic. The beauty of topic modeling is that the "top words" (words with highest probabilities in a topic) usually give a meaningful interpretation of a topic. The distributions are the underlying representation of the topics. The top words are usually used to show what topics are contained in the corpus.

By counting the number of words assigned to each topic, a topic proportion score can be computed for each document on each topic. The topic proportion scores then become a document feature that can be used in further analysis. However, the proportion scores are based on the statistical topic assignment of words. When documents are very short, such as most utterances in our chat data, the topic proportion scores won't be reliable. Cai et al. [4] argued that alternative ways to compute document topic scores are possible.

4.1 TASA Topic Model

Although our chat data set contained 15,670 utterances, the utterances were short and the corpus is not large enough to build a reliable topic model. To get a reliable model, we used a well known corpus provided by TASA (Touchstone Applied Science Associates). This corpus contained documents on seven known categories, including business, health, home economics, industrial arts, language arts, science and social studies. Our content topic, personality disorders, is obviously in the health category. Of course, not all topics in TASA are relevant to our study. Therefore, after building up the model, we need to select relevant topics. We will cover that in the next sub-section.

There are a total of 37,651 documents in TASA corpus, each of which is about 250 words long. Before we ran LDA, we filtered out very high frequency words and very low frequency words. High frequency words, such as "the", "of", "in", etc., won't contain much topic information. Rare words won't contribute to meaningful statistics. 28,483 words (it might be better to say "terms") were left after filtering. A model with 300 topics was constructed by LDA.

4.2 Topic score computation and topic selection

From the TASA topic model, we computed the word-topic probabilities based on the number of times a word was assigned to each of the 300 topics. Thus, each word is represented by a 300 dimensional probability distribution vector. For each chat in our chat corpus, we simply summed up the word probability vectors for the words appeared in each chat. That gave us 300 topic scores for each chat. Recall that, the chats were associated with a reading material and two quizzes. While the students were free to talk about anything, the content of the reading material and the quizzes set up the main chat topics, that is, personality disorders.

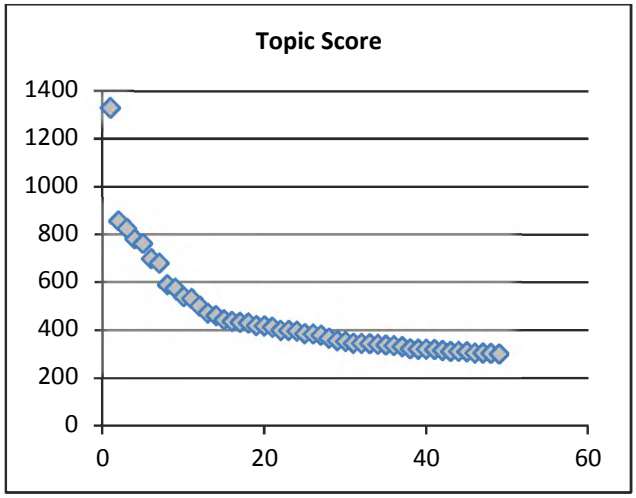


Figure 2. Sorted topic scores for topic selection.

The first thing we needed to do then was to investigate whether or not the "hot" topics from the computation made sense. To find that out, we computed the sum of all topic scores over all chats. The topics were sorted according the total topic score. The hottest topic had a total score higher than 1300, much higher than the second highest (less than 900). By examining the top words, this topic is about "illness", which is highly relevant to personality disorders. Six hot topics scored in the range from 600 to 900. They are about "outdoors", "biology", "people/social", "education" and "healthcare". The top words are listed below.

- **Illness:** health, disease, patient, body, diseases, medical, stress, mental, physical, heart, doctor, problems, cause, person, patients, exercise, illness, problem, nurse, healthy
- **Outdoors:** dog, energy, plants, earth, car, light, food, heat, words, animals, music, rock, language, children, air, uncle, city, sun, women, plant
- **Biology:** cells, cell, genes, chromosomes, traits, color, organisms, sex, egg, species, gene, body, male, female, parents, nucleus, eggs, sperm, organism, sexual
- **Psychology:** behavior, learning, theory, environment, feelings, sexual, physical, social, sex, human, research, person, animal, mental, response, positive, stress, personality, subject, reaction
- **People/Social:** joe, pete, mr, charlie, dad, frank, billy, tony, jerry, 'll, mom, 'd, going, 're, got, boys, looked, asked, paper, go
- **Education:** students, teacher, teachers, child, children, student, school, education, schools, learning, parents, tests, test, program, teaching, behavior, skills, reading, team, information
- **Healthcare:** patient, doctor, health, hospital, medical, dr, patients, nurse, disease, doctors, team, care, office, nursing, drugs, medicine, services, dental, diseases, help

"Illness", "biology", "psychology" and "healthcare" are the topics the learning materials involved. "Education" topic is about the education environment where the chat happened. "Outdoor" and "people/social" are off-task topics.

To get an idea about whether or not the topic scores were related to the learning gain, we aggregated the scores by person and computed the correlation between the total topic score and the learning gain for each topic. We were only interested in looking at the students with larger potential to learn, so we removed the data with pretest score greater than or equal to 0.5, leaving 624 students out of 844. The results (Table 1) showed that all topics were significantly correlated to learning gain. It doesn't seem to be great, because that seems to suggest that, whatever topic a student talked about, more a student talked, larger gain the student obtained. The real reason is that in the aggregation, all topic scores were summed up. Therefore, all topic scores were influenced by the chat length. So the correlation in Table 2 basically showed the chat length effect.

Table 2. Correlation between total topic scores and learning gain (N=624, pretest<0.5)

Topic	*Post-test*	*Pretest*	*Gain*
Illness	.183**	.116**	.132**
Outdoors	.216**	.133**	.154**
Biology	.159**	.125**	.105**
Psychology	.182**	.096*	.140**
People/Social	.115**	.022	.107**
Education	.175**	.118**	.121**
Healthcare	.157**	.130**	.097*

To remove the chat length effect, the simplest way is to divide all scores by the number of words (terms) in each chat. However, in this study, to be consistent with subsequent analysis, we normalized the topic scores to topic proportion scores by dividing each topic score for each utterance by the sum of all seven topic scores of the same utterance.

The results (Table 3) showed that the topic "people/social" had a significant negative correlation to learning gain. Others were not significant but were in the direction we would expect. "Illness", "biology", "psychology" and "healthcare" were positively correlated with gain scores, while "outdoors" and "people/social" topics were negatively correlated with gains scores. We observed almost no correlation for the "Education" topic. This seems to indicate that the aggregated topic scores have limited power in predicting learning. Therefore, we used ENA to examine the connections or association of these topics in the students discourse to

develop a predictive model of learning gains based on the use of these topics.

Table 3. Correlation between normalized topic proportion scores and learning gain (N=624, pretest<0.5)

Topic	*Post-test*	*Pretest*	*Gain*
Illness	.099*	0.077	0.067
Outdoors	-0.063	-0.043	-0.044
Biology	.085*	0.054	0.063
Psychology	0.067	0.019	0.058
People/Social	-.127**	-0.076	-.083*
Education	0.027	0.056	-0.002
Healthcare	0.073	.096*	0.027

5. EPISTEMIC NETWORK ANALYSIS

ENA measures the connections between elements in data and represents them in dynamic network models. ENA creates these network models in a metric space that enables the comparison of networks in terms of (a) difference graph that highlights how the weighted connections of one network differ from another; and (b) statistics that summarize the weighted structure of network connections, enabling comparisons of many networks at once.

ENA was originally developed to model cognitive networks involved in complex thinking. These cognitive networks represent associations between knowledge, skills, habits of mind of individual learners or groups of learners. In this study, we used ENA to construct network models. For each individual student, we constructed an ENA network using the selected seven topic scores for each utterance the student contributed to the group.

5.1 Process

While the process of creating ENA models is described in more detail elsewhere (e.g. [11; 17-19]), we will briefly describe how ENA models are created based on topic modeling. Here we defined network nodes as the seven topics identified from the topic model. We defined the connections between nodes, or edges, as the strength of the co-occurrence of topics within a moving stanza window (MSW) of size 5 [19]. To model connections between topics we used the products of the topic scores summed across all chats in the MSW. That is, for each topic, the topic scores are summed across all 5 chats in the MSW. Then ENA computed the product of the summed topic loadings for each pair topics to measure the strength of their co-occurrence. For example, if the sum of the topics scores across five chats was 0.5 for "illness", 0.3 for "psychology", and 0.2 for "healthcare", these scores would result in three co-occurrences, "illness-psychology", "illness-healthcare", and "psychology-healthcare", with scores of 0.15, 0.1, and 0.06, respectively.

Next ENA created adjacency matrices for each student that quantified the co-occurrences of topics within the students' discourse in the context of their chat group. Subsequently, the adjacency matrices were then treated as vectors in a high dimensional space, where each dimension corresponds to co-occurrence of a pair of topics. The vectors were then normalized to unit vectors. Notice that the normalization removed the effect of chat length embedded in the topic scores. A singular value decomposition (SVD) was then performed for dimensional reduction. ENA then projected a vector for each student into a low dimensional space that maximizes the variance explained in the data. Finally, the nodes of the networks, which in this case correspond to the seven selected topics generated from TASA corpus, were placed in the low dimensional space. The topic nodes were placed using an optimization algorithm such that the overall distances between centroids (centers of the mass of the networks) and the corresponding projected student locations was minimized. A critical feature of ENA is that these node placements are fixed, that is, the nodes of each network are in the same place for all units in the analysis. This fixing of the location of the nodes allows for meaningful comparisons between networks in terms of their connection patterns which allow us to interpret the metric space. As a result, ENA produced two coordinated representations: (1) the location of each student in a projected metric space, in which all units of analysis included in the model were located, and (2) weighted network graphs for each student, which explained why the student was positioned where it was in the space.

ENA also allows us to compare the mean network graphs and mean position in ENA space between different groups of students. In this study, we only considered the students with high potential to learn, i.e., the 624 students with pretest score < 0.5 (50% correct). Among these students, we compared the networks of low learning gain students (gain<-0.1, N=194) with the networks of high learning gain students (gain>0.43, N=105). We compared these groups using difference network graph, which was formed by subtracting the edge weights of the mean discourse network for the low gain group students from the mean discourse network from the high gain group. This difference network graph shows us which topic connections are stronger for each group. In addition, we conducted a *t*-test to test the difference between group means.

5.2 Results

Figure 3 shows mean discourse networks for students with low gain scores (left, red), students with high gain scores (right, blue), and a difference network graph (center) that shows how the discourse patterns of each group differs. Students with low gains had stronger connections between the "people/social" topic and all other topics except for "illness". More importantly, the connection that was the strongest for low gain students compared to high gain students was between "people/social" and "outdoors". Students with high gain scores made stronger connections between the topics of "illness", "psychology", "healthcare", "biology", and "education".

Table 4. Comparison of centroids between low gain and high gain students, $p = 0.047, t = 2.00$

	N	***Mean***	***SD***
High gain	105	0.033	0.220
Low gain	194	-0.048	0.322

Figure 4 shows centroids, or the centers of mass, of individual students' discourse networks and their means with low gain score students in red and high gain score students in blue. The differences between these two groups were significant on the x dimensions (see table 4). This means that the differences we saw in figure 2 and described above are statistically significant. In other words, the high learning gain students' discourse was more towards the right side of the ENA space and the low learning gain students' discourse was more towards the left side. That indicates that the discourse of students with high learning gains made more connections between on-task topics ("illness", "psychology", "healthcare", "biology", and "education"), while the discourse of

low gain students made more connections between off-task topics ("people/social" and "outdoors").

6. DISCUSSION

ENA makes it possible to visualize the chat dynamics to help researchers gain deeper understanding of what is going on in a collaborative learning environment. Differences in what topics students connect in discourse can predict learning outcomes. Previous use of ENA has relied on human coded data or use of regular expressions to classify data. Utilizing topic modeling can lead to fully automated ENA, making it more accessible to a wider group of researchers and allows ENA to be used with more and larger data sets.

The fact that the epistemic network predicts learning validates further application of ENA. For example, the turn by turn chat dynamics can be plotted as trajectories in the 2-D space, where the topics are placed. Investigating the trajectory patterns and their relationship to learning or socio-affective components are interesting future research directions.

We used a general topic model in this study. Many studies in the literature used LDA for topic modeling on relatively small corpora. This causes two problems. 1) LDA topic models built upon small corpora are not reliable, because LDA requires large number documents with relatively large size for each document. Inadequate corpus can result in misleading results. 2) Using a topic model that is not common would result in arbitrary interpretation. For example, the representation of "illness" from different corpus could be very different. Therefore, it is hard to compare the claims made to "illness" across different studies. Using a reliable, common topic models will set up a common language for different studies.

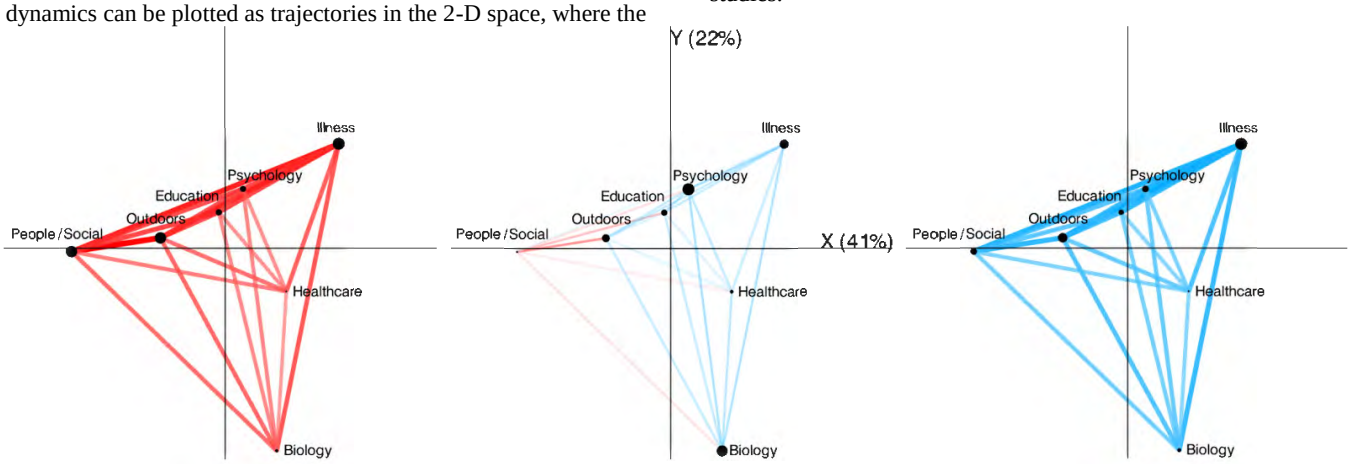


Figure 3: Mean discourse networks for students with low gain scores (left, red), students with high gain scores (right, blue), and a difference network graph (center).

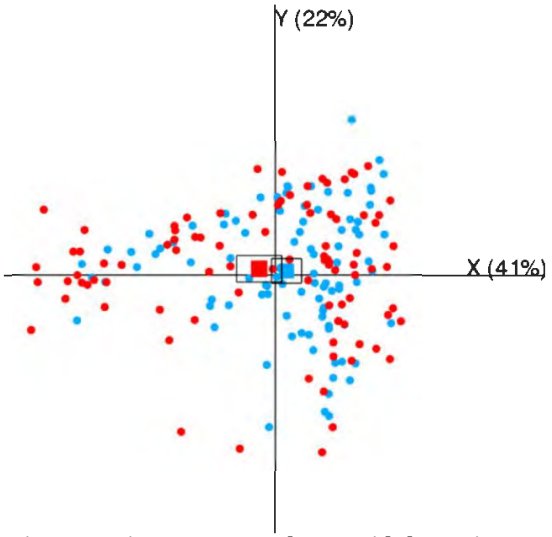


Figure 4: Discourse network centroids low gain score students red, high gain score students blue.

Topic scores for documents are usually inferred from topic models. While for longer documents, the topic scores can be used in many applications (e.g., text clustering [1]), the inferred topic proportion scores won't be useful for analyzing chats if we need to treat each utterance as a unit of analysis. It is not useful because chat utterances are too short. The statistical inference algorithm contains a high degree of randomness for short documents. As an extreme example, an utterance with a single word, would result in inferred topic proportion scores with "1" on one topic and "0" on others. The problem is that, this "1" was assigned to a topic with certain degree of uncertainty. That is, the topic this "1" was assigned to could be any topic. While aggregated analysis may not be sensitive to such uncertainty, detailed utterance by utterance analysis would suffer from it.

Our method of computing topic scores is based on the topic probability distribution over each word. We treat the topic distribution of each word as a vector. When computing the topic score, the simple sum of all word vectors gives scores to all topics. As we have pointed out, the summation algorithm will have a length effect. Therefore, when such topic scores are used, removing length effects through normalization is necessary. In this article, we did not use weighted sum as suggested in Cai et al. [4]. Comparing the effect of different weighting is beyond the scope of this paper.

When a general topic model is used, selecting topics relevant to the specific analysis becomes important. Our approach was to look at the total scores of utterances and find the "hot" topics by sorting the total topic scores. In our study, we had a quickly decreasing curve that helped us to select topics. We believe this would be the case for most studies using a model containing far more topics than the topics contained in the target data.

Although our study started with topic modeling to capture the "what" in the chats, the association networks constructed in the epistemic network analysis actually turned the "what" into a "how": how the topics in the chats associated with each other. This is conceptually similar to the cohesion features Dowell [7] and Cade [3] used.

Topic modeling emphasizes content words. When a topic model is built, stop words are usually removed. An interesting question is, what if we do the opposite: keep stop words and remove content words? Pennebaker (e.g., [13]) laid foundational work in this direction. The LIWC tool Pennebaker and his colleagues created provides over a hundred text measures by counting non-content words. LIWC measures could provide different features to epistemic network analysis and reveal different aspects of the chat dynamics.

7. ACKNOWLEDGMENTS

The research on was supported by the National Science Foundation (DRK-12-0918409, DRK-12 1418288), the Institute of Education Sciences (R305C120001), Army Research Lab (W911INF-12-2-0030), and the Office of Naval Research (N00014-12-C-0643; N00014-16-C-3027). 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. The Tutoring Research Group (TRG) is an interdisciplinary research team comprised of researchers from psychology, computer science, and other departments at University of Memphis (visit http://www.autotutor.org).

8. REFERENCES

[1] Alghamdi, R. and Alfalqi, K. 2015. A Survey of Topic Modeling in Text Mining. *IJACSA) International Journal of Advanced Computer Science and Applications*. 6, 1 (2015), 147–153.

[2] Blei, D.M., Edu, B.B., Ng, A.Y., Edu, A.S., Jordan, M.I. and Edu, J.B. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*. 3, (2003), 993–1022.

[3] Cade, W.L., Dowell, N.M.M. and Pennebaker, J. 2014. Modeling Student Socioaffective Responses to Group Interactions in a Collaborative Online Chat Environment. *Proceedings of the 7th International Conference on Educational Data Mining (EDM)*. 2, 21 (2014), 399–400.

[4] Cai, Z., Li, H., Graesser, A.C. and Hu, X. 2016. Can Word Probabilities from LDA be Simply Added up to Represent Documents? *Proceedings of the 9th International Conference on Educational Data Mining*. (2016), 577–578.

[5] von Davier, A.A. and Halpin, P.F. 2013. Collaborative Problem-Solving and the Assessment of Cognitive Skills: Psychometric Considerations. *ETS Research Report Series*. December (2013), 36 p.

[6] Dillenbourg, P. and Traum, D. 2006. Sharing Solutions: Persistence and Grounding in Multimodal Collaborative Problem Solving. *The Journal of the Learning Sciences*. 15, 1 (2006), 121–151.

[7] Dowell, N., Cade, W., Tausczik, Y., Pennebaker, J., and Graesser, A. 2014. What Works: Creating Adaptive and Intelligent Systems for Collaborative Learning Support. *Springer International Publishing Switzerland*. (2014), 124–133.

[8] Dowell, N.M.M., Skrypnyk, S., Joksimović, S., Graesser, A., Dawson, S., Gašević, D., Hennis, T. a., Vries, P. De and Kovanović, V. 2015. Modeling Learners ' Social Centrality and Performance through Language and Discourse. *Educational Data Mining - EDM'15* (2015), 250–257.

[9] Fiore, S.M., Rosen, M. a., Smith-Jentsch, K. a., Salas, E., Letsky, M. and Warner, N. 2010. Toward an understanding of macrocognition in teams: predicting processes in complex collaborative contexts. *Human factors*. 52, 2 (2010), 203–224.

[10] 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.

[11] Li, H., Samei, B., Olney, A., Graesser, A. and Shaffer, D. 2014. Question Classification in an Epistemic Game. *International Conference on Intelligent Tutoring Systems*. (2014).

[12] Pennebaker, J.W., Boyd, R.L., Jordan, K. and Blackburn, K. 2015. The Development and Psychometric Properties of LIWC2015. *Austin, TX: University of Texas at Austin*. (2015).

[13] Pennebaker, J.W., Chung, C.K., Frazee, J. and Lavergne, G.M. 2014. When Small Words Foretell Academic Success : The Case of College Admissions Essays. (2014), 1–10.

[14] Rosen, Y. 2014. Assessing Collaborative Problem Solving Through Computer Agent Technologies. *Encyclopedia of information science and technology*. 9, November (2014), 94–102.

[15] Sawyer, R.K. 2014. The new science of learning. *The Cambridge Handbook of the Learning Sciences*. 1–18.

[16] Scholand, A.J., Tausczik, Y.R. and Pennebaker, J.W. 2010. Assessing group interaction with social language network analysis. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 6007 LNCS, (2010), 248–255.

[17] Shaffer, D.W. 2006. Epistemic frames for epistemic games. *Computers and Education*. 46, 3 (2006), 223–234.

[18] Shaffer, D.W., Hatfield, D., Svarovsky, G.N., Nash, P., Nulty, A., Bagley, E., Frank, K., Rupp, A.A. and Mislevy, R.J. 2009. Epistemic Network Analysis: A Prototype for 21st-Century Assessment of Learning. *International Journal of Learning and Media*. 1, 2 (2009), 33–53.

[19] Siebert-Evenstone, A.L., Arastoopour, G., Collier, W., Swiecki, Z., Ruis, A.R. and Shaffer, D.W. 2016. In search of conversational grain size: Modeling semantic structure using moving stanza windows. *International Conference of the Learning Sciences*. (2016).

[20] Slavin, R.E. 1995. Cooperative Learning: Theory, Research and Practice (2nd Ed.). *The Nature of Learning*. (1995), 208.

[21] Tuulos, V.H. and Tirri, H. 2004. Combining Topic Models and Social Networks for Chat Data Mining. *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*. October (2004), 206–

213.

[22] Whitepaper, A.R. 2014. What Happens When We Learn Together. (2014).

[23] Yousef, A.M.F., Chatti, M.A., Schroeder, U., Wosnitza, M. and Jakobs, H. 2014. A Review of the State-of-the-Art. *Proceedings of the 6th International Conference on Computer Supported Education - CSEDU2014*. (2014), 9–20.

[24] Wang Z., Qiu B., Bai, W., Chuan, S. and Le, Y. 2014. Collapsed Gibbs Sampling for Latent Dirichlet Allocation on Spark. *JMLR: Workshop and Conference Proceedings*. 2004 (2014), 17–28.

Towards Closing the Loop: Bridging Machine-induced Pedagogical Policies to Learning Theories

Guojing Zhou, Jianxun Wang, Collin F. Lynch, Min Chi
Department of Computer Science
North Carolina State University
Raleigh, NC 27695
{gzhou3,jwang75,cflynch,mchi}@ncsu.edu

ABSTRACT

In this study, we applied decision trees (DT) to extract a compact set of pedagogical decision-making rules from an original *full* set of 3,702 Reinforcement Learning (RL)-induced rules, referred to as the DT-RL rules and Full-RL rules respectively. We then evaluated the effectiveness of the two rule sets against a baseline Random condition in which the tutor made random yet reasonable decisions. We explored two types of trees (weighted and unweighted) as well as two pruning strategies (pre- and post-pruning). We found that post-pruned weighted trees produced the best results with 529 DT-RL rules. The empirical evaluation was conducted in a classroom study using an existing Intelligent Tutoring System (ITS) named Pyrenees. 153 students were randomly assigned to three conditions. The procedure was the same for all students with domain content and required steps strictly controlled. The only substantive differences between the three conditions were the policy: (Full-RL vs. DT-RL vs. Random). Our result showed that as expected the machine induced policies (Full-RL and DT-RL) are significantly more effective than the random policy; more importantly, no significant difference was found between the Full-RL and DT-RL policies though the number of DT-RL rules is less than 15% of the number of the Full-RL rules and the former group also took significantly less time than the latter.

1. INTRODUCTION

Intelligent Tutoring Systems (ITSs) are interactive e-learning environments that support students' learning by providing instruction, scaffolded practice, and on-demand help. The system's behaviors can be viewed as a sequential decision-making process where at each step the system chooses an appropriate action from a set of options. *Pedagogical strategies* are the policies used to decide what action to take next in the face of alternatives. Each system decision will affect the user's subsequent actions and performance. Its impact on outcomes cannot always be immediately observed and the effectiveness of each decision depends upon the effectiveness of subsequent actions. Ideally, an effective learning environment will adapt its decisions to users' specific needs [1, 11]. However, there is no existing well-established theory on how to make these system decisions effectively. Generally speaking, prior research on pedagogical policies can be divided into two general categories: top-down or *theory-driven*, and bottom-up or *data-driven*.

In theory-driven approaches, ITSs employ hand-coded pedagogical rules that seek to implement existing cognitive or learning theories [1, 10, 17]. While existing learning literature gives helpful guidance on the design of pedagogical rules, such guidance is often too general to implement as effective immediate decisions. For example, the aptitude-treatment interaction (ATI) theory states that instructors should match their interventions to the aptitude of the learner [5]. While the principle behind this theory is understandable, it is not clear how to implement that rule for each decision. How do we represent learner's aptitude for each equation, how exact should be the system's adaptation, and so on.

Data-driven approaches, on the other hand, derive pedagogical policies directly from prior data. Here the policies specify the pedagogical decisions at a detailed level. Reinforcement Learning (RL), which we use here, is one popular approach that is able to derive pedagogical policies directly from student-system interaction logs. These policies are defined as a set of state-action mapping rules, which give the best decision to take in each state. The states are typically represented as sets of features and the actions are pedagogical actions such as presenting a worked example (WE) or requiring the student to solve problems (PS). When the system presents a worked example, the students will be given a detailed example showing a complete expert solution for the problem or the best step to take given their current solution state. In Problem Solving, by contrast, students are tasked with solving a problem using the ITS or with completing an individual problem-solving step.

For this project, our original complete RL-induced policy involves the following seven features representing the students' learning process from different perspectives[1].

[1]In the format of: [Feature-Name] (Discretization Procedure): Explanation of the feature.

1. **[nWESincePS]** **(**$0 \rightarrow 0; (0,1] \rightarrow 1; (1,+\infty) \rightarrow 2$**):** The number of worked example (WE) steps received since the last problem solving (PS) step.

2. **[timeInSession]** **(**$[0,2290] \rightarrow 0; (2290,4775] \rightarrow 1; (4775,7939] \rightarrow 2; (7939,+\infty) \rightarrow 3$**):** The total time spent in the current session.

3. **[avgTimeOnStepPS]** **(**$[0,29.01] \rightarrow 0; (29.01, 48.71] \rightarrow 1; (48.71,+\infty) \rightarrow 2$**):** The average amount of time spent on each PS step.

4. **[avgTimeOnStepSessionPS]** **(**$[0,23.51] \rightarrow 0; (23.51,36.56] \rightarrow 1; (36.56,55] \rightarrow 2; (55,+\infty) \rightarrow 3$**):** The average amount of time spent on each PS step in the current session.

5. **[nStepSinceLastWrongKC]** **(**$[0,1] \rightarrow 0; (1,7] \rightarrow 1; (7,25] \rightarrow 2; (25,+\infty) \rightarrow 3$**):** The number of steps received since the last wrong PS step on the current knowledge component (KC).

6. **[nWEStepSinceLastWrong]** **(**$[0,1] \rightarrow 0; (1,4] \rightarrow 1; (4,10] \rightarrow 2; (10,+\infty) \rightarrow 3$**):** The number of WE steps since the last wrong PS step.

7. **[nCorrectPSStepSinceLastWrongKCSession]** **(**$0 \rightarrow 0; (0,3] \rightarrow 1; (3,10] \rightarrow 2; (10,+\infty) \rightarrow 3$**):** The number of correct PS steps since the last wrong PS step on the current KC in the current session.

With this feature set, a state can be represented as a 7-dimensional vector where each element denotes a discretized feature value. Then, the rules can then be represented as:
(0:0:0:0:0:0:0) -> PS
(0:0:0:0:0:0:1) -> PS
(0:0:0:0:0:1:0) -> PS
(0:0:0:0:0:1:1) -> WE
In this study we discretized the features into three-four values producing a seven-feature state. This results in a state space of $3^2 * 4^5 = 9216$, that is 9216 rules in one RL-induced policy. While these types of polices can specify the exact action to take in each case, they are usually too narrow to be aligned to existing learning theories. Each of the rules covers only a very specific case and the relationship between rules is unknown. Thus it is impossible to explain the power of those rules from the perspective of learning theory. The opacity of those induced rules not only hinders us in improving data-driven methodologies when they go wrong, it also prevents us from advancing learning science research more generally. Moreover, it is possible that some of the decisions are environment-specific and may not generalize to other contexts. This in turn prevents translating these induced policies to environments other than the one from which they are induced. Therefore, a general method is needed to shed some light on the extracted detailed data-driven policies.

Decision tree (DT) induction is a robust data mining approach which can be used to extract a compact set of rules from a set of specific examples. It builds a tree-like hierarchical decision-making pattern which represents the knowledge it learned. Each path from root to leaf represents a single rule which may be dealt with separately. Prior studies have shown that DTs can match training examples in most cases, even with relatively small trees. Davidson et al., for example, built a DT for predicting the extinction risk of mammals [6]. Each of the species was described by 11 ecological features (e.g body mass, geographic range and population density) and were labeled with their extinction risk (threatened vs. non-threatened). Their tree contained 20 general rules which covered 4500 training examples, with a decision accuracy over 80%. Additionally, Reinchard et al. built a DT for predicting the invasiveness of woody plants [13]. The resulting DT encoded 15 rules from 235 examples, with a decision accuracy over 76%. Therefore, in our study, we will apply DT to extract general pedagogical decision-making rules from the detailed RL-induced policies.

In short, our primary research question is: *is DT an effective methodology for extracting more general pedagogical rules from the detailed RL-induced pedagogical rules?* In order to investigate this question, we will build DTs using the rules in a RL-induced policy as training examples and empirically evaluate the effectiveness of the extracted set of DT rules by comparing it to the full set of RL-induced rules in a classroom study. The state features in the RL-induced policies are the input features for the DT and the pedagogical actions are the output labels. In our empirical evaluation, we separate the pedagogical decisions from the instructional content, strictly controlling the content so that it is equivalent for all participants by 1) using an ITS which provides equal support for all learners; and 2) focusing on tutorial decisions that cover the same domain content, in this case WE versus PS.

2. BACKGROUND

2.1 Applying RL to ITSs

Beck et al. applied RL to induce pedagogical policies that would minimize the time students take to complete problems on AnimalWatch, an ITS for grade school arithmetic [2]. They trained the model with simulated students. The low cost of generated data allowed them to apply a model-free RL method, Temporal Difference learning. During the test phase, the induced policies were added to AnimalWatch and the new system was empirically compared with the original system. Their results showed that the policy group spent significantly less time per problem than their no-policy peers. Note that their primary goal was to reduce the amount of time per problem, however faster problem-solving does not always result in better learning performance. Nonetheless, their results showed that RL can be successfully applied to induce pedagogical policies for ITSs.

Iglesias et al., on the other hand, focused on applying RL to improve the effectiveness of an Intelligent Educational System that teaches students DataBase Design [8, 9]. They applied another model-free RL algorithm, Q-learning to induce policies that provide students with direct navigation support through the system's content. They used simulated students to induce the policy and empirically evaluated its effectiveness on real students. Their results showed that while the policy led to more effective system usage behaviors from students, the policy students did not outperform the no-policy peers in terms of learning outcomes.

Shen investigated the impact of both immediate and delayed reward functions on RL-induced policies and empirically evaluated the effectiveness of the induced policies within

an Intelligent Tutoring System called Deep Thought [15]. The induced pedagogical policies are used to decide whether the next task should be WE or PS. They found that some learners benefited significantly more from effective pedagogical policies than others.

Finally, Chi et al. applied model-based RL to induce pedagogical policies to improve the effectiveness of an Intelligent Natural Language Tutoring System for college-level physics called Cordillera [4]. The authors collected an exploratory corpus by training human students on an ITS that makes random decisions and then applied RL to induce pedagogical policies from the corpus. They showed that the induced policies were significantly more effective than the prior ones.

In short, prior studies have shown that RL-induced pedagogical policies can improve students' learning or reduce training time. However, all of these studies focused on the effectiveness of the RL-induced policies. None of them considered extracting more general rules from the induced policies.

2.2 Extracting General Rules

In addition to the work of Davidson et al. [6] and Reinchard et al. [13], DTs have been used for other tasks. Vayssiers et al., for example, applied Classification And Regression Trees to predict the presence of 3 species of oak in California [18]. Their training examples were Vegetation Type Map records for 2085 unique locations. Each record consisted of 25 climatic and geographic features as well as 3 labels showing the presence of the species (Quercus agrifolia, Quercus douglasii and Quercus lobata). One DT was induced for each type. The DTs were tested on another dataset which contains the same type of records for 2016 locations. For Quercus agrifolia, the induced tree had 10 leaf nodes and 94.9% of its predictions are correct for the locations that have the presence of this oak (sensitivity) while 86.7% of its predictions are correct for cases without the oak (specificity). For Quercus douglasii, the induced tree had 22 leaf nodes and a sensitivity and specificity of 87% and 79.9% respectively. For Quercus lobata, the tree had 6 leaves but reached a sensitivity of 77% and a specificity of 73.3%.

Thus, prior studies have shown that DT can effectively extract a small set of general decision-making rules from a large set of specific examples. However, all the examples used by these studies were observations of existing phenomena. So far as we know, this work is the only relevant research on the application of DT to extract a compact set of decision-making rules directly from full RL-induced rules and empirically evaluated the two sets of the rules.

2.3 Applying DT to RL

Prior research on incorporating DT with RL has largely focused on seeking a better representation of state space or policy for RL. Boutilier et al [3]. proposed representational and computational techniques for Markov Decision Processes (MDPs) to reduce the size of the state space. They used dynamic Bayesian networks and DTs to represent stochastic actions as well as DTs to represent rewards. Based upon this representation, they then developed algorithms to find conditional optimal policies. Their method was empirically evaluated on several planning problems and they showed significant savings in both time and space for some types of problems. Gupta et al. proposed the Policy Tree algorithm for RL. This algorithm is designed to directly induce a functional representation of the conditional optimal policies as a DT. They evaluated it on a variety of domains and showed that it was able to make splits properly [7].

In short, prior researchers have shown that properly combining DT with RL can result in a large amount of savings in time and space for finding good policies. However, none of these studies directly applied DT on RL-induced policies.

3. INDUCE FULL SET OF RL-POLICY

Previously, researchers have typically used the Markov Decision Process (MDP) [16] framework to model user-system interactions. The central idea behind this approach is to transform the problem of inducing effective pedagogical policies on what action the agent should take to the problem of computing an optimal policy for an MDP.

3.1 Markov Decision Process

An MDP is a mathematical framework for representing an RL task. It is defined by: a tuple $\langle S, A, T, R\rangle$. Where $S = \{S_1, S_2, ..., S_n\}$ denotes the state space; $A = \{A_1, A_2, ..., A_m\}$ represents a set of agent's possible actions; and $T : S \times A \times S \rightarrow [0, 1]$ is a transition probability table, where each element is $T^a_{S_iS_j} = p(S_j|S_i, a)$. This in turn indicates the probability of transiting from state S_i to state S_j by taking an action a while $R : S \times A \times S \rightarrow \mathbb{R}$ assigns rewards to state transitions given actions. The policy is defined as $\pi : S \rightarrow A$, mapping state S into action A with the goal of maximizing the expected reward.

After defining an MDP, we can transfer the student-system interaction dialog into the trajectory which can then be represented as follows:

$$S_1 \xrightarrow{A_1,R_1} S_2 \xrightarrow{A_2,R_2} S_3 \xrightarrow{A_3,R_3} ... \rightarrow S_N$$

Where $S_i \xrightarrow{A_i,R_i} S_{i+1}$ means that the tutor executed action A_i and received reward R_i in state S_i, and then transferred to the next state S_{i+1}. In general, the reward can be divided into two categories, immediate and delayed, where immediate rewards are received during the state transition, and delayed are available after reaching to goal state.

3.2 Training Datasets

Our training dataset was collected from three exploratory studies in which students were trained on an ITS which made random yet reasonable pedagogical decisions. The studies were given as homework assignments during CSC226: Discrete Mathematics, a core CS course offered at NCSU during the Fall 2014, Spring 2015 and Fall 2015 semesters. The dataset contains a total of 149 students' interaction logs. All students used the same ITS, followed the same general procedure, studied the same training materials, and worked through the same training problems. In order to model the students' learning process, we extracted a total of 142 state feature variables, which can be grouped into five categories:

1. **Autonomy (AM):** the amount of work done by the student: such as the number of problems solved so far *PSCount* or the number of hints requested *hintCount*.

2. **Temporal Situation (TS):** the time related information about the work process: such as the average time taken per problem *avgTime*, or the total time spent solving a problem *TotalPSTime*.
3. **Problem Solving (PS):** information about the current problem solving context, such as the difficulty of the current problem *probDiff*, or whether the student changes the difficulty level *NewLevel*.
4. **Performance (PM):** information about the student's performance during problem solving: such as the number of right application of rules *RightApp*.
5. **Student Action (SA):** the statistical measurement of student's behavior: such as the number of non-empty-click actions that students take *actionCount*, or the number of clicks for derivation *AppCount*.

3.3 Inducing RL Policies

In order to apply RL to induce pedagogical policies, we first defined the pedagogical decision-making problem as an MDP. The state representation includes all of the relevant features available at the beginning of each *step*. The actions are WE and PS at the *step* level. The transition tables were calculated on our training dataset, and our reward function includes two types of reward: delayed and immediate. Our most important reward is based on normalized learning gain (NLG) ($\frac{posttest-pretest}{1-pretest}$), which measures the students' learning gains *irrespective of their incoming competence*. This reward was given as a delayed reward as NLG scores can only be calculated after students finish the entire training process. However, Shen et al. [15] showed that giving immediate rewards can lead to the production of more effective policies when compared to delayed rewards. This is known as the credit-assignment problem. The more that we delay success measures from a series of sequential decisions, the more difficult it becomes to identify which of the decision(s) in the sequence are responsible for our final success or failure. Therefore, for the purposes of this study we also assigned immediate rewards based upon the students' performance during training on the system.

The value iteration algorithm was applied to find the optimal policy. This algorithm operates by finding the optimal value for each state $V^*(s)$. The optimal value for a given state is the expected discounted reward that the agent will gain if it starts in s and follows the optimal policy to the goal. Generally speaking, $V^*(s)$ can be obtained by the optimal value function for each state-action pair $Q^*(s,a)$ which is defined as the expected discounted reward the agent will gain if it takes an action a in a state s and follows the optimal policy to the end. The optimal state value $V^*(s)$ and value function $Q^*(s,a)$ can be obtained by iteratively updating $V(s)$ and $Q(s,a)$ via equations 1 and 2 until they converge:

$$Q(s,a) := R(s,a) + \gamma \sum_{s' \in S} p(S_j|S_i,a)V(s') \quad (1)$$

$$V(s) := \max_a Q(s,a) \quad (2)$$

Here, $p(S_j|S_i,a)$ is the estimated transition model T, $R(s,a)$ is the estimated reward model and $0 \leq \gamma \leq 1$ is a discount factor.

To induce effective pedagogical policies, we combined RL with various feature selections including 10 types of correlation-based methods and an ensemble method and capped the maximum number of state feature size to be eight. More details of our feature selection methods are described in [14]. The final resulting RL policy involves seven state features and 3706 rules.

4. EXTRACTING COMPACT DT-RL SETS

In order to extract a more compact set of decision-making rules from the full set of RL-induced rules, we implemented the ID3 algorithm to build DTs [12]. Each rule in the final RL-induced policy was used as a training example. Two types of decision trees were built: unweighted and weighted, as well as two types of pruning strategies were implemented: pre- and post-pruning. Next, we will discuss each of them in turn.

4.1 Unweighted vs. Weighted Tree

The decision to give a WE vs. PS may impact students' learning differently in different situations. We therefore built two types of decision trees: unweighted and weighted. *Unweighted* trees treated each decision equally while *weighted* trees take account of the relative importance of each pedagogical rule. When applying the value iteration algorithm to induce the optimal policy, we generate the optimal value function $Q^*(s,a)$, which gives the expected discounted reward each agent will gain if it takes an action a in a state s and follows the optimal policy to the end. For a given state s, a large difference between the values of $Q(s,\text{“}PS\text{”})$ and $Q(s,\text{“}WE\text{”})$ indicates that it is more important for the ITS to follow the optimal decision in the state s. We therefore used the absolute difference between the Q values for each state s to weight each RL pedagogical rule.

The ID3 algorithm builds a tree recursively from root to leaves. On each iteration of the construction process the algorithm will check the state of the dataset for the current branch. It will then select a test feature for the current node based upon the weighted information gain. The current node will then be expanded by adding branches to it, each of which represents a possible value for the selected feature. The data will be partitioned over the branches according to the value of the test feature. The selected feature cannot be used again by its children. Weighted information gain is defined by the difference between the weighted entropy of the examples before it is selected and after they are separated by feature value. The weighted entropy of a node can be calculated by equation 3

$$H(G) = -\sum_{i=1}^{J} p(i|G) log_2 p(i|G) \quad (3)$$

J is the total number of output label classes. In our case, it is the number of pedagogical actions (WE or PS) which is 2 . $p(i|G)$ is the weighted frequency defined by the equation: $p(i|G) = \frac{\sum_{x \in i} w_x}{\sum_{y \in G} w_y}$. $\sum_{x \in i} w_x$ is the total weight of the examples which are in node G and which belong to class i. And $\sum_{y \in G} w_y$ is the total weights of examples in node G.

The information gain of spliting the current set of training examples using feature F can be calculated by equation 4:

$$IG(F,G) = H(G) - \sum_{j=1}^{k} p(t_j|G) H(t_j) \quad (4)$$

$p(t_j|G)$ is the weighted frequency of the examples in node G: $p(t_j|G) = \frac{\sum_{x_F=t, x\in G} w_x}{\sum_{y\in G} w_y}$. $\sum_{x_F=t, x\in G} w_x$ is the total weights of examples in nodes G whose value of feature F is j and $\sum_{y\in G} w_y$ is the total weight of examples in nodes G.

4.2 Pre-Pruning and Post-Pruning

To control the size of rules induced by DT, we examined two types of pruning strategy: pre- and post-pruning. The pre-pruning is conducted during the process of building the tree and it used the information gain to determine whether to expand or to terminate. Only nodes with an information gain greater than a threshold times its depth: $IG(F,G) \geq \theta \times D_G$ will be expanded and others will be made as a leaf. θ is a fixed threshold and D_G is the depth of node G.

Post-Pruning is conducted after the whole decision tree is built and it used the error rate as the pruning measure. The error rate before a node is expanded is defined as: $e_G = \frac{\sum_{i\in I} w_i}{|G|}$. I is the set of the decisions incorrectly classified by node G and $|G|$ is the total number of examples in the node G. The error rate after a node is expanded is defined as: $e_C = \frac{\sum_{c\in C}\sum_{j\in I_c} w_i}{|G|}$. C is the set of children nodes of G after it is expanded and I_c is the set of the decisions incorrectly classified by the node c. In post-pruning, if the difference of a node's error rate from before to after split is less than a threshold, the node will be pruned by removing all of its branches to make it a leaf node.

4.3 The Compact Set of DT-RL Rules

In order to induce a compact set of DT-RL rules, we applied the DTs to the full set of 3706 RL-induced rules. The induced unweighted and weighted DTs without pruning has 2527 and 2456 rules (leaf nodes) respectively. Thus, without pruning, DTs are already able to extract a smaller set of rules: it reduced the total number of rules by over 1000.

Figure 1 shows the relationship between the number of leaf nodes (x-axis) and the inverted weighted accuracy (y-axis). Weighted accuracy(WA) is the weighted percentage of decisions correctly made, which can be calculated by the equation: $WA = \frac{\sum_{d_i\in T} w_i}{\sum_{d_i} w_i}$. T is the set of correct predictions made by a DT and w_i is the weight of decision i. The inverted weighted accuracy (IWA) is $IWA = WA^{-10}$, *the lower the better*. Since our goal is to find a good balance point between the IWA and the number of leaf nodes, we applied a widely used strategy called the Elbow Method, to select the best tree. As we can see in the figure, the elbows for the two unweighted tree approaches are around 800 and 1700 rules (x-axis) for the pre and post pruning respectively while the elbows for the two weighted tree approaches are around 250 and 500 for the pre and post pruning respectively. So it seems that weighted tree can extract more compact set of rules than the unweighted trees. While the weighted pre-pruning approach has around 250 rules, its IWA is much higher than the weighted post-pruning approach. Therefore, we chose the weighted tree with post-pruning strategy which has the an elbow at about 500 leaf nodes and reasonable IWA.

To further justify our DT choice, Table 1 shows the relationship between the pruning thresholds, WA and the number

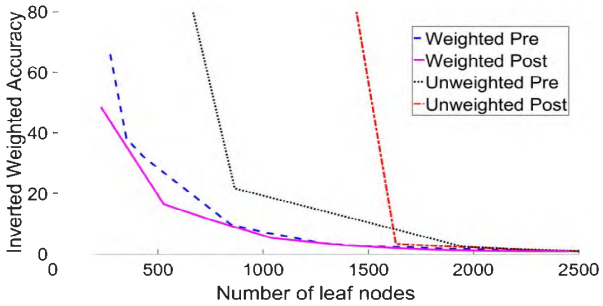


Figure 1: Leaf Nodes - Accuracy

of leaf nodes for the weighted tree with post-pruning. Table 1 shows that the tree with the closest number of leaves to 500 is the 529 one. It can be obtained by apply a pruning threshold of 0.8 and the result tree has a weighted accuracy of 0.76. The rules in the resulted tree will be the rules used in the DT-RL condition.

In short, we applied DT on RL-induced pedagogical policies to extract a more compact set of decision-making rules. The effectiveness of the original full set and the compact set of policies were empirically compared against a baseline policy which makes random yet reasonable decisions: PS vs. WE. Thus, we have three conditions:

1. Full-RL: the full set of 3706 RL-induced rules.
2. DT-RL: the compact set of 529 DT-induced RL rules.
3. Random: the random yet reasonable policy.

5. EMPIRICAL EXPERIMENT

Participants: This study was conducted in the undergraduate Discrete Mathematics course at the Department of Computer Science at NC State University in the Fall of 2016. 153 students participated in this study, which was given as their *final* homework assignment.

Conditions: Students in the study were assigned to three conditions via balanced random assignment based upon their course section and performance on the class mid-term exam. Since the primary goal of this work is to examine the effectiveness of the two RL based policies, we assigned more students to the Full-RL and DT-RL conditions than in the random condition. The final group sizes were: $N = 61$ (Full-RL), $N = 51$ (DT-RL), and $N = 41$ (Random).

Due to preparations for exams and length of the experiment, 126 students completed the experiment. 5 students were excluded from the subsequent analysis due to perfect pretest scores, working in group or gaming the system during the training. The remaining 121 students were distributed as follows: $N = 45$ for Full-RL; $N = 41$ for RL-DT; $N = 35$ for Random. We performed a χ^2 test of the relationship between students' condition and their rate of completion and found no significant difference among the conditions: $\chi^2(2) = 0.955, p = 0.620$.

Probability Tutor: Pyrenees is a web-based ITS for probability. It covers 10 major principles of probability, such as the Complement Theorem and Bayes' Rule. Pyrenees

Table 1: Weighted DT with Post-pruning

Threshold	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
WA	1.00	0.99	0.98	0.96	0.93	0.89	0.85	0.79	0.76	0.68
leaves	2456	2217	2029	1809	1608	1383	1043	758	529	231

provides step-by-step instruction and immediate feedback. Pyrenees can also provide on-demand hints prompting the student with what they should do next. As with other systems, help in Pyrenees is provided via a sequence of increasingly specific hints. The last hint in the sequence, the bottom-out hint, tells the student exactly what to do. For the purposes of this study we incorporated three distinct pedagogical decision modes into Pyrenees to match the three conditions.

Procedure: In this experiment, students were required to complete 4 phases: 1) pre-training, 2) pre-test, 3) training on Pyrenees, and 4) post-test. During the pre-training phase, all students studied the domain principles through a probability textbook, reviewed some examples, and solved certain training problems. The students then took a pre-test which contained 14 problems. The textbook was not available at this phase and students were not given feedback on their answers, nor were they allowed to go back to earlier questions. This was also true of the post-test.

During phase 3, students in all three conditions received the same 12 rather complicated problems in the same order on Pyrenees. Each main domain principle was applied at least twice. The minimal number of steps needed to solve each training problem ranged from 20 to 50. These steps included defining variables, applying principles, and solving equations. The number of domain principles required to solve each problem ranged from 3 to 11. All of the students could access the corresponding pre-training textbook during this phase. Each step in the problems could have been provided as either a WE or PS based upon the condition policy. Finally, all of the students completed a post-test with 20 problems. 14 of the problems were isomorphic to the pre-test given in phase 2. The remaining six were non-isomorphic complicated problems.

Grading Criteria: The test problems required students to derive an answer by writing and solving one or more equations. We used three scoring rubrics: binary, partial credit, and one-point-per-principle. Under the binary rubric, a solution was worth 1 point if it was completely correct or 0 if not. Under the partial credit rubric, each problem score was defined by the proportion of correct principle applications evident in the solution. A student who correctly applied 4 of 5 possible principles would get a score of 0.8. The one-point-per-principle rubric in turn gave a point for each correct principle application. All of the tests were graded in a double-blind manner by a single experienced grader. The results presented below are based upon the partial-credit rubric but the same results hold for the other two. For comparison purposes, all test scores were normalized to the range of [0,1].

6. EMPIRICAL RESULTS

Since both the Full-RL and DT-RL policies are based on an RL-induced policy, we combined the two conditions together as the *Induced group* to evaluate the effectiveness the RL-induced policy. The evaluation was conducted by comparing the Induced group with the baseline *Random condition* on learning performance and training time. Moreover, in order to further discover to what extent the compact policy retained the power of the full policy, we compared the Full-RL and DT-RL conditions on the same measures. Next, we will discuss each of the comparisons in turn.

6.1 Induced vs. Random

We measured Students' incoming competence via the pre-test scores collected before training took place. Table 2 shows a comparison between the Induced group and the Random group in terms of learning performance. The parenthesized values following the group names in row 1 denote the number of students in each group. The second row in this table shows the pre-test scores. The last column shows the pairwise t-test results. Pairwise t-tests on students' pre-test scores show that there is no significant difference between the two groups: $t(119) = -0.346$, $p = 0.730$, $d = 0.069$. Thus, despite attrition, the two groups remained balanced in terms of incoming competence. Next, we will compare the two groups in terms of learning performance in the post-test and training time.

Rows 2 - 4 in Table 2 show a comparison of the pre-test, isomorphic post-test (14 isomorphic questions), and adjusted post-test scores between the two groups along with the mean and SD for each. In order to examine the students' improvement through training on Pyrenees, we compared their scores on the pre-test and isomorphic post-test questions. A repeated measures analysis using test type (pre-test and isomorphic post-test) as factors and test score as the dependent measure showed a main effect for test type: $F(1, 119) = 98.75$, $p < 0.0001$. Further comparisons on group by group basis showed that on the isomorphic questions, both groups scored significantly higher in the post-test than in the pre-test: $F(1, 85) = 81.30$, $p < 0.0001$ for Induced and $F(1, 34) = 18.30$, $p = 0.0001$ for Random respectively. This suggests that the basic practice and problems, domain exposure, and interactivity of our ITS might help students to learn even when pedagogical decisions are made randomly.

In order to investigate the effectiveness of the induced policies, we compared students' overall learning performance, which was evaluated by their adjusted post-test scores, between the two groups. A one-way ANCOVA analysis was conducted on their overall post-test scores (20 questions), using the pretest scores as a covariate to factor out the influence of their incoming competence. The result shows a significant main effect: $F(1, 118) = 4.628$, $p = 0.033$. That is, the Induced group significantly outperformed the Random group on adjusted post-test scores, which is shown in

Table 2: Induced vs. Random

Cond	*Induced*(86)	*Random*(35)	T-test Result
Pre	.686(.194)	.699(.171)	$t(119) = -0.346,\ p = 0.730,\ d = 0.069$
Iso Post	.851(.155)	.812(.195)	$t(119) = 1.141,\ p = 0.256,\ d = 0.229$
Adjusted Post	.751(.144)	.689(.138)	$t(119) = 2.162,\ p = 0.033,\ d = 0.433$
Time	105.87(34.30)	111.18(27.33)	$t(119) = -0.815,\ p = 0.417,\ d = 0.163$
WE steps	205.74(62.73)	189.46(11.39)	$t(119) = 1.522,\ p = 0.131,\ d = 0.305$
PS steps	173.69(61.14)	190.26(10.28)	$t(119) = -1.591,\ p = 0.114,\ d = 0.319$
WE pct(%)	54.16(16.35)	49.89(2.78)	$t(119) = 1.532,\ p = 0.128,\ d = 0.307$

the fourth row of Table 2. Therefore, the results showed that the induced policies are significantly more effective than the random policy.

The fifth row in Table 2 shows the average amount of total training time (in minutes) students spent on our ITS for each group. Pairwise t-test showed no significant difference in training time between the two groups: $t(119) = -0.815$, $p = 0.417$, $d = 0.163$. The results suggest that when compared to the random policy, the induced policies generally do not have a significant different impact on students' training time.

The last three rows in Table 2 show the number of WE and PS steps given as well as the percentage of WE steps received by the Induced and the Random group. Pairwise t-tests showed that there is no significant difference between the two groups on these three measures.

6.2 Full-RL vs. DT-RL

We then performed the same comparison between the Full-RL and DT-RL conditions in order to examine the effectiveness of the DT-extracted compact policy. The second row in Table 3 shows the pre-test scores for each condition. A pairwise t-test on the scores shows no significant difference between the two conditions: $t(84) = -0.168$, $p = 0.867$, $d = 0.036$. Thus the two conditions were balanced in terms of incoming competence.

The pre-test, isomorphic post-test and adjusted post-test scores are shown in rows 2 - 4 of Table 3. A repeated measures analysis using test type (pre-test and isomorphic post-test) as factors and test score as dependent measure showed a main effect for test type: $F(1, 85) = 81.30$, $p < 0.0001$. Further comparisons on group by group basis showed that both conditions scored significantly higher in isomorphic post-test than in pre-test: $F(1, 44) = 42.16$, $p < 0.0001$ for Full-RL and $F(1, 40) = 39.16$, $p < 0.0001$ for DT-RL. These results suggest that the students can effectively learn from Pyrenees with the full and compact policies.

In order to discover to what degree the compact policy retained the effectiveness of the full policy, we compared the post-test scores between the two conditions. The results of a pairwise t-test showed no significant different between them on isomorphic post-test: $t(84) = 0.505$, $p = 0.615$, $d = 0.109$. We also conducted an ANCOVA analysis on the overall post-test scores using the pretest scores as a covariate and still found no significant different between the two conditions: $F(1, 83) = 0.348$, $p = 0.557$. In short, while on post-test scores, the DT-RL condition scored slightly lower than the Full-RL condition, the difference is not significant.

The fifth row of Table 3 shows the average amount of time students spent on training. As the row shows, the Full-RL condition spent significantly more time than the DT-RL condition: $t(84) = 3.829$, $p = 0.0002$, $d = 0.827$. Thus the Full-RL and DT-RL policies have significant different impact upon the students' training time.

The last three rows of Table 3 show the number of WE and PS steps given and the percentage of WE steps received by the Full-RL and the DT-RL condition. Pairwise t-tests showed that comparing to the DT-RL condition, the Full-RL condition received significantly fewer WE steps: $t(84) = -4.952$, $p < 0.0001$, $d = 1.069$; received a lower percentage of WE steps: $t(84) = -4.955$, $p < 0.0001$, $d = 1.070$; and completed more PS steps: $t(84) = 4.999$, $p < 0.0001$, $d = 1.079$. These results suggest that the pedagogical decisions made by the compact and full policies are substantively different.

7. DISCUSSION

In this study, we applied DT to extract a compact set of pedagogical rules from the full set of RL-induced rules and empirically evaluated the effectiveness of two sets of rules in a classroom study. Our goal was to shed some light on the RL-induced policies and we think this is only the first step towards narrowing the gap and building a bridge between machine-induced pedagogical policies and learning theories.

In order to find the best DT, we explored two types of tree: unweighted and weighted; and for each of them, we conducted two types of pruning strategy: pre- and post-pruning. After comparing the performance among them, we selected the weighted tree with the post-pruning strategy to perform the extraction of general decision-making rules. The RL-induced policy contains 3706 specific rules, and the compact DT-RL consisted of 529 rules with a weighted decision accuracy of 76%.

In our empirical experiment, we were able to strictly control the domain content and thus to isolate the impact of *pedagogy* from *content*. Based on this isolation, we compared students' performance with the Full-RL policy, the DT-RL policy and the baseline random policy. Our results showed that students in all three conditions learned significantly after training on Pyrenees, this suggests that the basic training of the ITS is effective, even when the pedagogical decisions are made randomly. To evaluate the effectiveness of the two machine induced policies (Full-RL policy and DT-RL policy), we combined the Full-RL and DT-RL condition as the Induced group and compared its learning performance with the Random group. Our results showed that the Induced

Table 3: Full-RL vs. DT-RL

Cond	Full-RL(45)	DT-RL (41)	T-test Result
Pre	.683(.205)	.690(.184)	$t(84) = -0.168,\ p = 0.867,\ d = 0.036$
Iso Post	.859(.145)	.842(.168)	$t(84) = 0.505,\ p = 0.615,\ d = 0.109$
Adjusted Post	.757(.144)	.739(.145)	$t(84) = 0.594,\ p = 0.554,\ d = 0.128$
Time	118.42(35.000)	92.10(27.95)	$t(84) = 3.829,\ p = 0.0002,\ d = 0.827$
WE steps	177.44(48.86)	236.80(62.03)	$t(84) = -4.952,\ p < 0.0001,\ d = 1.069$
PS steps	201.47(47.22)	143.20(60.57)	$t(84) = 4.999,\ p < 0.0001,\ d = 1.079$
WE pct(%)	46.77(12.78)	62.26(16.13)	$t(84) = -4.955,\ p < 0.0001,\ d = 1.070$

group significantly outperform the Random group. These results suggest that the machine induced policies are indeed more effective than the random policy.

Finally, in order to examine to what extent the compact DT-RL policy retained the power of the full RL-induced policy, we compared the learning performance of the Full-RL and the DT-RL conditions. Our results suggest that while some of the power was lost in the general rules extraction, the relative performance difference between the Full-RL and the DT-RL condition is not significant. In addition, our results on the pedagogical decisions made in training revealed that the compact DT-RL policy selected significant more WE than the Full-RL policy. This suggests that the two sets of policies indeed made materially different decisions. However, since the weighted DT took account of the importance of each rule, the DT-RL policy aims to retain maximal decision effectiveness from the Full-RL policy while the size of the former is less than 15% of the size of the Full-RL rules. In the future, we will apply existing learning theories to the decision-making process generated by decision tree to find a theoretical basis for the DT-induced general pedagogical decision-making rules.

8. ACKNOWLEDGEMENTS

This research was supported by the NSF Grant #1432156: "Educational Data Mining for Individualized Instruction in STEM Learning Environments" and #1651909: "Improving Adaptive Decision Making in Interactive Learning Environments".

9. 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] J. Beck, B. P. Woolf, and C. R. Beal. Advisor: A machine learning architecture for intelligent tutor construction. *AAAI/IAAI*, 2000:552–557, 2000.

[3] C. Boutilier, R. Dearden, and M. Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial intelligence*, 121(1):49–107, 2000.

[4] 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, 2011.

[5] L. J. Cronbach and R. E. Snow. *Aptitudes and instructional methods: A handbook for research on interactions.* Irvington, 1977.

[6] A. D. Davidson and et al. Multiple ecological pathways to extinction in mammals. *Proceedings of the National Academy of Sciences*, 106(26):10702–10705, 2009.

[7] U. D. Gupta, E. Talvitie, and M. Bowling. Policy tree: Adaptive representation for policy gradient. In *AAAI*, pages 2547–2553, 2015.

[8] A. Iglesias, P. Martínez, R. Aler, and F. Fernández. Learning teaching strategies in an adaptive and intelligent educational system through reinforcement learning. *Applied Intelligence*, 31(1):89–106, 2009.

[9] A. Iglesias, P. Martínez, R. Aler, and F. Fernández. Reinforcement learning of pedagogical policies in adaptive and intelligent educational systems. *Knowledge-Based Systems*, 22(4):266–270, 2009.

[10] K. R. Koedinger and et al. Intelligent tutoring goes to school in the big city. *IJAIED*, 8(1):30–43, 1997.

[11] P. Phobun and J. Vicheanpanya. Adaptive intelligent tutoring systems for e-learning systems. *Procedia-Social and Behavioral Sciences*, 2(2):4064–4069, 2010.

[12] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[13] S. H. Reichard and C. W. Hamilton. Predicting invasions of woody plants introduced into north america. *Conservation Biology*, 11(1):193–203, 1997.

[14] S. Shen and M. Chi. Aim low: Correlation-based feature selection for model-based reinforcement learning. *EDM*, 2016.

[15] S. Shen and M. Chi. Reinforcement learning: the sooner the better, or the later the better? In *UMAP*, pages 37–44. ACM, 2016.

[16] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[17] K. Vanlehn. The behavior of tutoring systems. *IJAIED*, 16(3):227–265, 2006.

[18] M. P. Vayssières, R. E. Plant, and B. H. Allen-Diaz. Classification trees: An alternative non-parametric approach for predicting species distributions. *Journal of vegetation science*, 11(5):679–694, 2000.

On the Influence on Learning of Student Compliance with Prompts Fostering Self-Regulated Learning

Sébastien Lallé
University of British Columbia
2366 Main Mall
Vancouver, BC V6T1Z4, Canada
lalles@cs.ubc.ca

Cristina Conati
University of British Columbia
2366 Main Mall
Vancouver, BC V6T1Z4, Canada
conati@cs.ubc.ca

Roger Azevedo
North Carolina State University
106 Caldwell Hall
Raleigh, NC 27695-8101, USA
razeved@ncsu.edu

Nicholas Mudrick
North Carolina State University
106 Caldwell Hall
Raleigh, NC 27695-8101, USA
nvmudric@ncsu.edu

Michelle Taub
North Carolina State University
106 Caldwell Hall
Raleigh, NC 27695-8101, USA
mtaub@ncsu.edu

ABSTRACT

In this paper, we investigate the relationship between students' learning gains and their compliance with prompts fostering self-regulated learning (SRL) during interaction with MetaTutor, a hypermedia-based intelligent tutoring systems (ITS). When possible, we evaluate compliance from student explicit answers on whether they want to follow the prompts, When such answers are not available, *we mine* several student behaviors related to prompt compliance. These behaviors are derived from students' eye-tracking and interaction data (e.g., time spent on a learning page, number of gaze fixations on that page). Our results reveal that compliance with some, but not all SRL prompts provided by MetaTutor do influence learning. These results contribute to gain a better understanding of how students benefit from SRL prompts, and provides insights on how to further improve their effectiveness. For instance, prompts that do improve learning when followed could be the focus of adaptation designed to foster compliance for those students who would disregard them otherwise. Conversely, prompts that do not improve learning when followed could be improved based on further investigations to understand the reason for their lack of effectiveness

Keywords

Intelligent tutoring systems; Self-regulated learning; Scaffolding; Compliance with prompts; Learning gains; Eye tracking; Linear regression; Hypermedia

1. INTRODUCTION

There is extensive evidence that the effectiveness of Intelligent Tutoring Systems (ITS) is influenced by how well students can regulate their learning, e.g., [13, 22]. Current research has shown that scaffolding self-regulated learning (SRL) strategies such as setting learning goals or assessing progress through the learning content can improve learning outcomes with an ITS, e.g., [1, 10, 22]. In particular, one of the most common approaches to scaffold SRL is to deliver *prompts* designed to guide students in applying specific SRL strategies as needed [22]. Previous work has focused on assessing the general effectiveness of such SRL prompts, for instance by comparing learning outcomes of students working with versions of the same ITS with and without the prompts. (e.g., [1, 19, 21]). Other work has investigated the extent to which students comply with the overall set of prompts generated by an ITS [16, 21]. However, there has been no reported study on the relationship between compliance with *specific* SRL prompts and learning outcomes. In this paper, we aim to fill this gap. Specifically, we explore the impact of student compliance with SRL prompts on learning gains with MetaTutor, an ITS designed to scaffold student SRL processes while learning about topics of the human circulatory system [1].

Our results show that student learning is influenced by compliance with some, but not all, of the SRL prompts delivered by MetaTutor. Overall, we found a positive impact on learning for compliance with prompts fostering learning strategies (revising a summary, reviewing notes), or planning processes (setting new learning goals). On the other hand, we found no impact on learning with prompts related to metacognitive monitoring processes (e.g., prompts to stay on or move away from the current page depending on student performance on a quiz on that page). Having information on the efficacy of each specific prompt in a ITS is important to guide further research on how to improve prompts that do not seem to improve learning when students follow them. Furthermore, prompts that foster learning when followed can become the focus of adaptive interventions designed to improve compliance for those students who would disregard these prompts if left to their own device.

The paper also provides initial insights into prompts design issues that affect how easy it is to evaluate compliance. In MetaTutor, some prompts explicitly asked students whether they wanted to follow the prompt, and then provided suitable affordance to accommodate a positive reply. Compliance with these prompts is easy to assess, but the additional interactions that they require might not always be possible, or might even be intrusive for some students. Other prompts did not require any specific response from the students. Thus, such prompts are in less danger of being intrusive, and provide for a more open-ended interaction. On the other hand, assessing compliance with these prompts is not trivial, because there is no clear definition of what compliance means. For example, one of the MetaTutor prompts asks students to re-read the current MetaTutor content page, but there is no obvious way to map this rather generic suggestion to a specific desired behavior (e.g., spend a specific amount of time on the page, read a specific number of words). We addressed this problem by running linear models to correlate a variety of student behaviors related to prompt compliance with learning. The behaviours we mined are based on both action and eye-tracking data (e.g., time spent on that page, gaze fixations on the content of the page), and our

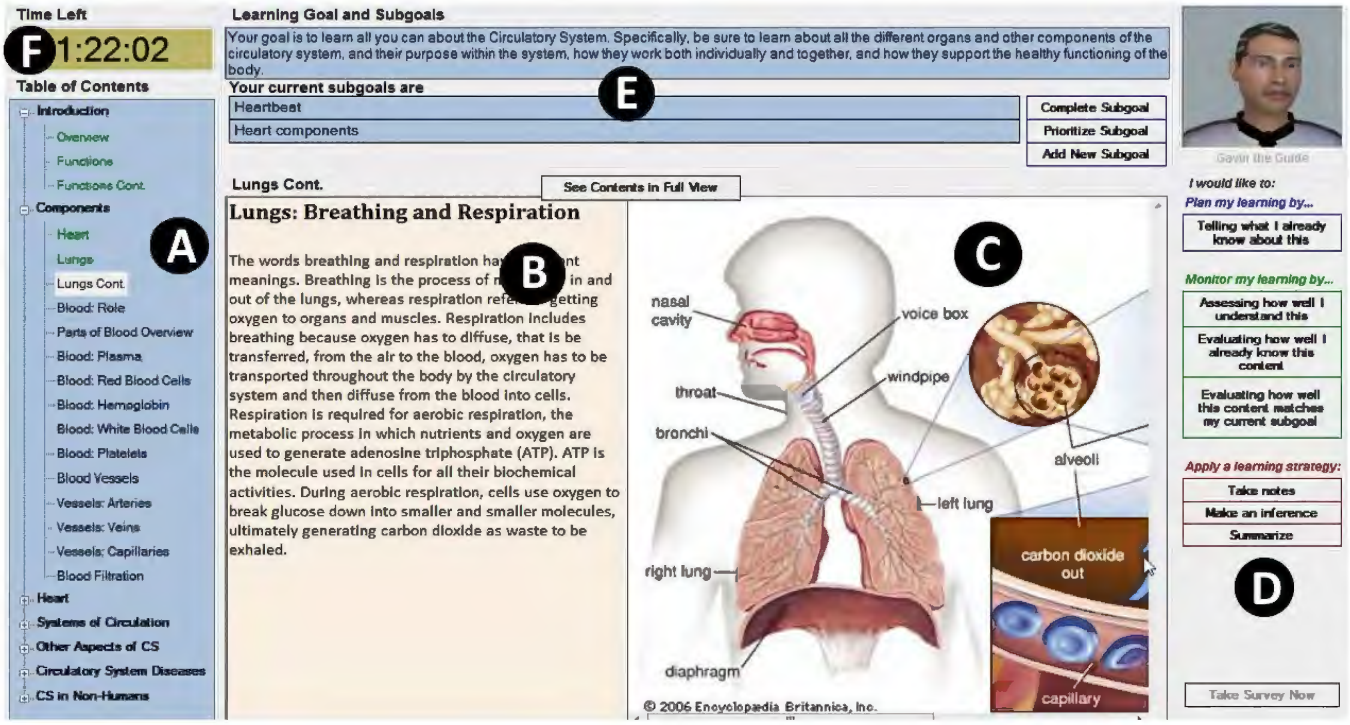


Figure. 1. Screenshot of MetaTutor.

results provide initial evidence that combining these two data sources can help to evaluate compliance. Thus, our findings represent a step toward research on how to evaluate compliance with prompts, both for the type of off line analysis presented in this paper, as well as for the real-time detection of compliance necessary if we want to have ITSs that adaptively help students follow prompts as needed.

The remainder of the paper starts with an overview of related work, followed by a description of MetaTutor and the study that generated the dataset we used for this research. Next, we illustrate how we mined data to evaluate compliance with MetaTutor's prompts, the statistical analysis we conducted, and our results.

2. RELATED WORK

There has been extensive work on assessing the effectiveness of scaffolding designed to support learning with ITSs. Scaffolding can include *prompts* or *hints* (i.e., interventions that guide the student in the right direction), *feedback* (evaluation of students answers, behavior or strategies), or *demonstration* (e.g., worked examples showing expert behavior) [22, 23]. Such scaffolding can be *domain-specific* to support the acquisition of domain-specific knowledge, or targeting domain-independent, meta-cognitive learning processes such as processes for self-regulated learning (SRL). There is extensive evidence that both domain-specific scaffolding (e.g., [3, 12, 18, 20]) and meta-cognitive scaffolding (e.g., [2, 10, 11, 21]) can improve the effectiveness of ITS. For example, domain-specific hints that explain how to solve the current problem step have been shown to improve skill acquisition in a variety of domains such as mathematics [20] and reading [3, 12]. At the meta-cognitive level, Roll et al. [21] tracked suboptimal help-seeking patterns (e.g., overuse of help) to deliver prompts and feedback on how to effectively use help. Prompts and feedback designed to help construct self-explanations during reading [10] or solving scientific problems [11] have been found to positively influence learning. Azevedo et al. [2] showed that SRL prompts and feedback effectively foster efficient use of SRL strategies while learning about biology.

Research has also examined student compliance with SRL prompts in ITS [5, 16]. Kardan and Conati [16] examined the benefit of providing a variety of prompts designed to help students progress within an interactive learning simulation. Overall they found that students largely complied with the prompts and that providing these prompts improved learning gains. However, they did not explore whether and how compliance with specific prompts influence learning outcomes, and which prompts are the most effective. Bouchet et al. [5] adapted the frequency of prompt delivery in MetaTutor based on whether students previously complied with prompts of the same type. However, their analysis uncovered no influence of such adaptive prompting strategy on learning gains. We extend the aforementioned work on prompt compliance by showing how learning gains are impacted by compliance with some, but not all SRL prompts in MetaTutor. Furthermore, whereas previous solely used interaction data to evaluate compliance, we also leverage eye-tracking data when compliance cannot be inferred directly from students' answers or actions (e.g., compliance with the prompts of reading a text further).

Eye-tracking has been used in ITS to model a variety of students traits and behavior, e.g., emotions [14], learning outcomes [15], metacognitive behavior [7], or mind wandering [4]. Eye tracking has also been used to capture students attention to prompts [6, 8] and to pedagogical agents [17]. Conati et al. [6] leveraged gaze data to detect whether students processed domain-specific textual prompts in an educational game for math, and found that reading the prompts more extensively improved game performance. Lallé et al. [17] used gaze data to capture student visual attention to pedagogical agents in MetaTutor, and found that student learning gains are significantly influenced by specific metrics for visual attention (fixation rate, longest fixation). Eye-tracking has also

been used to add real-time adaptive prompts to Guru, an agent-based ITS for learning biology [9]. In that work, audible prompts designed to reorient student attention towards the screen were triggered if a student had not looked at the screen for more than 5s while Guru was providing scaffolding. This research showed that this gaze-reactive feedback can improve learning with Guru. In our work, we mine eye-tracking data to evaluate compliance with specific SRL prompts, and examine whether and how compliance with such SRL prompts influences learning gains.

3. METATUTOR

MetaTutor [1] is a hypermedia-based ITS containing multiple pages of content about the circulatory system, as well as mechanisms to help students self-regulating their learning with the assistance of multiple speaking pedagogical agents (PAs). When working with MetaTutor, students are given the overall goal of learning as much as they can about the human circulatory system. The main interface of MetaTutor (see Fig. 1) includes a table of contents (Fig. 1A), the text of the current content page (Fig.1B), a miniature image allowing the student to display a diagram along with the text (Fig. 1C), the current goals and subgoals to learn about (Fig. 1E), a timer indicating how much time remains in the learning session (Fig. 1F), and an SRL palette (Fig. 1D). This palette is designed to scaffold students self-regulatory processes by providing buttons they can select to initiate specific SRL activities (e.g., making a summary, taking a quiz, setting subgoals). Further SRL scaffolding is provided by three PAs in the form of *feedback* on student performance on these SRL activities (e.g., performance on quiz or on the quality of their summaries), as well as *prompts* designed to guide these activities as needed. The PAs deliver these prompts based on student behavior (e.g., time spent on page, number of pages visited).

Specifically, *Pam the Planner* prompts planning processes primarily at the beginning of the learning session by suggesting to add a new subgoal and, if needed, which one to choose (e.g., path of blood flow, heart components). *Mary the Monitor* scaffolds students' metacognitive monitoring processes by making them take quizzes on the target material when they appear to be ready for them. Based on quiz outcomes, Mary prompts students to evaluate the relevance of the current content and subgoal to their knowledge, and suggests how to move through the available material and sub goals accordingly. *Sam the Strategizer* prompts students to apply the learning strategies consisting of summarizing the content studied so far or reviewing notes they have taken on the content[1].

All PAs provide audible assistance through the use of a text-to-speech engine (Nuance). The PAs are visually rendered using Haptek virtual characters, which generate idle movements when the PAs are not speaking (subtle, gradual head and eye movements), as well as lip movements during speech.

4. USER STUDY

The data used for the analysis presented in this paper were collected via a user study designed to gain a general understanding of how students learn with MetaTutor [1]. The study included the collection of a variety of multi-channel trace data (e.g., eye tracking, log files, physiological sensors). In this paper, we focus on using interaction and eye-tracking data to track compliance with the SRL prompts provided by MetaTutor, and study the relationship among compliance with the prompts and learning gains.

Twenty-eight college students participated in the study, which consisted of two sessions conducted on separate days. During the first session, lasting approximately 30-60 minutes, students were administered several questionnaires, including a 30-item pretest to assess their knowledge of the circulatory system. During the second session lasting approximately three hours, students first underwent a calibration phase with the eye tracker (SMI RED 250) as well as a training session on MetaTutor. Each student was then given 90 minutes to interact with the system. Finally, students completed a posttest analogous to the pretest, followed by a series of questionnaires about their experience with MetaTutor.

5. DATA ANALYSIS

5.1 Evaluating Compliance with Prompts

In our analysis we categorize prompts into two types based on how compliance can be evaluated. The first type includes prompts for which compliance can be explicitly assessed from students subsequent responses (*explicit compliance prompts*); the second type includes prompts for which compliance needs to be inferred by mining a variety of behaviors (*inferred compliance prompts*).

Explicit compliance prompts are those that:

- Require students to answer "yes" or "no" (using a dialogue panel that becomes active at the bottom of the display). If students answers yes, the only action they can perform in the MetaTutor interface is the one they agreed upon (e.g., adding a specific subgoal suggested by the agent, making or revising a summary, moving to a previously added subgoal or staying on the current one)[2].
- Require students to take a specific action within a specific time frame (i.e., open the diagram while they are on the current page, and review notes by the end of the learning session).

Table 1 lists the explicit compliance prompts considered in this analysis.

Inferred compliance prompts are those for which the PAs do not force students to provide an explicit answer. Specifically, after the agent utters one of these prompts, the student simply clicks on "continue" in the same dialogue panel, and can either ignore the prompted action, or comply at some point. These prompts (listed in Table 2) include all prompts related to staying on or moving away from the current page, as well as initiating the action of adding a new subgoal.

5.2 Statistical Analysis

Our analysis aims to investigate if and how compliance with MetaTutor's SRL prompts influence learning. The variable we

[1] More details about the design of the agents can be found in [1].

[2] For the "stay on current subgoal" prompt, students are not forced to comply after answering "yes", but we have listed it in this category because student are still required to explicitly answer "yes" or "no" to the PAs as for whether they want to follow the prompt or not.

Table 1. List of explicit compliance prompts provided in MetaTutor (grouped by type of prompted SRL processes).

Prompt label	Description	Prompts for
Suggest subgoal	Recommend possible subgoals to learn about while the students is adding new subgoal.	Planning processes
Moving to next subgoal	Recommend moving on to another subgoal when the student did well on a quiz related to the current subgoal.	Metacognitive monitoring processes
Stay on subgoal	Recommend to learn more about the current subgoal when the student did not do well enough on a quiz related to that subgoal.	
Open diagram	Recommend opening the diagram when it is relevant to the current subgoal.	
Summarize	Recommend making a summary of the current page when the student has spent enough time on that page.	Learning strategies
Revise summary	Recommend revising the summary submitted by the student when there are issues with the summary (e.g., the summary is too long or too short).	
Review notes	Recommend reviewing notes taken on the learning content when approaching from the end of the session.	

Table 2. List of inferred compliance prompts provided in MetaTutor (grouped by type of prompted SRL processes).

Prompt label	Description	Prompts for
Add subgoal	Recommend adding a new subgoal to learn about when a student has no active subgoal.	Planning processes
Move to next page	Recommend moving on to another page when the student did well on a quiz related to the current page.	Metacognitive monitoring processes
Stay on page	Recommend staying on the current page when the student did not well enough on a quiz related to that page.	

adopted to measure learning in our analysis is *proportional learning gain*, defined as:

$$\frac{\text{posttest score ratio} - \text{pretest score ratio}}{1 - \text{pretestscore ratio}}$$

Table 3 reports statistics for pre- and post-test scores, as well as for the corresponding learning gains.[3]

Table 3. Descriptive statistics for pretest, posttest, and learning gain.

Measures of learning	M	SD	Median
Pretest	18.6	4.2	19
Posttest	21.4	4	21
Proportional learning gain	15.3	50.2	20

We conducted two separate analyses for explicit and inferred compliance prompts, described next.

Explicit compliance prompts. Since compliance is directly observed in the data for explicit compliance prompts (listed in Table 2), we computed a *compliance rate* for each of these prompts as follow:

$$\frac{\text{Number of prompts followed}}{\text{Total number of prompts delivered}}$$

[3] The increase from pretest to post-test is statistically significant indicating that MetaTutor is overall effective at fostering learning, as further discussed in [1].

Table 4 shows the compliance rate averaged across students for each of the seven explicit compliance prompts in MetaTutor, and the number of prompts delivered.

Table 4. Descriptive statistics of the number of explicit compliance prompts delivered, as well as on compliance rate.

Prompt	Total number of prompts delivered	Compliance rate *Mean (SD)*
Suggest subgoal	60	.90 (.25)
Move next subgoal	25	.85 (.34)
Stay on subgoal	44	.27 (.37)
Open diagram	77	.21 (.32)
Summarize	105	.32 (.41)
Revise summary	59	.76 (.37)
Review notes	28	.46 (.51)

To investigate the impact of compliance with explicit compliance prompts on learning, we ran a multiple linear regression model with *proportional learning gain* as the dependent variable, as well as the *compliance rate* for each of the seven explicit compliance prompts, and the *total number of prompts received* as the factors. For post-hoc analysis we ran pairwise *t*-test comparisons, and *p*-values were adjusted with the Holm-Bonferroni approach to account for multiple comparisons.

Inferred compliance prompts. As stated above, for inferred compliance prompts (listed in Table 5), students are not forced to explicitly accept or ignore the prompt. This means that compliance with those prompts has to be assessed from student behaviors following the prompts. One approach we considered was to make this assessment binary, as we did for explicit compliance prompts, by establishing thresholds for relevant behaviors. For instance, compliance with the prompt to re-read the current page could be assessed to be true if the student stays on the page for a fixed number of seconds after receiving this prompts. However, it

is difficult to fix these thresholds in an informed manner, as they may depend on the student (e.g., on a student's readings speed, existing understanding of the page, etc.), and on the object of the prompt (e.g., on the length or difficulty of the page to be re-read). It is also difficult to decide which specific behaviors should be considered for compliance, as several might be relevant (e.g., time spent on a page, specific attention patterns on a page).

Thus, for the subsequent analysis, we avoided committing to specific thresholds and behaviors, and we opted instead for performing regression analyses to try to relate multiple relevant compliance behaviors to learning.

We started by building *data windows* that capture student data from the delivery of each inferred compliance prompt in Table 2, to the following actions:

- "Moving to another page" for the *move to next page* and *stay on page* prompts;
- "Adding a new subgoal" for the *add new subgoal* prompt.

We used these data windows to derive three behavioral measures related to compliance:

- *Window length*, capturing how long students spent before moving on to another page or adding a new subgoal;
- *Number of fixations*[4] made on MetaTutor's learning content (text and diagram), as captured by eye tracking. We use this measure to understand whether students read the page and/or processed the diagram;
- *Number of SRL strategies* initiated by the student by pressing the corresponding buttons in the SRL palette (see Fig. 1 D).

Higher values of these measures (i.e., long windows, high number of fixations on the page and high number of SRL strategies used) are possible indicators that the student is processing the current page, e.g., the student is thinking about or reading the content (as captured by the length of the data window and number of fixations on the page), or using SRL strategies on the current page. Thus, we hypothesized that higher values of these measures could reveal compliance with *stay on page* prompts, whereas lower values could reveal compliance with prompts instructing students to *move on*. Similarly, because prompts to *add a subgoal* requires moving on from the learning content to actually add a subgoal, we expected a short window, a small number of fixations on the page, and a small number of SRL strategies to indicate compliance.

It should be noted that we could have generated other eye-tracking measures, such as fixation duration on the text or the number of transitions from the text to other components of the MetaTutor's interface. However, because valid eye-tracking data were collected for only 16 students out of the 28 who participated in the study, resulting in a rather small dataset, we focused on the most promising behavioral measures that could be related to compliance, as a proof of concept. Table 5 shows the amount of inferred compliance prompts delivered to those 16 students.

Table 5. Number of inferred prompts delivered.

Prompt	Total number of prompts delivered
Add a subgoal	34
Stay on page	117
Move to next page	326

We leveraged the three aforementioned measures of student behavior to investigate if complying with inferred compliance prompts influences learning, and if so, how. Specifically, for each of the three inferred compliance prompts, we ran a multiple linear regression model with *proportional learning gain* as the dependent variable, as well as the *window length*, *number of SRL strategies performed*, and *number of fixations on the learning content* as the factors. As done for explicit compliance prompts, we used pairwise *t*-test comparisons for post-hoc analysis, and all *p*-values were adjusted with the Holm-Bonferroni approach.

6. RESULTS

We describe below the significant[5] effects found in our analysis, first for explicit compliance prompts, and second for inferred compliance prompts.

6.1 Effects for Explicit Compliance Prompts

Our statistical analysis uncovered significant main effects of *compliance rate* for three explicit compliance prompts:

- *Revise summary* ($F_{1,20} = 6.17$, $p=.02$, $\eta_p^2 = .15$), shown Fig. 2a.
- *Review notes* ($F_{1,20} = 7.43$, $p=.013$, $\eta_p^2 = .16$), shown Fig. 2b.
- *Suggest subgoal* ($F_{1,20} = 11.4$, $p=.003$, $\eta_p^2 = .27$), shown Fig. 2c.

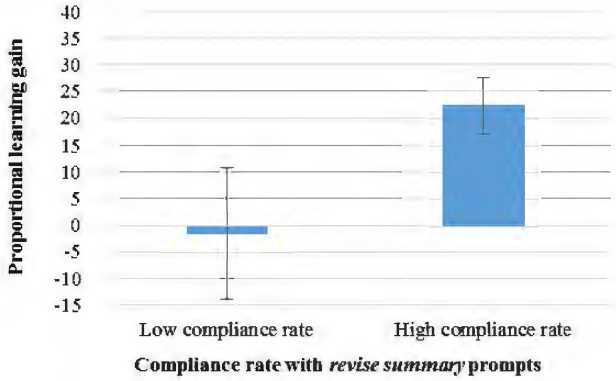
These three main effects and related pairwise comparisons all reveal that students learned more when they complied more with these prompts than when complying less.

These results for *revise summary* and r*eview notes* are consistent with previous findings showing these learning strategies can be beneficial for learning [17, 22, 24], and extend them by showing that prompting these strategies is effective when students comply with the prompts. Notably, we found a significant effect for prompts to *revise summary*, but not for prompts to *summarize*. This indicates that solely prompting to summarize is not enough to improve learning, and that guiding the students through the process of making a good summary is necessary. Results for *suggest subgoal* indicate that recommending a particular learning subgoal is useful, possibly because it is difficult for students to choose good subgoals by themselves.

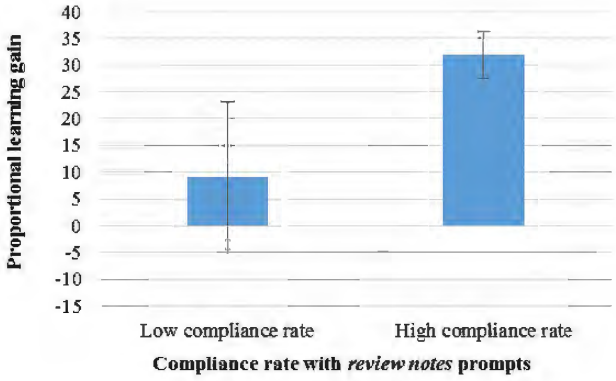
These results suggest to examine ways to improve compliance with prompts to *revise summary, review notes* and *suggest subgoal*, since our analysis reveals that not complying with them hinders learning. For instance, MetaTutor could foster compliance with these prompts by explaining how they can help the students, or conversely force the students to follow these prompts.

[4] Fixation is defined as gaze maintained at one point on the screen for at least 80ms.

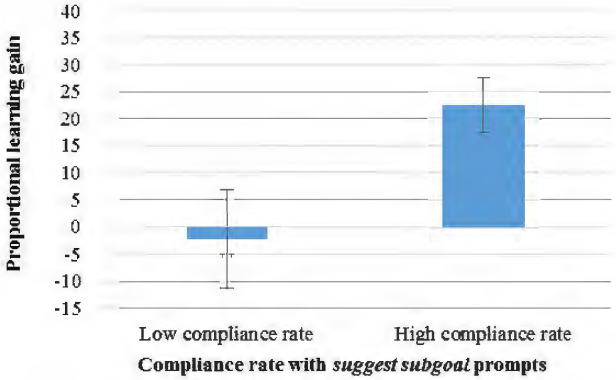
[5] We report statistical significance at the 0.05 level throughout this paper, and effect sizes as small for $\eta_p^2 \geq 0.02$, medium for $\eta_p^2 \geq 0.13$, and large for $\eta_p^2 \geq 0.26$.



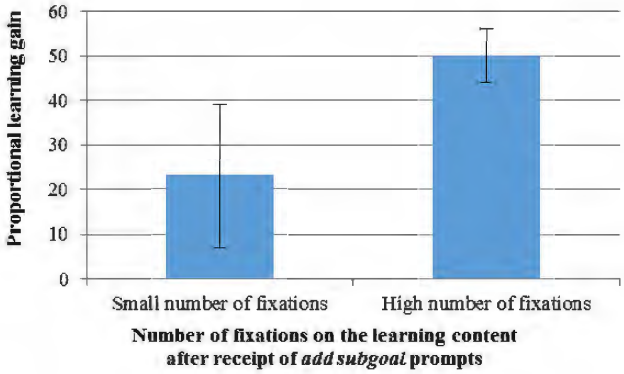
a. Main effect of compliance rate with "revise summary".



b. Main effect of compliance rate with "review notes".



c. Main effect of compliance rate with "suggest subgoal".



d. Main effect of fixation on page after reception of "add subgoal".

Figure 2. Main effects found in this analysis, for explicit compliance prompts (charts a, b, c) and inferred conpliance prompts (chart d). Error bars show 95% confidence interval.

We found no significant effects and small effect sizes (see Appendix A) for the four remaining prompts, namely *summarize*, *stay on subgoal* or *move to next subgoal*, and *open the diagram*. These results indicate it is important to study the effectiveness of SRL prompts individually, to identify those for which compliance does not improve learning. Based on these findings, it is justified to further investigate why complying with these prompts is not beneficial for learning in MetaTutor, and revise the prompts accordingly. For example, it might be due to the nature of the prompts, their timing, their frequency, their wording, and so forth.

6.2 Effects for Inferred Compliance Prompts

We found a main effect of *fixation on learning content* for the "add subgoal" prompts ($F_{1,3} = 13$, $p = .03$, $\eta_p^2 = .29$), shown in Fig. 2d. This effect and related pairwise comparisons reveal that students learned more when they fixate more on the current page than when fixating less. Since students were instructed to add a new subgoal rather than process the current page, this finding suggests that complying with this prompt might not be effective for learning with MetaTutor, possibly because of the timing of this prompt, its frequency or its wording. Although only seven students with valid gaze data received this prompt, the effect size is large, suggesting it is worth conducting further analysis to ascertain whether and why complying with this prompt is not beneficial for learning.

We found no effects and small effect sizes (see Appendix B) for the other inferred compliance prompts, namely *stay on page* and *move to next page*, two prompts related to metacognitive monitoring processes. We cannot make final conclusions on the pedagogical effectiveness on these prompts based on these results, because the dataset is not large and for this reason we did not include in the analysis other features that could indicate compliance (for example other eye-tracking measures such as fixation duration on text or gaze transitions from the text to other components of MetaTutor). However, it should be noted that we also found no effect for the explicit compliance prompts that foster metacognitive monitoring processes (*stay on subgoal*, *move to next subgoal*, and *open the diagram*, see previous section). This lack of effect for all prompts fostering metacognitive monitoring, even when compliance is explicitly assessed, suggests that these prompts are not beneficial for learning with MetaTutor. This could be due to the way these prompts are currently implemented in MetaTutor (e.g., their wording, timing delivery or frequency), or to the nature or the prompts itself. Our results nonetheless justify to run further analysis to ascertain whether (and why) prompts fostering metacognitive monitoring are not effective, and revise them as needed.

7. CONCLUSION

In this research we investigated the relationship between compliance with prompts designed to support the use of self-regulated learning (SRL) processes and learning gains while learning about

the human circulatory system with MetaTutor. We identified two approaches to evaluate compliance to MetaTutor's prompts:

(i) Assess compliance from students' subsequent response to the prompts when students are forced to express compliance (e.g., by answering "yes" or "no" to a prompt);

(ii) Run linear models to examine the influence on learning of a variety of student behaviors related to prompt compliance, when compliance is not elicited by MetaTutor. The behaviors we mined are based on both interface and eye-tracking data (e.g., time spent on that page, gaze fixations on the content of the page).

Our results revealed that student learning gains are influenced by compliance with some, but not all SRL prompts provided by MetaTutor. Specifically, we found a positive influence on learning for prompts that foster learning strategies (*revise a summary* and *review notes*) as well as prompts that recommend setting a specific learning subgoal. Based on these findings, it is worth exploring ways to improve compliance with these prompts. In particular, in future research we plan to examine whether forcing students to comply with these prompts or providing detailed explanations on how the prompted SRL strategies can be useful can improve learning.

We found that compliance with the other MetaTutor's prompts studied in this analysis does not improve learning. This finding reveals that assessing compliance to SRL prompts individually is useful to identify prompts that may not be effective at supporting learning. In particular, we found no results for all prompts related to metacognitive monitoring processes (e.g., staying on/moving away from the current page), suggesting to examine further why complying with these prompts do not influence learning with MetaTutor. For example, it could be due to their timing and frequency, their wording, their nature, and so forth.

In this paper we also addressed the challenge of evaluating compliance with rather open-ended prompts for which there is no clear definition of compliance. Specifically we ran a linear regression analysis to relate relevant compliance behaviors to learning. Such behaviors were derived from a combination of student interaction and eye-tracking data after receipt of a prompt (e.g., time spent and amount of gaze fixations on a page can reveal compliance with prompt to read that page). Preliminary results show that such interaction-based and eye-tracking-based measures can help evaluate compliance. In future research, we plan to investigate further behavioral measures relevant to assessing compliance, such as tracking eye gaze patterns on the different components of MetaTutor as well as transitions between those components.

Lastly, we plan to investigate the possibility of detecting in real time compliance with SRL prompts for which we found a positive effect on learning, using eye-tracking and interaction data. Such real-time detection could inform the design of adaptive prompts to foster compliance for those students who might otherwise disregard these prompts. For instance, adaptive prompts could force students to follow them or explain how the prompted SRL processes can improve learning. Evaluating such adaptive prompts fostering SRL processes would provide further insights on how students comply with and benefit from SRL prompts.

8. ACKNOWLEDGMENTS

This publication is based upon work supported by the National Science Foundation under Grant No. DRL-1431552 and the Social Sciences and Humanities Research Council of Canada. 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 National Science Foundation or the Social Sciences and Humanities Research Council of Canada.

9. REFERENCES

[1] Azevedo, R., Harley, J., Trevors, G., Duffy, M., Feyzi-Behnagh, R., Bouchet, F. and Landis, R. 2013. Using trace data to examine the complex roles of cognitive, metacognitive, and emotional self-regulatory processes during learning with multi-agent systems. *International handbook of metacognition and learning technologies*. Springer, 427–449.

[2] Azevedo, R., Martin, S.A., Taub, M., Mudrick, N.V., Millar, G.C. and Grafsgaard, J.F. 2016. Are Pedagogical Agents' External Regulation Effective in Fostering Learning with Intelligent Tutoring Systems? *Proceedings of the 13th International Conference on Intelligent Tutoring Systems* (Zagreb, Croatia, 2016). Springer, 197–207.

[3] Beck, J., Chang, K., Mostow, J. and Corbett, A. 2008. Does help help? Introducing the Bayesian Evaluation and Assessment methodology. *Proceedings on the 9th International Conference on Intelligent Tutoring Systems* (Montréal, QC, Canada, 2008). Springer, 383–394.

[4] Bixler, R. and D'Mello, S. 2015. Automatic Gaze-Based Detection of Mind Wandering with Metacognitive Awareness. *Proceedings of the 23rd International Conference on User Modeling, Adaptation and Personalization* (Dublin, Ireland, 2015). Springer, 31–43.

[5] Bouchet, F., Harley, J.M. and Azevedo, R. 2016. Can Adaptive Pedagogical Agents' Prompting Strategies Improve Students' Learning and Self-Regulation? *Proceedings of the 13th International Conference on Intelligent Tutoring Systems* (Zagreb, Croatia, 2016). Springer, 368–374.

[6] Conati, C., Jaques, N. and Muir, M. 2013. Understanding attention to adaptive hints in educational games: an eye-tracking study. *International Journal of Artificial Intelligence in Education*. 23, 1–4 (2013), 136–161.

[7] Conati, C. and Merten, C. 2007. Eye-tracking for user modeling in exploratory learning environments: An empirical evaluation. *Know.-Based Syst.* 20, 6 (2007), 557–574.

[8] D'Mello, S., Olney, A., Williams, C. and Hays, P. 2012. Gaze tutor: A gaze-reactive intelligent tutoring system. *International Journal of Human-Computer Studies*. 70, 5 (2012), 377–398.

[9] D'Mello, S., Olney, A., Williams, C. and Hays, P. 2012. Gaze tutor: A gaze-reactive intelligent tutoring system. *International Journal of Human-Computer Studies*. 70, 5 (2012), 377–398.

[10] Graesser, A. and McNamara, D. 2010. Self-regulated learning in learning environments with pedagogical agents that interact in natural language. *Educational Psychologist*. 45, 4 (2010), 234–244.

[11] Hausmann, R.G. and Vanlehn, K. 2007. Explaining self-explaining: A contrast between content and generation. *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (Los Angeles, CA, USA, 2007). Springer, 417–424.

[12] Heiner, C., Beck, J. and Mostow, J. 2004. Improving the help selection policy in a Reading Tutor that listens. *Proceedings of the InSTIL/ICALL Symposium on NLP and Speech Technologies in Advanced Language Learning Systems* (Venice, Italy, 2004), 195–198.

[13] Jacobson, M.J. 2008. A design framework for educational hypermedia systems: Theory, research, and learning emerging scientific conceptual perspectives. *Educational technology research and development*. 56, 1 (2008), 5–28.

[14] Jaques, N., Conati, C., Harley, J.M. and Azevedo, R. 2014. Predicting Affect from Gaze Data during Interaction with an Intelligent Tutoring System. *Proceedings of the 12th International Conference on Intelligent Tutoring Systems* (Honolulu, HI, USA, 2014). Springer, 29–38.

[15] Kardan, S. and Conati, C. 2012. Exploring gaze data for determining user learning with an interactive simulation. *Proceedings of the 20th international conference on User Modeling, Adaptation, and Personalization* (Montréal, QC, Canada, 2012). Springer, 126–138.

[16] Kardan, S. and Conati, C. 2015. Providing adaptive support in an interactive simulation for learning: An experimental evaluation. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, South Korea, 2015). ACM, 3671–3680.

[17] Lallé, S., Taub, M., Mudrick, N.V., Conati, C. and Azevedo, R. 2017. The Impact of Student Individual Differences and Visual Attention to Pedagogical Agents during Learning with MetaTutor. *Proceedings of the 18th International Conference on Artificial Intelligence in Education* (Wuhan, China, 2017). Springer (to appear).

[18] McNamara, D.S., Boonthum, C., Levinstein, I.B. and Millis, K. 2007. Evaluating self-explanations in iSTART: Comparing word-based and LSA algorithms. *Handbook of latent semantic analysis*. Psychology Press. 227–241.

[19] Najar, A.S., Mitrovic, A. and McLaren, B.M. 2014. Adaptive Support versus Alternating Worked Examples and Tutored Problems: Which Leads to Better Learning? *Proceedings of the 22nd International Conference on User Modeling, Adaptation, and Personalization* (Aalborg, Denmark, 2014). Springer, 171–182.

[20] Poitras, E.G. and Lajoie, S.P. 2014. Developing an agent-based adaptive system for scaffolding self-regulated inquiry learning in history education. *Educational Technology Research and Development*. 62, 3 (2014), 335–366.

[21] Ritter, S., Anderson, J.R., Koedinger, K.R. and Corbett, A. 2007. Cognitive Tutor: Applied research in mathematics education. *Psychonomic bulletin & review*. 14, 2 (2007), 249–255.

[22] Roll, I., Aleven, V., McLaren, B.M. and Koedinger, K.R. 2011. Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction*. 21, 2 (2011), 267–280.

[23] Roll, I., Wiese, E.S., Long, Y., Aleven, V. and Koedinger, K.R. 2014. Tutoring self-and co-regulation with intelligent tutoring systems to help students acquire better learning skills. *Design Recommendations for Intelligent Tutoring Systems, Volume 2*. U.S. Army Research Laboratory. 169–182.

[24] Shute, V.J. 2008. Focus on formative feedback. *Review of educational research*. 78, 1 (2008), 153–189.

[25] Trevors, G., Duffy, M. and Azevedo, R. 2014. Note-taking within MetaTutor: interactions between an intelligent tutoring system and prior knowledge on note-taking and learning. *Educational Technology Research and Development*. 62, 5 (2014), 507–528.

APPENDIX A

All statistical results for explicit compliance prompts (discussed in Section 6.1). Bold indicates a significant effect.

Prompt	F value	*p*-value	Effect size
Suggest subgoal	**$F_{1,20} = 11.4$**	***p*=.003**	**η_p^2=.27**
Review notes	**$F_{1,20} = 7.43$**	***p*=.013**	**η_p^2 =.16**
Revise summary	**$F_{1,20} = 6.17$**	***p*=.02**	**η_p^2 =.15**
Summarizing	$F_{1,20} = 1.76$	*p*=.20	η_p^2 =.06
Move on subgoal	$F_{1,20} = 0.92$	*p*=.35	η_p^2 =.02
Stay on subgoal	$F_{1,20} = 1.47$	*p*=.24	η_p^2 =.01
Open diagram	$F_{1,20} = 0.71$	*p*=.41	η_p^2 =.08

APPENDIX B

All statistical results for explicit compliance prompts (discussed in Section 6.2). Bold indicates a significant effect.

Prompt	Measure	F value	*p*-value	Effect size
Add subgoal	Window length	$F_{1,3} = .91$	$p = .41$	$\eta_p^2 = .04$
	#fixations on page	**$F_{1,3} = 13$**	**$p = .03$**	**$\eta_p^2 = .29$**
	#SRL strategies	$F_{1,3} = .02$	$p = .90$	$\eta_p^2 = .01$
Move on page	Window length	$F_{1,10} = .00$	$p = .98$	$\eta_p^2 = .00$
	#fixations on page	$F_{1,10} = .03$	$p = .86$	$\eta_p^2 = .00$
	#SRL strategies	$F_{1,10} = .40$	$p = .54$	$\eta_p^2 = .01$
Stay on page	Window length	$F_{1,10} = .34$	$p = .57$	$\eta_p^2 = .01$
	#fixations on page	$F_{1,10} = .07$	$p = .79$	$\eta_p^2 = .03$
	#SRL strategies	$F_{1,10} = .004$	$p = .95$	$\eta_p^2 = .02$

Assessing Computer Literacy of Adults with Low Literacy Skills

Andrew M. Olney
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
aolney@memphis.edu

Dariush Bakhtiari
Department of Educational Psychology, Special Education, and Communication Disorders
Georgia State University
Atlanta, GA 30302
dbakhtiari1@gsu.edu

Daphne Greenberg
Department of Educational Psychology, Special Education, and Communication Disorders
Georgia State University
Atlanta, GA 30302
dgreenberg@gsu.edu

Art Graesser
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
a-graesser@memphis.edu

ABSTRACT

Adaptive learning technologies hold great promise for improving the reading skills of adults with low literacy, but adults with low literacy skills typically have low computer literacy skills. In order to determine whether adults with low literacy skills would be able to use an intelligent tutoring system for reading comprehension, we adapted a 44 task computer literacy assessment and delivered it to 114 adults with reading skills between 3rd and 8th grade levels. This paper presents four analyses on these data. First, we report the pass/fail data natively exported by the assessment for particular computer-based tasks. Second, we undertook a GOMS analysis of each computer-based task, to predict the task completion time for a skilled user, and found that it negatively correlated with proportion correct for each item, $r(42) = -.4,\ p = .01$. Third, we used the GOMS task decomposition to develop a Q-matrix of component computer skills for each task, and using logistic mixed effects models on this matrix identified five component skills highly predictive of the success or failure of an individual on a computer task: function keys, typing, using icons, right clicking, and mouse dragging. And finally, we assessed the predictive value of all component skills using logistic lasso.

Keywords

adult literacy, computer literacy, GOMS, Q-matrix, mixed model, lasso

1. INTRODUCTION

Of adults with the lowest literacy levels, 43% live in poverty, and low literacy costs the U.S. economy $225 billion annually [14]. The need for literacy interventions is matched by the complexity of delivering interventions to this population. Low literacy adults have difficulty attending face to face programs at literacy centers because of work, child care, and transportation [5], and even when these challenges are met, two-thirds of literacy centers have long waiting lists [14]. Adaptive computer-based interventions for literacy hold promise to overcome these challenges. Such interventions can be deployed in homes and local libraries, in addition to literacy centers. However, computer-based interventions raise another question: can adults with low literacy skills use computers well enough to benefit? Several surveys suggest that this might be a problem. The demographics most affected by low literacy are the same demographics least likely to use the Internet (over age 50, making less than $30 thousand a year, and with less than a high school education [1]).

Several decades of research have investigated computer literacy using self-report measures as well as objective tests, i.e. multiple choice, and find that self-report measures tend to exaggerate proficiency while objective tests are more reliable (see [3] for a review). For an adult literacy population, however, multiple-choice tests delivered as print create additional concerns as to whether the questions themselves can be comprehended. Recently a new type of assessment, known as the Northstar Digital Literacy Assessment (the Northstar), has been created that directly measures ability to perform computer tasks [13]. Unlike multiple choice assessments, the Northstar can simulate a computer desktop, use voice prompts to instruct users to perform tasks on that desktop, and then record their mouse clicks and keystrokes to determine if the task has been completed. Almost all of the tasks can be completed without reading by listening to the voice prompt instructions. The few tasks that do involve reading are word recognition tasks rather than sentence reading, e.g. a task to log in may require the user to copy a name and password to the appropriate boxes and so require reading of "Username," "Password," and the corresponding fillers. The Northstar has been adopted as the computer literacy standard for adult basic education in the

state of Minnesota, which further supports its appropriateness for assessing the computer literacy skills of adults with low literacy skills.

The present study investigated the computer literacy skills of adults with low literacy skills for the purpose of developing an intelligent tutoring system for reading comprehension for this population [7]. It includes a set of Northstar items that were collected to cover a range of potential interface and interaction components. In the remainder of the paper we describe the data collection procedure and four analyses performed, including pass/fail frequencies for each task, relation of these frequencies to GOMS-predicted execution times for skilled users, a logistic mixed-model using a Q-matrix decomposition of the tasks into component skills, and a logistic lasso model to assess the predictive value of component skills. From these analyses we identify specific tasks that are problematic for adults with low literacy skills as well as component skills that make it more likely adults with low literacy skills will succeed or fail at a computer-based task.

2. ANALYSIS 1: PROPORTION CORRECT

2.1 Participants

Participants (N = 114) were recruited through adult literacy centers in Atlanta, GA and Toronto, ON, from classes where the reading level was between 3rd and 8th grade. Reading level was determined by the centers using their "business as usual" assessments. Demographic surveys were completed by 90 participants (79% completion rate). Completed surveys indicated that participants were slightly more female than male (55 vs. 35) and that participant age ranged from 17 to 69 ($M = 42.74,\ SD = 13.73$).

2.2 Materials

Forty-four items were selected from four (out of seven) of the Northstar modules available at the time of the study: Basic Computer Skills (21), WWW (13), Windows (6), and Email (4). Task descriptions are given in Table 1. Basic Computer Skills covered such topics as turning a computer on, identifying components of a computer, files and folders, menus, and windows. WWW focused on browser-based activities like searching, search results, browser functionalities, and logging in. Although the Windows module focused on Windows overall, the items selected were fairly generic to any windowed operating system and mostly pertained to desktop applications. Email questions used a webmail interface (browser-based email client) and queried how one would create a new email, send an email, or similar email task. Because Northstar modules are integrated assessments, the Northstar Project compiled the items we selected into a custom assessment for us.

2.3 Procedure

Participants first completed informed consent and then the demographic survey. Both informed consent and demographic survey were read aloud to participants to ensure comprehension. Participants were then asked to sit in front of a computer to take the Northstar assessment. The assessment was delivered in the browser using Adobe Flash. At the start of the assessment, a 3-minute orientation video was played explaining how to answer questions in the assessment. If the participant was confused about what to do, an experimenter was available to answer questions. Each question consisted of an voice prompt defining the task, which was also written at the top of the screen. A replay button was available to repeat the prompt. Participants could select, click, type, drag, etc. on the interface in an attempt to perform the task. If the participant did not know how to complete the task, they could press an "I Don't Know" button, at which point the system scored their attempt as a failure. Attempts were only scored as a success if the participant completed the task in the manner requested in the prompt. The completion of each task initiated the next task until the assessment was complete.

2.4 Results & Discussion

The Northstar records success/failure of each participant on each task, and these data are reported in detail elsewhere [2]. Here we briefly note that the proportion of correct responses for each task is quite wide, ranging from .19 to .98. Tasks in which participants performed particularly well (proportion correct above .80) include identification tasks (e.g. for mouse, keyboard, headphone jack, and websites), turning on a computer or monitor, and common operations like recycling a file, using checkboxes, dragging, scrolling, and using hyperlinks. Tasks in which participants performed poorly (proportion correct below .60) include identification of various keys, double- or right-clicking, typing web addresses, signing into email, and composing email.

The proportion correct results from the Northstar indicate the adults with low literacy skills can power on their device and perform a variety of basic operations. To the extent that these tasks exactly matched tasks that would be performed in a computer-based literacy intervention, like an intelligent tutoring system, this level of results is quite useful. However, for some tasks there is not an exact match, and the implications of the proportion correct results are less clear. For example, difficulties performing tasks using Word, Excel, or webmail may reflect problems with those specific interfaces that may not transfer to other programs. Understanding these more nuanced relationships would require a deeper analysis than is afforded by Northstar's success/failure output.

3. ANALYSIS 2: GOMS MODELING

The purpose of this analysis was to explore whether the success rate of the Northstar tasks could be modeled using GOMS (Goals, Operators, Methods, & Selection rules), a well-known computational technique for modeling expert user performance on a task [10]. GOMS decomposes a particular computer task, e.g. saving a file, into goals and subgoals, perceptual, cognitive, and motor actions in service these goals, methods or sequences of operators that achieve a goal, and selection rules that choose between alternative methods. An important assumption of GOMS is that the users are expert at the computer task in question. Therefore GOMS models of execution time represent the upper bound of performance after a user has learned the interface and practiced it many times. The expert assumption of GOMS is violated in the adult literacy population, making the outcome of this analysis non-obvious. If the GOMS model predictions of execution time were related to our adult's performance, that would provide evidence that GOMS modeling

Table 1: Northstar Tasks

Click on the monitor	Recycle file	Click stop loading
Click on the keyboard	Checkboxes	Select search engines
Click on the system unit	Organize folder options	Google query
Click on the headphone jack	Start menu, lauch program	Google scroll
Click on picture of a mouse	Turn up audio slider	Use hyperlink
Newline key	Mute audio	Maximize window
Caps key	Select browser icons	Minimize window
Shift key	Click on the website	Open Excel
Backspace key	Drag item in browser	Open Word using taskbar
Up arrow	Click on address bar	Close Word
Turn on monitor	Type the web address	Select login and password
Turn on computer	Click homepage button	Choose secure password
Log on to computer	Click browser back button	Sign into email
Double click on Documents	Click browser refresh	Compose email
Right click menu	Click browser forward	

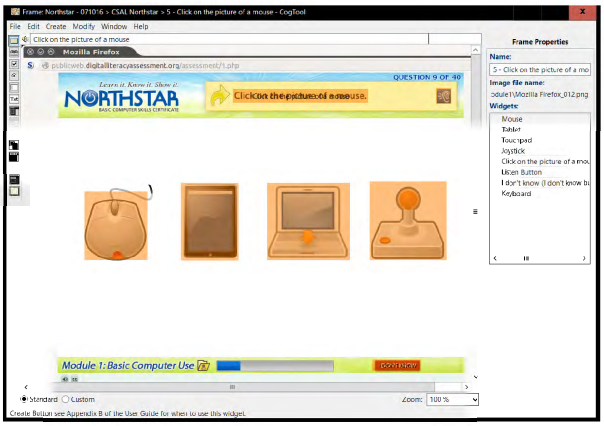


Figure 1: A CogTool annotation of a Northstar task. Annotations appear as semi-transparent orange boxes over the Northstar interface.

has some validity for this population.

3.1 Procedure

The CogTool system was used to perform a GOMS analysis [11, 9]. CogTool allows the easy creation of GOMS models by annotating an existing user interface, and then recording a demonstration of the task against than annotated interface. Figure 1 shows the CogTool interface for the "Click on the mouse" task. For example, when the Northstar task required clicking on an icon, button, or other interface element as in Figure 1, a CogTool button annotation was overlaid on the interface, and then in demonstration mode the modeler would demonstrate the task by clicking on the annotated button. From this demonstration on the annotation, CogTool builds a GOMS model that includes the perceptual, cognitive, and motor tasks required to perform the task. Similar annotations were made for auditory directions, keyboard input, and other kinds of interface actions. Once a task was annotated and demonstrated, a CogTool simulation was run on GOMS model to generate a predicted execution time of expert performance. Annotations, demonstrations, and execution time predictions were performed for all 44 Northstar items used in Analysis 1.

3.2 Results & Discussion

GOMS-predicted execution times for Northstar tasks ranged from 3.0 to 17.1 seconds ($M = 6.88,\ SD = 4.07$). These execution times were significantly negatively correlated with proportion correct, $r(42) = -.40,\ p = .01,\ CI_{95}[-.61, -.10]$, indicating that tasks predicted to take an expert longer to accomplish were more likely to be answered incorrectly by low literacy adults. Tasks that take longer are inherently more complex and require more operations to complete. These results suggest that GOMS has some validity for modeling the performance of adults with low literacy skills even though it was not intended for this purpose. However, by themselves these results convey little additional insight. The GOMS-predicted execution times, generated by CogTool, are still at the task level rather than the component skills required to achieve each task. This is partly because the orientation of CogTool is to produce execution times and partly because of the expert orientation of GOMS. For example, in GOMS the factors involved in clicking a button are the perceptual (size, location) and motor operations involved, but in Northstar, some "buttons" are tapping specific types of knowledge, like identifying hardware, understanding icons, or various keys on a keyboard. The different types of knowledge behind the various CogTool annotations are not represented or considered in the GOMS analysis it provides.

4. ANALYSIS 3: Q-MATRIX & LOGISTIC MIXED MODELS

We would like to understand how the component skills underlying Northstar tasks differentially affect the probability a low literacy adult will perform the task correctly. In educational data mining, component skills are typically modeled using a Q-matrix analysis [4]. In its simplest form, a Q-matrix analysis constructs a problem by skill matrix such that a $cell_{ij}$ in the matrix represents whether $skill_i$ is needed to solve $problem_j$: $cell_{ij} = 1$ if $skill_i$ is needed to solve $problem_j$, and $cell_{ij} = 0$ if $skill_i$ is not needed to solve $problem_j$. Analysis 2 provides a useful guide towards the creation of a Q-matrix for the Northstar tasks, as it has already captured each component action required to perform

Table 2: Component skills coded from GOMS

Component Skill	Probability Correct Given Skill
Checkboxes	.89
Mouse Drag	.86
Hardware Identify	.83
Hardware Function	.78
Complex Scrolling	.74
Browser Functions	.66
Left Click	.64
Use Icons	.61
Double Click	.58
Window Functionality	.56
Program Brands	.55
Desktop Concept	.53
Select Menu	.50
Good Login Info	.50
Login Info	.48
Keyboard Function	.46
Simple Typing	.43
Right Click	.19

each task. What it lacks in some cases, however, is an annotation of the knowledge behind each component action.

4.1 Procedure

The first author recoded the GOMS task annotations with 18 novice-relevant component skills. The coding was done in one pass, and component skills were defined on the fly. Component skills that occurred in only one task were then removed as they offer no predictive utility for other tasks. The appropriateness of the component skills was evaluated by correlating the total number of component skills needed in each task with the GOMS execution time and the proportion correct for the respective task. We used a logistic mixed model to predict the correctness of each participant on each task as a function of the presence of component skills for that task. This analysis addresses the question as to whether there is an effect (main effect) of the presence of component skills on the likelihood that an adult with low literacy skills will be able to perform the task correctly. Using a logistic mixed model in this way has strong similarities to cognitive psychometric models like Diagnostic Classification Models [16] or more specifically a mixed model implementation of linear logistic test models [15].

In the logistic mixed model, random slopes were initially included but failed to converge. Random intercepts for task and participant are theoretically motivated, and backward selection of these effects using Akaike information criterion (AIC) achieved a minimum when these effects were included, indicating that these intercepts should remain in the model. These random intercepts can be considered as per-task difficulty not captured by component skills and per-subject ability, respectively. The initial model that included Left Click was rank deficient, so Left Click, which appears in most tasks, was removed from the final model. Additionally, the total number of component skills in each task (i.e. column sums of the Q-matrix) was initially considered as a predictor of correctness, but was excluded based on extremely high collinearity, having a variance inflation factor of over 40.

4.2 Results & Discussion

The component skills and the conditional probability that a task will be correctly performed if the component skill is present are shown in Figure 2. Total component skills per task was marginally positive correlated with GOMS execution time, $r(42) = .27,\ p = .07,\ CI_{95}[-.02, .53]$, suggesting that tasks with more component skills take longer to perform. Total component skills per task was significantly negatively correlated with proportion correct, $r(42) = -.35,\ p = .02,\ CI_{95}[-.59, -.06]$, indicating that tasks with more component skills are more difficult to perform correctly. The correlation between predicted execution time and proportion correct was not significantly different from the correlation between total component skills and proportion correct, $t(82) = .18,\ p = .86$, indicating that the Q-matrix decomposition of component skills is comparable to the GOMS execution time in terms of its relationship to proportion correctness. Altogether these correlation results provide additional evidence that the Q-matrix decomposition is appropriate.

The logistic mixed model had a marginal R^2 of .18 (fixed effects only) and a conditional R^2 of .47 (including random effects) [12]. We found a positive main effect of Mouse Drag, $\hat{\beta} = 2.06,\ SE = .90,\ p = .02$, such that tasks with a Mouse Drag component were 7.87 times as likely to be answered correctly, $CI_{95}[1.36, 45.50]$, and a marginal main effect of Hardware Identify, $\hat{\beta} = .89,\ SE = .53,\ p = .10$, such that tasks with a Hardware Identify component were 2.44 times as likely to be answered correctly, $CI_{95}[.86, 6.94]$. We found negative main effects for Keyboard Function, $\hat{\beta} = -1.31,\ SE = .51,\ p = .01$, Use Icon $\hat{\beta} = -1.35,\ SE = .55,\ p = .01$, Simple Typing $\hat{\beta} = -1.91,\ SE = .64,\ p = .003$, and Right Click $\hat{\beta} = -3.20,\ SE = 1.34,\ p < .02$, such that tasks with a Keyboard Function component were .27 times as likely to be answered correctly, $CI_{95}[.10, .73]$, tasks with a Use Icon component were .26 times as likely to be answered correctly, $CI_{95}[.09, .75]$, tasks with a Simple Typing component were .15 times as likely to be answered correctly, $CI_{95}[.04, .52]$, and tasks with a Right Click component were .04 times as likely to be answered correctly, $CI_{95}[.00, .56]$.

We found that Mouse Drag was extremely predictive of success. The reason is unclear, but we hypothesize that the frequency of mouse dragging in many computer tasks may have afforded participants the opportunity to become expert in this skill. Mouse dragging has some similarity to swiping on a smartphone or tablet interface, so it may be that expertise with other devices has transferred into the Northstar tasks. Amongst the components that predict failure, perhaps the most intuitive are Keyboard Function and Simple Typing. Typing is a complex skill that takes practice to master. Function keys are difficult in that they don't themselves produce a character, but either operate on a character on the screen (Delete) or work in combination with another key to modify it (Shift). The negative effects associated with Use Icon and Right Click are somewhat surprising. Icons come in many different variations, and so it is possible that the negative Use Icon effect is attributable to a lack of knowledge of specific icons or perhaps to the conventions of icons generally. Right Click is possibly rare and usually brings up a context menu with commands that are often available elsewhere, making it more relevant for power users but perhaps less so to novice users.

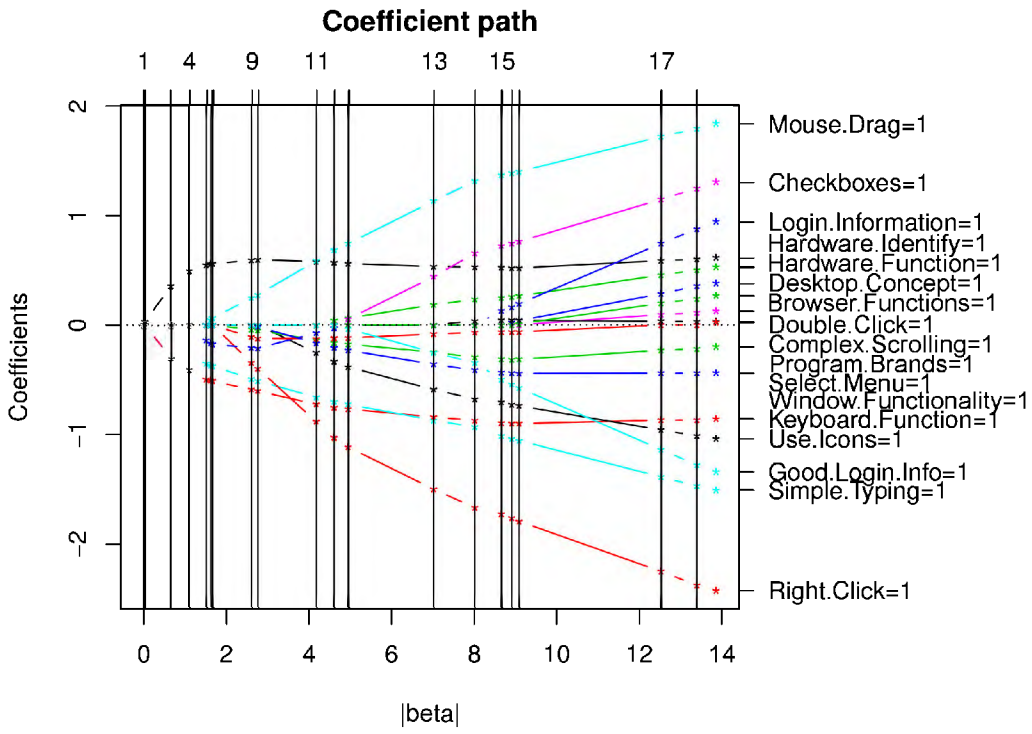


Figure 2: The coefficient path for the lasso model. As the L1 sparsity threshold increases along the x-axis, more coefficients are non-zero.

5. ANALYSIS 4: Q-MATRIX LASSO

Analysis 3 provides a more traditional analysis of significant predictors in our study, but must be interpreted with caution with respect to generalizing to new data. It may be that insignificant predictors in Analysis 3 nevertheless have predictive value on new data. The problems of relying on p-values or criteria like AIC to select variables are well known [8]. To explore the predictive potential of the Q-matrix component skills, we created a lasso model (least absolute shrinkage and selection operator [18]), a form of regression that promotes sparsity (i.e. zero coefficients) and predictive accuracy simultaneously. While not necessarily the best predictive model (cf. gradient boosting [6]), lasso has the advantage of being simple to interpret, and thus our results can guide what variables to use in future models.

5.1 Procedure

A logistic regression base model without random effects was initialized with 17 component skills (Left Click excluded) and submitted to lasso. Because lasso has a free parameter, λ, that controls sparsity of the regression, a lasso analysis varies the level of λ and generates regression coefficient estimates at each level. This sequence of regression coefficients is known as the regularization path. The value of λ that minimized prediction error was estimated using both cross validation and AIC.

5.2 Results & Discussion

The coefficient (regularization) path for the lasso model is shown in Figure 2 and the corresponding AIC curve is shown in Figure 3. In Figure 2, the center line represents coefficients having zero values. As the L1 sparsity threshold ($|beta|$) increases, more coefficients become non-zero. For selecting the optimal λ that minimizes overall prediction error, ten-fold cross validation and AIC yielded congruent results. AIC results are depicted in the curve in Figure 3, which shows that that AIC improves as $|beta|$ increases, coming to a minimum at $|beta| = 13.40$. Accordingly, most coefficients for the optimal lasso model are non-zero.

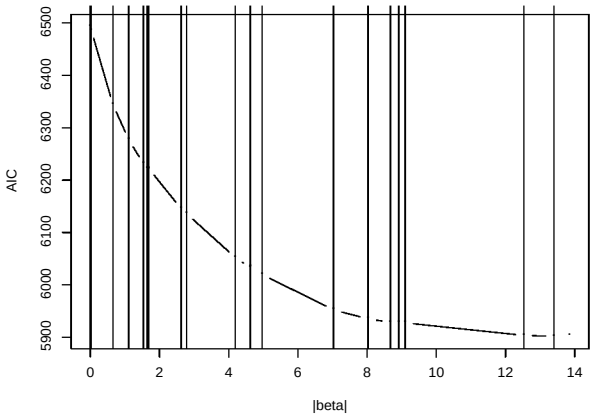


Figure 3: The AIC curve for the lasso model. Lower values of AIC indicate better model fit.

Table 3: Lasso component skill coefficients

Component Skill	$\hat{\beta}$	$exp(\hat{\beta})$
Mouse Drag	1.80	6.02
Checkboxes	1.27	3.55
Login Information	.88	2.41
Hardware Identify	.60	1.82
Hardware Function	.50	1.65
Desktop Concept	.35	1.43
Browser Functions	.24	1.27
Double Click	.11	1.12
Complex Scrolling	.03	1.03
Program Brands	.00	1.00
Select Menu	-.21	.81
Window Functionality	-.44	.64
Keyboard Function	-.86	.42
Use Icons	-1.01	.36
Good Login Info	-1.28	.28
Simple Typing	-1.47	.23
Right Click	-2.39	.09

Table 3 gives the $\hat{\beta}$ coefficients (log odds) for the AIC-optimal model as well as the odds ratio $exp(\hat{\beta})$ for each coefficient. The coefficients converted to odds ratios have the same interpretation as in the logistic mixed model, e.g. tasks with a Mouse Drag component are 6.02 times as likely to be answered correctly as those without. Although the logistic lasso model does not include random intercepts corresponding to task difficulty and subject ability, the magnitudes of coefficients in the logistic lasso are highly comparable to the logistic mixed model. However, the strength of the coefficients in the logistic lasso are weaker, in general, than in the logistic mixed model, suggesting that the logistic mixed model may be slightly over-fitted. For example, according to the logistic mixed model, Mouse Drag tasks are 7.87 times as likely to be answered correctly, but according to the logistic lasso model, Mouse Drag tasks are only 6.02 times as likely to be answered correctly; similarly Right Click containing tasks in the mixed model are .04 times as likely to be answered correctly compared to .09 times as likely in the logistic lasso. These results suggest that while the logistic mixed model might be more appropriate for assessment purposes, as it additionally estimates task difficulty and subject ability, the logistic lasso model might be more appropriate for predicting the effects of component skills on success rates for new tasks.

6. GENERAL DISCUSSION

Together, our results suggest that not only are there specific Northstar tasks that are informative with regard to building an adaptive computer-based intervention for adults with low literacy skills but also that these tasks can themselves be decomposed into component skills that can be further used for this purpose. The main effects of Analysis 3 and coefficient rankings of Analysis 4 are consistent and complimentary with the proportion correct results in Analysis 1. The marginal main effect for Hardware Identify explains the high proportion correctness for identification tasks for mouse, keyboard, and headphone jack, and the main effect for Mouse Drag explains the high proportion correctness for recycling a file (dragging to the Recycle Bin), dragging, and scrolling (by dragging a scroll bar). These correctness-enhancing main effects are also reflected in odds ratios greater than one in Analysis 4. Similarly the main effects for Keyboard Function and Simple Typing explain the low proportion correctness for identifying various keys, typing web addresses, signing into email, and composing email, and these main effects are likewise reflected in odds ratios less than one in Analysis 4. In these cases we infer that the problem is not specific to the interface in question, e.g. email, but rather that there is a deficiency in a component skill needed for the task taking place in the context of that interface.

The implications for building adaptive computer-based interventions for adults with low literacy skills are clear. First, it is important to keep typing to a minimum, either by having users select response options or by using speech recognition. Second, right clicking should be eliminated or at least made optional. Third, icons should be close to icon archetypes. And finally, mouse dragging is a good skill around which to build user interaction. Interestingly, all of these implications seem to point to tablet and smartphone platforms, which have a minimum of typing (and built in speech interfaces), no right clicking, minimal icons in-app, and plenty of swiping/dragging. Moreover, smartphone ownership has been rapidly increasing – now 64% of households earning below $30 thousand own a smartphone [17]. It may be the case that deploying interventions on smartphones and tablets better makes use of both the computer literacy strengths and the material resources of low literacy adults.

7. ACKNOWLEDGMENTS

This research was supported by the Institute of Education Sciences (IES; R305C120001). 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.

8. REFERENCES

[1] M. Anderson and A. Perrin. 13% of Americans don't use the internet. Who are they? Technical report, Pew Research Center, 2016.

[2] D. Bakhtiari, A. Olney, and D. Greeberg. Computer literacy skills of adult learners. In preparation.

[3] J. A. Ballantine, P. M. Larres, and P. Oyelere. Computer usage and the validity of self-assessed computer competence among first-year business students. *Computers & Education*, 49(4):976 – 990, 2007.

[4] T. Barnes, D. Bitzer, and M. Vouk. Experimental analysis of the q-matrix method in knowledge discovery. In *International Symposium on Methodologies for Intelligent Systems*, pages 603–611. Springer, 2005.

[5] H. Beder and P. Medina. Classroom dynamics in adult literacy education. ncsall research brief. Technical report, National Center for the Study of Adult Learning and Literacy, 2002.

[6] J. H. Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002.

[7] A. C. Graesser, Z. Cai, W. O. Baer, A. M. Olney,

X. Hu, M. Reed, and D. Greenberg. Reading comprehension lessons in AutoTutor for the Center for the Study of Adult Literacy. In S. A. Crossley and D. S. McNamara, editors, *Adaptive Educational Technologies for Literacy Instruction.*, pages 288–293. Routledge, 2016. DOI: 10.4324/9781315647500 DOI: 10.4324/9781315647500.

[8] F. Harrell. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Graduate Texts in Mathematics. Springer, 2001.

[9] B. E. John. Cogtool: Predictive human performance modeling by demonstration. In *Proceedings of the 19th Conference on Behaviour Representation in Modeling and Simulation*, pages 83–84, 2010.

[10] B. E. John and D. E. Kieras. The goms family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 3(4):320–351, 1996.

[11] B. E. John, K. Prevas, D. D. Salvucci, and K. Koedinger. Predictive human performance modeling made easy. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 455–462, New York, NY, USA, 2004. ACM.

[12] S. Nakagawa and H. Schielzeth. A general and simple method for obtaining r2 from generalized linear mixed-effects models. *Methods in Ecology and Evolution*, 4(2):133–142, 2013.

[13] N. D. L. Project, 2016.

[14] ProLiteracy. U.S. adult literacy facts. Technical report, 2017.

[15] F. Rijmen, P. D. Boeck, and K. U. Leuven. The random weights linear logistic test model. *Applied Psychological Measurement*, 26(3):271–285, 2002.

[16] A. A. Rupp and J. L. Templin. Unique characteristics of diagnostic classification models: A comprehensive review of the current state-of-the-art. *Measurement: Interdisciplinary Research and Perspectives*, 6(4):219–262, 2008.

[17] A. Smith. Record shares of americans now own smartphones, have home broadband. Technical report, Pew Research Center, 2017.

[18] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

Towards reliable and valid measurement of individualized student parameters

Ran Liu
Human-Computer Interaction Institute
Carnegie Mellon University
ranliu@cmu.edu

Kenneth R. Koedinger
Human-Computer Interaction Institute
Carnegie Mellon University
koedinger@cmu.edu

ABSTRACT

Research in Educational Data Mining could benefit from greater efforts to ensure that models yield reliable, valid, and interpretable parameter estimates. These efforts have especially been lacking for individualized student-parameter models. We collected two datasets from a sizable student population with excellent "depth" – that is, many observations for each skill for each student. We fit two models, the Individualized-slope Additive Factors Model (iAFM) and Individualized Bayesian Knowledge Tracing (iBKT), both of which individualize for student ability and student learning rate. Estimates of student ability were reliable and valid: they were consistent across both models and across both datasets, and they significantly predicted out-of-tutor pretest data. In one of the datasets, estimates of student learning *rate* were reliable and valid: consistent across models and significantly predictive of pretest-posttest gains. This is the first demonstration that statistical models of data resulting from students' use of learning technology can produce reliable and valid estimates of individual student learning rates. Further, we sought to interpret and understand what differentiates a student with a high estimated learning rate from a student with a low one. We found that learning rate is significantly related to estimates of student ability (prior knowledge) and self-reported measures of diligence. Finally, we suggest a variety of possible applications of models with reliable estimates of individualized student parameters, including a more novel, straightforward way of identifying wheel spinning.

Keywords

Explanatory models, model interpretability, individualized parameters, 3, Additive Factors Model, individualized Bayesian Knowledge Tracing

1. INTRODUCTION

In Educational Data Mining, statistical models are typically evaluated based on fit to overall data and/or predictive accuracy on test data. While this is an important initial step in evaluating the contributions of advancements in statistical and cognitive modeling, research in the field could benefit from greater efforts to ensure that models are reliable and valid. More reliable and valid models offer more explanatory power, contributing to the advancement of learning science. They also inspire greater confidence that deploying model advancements in future tutoring systems will genuine result in the hypothesized improvements to learning.

Some recent work has been done towards interpreting, validating, and acting upon cognitive/skill modeling improvements [7, 8, 10, 11, 17]. Educational data mining efforts oriented around personalizing student constructs [3, 12, 13, 14, 18], however, have remained focused on improving predictive accuracy and/or demonstrating hypothetical time savings. Little has been done to validate or understand the estimates that models with individualized or clustered student parameters produce. Anecdotally, efforts to do so have shown that these individualized student parameter estimates, or discovered student clusters, are often difficult to interpret.

It is especially critical to examine the reliability and validity of parameter estimates for modeling advancements that dramatically increase the parameter count, as is generally true for individualized student-parameter models. More parameters create greater degrees of freedom and increase the likelihood that the model may be underdetermined by the data.

We focus on the question: To what degree can we trust a model's parameter estimates to correctly represent the constructs they are supposed to?

Key to expecting reliable, valid estimates of student-level constructs is not just big data in the "long" sense, but big data in the "deep" sense. Oftentimes, the datasets used in secondary analyses in EDM are large in terms of total number of students (or total observations) but highly sparse in terms of observations per skill, per student. These features make it difficult to get reliable measurements of constructs at the individual student level, particularly constructs related to learning over time.

Here, we collected two datasets from a sizable student population (196 students) with excellent "depth" – that is, many observations for each skill for each student. We then fit two models that individualize for student ability and student learning rate (the Individualized-slope Additive Factors Model [9] and Individualized Bayesian Knowledge Tracing [18]). We assess the models' fit to data and predictive accuracy. We also move beyond these metrics to examine the reliability of the models' estimates of student ability and student learning rate. Additionally, we externally validate the parameter estimates against out-of-tutor assessment data.

We further interpret and understand the constructs by visualizing representative student learning trajectories, examining the relationship between estimated student ability and student learning rate, and the relationship between those constructs and self-reported data on motivational attributes. Finally, we propose some useful applications of reliable and valid individualized student-parameter models, including a new way to detect wheel spinning.

2. PRIOR WORK

Prior work on individualizing student parameters has focused on variants of Bayesian Knowledge Tracing (BKT) [3]. This work includes modeling the parameters separately for each individual student instead of separately for each skill [3], individualizing the P(Init) ("initial knowledge") parameter for each student [13], and individualizing both P(Init) and P(Learn) ("learning rate") to the

base BKT model [18]. These models have generally focused on assessing predictive accuracy improvements relative to their respective non-individualized baseline models.

There have also been some "time savings" analyses [12, 18] that evaluate the hypothetical real world impact that individualizing statistical model fits could have. These analyses report the effect of fitting individualized BKT models, compared to traditional BKT, on the hypothetical number of under- and over- practice attempts that would be predicted for each student. Results generally have indicated that many more practice opportunities are needed for models to infer the same level of knowledge when using whole-population parameters rather than individual student parameters. These analyses show that individualized models differ in their hypothetical decision points if they were to be applied to drive mastery-based learning, but they do not in and of themselves interpret the individualized parameter estimates, nor do they assess the reliability and validity of such estimates.

In a previous effort to better understand individualized student learning rate parameters [9], we examined predictive accuracy and parameter reliability in an extension of the Additive Factors Model [2] applied to existing educational datasets. We did not find evidence that individualizing student rate parameters consistently improved predictive accuracy improvements, nor could we validate the parameter estimates on out-of-tutor assessment data. However, the datasets we analyzed either contained a small number of students or were largely sparse in observations for student-skill pairs, with the exception of two datasets. These two datasets happened to be the ones on which the Individualized-slope Additive Factors Model *did* achieve higher predictive accuracy. Thus, we wondered if the sparsity of the datasets were the primary limitation, rather than the modeling advancement itself. This idea is corroborated by the fact that pooling students into "groups" rather than generating individualized estimates worked well on those datasets [9].

For the present modeling work, we collected our own data in order to ensure the data features that we believe are necessary for reliable, valid, and potentially meaningful estimates of constructs at the individual student level.

3. METHODS

It is common in EDM to do secondary analyses across multiple datasets. However, it can be difficult to find datasets that (1) contain a sizable number of students, (2) contain many observations for each skill for each student (i.e., are not sparse), (3) contain students spanning a range of abilities in the domain covered by the tutor, and (4) contain data from out-of-tutor assessment data that is well-mapped to the content in the tutor.

For the present work, we wanted to use as close to an "ideal" dataset as possible for estimating student parameters. We collected our own dataset with a sizable number of students (196), many observations (5-50, depending on the skill) for each skill for each student. In addition, we ensured that a wide range of student ability levels was represented in our data to allow for the possibility that models could capture this variability.

3.1 Data Collection

196 students, spanning 10 classes taught by three different teachers, enrolled in high school geometry participated in two studies conducted about a month apart. A range of student abilities were included in the study. Two of the 10 classes were "Honors" and three of the 10 classes were "Inclusion". Honors classrooms are intended for students who have strong theoretical interests and abilities in mathematics. Inclusion classrooms are "general education" classrooms designed to provide the opportunity for individuals with disabilities and special needs to learn alongside their non-disabled peers.

Students spent five consecutive days participating in each study during their regular geometry class periods. On the first and last days, they took a computerized pretest and posttest, respectively. During the middle three days, they worked within an intelligent tutoring system [19] designed to give them practice on their current chapter's content. This procedure applied to both studies, one of which covered the students' Chapter 3 content (Parallel Lines Cut by a Transversal, Angles & Parallel Lines, Finding Slopes of Lines, Slope-Intercept Form, Point-Slope Form) and the other of which covered the students' Chapter 4 content (Classifying Triangles, Finding Measures of Triangle Sides & Angles, Triangle Congruence Properties). Figure 1 shows an example problem interface from the intelligent tutoring system, which was designed using Cognitive Tutor Authoring Tools [1].

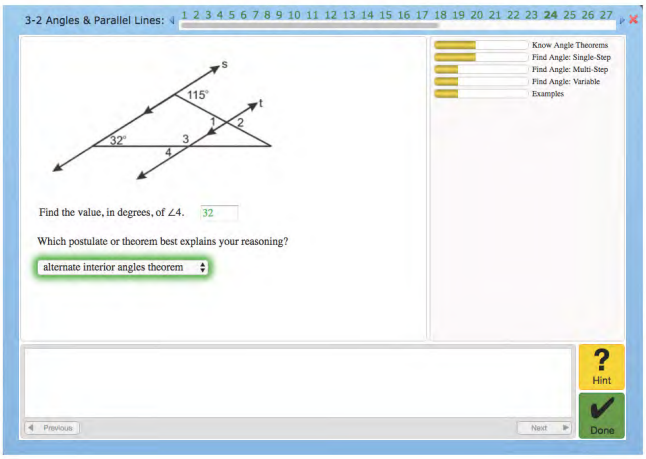


Figure 1. Example problem interface from the intelligent tutoring system used for data collection.

We also collected self-report survey data on motivational factors falling along three dimensions. These were Competitiveness (e.g., "In this unit, I am striving to do well compared to other students" and "In this unit, I am striving to avoid performing worse than others"), Effort (e.g., "I am striving to understand the content of this unit as thoroughly as possible" and "I work hard to do well in this class even if I don't like what we are doing"), and Diligence (e.g., "when class work is difficult, I give up or only study the easy parts" [inverted scale] and "I am diligent"). Self-report measures were indicated on a Likert scale from 1-7.

A key reason we collected two datasets, covering two distinct chapters of the curriculum, is that we were interested in investigating the consistency of student-level parameter estimates across different content, time, and contexts. We discuss this further, along with preliminary results, in Section 4.4.1.

3.2 Statistical Models

3.2.1 The Individualized-slope Additive Factors Model (iAFM)

The Additive Factors Model (AFM) [2] is a logistic regression model that extends item response theory by incorporating a growth or learning term.

$$\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \theta_i + \sum_{k \in KCs} Q_{jk}(\beta_k + \gamma_k T_{ik}) \quad (1)$$

This statistical model (Equation 1) gives the probability p_{ij} that a student i will get a problem step j correct based on the student's baseline ability (θ_i), the baseline easiness (β_k) of the required knowledge components on that problem step (Q_{jk}), and the improvement (γ_k) in each required knowledge component (KC) with each additional practice opportunity. This KC slope, or "learning rate," parameter is multiplied by the number of practice opportunities (T_{ik}) the student already had on it. Knowledge components (KCs) are the underlying facts, skills, and concepts required to solve problems [6].

Individualized-slope AFM (iAFM) builds upon this baseline model by adding a per-student learning rate parameter (δ_i). This parameter represents the improvement (δ_i) by student i with every additional practice opportunity with the KCs required on problem step j.

$$\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \theta_i + \sum_{k \in KCs} Q_{jk}(\beta_k + \gamma_k T_{ik} + \delta_i T_{ik}) \quad (2)$$

The KC and student learning rate parameters are both multiplied by the number of opportunities (T_{ik}) the student already had to practice that KC.

3.2.2 Individualized Bayesian Knowledge Tracing (iBKT)

Bayesian Knowledge Tracing (BKT [3]) is an algorithm that models student knowledge as a latent variable using a Hidden Markov Model. The goal of BKT is to infer, for each skill, whether a student has mastered it or not based on his/her sequence of performance on items requiring that skill. It assumes a two-state learning model whereby each skill is either *known* or *unknown*. There are four parameters that are estimated in a BKT model: the initial probability of knowing a skill a priori – p(Init), the probability of a skill transitioning from not known to known state after an opportunity to practice it – p(Learn), the probability of slipping when applying a known skill – p(Slip), and the probability of correctly guessing without knowing the required skill – p(Guess). Fitting BKT produces estimates for each of these four parameters for every skill in a given dataset. BKT models are usually fit using the expectation maximization method (EM), Conjugate Gradient Search, or discretized brute-force search.

Individualized Bayesian Knowledge Tracing (iBKT [18]) builds upon this baseline BKT model by individualizing the estimate of the probability of initially knowing a skill, p(Init), and the transition probability, p(Learn), for each student. To accomplish the student-level individualization of these parameters, each of them is split into skill- and student-based components that are summed and passed through a logistic transform to yield the final parameter estimate. Details on the decomposition of p(Init) and p(Learn) into skill- and student-based components are described in [18].

4. RESULTS

4.1 Model Fit & Predictive Accuracy

As a first pass evaluation of the two individualized models, we assessed them using Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), which are standard metrics for model comparison, and 10 independent runs of split-halves cross validation (CV). Although 10-fold cross validation has been popular in the field, [4] showed that it has a high type-I error due to high overlap among training sets and recommended at least 5 replications of 2-fold CV instead.

Here, the comparison of interest is each individualized model against its non-individualized counterpart. We do not encourage a literal comparison between the predictive accuracies of the two classes of models due to differences in whether they use incoming test data towards their predictions on later test data (BKT/iBKT do, and AFM/iAFM do not).

Both iAFM and iBKT outperform their non-individualized counterparts by all metrics, with the exception of BKT having a better BIC value than iBKT for the Chapter 4 dataset. This is not surprising, as BIC is known to over-penalize for added parameters. We recommend cross validation as a better indicator that iBKT is the true better fitting model in this case.

Counter to the majority of findings reported in [9], iAFM achieved higher predictive accuracy than AFM in both datasets here. This further supports the idea that the "depth" of the dataset is a critical factor in whether an individualized student-parameter model can explain unique variance in the data.

Table 1. Summary of Model Fit and Predictive Accuracy metrics comparing AFM vs. iAFM and BKT vs. iBKT. Cross-validation values are mean RMSE values across 10 runs, with standard deviations included in parentheses.

Data Set	Model	AIC	BIC	CV *Test* RMSE (10-Run Average)
Ch. 3	AFM	57229	57283	0.38440 (0.0039)
	iAFM	**55931**	**56003**	**0.37868 (0.0044)**
	BKT	66714	67473	0.4222 (0.0005)
	iBKT	**56325**	**60479**	**0.3777 (0.0006)**
Ch. 4	AFM	18059	18106	0.41037 (0.0048)
	iAFM	**17863**	**17925**	**0.40789 (0.0050)**
	BKT	19908	**20376**	0.44091 (0.0014)
	iBKT	**18285**	21809	**0.40725 (0.0018)**

4.2 Reliability of Student Parameters

Next, we examined the degree to which we can rely on these parameters to reasonably estimate the constructs that they should be estimating. We believe that a strong relationship between the parameter estimates of two statistical models with entirely different architectures is a high bar for testing reliability. That is, if a student genuinely displayed evidence of high overall ability in a dataset (relative to his/her peers), then both iAFM and iBKT should estimate that to be the case.

Because of known and observed nonlinear relationships between logistic regression and Bayesian Knowledge Tracing parameter estimates, we measured correlation based on Spearman's coefficient (r_s), which is based on rank order.

We observed strong and statistically significant correlations between iAFM Student Intercept and iBKT Student p(Init) parameter estimates (Figure 2, top row). We also observed a strong and statistically significant correlation between iAFM Student Slope and iBKT Student p(Learn) parameter estimates for one of the two datasets (Chapter 4). This correlation was much milder, though still significant, for the other dataset (Chapter 3).

We hypothesize that this difference between datasets may be due to the presence of more difficult KCs in Chapter 4. A dataset with more difficult items should provide more sensitive measures of individual differences in improvement, since it avoids ceiling effects. Indeed, this was the case: the mean KC easiness parameter estimate (β_k) for chapter 4 was 0.799 (which translates to a

probability of 0.69), compared to 1.253 for chapter 3 (which translates to a probability of 0.78). When students are practicing many opportunities at ceiling (which was the case in particular for chapter 3, based on exploratory analyses of the data), the individualized models will often assign them a lower "learning rate" due to an essentially flat learning trajectory.

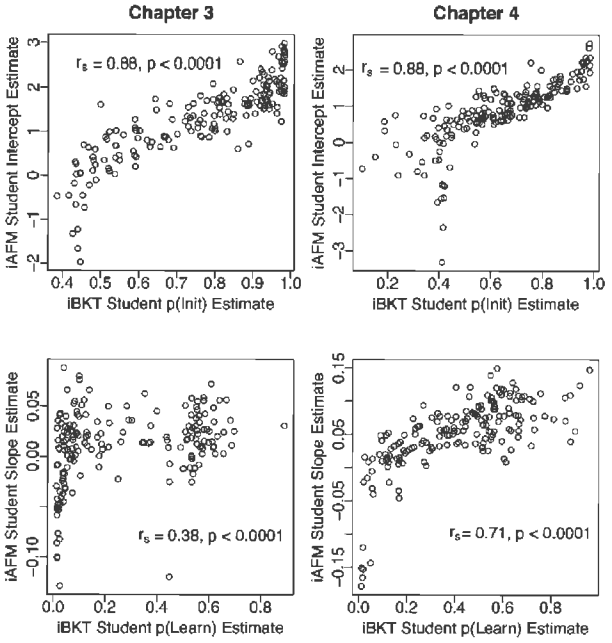


Figure 2. Relationships between iAFM Student Intercept and iBKT Student p(Init) parameter estimates (top row), and between iAFM Student Slope and iBKT Student p(Learn) parameter estimates (bottom row), for the two datasets.

4.3 Validity of Student Parameters

To assess the validity of student parameter estimates, we related them to out-of-tutor assessments of the relevant student constructs. In this case, we validated parameter estimates using pretest and posttest assessment data collected in the study.

4.3.1 Estimates of Student Ability

The Student Intercept (θ_i) parameter of iAFM and the Student p(Init) parameter of BKT are designed to estimate baseline student ability, as least for the knowledge domain represented in the dataset. To validate the models' estimates of this construct, we examined relationships between the model estimates and students' pretest scores, which are an out-of-tutor assessment of student initial ability for the skills covered by the tutor.

We report standard Pearson correlation coefficients here, since the relationships between pretest scores and the parameter estimates did not appear to be particularly nonlinear.

Figure 3 illustrates a summary of these relationships. Both models' estimates of the student ability construct were strongly and significantly correlated with pretest scores.

In addition, adding an individualized student slope *improved* the validity of the model's estimate of student ability (a parameter that's modeled in both AFM and iAFM). We compared the correlations between AFM's intercept estimates to pretest scores (Chapter 3: r = 0.62, p < 0.0001, Chapter 4: r = 0.58, p < 0.0001) to iAFM's intercept estimate / pretest score correlations (Chapter 3: 0.74, p < 0.0001, Chapter 4: r = 0.66, p < 0.0001).

This has several interesting implications for educational applications. First, it suggests that formative assessment via modeling of process data as learning unfolds is a reasonable method of assessment.

It also suggests that detailed assessment data (e.g., from a pretest) could be used to reasonable effect to improve different students' "on-line" estimates of students' knowledge of KCs. For example, combining KC parameter estimates (derived from model-fitting to prior domain-relevant data) with student intercept priors based on pretest assessment data would allow a model like AFM to generate individualized predictions of how much each student needs to practice to reach mastery.

In addition, these results suggest that individualized BKT models could use pretest assessment data to "set" reasonably valid student-specific p(Init) values before collecting any within-tutor data from those students.

In considering the degree to which these results may generalize, it is important to note that the pretests in the present datasets were specifically designed to map closely to the practice problems in the intelligent tutor. Pretests contained 1-2 questions for each KC that was practiced in the tutor, and the items were similar to those encountered within the tutor.

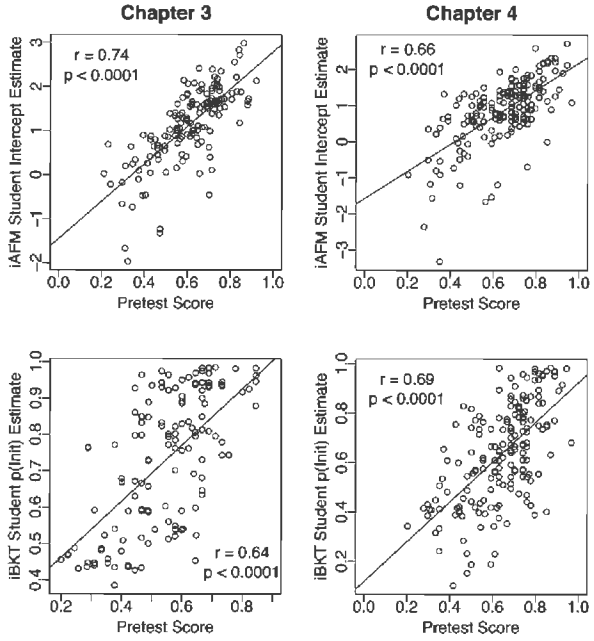


Figure 3. Relationships between out-of-tutor pretest scores and iAFM/iBKT estimates of student ability based on within-tutor data.

4.3.2 Estimates of Student Learning Rate

Given that the only external assessment data collected were a pretest and posttest, we sought to validate the construct of student learning rate (as estimated by the models) on pretest-posttest gains. Students were given roughly the same amount of time to engage with the tutors, so those with accelerated learning rates might be expected to gain more knowledge in the time available.

Thus, we examined the degree to which student learning rate estimates predicted pretest-posttest gains while controlling for pretest scores. We controlled for pretest scores because they have been shown to negatively predict learning gains due to assessment

ceiling effects. That is, students who start out performing well on the pretest have less "room for improvement".

For the Chapter 3 dataset, iAFM Student Slope (δ_i) estimates did not significantly predict learning gains. In a linear regression predicting pretest-posttest gains, pretest scores were a significant predictor (β=-0.189, p=0.005) and Student Slope estimates were not (β=0.396, p=0.144). iBKT Student p(Learn) estimates did not significant predict learning gains. In a linear regression predicting pretest-posttest gains, pretest scores were a significant predictor (β=-0.226, p=0.005) and Student Slope estimates were not (β=0.062, p=0.218).

For the Chapter 4 dataset, iAFM Student Slope (δ_i) estimates significantly predict learning gains. In a linear regression predicting pretest-posttest gains, pretest scores (β=-0.641, p<0.0001) and Student Slope estimates (β=0.576, p=0.007) were both significant predictors. iBKT Student p(Learn) estimates also significantly predict learning gains. In a linear regression predicting pretest-posttest gains, pretest scores (β=-0.645, p<0.0001) and p(Learn) estimates (β=0.133, p=0.004) were both significant predictors.

For one of the two units (Chapter 4), we observed that student learning rate estimates were validated on external assessments of learning gain. Interestingly, this is the same unit for which we observed a strong cross-model reliability in student learning rate estimates. Thus, we have converging evidence that student learning rates estimates for the Chapter 4 dataset are both reliable and valid.

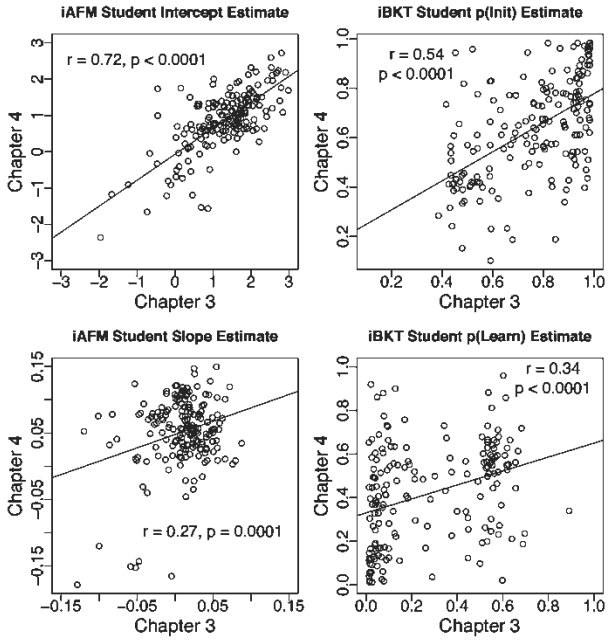


Figure 4. Relationships between student parameter estimates across the two datasets (same student population).

4.4 Towards Understanding & Using Student Parameter Estimates

4.4.1 Consistency of individual student constructs across datasets

A core motivating question for collecting two datasets on the same group of students was: How consistent are iAFM and iBKT model estimates of the student ability and student learning rate constructs across units?

Figure 4 summarizes this relationship. Estimates of student ability are fairly consistent, especially as estimated by iAFM. It seems sensible to interpret this as suggesting that overall student ability on Chapter 3 content is strongly related to overall student ability on Chapter 4 content, as we have shown estimates of student ability to be both reliable and valid.

Estimates of student learning rate are less consistent. This may either be due to the fact that Chapter 3 estimates of student learning rate were neither very reliable nor very valid. Alternatively, the differences in student learning rate estimates across the two chapters may also be due to the fact that students genuinely learn different material at different rates. Unfortunately, we cannot resolve this question with the present data. We are currently collecting more datasets from this same group of students. If we obtain more reliable and valid student learning rate estimates in future data from this group of students, we can more confidently address this question in future research.

4.4.2 Understanding student learning rate estimates

Given that we established the reliability and validity of iAFM and iBKT's parameter estimates for the Chapter 4 dataset were reasonably reliable and valid, we sought to dig deeper into the explanatory power of these estimates. To this end, we conducted exploratory analyses on the Chapter 4 data to (1) visualize the learning trajectories of students with the highest vs. lowest estimated learning rates, (2) understand the relationships between estimated learning rates and prior-knowledge and motivational factors, and (3) understand the degree of variability in estimated learning rate across students.

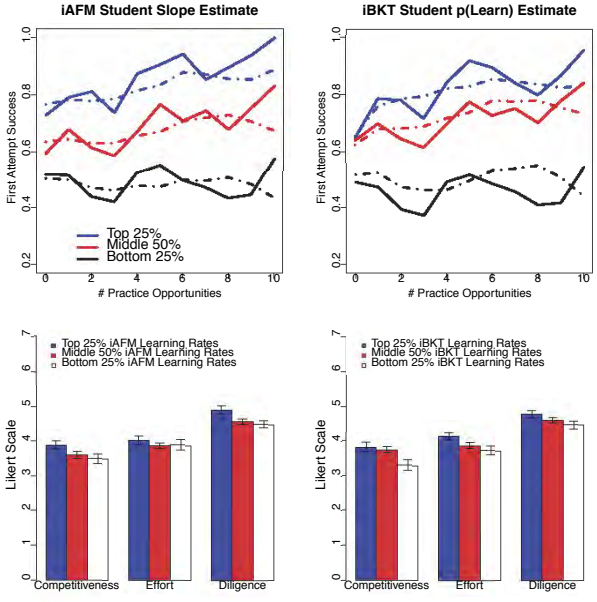


Figure 5. *Top Row*: Early-opportunity learning trajectories of students, grouped based on iAFM (Left) and iBKT (Right) estimated learning rates. Solid lines are actual data; dotted lines are each respective model's predicted performance. *Bottom Row*: Mean self-report Likert scale ratings of questions measuring dimensions of competitiveness, effort, and diligence. Grouped based on iAFM (Left) or iBKT (Right) estimated learning rates. Error bars show standard errors on the means.

Figure 5 (top row) shows the aggregate learning trajectories for students split based either on their iAFM Student Slope estimates (top left) or their iBKT Student p(Learn) estimates (top right). The top 25% of student parameter estimates are plotted in blue, the middle 50% (between 1st and 3rd quartiles) are plotted in red, and the lower 25% are plotted in black. Dotted lines represent each respective model's *predicted* earning trajectories.

One striking pattern, especially in the iAFM learning trajectories (top left), is the apparent relationship between average success on initial practice opportunities (i.e., prior knowledge) and estimated learning rate through the remaining opportunities. This observation is corroborated by a strong and significant correlation between iAFM Student Intercepts and iAFM Student Slopes ($r=0.78$, $p<0.0001$). One might interpret this to suggest that students who enter into the tutor with greater prior knowledge will be poised to gain more from the tutor (i.e., "the rich get richer"). Alternatively, students may have higher overall knowledge *because* they are fast learners. There may also be individual trait-based variables that positively drive both learning rate and overall achievement.

To explore the relationships between measures of traits relevant to learning, we analyzed self-report survey data grouped by three factors (as described in Section 3.1): Competitiveness, Effort, and Diligence. The relationship between these measures and the high, medium, and low learning rate estimates from iAFM and iBKT are shown in Figure 5 (bottom row). There appears to be a relationship between the means of each self-report measure and the general range that the learning rate estimate falls in.

We analyzed the continuous relationship between students' mean self-report rating along each dimension and their iAFM learning rate estimates. In a linear regression predicting iAFM Student Slopes, Competitiveness and Effort were not significant predictors but Diligence ($\beta=0.016$, $p=0.007$) was. In a similar linear regression predicting iAFM Student Intercepts, again Diligence was the only significant predictor ($\beta=0.02$, $p=0.04$). Thus, among self-reported measures, the strongest dimension predicting both student ability/prior knowledge *and* student learning rate was the Diligence measure. Future work using causal modeling is warranted to discover the true nature of causality among these student-level constructs.

Finally, we investigated the degree of variability in estimated learning rate across students. The first quantile of student learning rates from iAFM is 0.03 logits and the third quantile of rates from iAFM is 0.08 logits. These can be conceptualized as canonical "slow" and "fast" learners. If we were to assume starting at around 70% performance (which comes from the model's global intercept estimate), it would take the "slow" (0.03 logits) student approximately 25 opportunities to reach mastery (defined as 85%, the performance equivalent of a p(Know)=0.95, factoring in the guess and slip probabilities we used in the actual tutor). It would take the "fast" (0.08 logits) student approximately 11 opportunities to reach the same place.

4.4.3 Identifying wheel spinners

The current definition of "wheel spinning" put forth in the Educational Data Mining community is the "phenomenon in which a student has spent a considerable amount of time practicing a skill, yet displays little or no progress towards mastery" [5]. There has been some controversy around the ideal way to measure mastery (e.g., 3 corrects in a row vs. reaching a certain p(Know) in knowledge tracing). Furthermore, some students may be classified as wheel spinners based on not mastering in a certain number of opportunities but they may still be making progress.

We propose that reliable and validated estimates of individual student learning rate parameters, combined with KC learning rate parameters, could be used to estimate wheel spinning student/KC pairs in way that is agnostic to mastery status. Specifically, if the combined student and KC learning rate parameters in iAFM predict *no* improvement or negative improvement across additional practice opportunities, and aren't already at a high level of performance on their first opportunity (here we considered this to be 80% or above), we could consider the student to be wheel spinning on the KC. This method of estimating wheel spinning would be particularly useful for datasets with sparse data on some student-KC pairs, as it is not performance-dependent after the model has been fit to the full dataset.

Based on this operationalized definition, we found that approximately 15% of student-KC pairs in the Chapter 4 dataset are estimated to be wheel spinning. That is, those students are not making progress on those KCs. This is a substantially lower estimate than the 25% reported by a recent wheel spinning detector in [5]. An interesting route for future work would be to do a direct comparison of the wheel spinning detector presented in [5] and our proposed student/KC learning rate identifier within the same dataset. This would allow for testing the possibility that some students who are still making progress, albeit extremely slowly, may be prematurely labeled as "wheel spinners" by [5].

5. SUMMARY & LIMITATIONS

Previous efforts towards more explanatory, interpretable, and actionable modeling advancements in the realm of skill/knowledge component model discovery have been promising in their potential and demonstrated impact on learning science and education. The present paper represents a novel effort to bring these deeper modeling approaches, focused on ensuring explanatory power, to the realm of individualized student-parameter models.

Towards improving the reliability and validity of individualized student estimates, we collected two datasets from the same student population. Both datasets were "deep" along the dimension of student-KC observations. We fit iAFM and iBKT to both datasets and showed that the models outranked their non-individualized counterparts in terms of fit to data and predictive accuracy. Importantly, we moved beyond these metrics to show that estimates of student ability were highly reliable (iAFM and iBKT yielded strongly correlated estimates) and valid (estimates significantly predicted pretest data).

This demonstration of confidence in the student ability estimates from iBKT, but even more so iAFM, has promising implications for the possibility of individualizing the student models that determine mastery in intelligent tutoring systems at *least* in terms of overall student ability/knowledge. Our results also suggest that it would be reasonable to fix such student ability parameters, or set priors on them, based on either well-mapped pretest assessment data or prior (deep) data from those students' learning.

We also showed that estimates of student learning rate per practice opportunity were reliable and valid in one of the two datasets (Chapter 4). This is the first evidence, to our knowledge, of obtaining both reliable and valid student learning rates through a statistical model with *individualized* student parameters. We believe that this success is largely related to the amount and quality of per-student data we collected.

With the confidence of having reliable and valid parameter estimates, we then proceeded to further investigate potential explanations for differences in student learning rates within the

Chapter 4 dataset. We found a strong and significant relationship between student ability and improvement rate as well as an additional effect of diligence, based on self-report measures. Further research is warranted to distill the causal relationships between these constructs.

Knowing that a model's estimates of individualized student parameters not only fit data well, but are reliable and valid, provides greater confidence for applying the model to (1) interpret the parameter estimates to understand characteristics of students, and (2) use the model to individualize the trajectory of mastery estimation for future students.

Even though both iBKT and iAFM outperformed their non-individualized counterparts in predicting performance in the Chapter 3 dataset, we did not find strong evidence of reliability and validity of the student-specific parameter estimates. Thus, we did not rely on that dataset to help us understand individual differences in learning rates. For the same reason, we could not confidently attribute the differences, in estimated student learning rates across the datasets, to *true* differences in students' learning rates for the two chapters' material.

Although considering reliability and validity of models' parameter estimates sets a higher bar than predictive accuracy for evaluating modeling advances, we believe those to be important characteristics of a model that is to be explanatory, interpretable, and/or actionable. Here, we have demonstrated that with a sufficiently good dataset, iAFM and iBKT are individualized student models that *can* produce reliable and valid parameter estimates.

Since our present work was limited to two datasets on one population of students, it is unclear the degree to which our modeling results will generalize, especially given that at least iAFM does not produce reliable, valid parameter estimates on more sparse datasets [9]. In addition, these results are limited to two specific statistical models produce individualized estimates student-level parameters, with a particular focus on individual differences in learning rate. There are other classes of models that could be extended to estimate differences in learning rate: for example, producing individualized estimates of the differential effects of success versus failure [15]. This would be an interesting focus for future work on this topic.

Nevertheless, we have laid a foundation of methodology by which reliability and validity of parameter estimates, whether student- or KC-level, can be assessed. We have also demonstrated ways of using the reliable and valid student parameter estimates from iAFM and iBKT to yield interesting insights about student learning.

6. ACKNOWLEDGMENTS

We thank the Institute of Education Sciences for support to RL (training grant #R305B110003) and the National Science Foundation for support to Carnegie Mellon University's LearnLab (#SBE-0836012).

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] Cen, H., Koedinger, K.R., & Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. Intelligent Tutoring Systems, 164-175.

[3] Corbett, A.T., & Anderson, J.R. (1995). Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction, 4, 253-278.

[4] Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation, 10(7), 1895–1923.

[5] Gong, Y. & Beck, J. (2015). Towards Detecting Wheel-Spinning: Future Failure in Mastery Learning. In Proceedings of Learning At Scale '15.

[6] Koedinger, K.R., Corbett, A.C., & Perfetti, C. (2012). The Knowledge-Learning-Instruction (KLI) framework: Bridging the science-practice chasm to enhance robust student learning. Cognitive Science, 36(5), 757-798.

[7] Koedinger, K.R., McLaughlin, E.A., & Stamper, J.C. (2012). Automated Student Model Improvement. 5th International Conference on EDM.

[8] Koedinger, K. R., Stamper, J. C., McLaughlin, E. A., & Nixon, T. (2013). Using data-driven discovery of better cognitive models to improve student learning. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED '13), 9–13 July 2013, Memphis, TN, USA (pp. 421–430). Springer.

[9] Liu, R., & Koedinger, K. R. (2015). Variations in learning rate: Student classification based on systematic residual error patterns across practice opportunities. 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 Education Data Mining (EDM2015), 26–29 June 2015, Madrid, Spain (pp. 420–423). International Educational Data Mining Society.

[10] Liu, R., & Koedinger, K. R. (under review). Closing the loop: Automated data-driven skill model discoveries lead to improved instruction and learning gains.

[11] Liu, R., Koedinger, K. R., & McLaughlin, E. A. (2014). Interpreting model discovery and testing generalization to a new dataset. In J. Stamper, Z. Pardos, M. Mavrikis, & B. M. McLaren (Eds.), Proceedings of the 7th International Conference on Educational Data Mining (EDM2014), 4–7 July, London, UK (pp. 107–113). International Educational Data Mining Society.

[12] Lee, J.I., & Brunskill, E. (2012). The Impact on Individualizing Student Models on Necessary Practice Opportunities. 5th International Conference on EDM.

[13] Pardos, Z.A., & Heffernan, N.T. (2010). Modeling individualization in a bayesian networks implementation of knowledge tracing. User Modeling, Adaptation, and Personalization, 255-266.

[14] Pardos, Z. A., Trivedi, S., Heffernan, N. T., & Sárközy, G. N. (2012). Clustered knowledge tracing. In S. A. Cerri, W. J. Clancey, G. Papadourakis, K.-K. Panourgia (Eds.), Proceedings of the 11th International Conference on Intelligent Tutoring Systems (ITS 2012), 14–18 June 2012, Chania, Greece (pp. 405–410). Springer.

[15] Pavlik, P.I., Cen, H., & Koedinger, K.R. (2009). Performance factors analysis–a new alternative to knowledge tracing. AIED, 531–538.

[16] Shmueli, G. (2010). To explain or to predict? *Statistical Science*, *25*(3), 289–310. doi:10.1214/10-STS330

[17] Stamper, J., & Koedinger, K. R. (2011). Human-machine student model discovery and improvement using data. *Proceedings of the 15th International Conference on*

Artificial Intelligence in Education (AIED '11), 28 June–2 July, Auckland, New Zealand (pp. 353–360). Springer.

[18] Yudelson, M.V., Koedinger, K.R., & Gordon, G.J. (2013). Individualized bayesian knowledge tracing models. AIED, 171-180.

[19] VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education, 16*, 227–265.

The Misidentified Identifiability Problem of Bayesian Knowledge Tracing

Shayan Doroudi
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15206
shayand@cs.cmu.edu

Emma Brunskill
Computer Science Department
Stanford University
Stanford, CA 94305
ebrun@cs.stanford.edu

ABSTRACT

In this paper, we investigate two purported problems with Bayesian Knowledge Tracing (BKT), a popular statistical model of student learning: *identifiability* and *semantic model degeneracy*. In 2007, Beck and Chang stated that BKT is susceptible to an *identifiability problem*—various models with different parameters can give rise to the same predictions about student performance. We show that the problem they pointed out was not an identifiability problem, and using an existing result from the identifiability of hidden Markov models, we show that under mild conditions on the parameters, BKT is actually identifiable. In the second part of the paper, we discuss a problem that has been conflated with identifiability, but which actually does arise when fitting BKT models, *semantic model degeneracy*—the model parameters that best fit the data are inconsistent with the conceptual assumptions underlying BKT. We give some intuition for why semantic model degeneracy may arise by showing that BKT models fit to data generated from alternative models of student learning can have semantically degenerate parameters. Finally, we discuss the potential implications of these insights.

Keywords

Bayesian Knowledge Tracing, identifiability, semantic model degeneracy

1. INTRODUCTION

Bayesian Knowledge Tracing (BKT) is a popular model of student learning that tries to predict the probability that a student knows a skill and the probability that a student will answer questions based on the skill correctly. The BKT model is a two state hidden Markov model (HMM) that posits students have either mastered a skill or not, and at every practice opportunity, a student who has not mastered the skill has some chance of attaining mastery. If a student has mastered a skill, they will answer a question correctly unless they "slip" with some (ideally small) probability, and if the student has not mastered the skill, they can only guess correctly with some (ideally small) probability. In 2007, Beck and Chang stated that BKT is not identifiable, meaning that different settings of the four BKT parameters can lead to identical predictions about a student's performance [7]. Whether or not BKT is identifiable is an important issue, because if BKT is not identifiable, it means that we would fundamentally need other criteria (beyond accurately modeling student performance data) to fit BKT models.

However, in this paper, we show that BKT is actually an identifiable model, under mild conditions on the parameters that should always be satisfied in practical settings. This result follows from BKT being a special case of a hidden Markov model and therefore it inherits identifiability results that prior work has proven for HMMs. This implies no additional criteria beyond predictive accuracy are needed to identify a single BKT model that best explains observed student performance, under the assumption that learning can accurately be modeled by a BKT. We then describe three potential issues with BKT models that may have been misconstrued as an identifiability problem in the literature. Note that our goal is by no means to criticize prior researchers, as such researchers helped identify some important limitations of Bayesian Knowledge Tracing, but these limitations do not stem from a lack of identifiablity.

In the second part of this paper, we focus on one of the issues that has been conflated with identifiability, but which actually does arise when fitting BKT models, *semantic model degeneracy*—the model parameters that best fit the data are inconsistent with the conceptual assumptions underlying BKT. We give a critical look at the types of semantic model degeneracy in the literature and then give some intuition for why this problem may arise by showing that BKT models fit to data generated from alternative models of student learning can have degenerate parameters. We further show that fitting models to sequences of different lengths generated from the same underlying model can result in different forms of semantic degeneracy. We show that these insights can have important implications on how these models should be used.

2. BAYESIAN KNOWLEDGE TRACING

The Bayesian Knowledge Tracing model is a two-state hidden Markov model that keeps track of the probability that a student has mastered a particular skill and the probability that the student will be able to answer a question on that skill correctly over time. At each practice opportunity $i \geq 1$ (i.e., when a student has to an answer a question corresponding to the skill), the student has a latent knowledge state $K_i \in \{0, 1\}$. If the knowledge state is 0, the student has not mastered the skill, and if it is 1, then the student has mastered it. The student's answer can either be correct or incorrect: $C_i \in \{0, 1\}$ (where 0 corresponds to incorrect and 1 corresponds to correct). After each practice opportunity, the student is assumed to master the skill with some probability. The BKT model is parametrized by the following four parameters:

- $P(L_0) = P(K_1 = 1)$: the initial probability of knowing the skill (before the student is given any practice opportunities)
- $P(T) = P(K_{i+1} = 1|K_i = 0)$: the probability of mastering a skill at each practice opportunity (if the student has not yet mastered the skill)
- $P(G) = P(C_i = 1|K_i = 0)$: the probability of guessing
- $P(S) = P(C_i = 0|K_i = 1)$: the probability of "slipping" (answering incorrectly despite having mastered the skill)

3. IDENTIFIABILITY

In their 2007 paper, Beck and Chang claimed that BKT is not identifiable, illustrating this with a particular example of three different BKT models [7]. For concreteness we include these models in Table 1. The authors consider the case of predicting the probability of correctness under these three models as the students receive practice opportunities, but in absence of any observation about the student's performance. They use plots as in Figure 1 to claim that the three models make very different predictions about student knowledge (Figure 1 (a)), but make identical predictions about student performance (Figure 1 (b)). They claim,

> All three of the sets of parameters instantiate a knowledge tracing model that fit the observed data equally well; statistically there is no justification for preferring one model over another. This problem of multiple (differing) sets of parameter values that make identical predictions is known as identifiability.

However, this is not correct since no data was used to fit these curves; the curves are predicting the probability that a student will know the skill or will answer the skill correctly at each practice opportunity i, *when we have no prior performance or data on the student.* In order to take past data from a student into account, we actually want to predict $P(K_i = 1|C_1, \ldots C_{i-1})$ and $P(C_i = 1|C_1, \ldots C_{i-1})$ and this is indeed what we do in practice when doing knowledge tracing; we make predictions based on our past observations. Figure 2 shows the curves predicting these conditional probabilities for a particular sequence of correct/incorrect answers for a student (namely we use $(1, 0, 0, 0, 0, 0, 0, 1, 1)$). We find that even when we condition on a single observation (i.e., for $P(C_2 = 1|C_1)$), the three models make vastly different predictions, and as we collect more data, the models continue to make very different predictions. In fact, except for $P(C_1 = 1)$, the models never agree on the probability that a student would answer the step correctly.

Formally, a model is said to be *identifiable* if there are no two distinct sets of model parameters θ and θ' that can give rise to the same joint probability distribution over observations under that model. As far as inference is concerned, identifiability means that the likelihood function of the model has only one global maximum, so inference of the true model parameters is possible. In the case of BKT, the model would be identifiable if for any two distinct sets of BKT parameters, θ and θ',

$$P_\theta(C_1, C_2, \ldots, C_n) \neq P_{\theta'}(C_1, C_2, \ldots, C_n)$$

for some $n \geq 1$. What Beck and Chang show is that there can be infinitely many models that share the same set of marginal distributions $P(C_1), P(C_2), \ldots, P(C_n)$. This does not mean the model is unidentifiable. As we saw from Figure 2, the conditional distribution $P(C_n|C_1, \ldots, C_{n-1})$ is quite different for each model, and so the joint distribution $P(C_1, \ldots, C_n)$ is also very different for the three models.

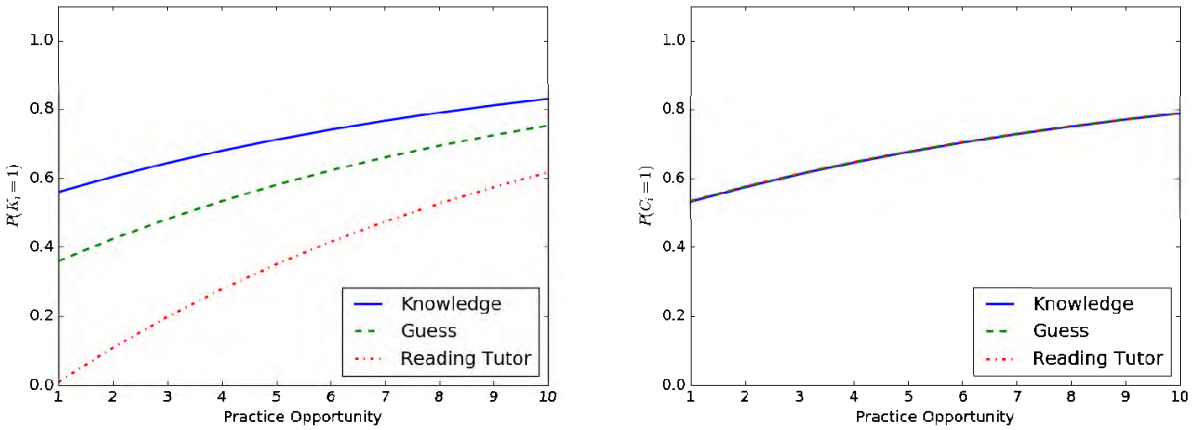
It turns out there has been a substantial amount of work, going back 50 years and continuing to this day, on finding the conditions for which hidden Markov models are identifiable [15, 1, 2, 17, 10]. Although much of the literature focuses on particular types of HMMs (e.g., stationary, irreducible) that do not include the standard BKT model, Anandkumar et al. have recently shown that, subject to some non-degeneracy conditions, a large class of HMMs, which includes BKTs, is identifiable with just the joint probability distributions for up to three sequential observations [4]. That is, knowing $P(C_1)$, $P(C_1, C_2)$, and $P(C_1, C_2, C_3)$ is enough to infer the unique model parameters, subject to non-degeneracy conditions. In our context, the conditions are that $P(L_0) \notin \{0, 1\}$, $P(T) \neq 1$, and $P(G) \neq 1 - P(S)$. This suggests that as long as we have more than two observations per student, BKT models with reasonable parameters are identifiable and there is a single global maximum to the likelihood function. Feng recently independently showed the same result directly for BKT models, except without requiring the condition that $P(L_0) \neq 0$ [9]. One advantage of relying on general identifiability results for HMMs is that we can use the same results to show the conditions under which related student models that can also be modeled as HMMs are identifiable[1].

This misuse of the term "identifiability" has lead to multiple subsequent papers in the educational data mining community throughout the past decade which have similarly given a mistaken description of the underlying phenomena [5, 16, 13, 12]. Two papers, however, have correctly identified that the

[1]For example, for the BKT model with forgetting, where $P(F) = P(K_{i+1} = 0|K_i = 1) \neq 0$, we can show that the model is identifiable with the same conditions, except that we require $P(T) \neq 1 - P(F)$ instead of $P(T) \neq 1$. We can also easily show the conditions under which multi-state extensions of BKT such as the model introduced in Section 4.2 are identifiable. These conditions can be derived from Condition 3.1 and Proposition 4.2 of [4]. See also the note under Proposition 3.4 of [3].

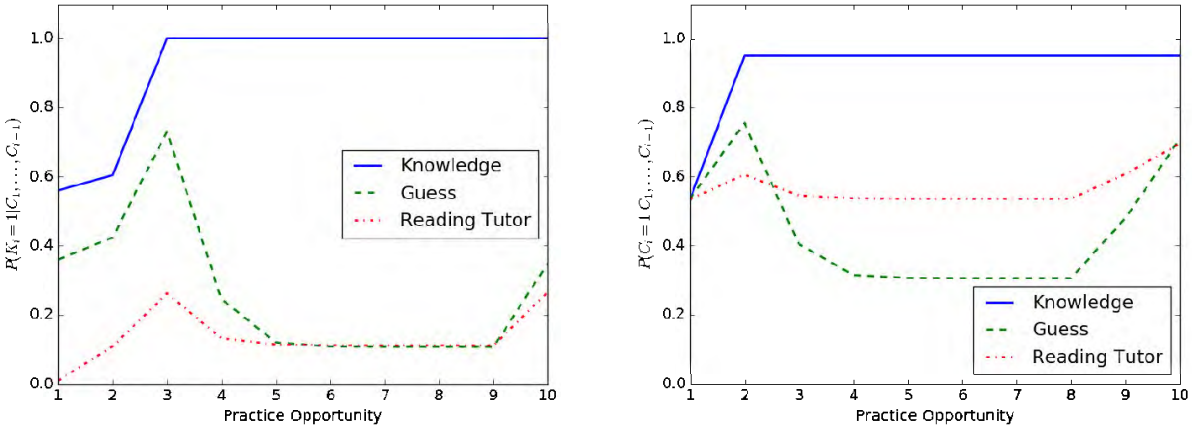
	Model		
Parameter	Knowledge	Guess	Reading Tutor
$P(L_0)$	0.56	0.36	0.01
$P(T)$	0.1	0.1	0.1
$P(G)$	0	0.3	0.53
$P(S)$	0.05	0.05	0.05

Table 1: The three BKT models used by Beck and Chang [7] to claim BKT is unidentifiable. The models are chosen to have very different semantic interpretations. The Knowledge model requires the student to master the skill to get it correct, the guess model relies on the student guessing, and the Reading Tutor model has an even higher probability of guessing, but it was based on models actually used by the Reading Tutor [14].



(a) Learning Curve (b) Performance Curve

Figure 1: Hypothetical learning and performance curves for three models from [7], in absence of any data.



(a) Learning Curve (b) Performance Curve

Figure 2: Learning and performance curves for three models from [7] conditioned on all past observations for a student whose observed trajectory is as follows: $(C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9) = (1, 0, 0, 0, 0, 0, 0, 1, 1)$

"identifiability problem" is limited to the case where there is no data [18, 11]. Even though this is not a statistically precise claim, it does show that some researchers have the correct understanding behind the phenomenon. Van de Sande distinguishes between the two cases where predictions are made in the absence of data and where they are made in the presence of data, and claims that the source of the identifiability problem in the former case is that the predictions can be completely determined by three parameters, so there is a degree of freedom [18]. When we are making predictions, however he claims there is no identifiability problem, because $P(K_i|C_i)$ depends on four parameters [18]. While he has correctly identified the absence of an identifiability problem in the presence of data, we believe that there is still confusion about the identifiability problem in the community (e.g., some of the papers that show a misunderstanding of the issue are more recent than [18]). We hope to make the absence of an identifiability problem more clear and elucidate the phenomena and misconceptions surrounding it. Gweon et al. also distinguish between two cases which they refer to as the BKT model without measurement and the BKT model with measurement, and show, as van de Sande did, that the former depends on three parameters (hence the "identifiability problem") whereas the latter depends on all four [11]. However, they claim this does not necessarily mean that the BKT model with measurement does not suffer from an identifiability problem, and actually claim that it still does suffer from an identifiability problem, because empirically, they found that for some data, fitting BKT models many times resulted in a wide spread of possible parameters [11]. However, this cannot be due to the presence of an multiple global maxima, which we have shown cannot exist, and hence must be due to multiple local optima.

The work closest to ours is Feng's recently published dissertation [9]. The author gives a similar explanation to ours for why Beck and Chang's claim was incorrect and also proves that the BKT model is identifiable directly [9]. However, we believe the exposition there is perhaps less accessible to the educational data mining community and will likely not obtain the visibility needed to clear the misunderstandings surrounding the identifiability of BKT. In this paper, we not only focus on identifying the misidentified identifiability problem, but also understanding the confusion surrounding it as well as pointing out actual issues with fitting BKT models that have been conflated with identifiability. This is the focus of the rest of the paper.

There are three potential sources of confusion that we believe could be and have been misconstrued as an identifiability problem:

1. *A priori predictions.* That multiple models, which make very different claims about student's knowledge state over time, could predict the same probability that students answer questions correctly over time *in the absence of data.* This is the problem that Beck and Chang conflated with identifiability, and many researchers thereafter also treated as identifiability. As we showed above, van de Sande, Gweon et al. and Feng correctly identified what is happening here [18, 11, 9].

2. *Multiple local optima.* It is well known that the expectation-maximization algorithm that is commonly used to fit BKT models is suceptible to converging to local optima of the likelihood function rather than converging to the global optimum. While Beck and Chang clearly did not conflate this with the identifiability issue, we saw that other researchers such as Gweon et al. have possibly conflated the two. In order to avoid local optima, one can use a grid search over the entire parameter space or run multiple iterations of the expectation-maximization algorithm with different initializations of the parameters.

3. *Semantic model degeneracy.* Baker et al. identified another problem with BKT models, which they termed model degeneracy [5]. A model is said to be semantically degenerate[2] when it is inconsistent with the conceptual assumptions underlying the BKT model. The problem is when the model that best fits our data is semantically degenerate. Even though Baker et al. clearly contrasted this to the (supposed) identifiability problem, we claim that this is the problem that Beck and Chang attempted to fix in their paper. We will now focus on better understanding this problem.

4. SEMANTIC MODEL DEGENERACY

In their paper, Beck and Chang propose a way to get around the identifiability problem. They propose using Dirichlet priors to encode prior beliefs about the BKT parameters, which will in turn bias the model search towards more reasonable parameters [7]. They motivate their method as follows:

> We have more knowledge about student learning than the data we use to train our models. As cognitive scientists, we have some notion of what learning "looks like." For example, if a model suggest that a skill gets worse with practice, it is likely the problem is with the modeling approach, not that the students are actually getting less knowledgeable. The question is how can we encode these prior beliefs about learning?

The problem they appear to be describing is that some models have parameters that do not match our intuitions of student learning, i.e., they are exactly describing the issue of semantic model degeneracy (and not that of unidentifiability). Baker et al. later provide another solution to tackling semantic model degeneracy by using contextual features to estimate the guess and slip parameters [5]; however, interestingly they did not view Beck and Chang's original solution as a way of tackling semantic model degeneracy, treating it as a way to tackle identifiability as the authors originally claimed.

Having shown that identifiability is not an issue with BKT, and given that there are easy ways to tackle the existence of local optima, we believe semantic model degeneracy is perhaps the most important problem with respect to fitting BKT models that needs to be better understood and tackled. Essentially, the problem arises because the BKT is simply a

[2]We refer to this property as semantic model degeneracy to distinguish it from mathematically degenerate parameters that would result in BKT models being unidentifiable, as described above.

particular form of a two-state hidden Markov model and it will try to fit the best two state hidden Markov model it can to the data; our model fitting procedures do not understand that the $K_i = 1$ state is supposed to correspond to mastering a skill, and so it might fit a model that does not match our intuitions of mastery. We will try to understand this in more detail below, but first we aim to characterize the types of semantic model degeneracy that have been pointed out in the literature.

4.1 Types of Semantic Model Degeneracy

Baker et al. distinguish between two forms of semantic model degeneracy: *theoretical degeneracy* and *empirical degeneracy* [5]. They define a model to be theoretically degenerate when either the guess or the slip parameter is greater than 0.5. They define a model to be empirically degenerate if one of two things occur: (1) for some large enough n the model's estimate of the student having mastered the skill decreases after the student gets the first n skills correct or (2) for some large enough m, the student does not achieve mastery (our estimate of the student having mastered the skill does not go beyond 0.95) even after m consecutive correct responses [5]. The authors arbitrarily chose the values $n = 3$ and $m = 10$. Note that the first form of empirical degeneracy is only possible if $1 - P(S) < P(G)$ (i.e., the student is more likely to answer a question correctly if they have not mastered a skill than if they have mastered a skill), as was shown by van de Sande [18]. This is true, even for $n = 1$. Thus, this first notion of empirical degeneracy is equivalent to $P(G) + P(S) > 1$, which implies either $P(S) > 0.5$ or $P(G) > 0.5$, meaning that it always implies theoretical degeneracy! Huang et al. have noted that while $P(G) + P(S) > 1$ definitely implies semantically degenerate parameters as it contradicts mastery, the condition that $P(G) < 0.5$ and $P(S) < 0.5$ may not always be necessary for the parameters to be semantically meaningful, since, for example, there may be some domains where the student can guess the correct answer easily [12]. We agree that suggesting $P(G) < 0.5$ is degenerate does seem somewhat arbitrary depending on the domain; however, we do think $P(S) > 0.5$ should be characterized as a form semantic degeneracy, because, as Baker et al. claimed, it does not make sense for a student who has mastered a skill to answer questions of that skill incorrectly most of the time—that goes against our intuitions of what mastery means. In any case, it does not seem like the distinction between theoretical and empirical degeneracy is a clear one, so we suggest categorizing the forms of semantic model degeneracy by what they suggest about student learning:

- *Forgetting*: This is a result of $P(G) + P(S) > 1$, which suggests that not only are students not learning, but that students have some probability of losing their knowledge over time. Another way to view this degeneracy is that the state we would conceptually call the mastery state is now the state where performance is worse.
- *Low Performance Mastery*: This is a result of $P(S) > 0.5$. Alternatively, we can set our threshold for low performance mastery to be lower (e.g., P(S) > 0.4).
- *High Performance Guessing*: This is a result of $P(G) > t$, where t is some threshold. As mentioned earlier, this seems like a weak form of degeneracy, as students can often guess an answer easily even if they have not mastered a skill, but we can set t to a large enough value, to make this a form of model degeneracy.
- *High Performance $\not\Rightarrow$ Learning*: This is the second form of empirical degeneracy given by Baker et al. [5]: for some choice of m, the probability that the student has achieved mastery is less than some threshold p (typically taken to be 0.95) after m consecutive correct responses

4.2 Sources of Semantic Model Degeneracy

We will now consider a possible explanation for why BKT models are so prone to semantic model degeneracy (which we believe to be part of the reason that researchers look towards identifiability and local optima to explain the strange parameters that result from fitting BKT models). First of all, note that forgetting degeneracy will occur whenever students actually do forget or when they learn misconceptions; it is not unreasonable to believe that students will sometimes learn and reinforce a misconception, causing their knowledge of some skill to decrease over time. Thus, while this form of degeneracy technically violates our notion of mastery, it is to be expected if we switch the semantic interpretation of the two states and suppose that students forget instead of learn. We now consider sources of the other forms of semantic model degeneracy. We claim that such forms of semantic model degeneracy can result from not accurately being able to capture the complexity of student learning with a two state HMM. When this is the case, fitting the data with a two state HMM will result in trying to find the best fit of the data for a two state HMM, and not to come up with a model that tries to accurately model the data while also matching our intuitions about what it means for a student to have mastered a skill.

To support our claim, suppose student learning is actually governed by a 10-state HMM with ten consecutive states representing different *levels* of mastery. From each state, the student has some probability of transitioning to the next state (slightly increasing in mastery), and from each state, the student has a probability of answering questions correctly, and this probability strictly increases as the student's level of mastery increases. Specifically consider the model presented in Table 2. Now suppose we try to use a standard BKT model to fit data generated from this alternative model of student learning. The first two columns of Table 3 show the parameters of BKT models fit to 500 sequences of 20 practice opportunities or 100 sequences of 200 practice opportunities, both generated from the the model in Table 2. Notice that the model fits (nearly) degenerate parameters in both cases. When we only have 20 observations per student, the model estimates a very high slip parameter; this is because it has to somehow aggregate the different latent states which correspond to different levels of mastery, and since not many students would have reached the highest levels of mastery in 20 steps, it is going to predict that students who have "mastered" the skill are often getting it wrong. However, what's more interesting is that for the same model, if we simply increase the number of observations per student from 20 to 200, we find that the slip parameter is reasonably small, but now the guess probability is 0.49! This is because, by

	State i									
Parameter	0	1	2	3	4	5	6	7	8	9
$P(K_0 = k)$	0.1	0.1	0.1	0.2	0.2	0.3	0	0	0	0
$P(C_i = 1 \mid K_i = k)$	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$P(K_i = k+1 \mid K_i = k)$	0.4	0.3	0.2	0.1	0.05	0.05	0.05	0.05	0.05	-

Table 2: Alternative model of student learning where there are ten levels of mastery.

	10-State HMM		AFM	
Parameter	20	200	20	200
$P(L_0)$	0.30	0.001	0.09	0.001
$P(T)$	0.05	0.02	0.05	0.05
$P(G)$	0.27	0.49	0.14	0.28
$P(S)$	0.44	0.13	0.46	0.03

Table 3: BKT models fit to data generated from the model described in Figure 2 and an additive factors model described in the text. The first column for each model is fit to 500 sequences of 20 practice opportunities, while the second column is fit to 100 sequences of 200 practice opportunities. The models were fit using brute-force grid search over the entire parameter space in 0.01 increments for the parameters using the BKT Brute Force model fitting code [6].

this point most students have actually reached the highest level of mastery, so to compensate for the varying levels of mastery that occurred earlier in student trajectories, the model will have to estimate a high guess parameter. So we find that not only can alternative models of student learning lead to fitting (near) degenerate parameters, but varying the number of observations can lead to different forms of degeneracy! This is a counterintuitive phenomenon that we believe is not the result of not having enough data (students) to fit the models well, but rather the result of the mismatch between the true form of student learning and the model we are using the fit student learning.

We find similar results if we fit a BKT model to data generated from another alternative model of student learning that is commonly used in the educational data mining community, the additive factors model (AFM) [8]. In particular, we used the model

$$P(C_i = 1) = \frac{1}{1 + exp(-\theta + 2 - 0.1i)}$$

where $\theta \sim \mathcal{N}(0, 1)$ is the student's ability[3]. The second two columns of Table 3 show the parameters of BKT models fit to data generated from this model. We again find that when using only data with 20 practice opportunities, we fit a high slip parameter, but when we using data with 200 practice opportunities, we fit a higher guess parameter and a very small slip parameter.

Additionally, notice that for the parameters fit to the 10-state HMM, the probability of transitioning to mastery is very small when we fit to sequences with 200 practice opportunities. Since the transition probability is small and the guess probability is large, we also have high performance $\not\Rightarrow$ learning degeneracy for this model for $m = 10$. That is,

$$P(K_{11} = 1 \mid C_1 = 1, C_2 = 1, \ldots, C_{10} = 1) \approx .89 < 0.95$$

This is yet another form of degeneracy that does not exist in the model fit to sequences of 20 practice opportunities. Furthermore, notice that when we have 200 observations, the probability of transitioning to mastery is smaller than $P(K_i = k+1 \mid K_i = k)$ for all states i in the model that generated the data (Table 2). Again, this is because the best fitting BKT model will aggregate low performing states and high performing states, so a single transition in the BKT model between these two aggregate states will have to loosely correspond to the student transitioning several times in the actual 10-state HMM. Thus, while the learned BKT model makes it appear as though learning happens very slowly, according to the true student model, learning actually occurs much more often but in more progressive increments. This suggests that if we use some automated technique to detect if a skill is useful for student learning, we may conclude it is not, if we do not allow for the possibility that students are learning progressively.

These observations have important implications for how learned models can be used in practice. Using such a BKT model to predict student mastery can lead to problematic inferences. For example, for the first model in Table 3, the BKT model assumes that when a student has reached mastery, they have a 56% chance of answering a question correctly, whereas a student who has actually mastered the skill will have a 90% chance of answering correctly (see Table 2). Thus, an intelligent tutoring system that uses such a BKT model to determine when a student has had sufficient practice on a problem, will likely give far fewer problems to the student than they actually need in order to reach mastery!

There are several potential ways that future work can proceed in light of these findings. One is that we should be giving our model fitting procedures more domain knowledge about the kind of model we want it to fit. This is essentially what Beck and Chang did by using Dirichlet priors [7] and what Baker et al. did by estimating the guess and slip parameters using context [5]. But perhaps there are other ways of doing this where we do not need to give context-dependent domain knowledge to the model per se, but rather come up with a model that realizes the difference between a student having mastered a skill or not (which the BKT model cannot do). However, this may not be ideal in some cases where student learning cannot accurately be modeled by BKT with semantically plausible parameters. For example when we have forgetting degeneracy, we should probably not force

[3]This model suggests that students who are two standard deviations above the mean initially will answer correctly half the time, and after 20 practice opportunities the average student will answer correctly half the time.

the parameters to suggest learning is occurring when it may not be. Another way to proceed is to consider alternative student models, which is an active area of educational data mining research. Perhaps, obtaining semantically degenerate parameters from a fit should signal that our students may be learning in more complicated ways than the simple BKT model can predict, and so we should try to find alternative models that fit our data better without yielding semantically degenerate parameters. Finally, even if our model is semantically degenerate, it does not necessarily make the BKT model useless. The result of fitting a BKT model is that we get the best fit of the data given that we are modeling the data with a two-state HMM (if we disregard local optima). Presumably, such a model can give us some insights about student learning even if it is not modeling student mastery. So perhaps we can use such semantically degenerate models to understand student learning rather than to predict student mastery.

5. CONCLUSION

We have explored the issues of identifiability and semantic model degeneracy in Bayesian Knowledge Tracing. We have shown that what researchers posited was an identifiability problem is actually not an identifiability problem, and by using a result from the literature on learning hidden Markov models, we showed that an identifiability problem does not exist for BKT models (with the exception of some mathematically degenerate cases that should not come up in practice). We then examined the various issues with fitting BKT models that have been conflated with identifiability. We offered what we believe to be new insights on one potential source of semantic model degeneracy. We believe analyzing the sources of semantic model degeneracy in more detail can be a fruitful direction for future research. For example, it could be useful to know what BKT parameters result from fitting various other popular models of student learning. It would also be informative to see if we can find automated ways of detecting which assumptions of BKT are not met in our data (e.g., the number of levels of mastery, the independence of different skills). Such analyses could help in devising better student models, and ultimately may lead to a better understanding of student learning.

6. ACKNOWLEDGEMENTS

The research reported here was supported, in whole or in part, 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.

7. REFERENCES

[1] E. S. Allman, C. Matias, and J. A. Rhodes. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, pages 3099–3132, 2009.

[2] Y. An, Y. Hu, J. Hopkins, and M. Shum. Identifiability and inference of hidden markov models. Technical report, Technical report, 2013.

[3] A. Anandkumar, R. Ge, D. J. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

[4] A. Anandkumar, D. Hsu, and S. M. Kakade. A method of moments for mixture models and hidden markov models. In *Conference on Learning Theory*, pages 33–1, 2012.

[5] R. S. Baker, A. T. Corbett, and V. Aleven. Improving contextual models of guessing and slipping with a truncated training set. *Human-Computer Interaction Institute*, page 17, 2008.

[6] R. S. Baker, A. T. Corbett, S. M. Gowda, A. Z. Wagner, B. A. MacLaren, L. R. Kauffman, A. P. Mitchell, and S. Giguere. Contextual slip and prediction of student performance after use of an intelligent tutor. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 52–63. Springer, 2010.

[7] J. E. Beck and K.-m. Chang. Identifiability: A fundamental problem of student modeling. In *International Conference on User Modeling*, pages 137–146. Springer, 2007.

[8] H. Cen. *Generalized learning factors analysis: improving cognitive models with machine learning*. Carnegie Mellon University, 2009.

[9] J. Feng. *Essays on learning through practice*. PhD thesis, The University of Chicago, 2017.

[10] É. Gassiat, A. Cleynen, and S. Robin. Inference in finite state space non parametric hidden markov models and applications. *Statistics and Computing*, 26(1-2):61–71, 2016.

[11] G.-H. Gweon, H.-S. Lee, C. Dorsey, R. Tinker, W. Finzer, and D. Damelin. Tracking student progress in a game-like learning environment with a monte carlo bayesian knowledge tracing model. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 166–170. ACM, 2015.

[12] Y. Huang, J. Gonzalez-Brenes, R. Kumar, and P. Brusilovsky. A framework for multifaceted evaluation of student models. In *Proceedings of the 8th International Conference on Educational Data Mining*. University of Pittsburgh, 2015.

[13] J. I. Lee and E. Brunskill. The impact on individualizing student models on necessary practice opportunities. *International Educational Data Mining Society*, 2012.

[14] J. Mostow and G. Aist. Smart machines in education. chapter Evaluating Tutors That Listen: An Overview of Project LISTEN, pages 169–234. MIT Press, Cambridge, MA, USA, 2001.

[15] T. Petrie. Probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics*, 40(1):97–115, 1969.

[16] S. Ritter, T. K. Harris, T. Nixon, D. Dickison, R. C. Murray, and B. Towle. Reducing the knowledge tracing space. *International Working Group on Educational Data Mining*, 2009.

[17] P. Tune, H. X. Nguyen, and M. Roughan. Hidden markov model identifiability via tensors. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2299–2303. IEEE, 2013.

[18] B. van De Sande. Properties of the bayesian knowledge tracing model. *JEDM-Journal of Educational Data Mining*, 5(2):1–10, 2013.

Short Papers

An Effective Framework for Automatically Generating and Ranking Topics in MOOC Videos

Jile Zhu
School of EECS, Peking University
zhujile0918@pku.edu.cn

Xiang Li
School of EECS, Peking University
lixiang.eecs@pku.edu.cn

Zhuo Wang
School of EECS, Peking University
wangzhuo@pku.edu.cn

Ming Zhang*
School of EECS, Peking University
mzhang@net.pku.edu.cn

ABSTRACT

Although millions of students have access to varieties of learning resources on Massive Open Online Courses (MOOCs), they are usually limited to receiving rapid feedback. Providing guidance for students, which enhances the interaction with students, is a promising way to improve learning experience. In this paper, we consider to show students the emphasis of lectures before their learning. We propose a novel framework that automatically generates and ranks the topics within the upcoming chapter. We apply the Latent Dirichlet Allocation (LDA) model on the subtitles of lectures to generate topics. We then rank the importance of these topics through a particular PageRank method, which also leverages structural information of lectures. Experimental results demonstrate the effectiveness of our approach, with a 18.9% improvement in Mean Average Precision (MAP). At last, we simulate two cases to discuss how can our framework guide students according to their learning status.

Keywords

Massive Open Online Courses (MOOCs); Guidance for Students; Topic Model; PageRank.

1. INTRODUCTION

With recent developments of Massive Open Online Courses (MOOCs), millions of students have access to abundant high-quality learning resources at their convenience and with no cost. Despite all the advantages, students on MOOCs are usually limited to receiving rapid feedback, and the lack of interaction with instructors and peers would reduce their learning experience [6, 16]. Previous explorations of course design and intervention have shown the guidance would improve student learning experience and performance [3, 11]. However, few works researched on providing guidance at the early stage of learning process. According to the strategy of learning design, Conole suggested teachers design a vision for the course in terms of knowledge [6].

Traditionally, teachers emphasize important concepts in classes. But in MOOCs, not all the teachers underline the key points when giving the lectures. Moreover, even if teachers have repeated the key points in the videos, MOOC students are prone to miss such information. A study of edX student habits found that even certificate-earning students only viewed the first 4.4 minutes of 12 to 15 minute videos [7].

With guidance that highlights the most important topics, students can have an vision of key points before watching lectures, or briefly review these knowledge if they are going to take assignments. Specifically, important topics are more likely to be involved in assignments in the perspective of students [2, 10], so that such guidance will be valuable for those who have little leisure time but want to complete the course. Thus, such automatic guidance is helpful for students to know the emphasis of upcoming lectures.

Previous studies in knowledge tracing represented key points as knowledge components, which are inferred from student performance on assignment items [9]. Besides, some works in MOOCs simply defined knowledge components as one single problem or chapter [15, 17]. However, most MOOCs don't have enough problem items for accurate definition. Different from these works, our framework generates topics from video subtitles, which is more general for MOOCs. Moreover, our work is the first to rank these topics, by leveraging both textual and structural information of videos.

Our work focuses on automatically providing students with guidance at the early stage of learning process. We propose a novel framework that takes the video subtitles as inputs and suggests students the most important topics within the upcoming chapter. To address such a task, we decompose it into the following three steps: (1) Generate topics from subtitles by LDA model; (2) Decide the importance of phrases based on a particular PageRank method; (3) Smooth the PageRank value and measure the importance of topics. The experiments show the effectiveness of our algorithm, which improves by 18.9% in Mean Average Precision (MAP). We also use two cases to illustrate how our framework help different students according to their learning status. The main contributions of our work are listed as:

- We design a novel framework for MOOCs that automatically provides students with a vision of important topics at the early stage of their learning.
- We propose a particular PageRank method to rank the importance of topics within the upcoming chapter.
- The experiments and simulated cases show the effectiveness of our algorithm and how it works.

*Ming Zhang is the corresponding author.

2. RELATED WORK

2.1 Design and Intervention

Students participate in MOOCs through the interactions with lectures, assignments, and forums. Interventions were designed to enhance their engagement and learning experience. Previous work explored the effect of video production on student engagement [8], suggested detecting confusion in forums [18], and showed that immediate feedback of assignments can improve learning performance [11]. However, most of recent works designed the interventions for students during or after their learning process.

Basu et al.[3] presented an intervention that assists students in understanding detailed specification of assignments before their attempts. However, this work addressed the problem of assignments, but not learning by watching lectures. Our work focuses on providing guidance for students with a vision of the key points they are going to learn.

2.2 Topic Model

To automatically summarize the content of lectures, NLP techniques are commonly used to extract the keyphrases in the text. Topic model is designed for discovering the latent topics from a collection of documents. Among different algorithms, Latent Dirichlet allocation (LDA) is the most common topic model currently in use [4].

For MOOCs, the works concentrating on knowledge tracing defined the knowledge component as a chapter or a problem item[15, 17], but such representation deviates from common sense. Inspired by the work from Matsuda et al.[12], which applied LDA model on assignment items and viewed the auto-generated clusters as knowledge component candidates, we transfer this method to the videos in MOOCs. In our work, we generate latent topics from video subtitles, and define each topic as a probability distributions over phrases.

2.3 Ranking Model

Students are unlikely to post questions before their learning, especially in MOOCs. Therefore, in order to provide guidance at early stage, we should rank the topics through the content analysis of the lectures. PageRank is a graph-based ranking algorithm and it is a common way to measure the relative importance of items [14].

Some variants, like TextRank, created an undirected phrase graph from natural language texts for text processing, such as keyphrase extraction, extractive summarization [5, 13]. Different from these works, we view the MOOC video subtitles as the documents and leverage the structural relation between lectures. More specifically, we design a novel method to construct the phrase graph, which assigns phrase relations in different documents with different weights.

3. DATA PREPARATION

Recent MOOC providers also allow registered users to download the lecture videos and subtitle files. Therefore, it is convenient for researchers to analyze the video content as documents, using natural language processing (NLP) techniques. The dataset for this paper consists of a Coursera course "Data Structure and Algorithm". The filmed lectures are hierarchically organized.

To analyze the content of the lectures, we first extract noun-phrases from each subtitle for preprocessing, based on Python library *TextBlob*. Previous studies demonstrated that nouns and noun-phrases tend to produce keywords that typically express what the content is about [1]. Thus, the lectures can be represented as lists of consecutive phrases. There are 3,964 different phrases in total, and each lecture has an average length of 129.4 (including repeated phrases). Besides, the course sets up a quiz for every single chapter and two exams. The questions in these assignments are randomly sampled from a problem set, which contains 254 different items.

4. METHODS

The main objective of our research is to automatically provide students with guidance before their learning, which tells them the most important topics of the upcoming chapter. Based on such guidance, students can have a vision of the course, or check whether they have achieved these topics before they take an assignment. In brief, we propose a novel framework for MOOCs that takes a set of subtitles as inputs and returns a ranked list of topics ordered by their importance. Figure 1 shows the overall architecture of our framework, which can be decomposed into three steps.

In the first step, we use LDA model to generate topics from the subtitles of lectures. In the second step, we define a particular PageRank method for ranking the importance of phrases. Finally, we apply three transfer functions to reassign the importance value of phrases and measure the importance of topics.

4.1 Generating Topics from Subtitles

Then, we aim to generate topics for each chapter separately. Inspired by previous work, which applied LDA model on assessment items [12], we transfer this method to the subtitles of videos in MOOCs. LDA model is a generative probabilistic model that allows a set of observations to be explained by unobserved groups [4]. It is known to discover latent topics of a set of documents. In our cases, we denote lectures as documents and phrases as words. Specifically, the model takes the phrase lists from a chapter as inputs, and returns a set of latent topics, where each topic is characterized by a distribution over phrases.

In practice, we implement the model based on a Python library "lda". The number of iteration is set at 200 and the number of topics is dynamic with the number of lectures in the chapter, considering that different chapters have different number of topics. In addition, if the topics have been predefined by experts (given n keywords for each topic), we can also take such information as an alternative, instead of generating topics by LDA model. Specifically, to construct probability distributions over phrases as topics, it just needs to set the probabilities of corresponding phrases as $1/n$ and set the others as 0.

The output of this step for each chapter is a set of latent topics, in the form of probability distribution over phrases. To have an intuitive sense, we display each topic as a tuple, including three phrases with the highest probability in the distribution. Table 1 shows the topics generated from "Graph", which is one of the chapters in this course.

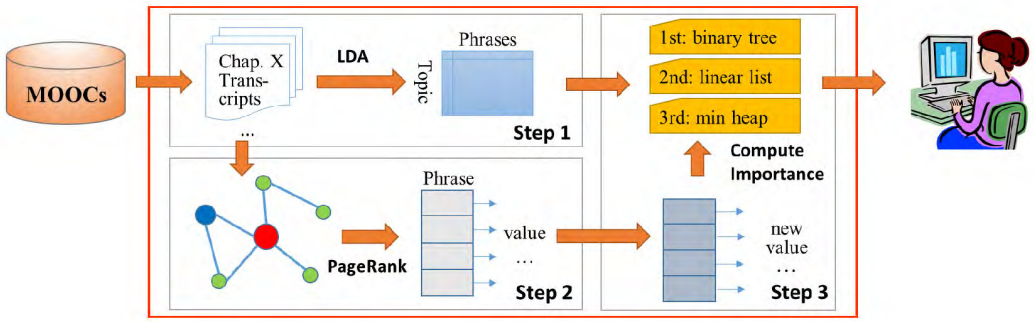


Figure 1: Overview of the framework that takes subtitles of MOOCs as inputs, and generates a ranked list of topics to students.

Chapter 8: Graph
(Kruskal algorithm, algorithms, data structure)
(adjacency list, adjacent matrix, list contains)
(MST, Prim algorithm, minimum weight edge)
(DAG, start node, data structure)
(DFS, topological sort, post process)
(Dist, shortest path, source node)
(old value, time complexity, Dijkstra)

Table 1: The topics generated by LDA model in Chapter "Graph".

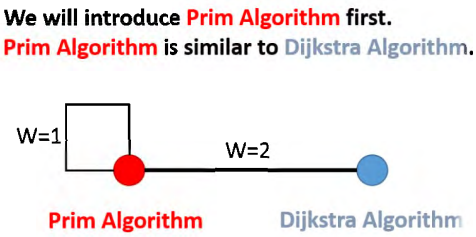


Figure 2: A sample graph built for a slice of subtitles, which is printed above the graph.

4.2 Ranking the Importance of Phrases

Our basic intuition is that important phrases are more likely to be mentioned in class. Moreover, when teachers talk about a new topic, they often briefly retrospect corresponding topics as comparisons, which enables us to connect a relation between phrases in different chapters. Based on these latent relation, we design a particular PageRank method, which leverages both textual and structural information of lectures, to rank the importance of phrases within chapters.

Our ranking algorithm can be decomposed into three processes. The first is to construct a phrase graph for each chapter. Then, for each chapter, we combine all the graphs generated by previous chapters that have been released before. At the end, we define a random walk on the graph to compute the importance magnitude of phrases. The output of this step is a ranked list of phrases, along with the value of their importance.

4.2.1 *Construction*

Intuitively, we consider that two important phrases occurring on close position suggest they have a relation between each other. PageRank is an algorithm for measuring the importance of website pages based on the webgraph [14]. In our cases, we denote the phrases as nodes and connect two phrases if they are close in the lecture.

Formally, we define an undirected graph $G_k = (V_k, E_k)$ in the k^{th} chapter, where $V_k = \{v_1, v_2, ..., v_{n_k}\}$ denotes the set of phrases. $L_k = \{l_1, l_2, ..., l_{m_k}\}$ denotes the lectures in the k^{th} chapter. We follow the TextRank [13] to construct the basic phrase graph for each chapter, which defines an edge as if the distance between the offset positions of two phrases is less than a preset parameter c (we set it as 8 during the experiments). We define the weight of edge as the times of co-occurrence between two phrases. Self-loop is allowed in our algorithm. The formula for the edge weight between phrases v_i and v_j is

$$w_k(v_i, v_j) = \sum_{s=1}^{m_k} \sum_{v_i \in l_s, v_j \in l_s} I\left\{dist(v_i, v_j) < c\right\},$$

where I is an indicator function and $dist(v_i, v_j)$ denotes the offset difference between v_i and v_j. The formula implies that two phrases appearing in the lectures more frequently and simultaneously result in a higher value of edge weight. For instance, Figure 2 shows a sample graph built for a slice of subtitle.

4.2.2 *Combinaton*

For teachers usually avoid repeating topics which have been discussed before, the relation of phrases will be insufficient if we only consider current chapter. For example, considering a paragraph of Chapter "Binary Tree", "We use a queue to implement BFS, ..., binary linked list is a way to store binary tree.", the phrases "BFS" and "binary tree" will not be connected, unless we combine Chapter "Stack and Queue" to connect "queue" and "linked list". Thus, when phrases propagate information over the graph, some important phrases do not associate with each other directly, but build an path through some "hubs". Based on these considerations, in order to supplement more relationships in current phrase graph, we combine it with those of previous chapters.

Therefore, we propose a weighted method for the combination of graphs. Specifically, when we rank the phrases in a chapter, we combine the current phrase graph with those constructed by all other chapters that have been released. We sum the weights of two phrases in different graphs by utilizing a damping factor α, which gives a lower weight to an earlier chapter. Formally, edge weights in the k^{th} chapter are formulated as

$$W_k(v_i, v_j) = \sum_{t=1}^{k} \alpha^{k-t} w_t(v_i, v_j).$$

4.2.3 Computation

The PageRank value transferred from a given node to the targets of its neighbors upon the next iteration is divided by all adjacent nodes, according to their edge weights. We set the number of iteration times as 20, which is enough to ensure the convergency in our experiments. And we set the damping factor d to 0.85, which is represented as the transition probability. For each chapter, the output of this model is a ranked list of phrases with the PageRank value.

Formally, the iterative process can be described as the following equations. We first initialize all phrases with the same value as $PR_k(v_i; 0) = \frac{1}{N}$, where N is the total number of nodes. At each time step, the computation yields

$$PR_k(v_i; t+1) = \frac{1-d}{N} + d \sum_{v_j \in M(v_i)} \frac{PR_k(v_j; t) W_k(v_i, v_j)}{\sum_{v_s \in M(v_i)} W_k(v_i, v_s)},$$

where $PR_k(v_i; t)$ denotes the PageRank value of v_i at time t in the k^{th} chapter, and $M(v_i)$ denotes the set of nodes adjacent to v_i. The computation process ensures that the sum of overall PageRank values identically equals to 1 at any time step.

4.3 Measuring the Importance of Topics

However, PageRank method only concerns about relative importance and exaggerates the difference between top phrases. To avoid the situation where one phrase plays a dominant role on the importance of topics, we propose three commonly-used distributions to smooth the result: linear function, sigmoid function and Gaussian function. The gradient of these functions are more gentle, so as to alleviate the "slump" at first several phrase importance in the original ranking. The comparison of the phrase importance distribution between original PageRank value and three new functions is shown in Figure 3.

Thus, we have got a ranking of phrase importance with a more gentle slope. We multiply the phrase distribution of topics and the vector of phrase importance. The product can be viewed as the importance magnitude of the topics in this chapter. The formula is shown as:

$$Imp(Topic) = \sum_{phrase \in Topic} Imp(phrase) F(p(phrase)),$$

where $p(phrase)$ denotes the probability of *phrase* occurring in *Topic* and F denotes one of the transfer functions. Eventually, we sort the topics by their importance, and output a ranked list of topics as the final result of this chapter.

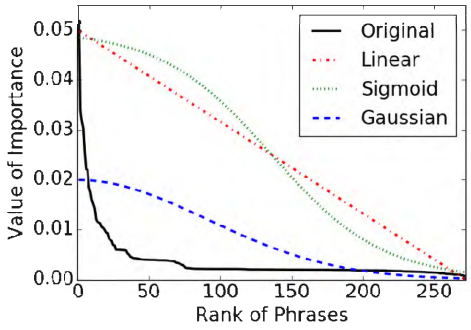


Figure 3: The comparison of the distributions of phrase importance between original PageRank value and three transfer functions that aims to smooth the result of original ranking.

5. EXPERIMENTS

In this section, we evaluate our framework by identifying the most important topics for each chapter. We examine the performance of our algorithm by comparing with four baselines. The ground truth labels come from the problem set annotated by three domain experts. Three metrics are used to evaluate the effect of our ranking algorithm.

5.1 Setups

Our framework first generates several topics from the subtitles in each chapter. Then, we compute the importance of these topics by our algorithm and get a ranking list. These topics are also sorted by ground truth labels, which leads to an ideal ranking. Based on these two rankings, we then compute the metric score of our ranking in this chapter. At last, we take the average among chapters as the performance of our algorithm. Besides, we also try different variants of our algorithm by taking different transfer functions and altering the damping factor.

5.2 Baseline Algorithms

To evaluate the performance of our algorithm, we take four commonly-used strategies as baselines to rank the importance of phrases: (1) Random; (2) Bag-of-Words; (3) TF-IDF; (4) TextRank. For the comparability, these baselines also adopt the topics generated from LDA model as ranking items.

Random Strategy simply ranks the topics by random selection. *Bag-of-Words Strategy* views the frequency of each phrase as the importance in a certain chapter. One shortage of the Bag-of-Words is that some phrases having a high raw count in every chapter do not obviously overweigh than other phrases. *TF-IDF Strategy* is a numerical statistic that addresses this problem by weighting the phrase frequencies through the inverse of document frequency. *TextRank Strategy* in our experiments is followed by [13], which leverages neither previous chapters nor transfer functions.

5.3 Ground Truth and Metrics

For students who want to complete the course are more likely to finish the quizzes and exams [2, 10], we think they pay

Type	Algorithm	nDCG	MAP	τ_B
Baseline	Random	0.838	0.586	0.000
	BoW	0.867	0.631	0.007
	TF-IDF	0.850	0.580	-0.039
	TextRank	0.869	0.640	-0.010
Ours	PR-Linear	0.871	0.645	0.211
	PR-Sigmoid	0.883	0.649	0.256
	PR-Gaussian	0.878	0.613	0.144
	α-PR	0.900	0.749	0.263
	α-PR-Linear	**0.920**	0.752	0.237
	α-PR-Sigmoid	0.917	**0.761**	**0.266**
	α-PR-Gaussian	0.906	0.747	0.255

Table 2: The comparison of performance between four baselines and our algorithm. For all metrics, a higher value means a better performance.

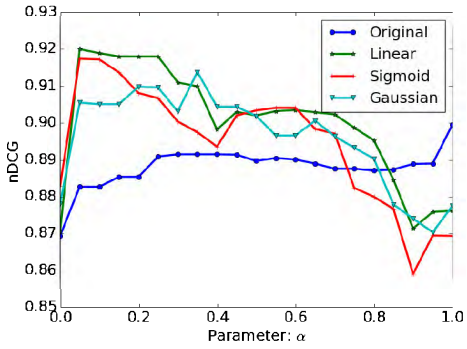


Figure 4: The change of nDCG in different PageRank variants, with α tuned from 0 to 1.

a higher value on the topics which count for more in the assignments. Thus, in this paper, we define the importance of a topic as *"the number of problems that involve this topic"*.

Three domain experts in computer science independently annotated the relevance between the problems and the topics. Specifically, given the problem set and the topics we generated, raters labeled each topic with all the problems whose content is related to this topic. The Cohen's Kappa for the annotations was 0.535 (in the range of $[-1, 1]$), which indicated moderate agreement on inter-reliability. Considering the different understanding of generated topics between raters, we took the union set of problems selected by three raters as the final result. Then, we define the number of problems in this set as ground truth. This process induces a human-generated ranking, which is then compared to the ranking computed by our algorithm. We use three kinds of metrics to evaluate the effectiveness of our ranking algorithm: nDCG, MAP and Kendall's τ, which are widely used for ranking model.

5.4 Results

5.4.1 Performance Comparison

Table 2 shows the comparison of performance between baselines and our algorithms. We report seven variants of our algorithm, which differ in whether combines previous chapters as additional information and which transfer function is used for smoothing. We find that all the variants outperform the baselines. The best variant (α-PR-Sigmoid) yields a 18.9 percent boost of MAP score, compared with TextRank. The results also show the consistency among different metrics. Besides, the methods which combine the content of previous chapters have a significant improvement, compared with those not combine. In addition, we find the transfer functions effective no matter whether or not the method combines the previous chapters.

We then discuss the possible reasons why our algorithms beat the baselines, especially Bag-of-Words and TF-IDF. Firstly, we think *PageRank methods leverage the relation between phrases.* The PageRank method suggests that the phrase is important if the neighbors linked to it are important, so that an important phrase can be explored even if it does not occur so often. Then, *combining previous chapters provides the phrase graph with richer structure information.* One reliable explanation is that some phrases and relations not appearing in the current chapter play a role as "hubs" that connect two important phrases. At last, *transfer functions alleviate the bias from PageRank.* For the importance of top phrases have been exaggerated in PageRank, the topics having these phrases with a higher probability will surpass the others.

5.4.2 Parameter Analysis

When we combine graphs of previous chapters, the damping factor α should be preset. The analysis of α is shown in Figure 4. The situations are almost consistent when using different metrics. Note that when α equals to 0, the method will degrade into those not combining the previous chapters.

We observe an interesting phenomenon that as α tuned from 0.05 to 1.00, the performance trends downward when using transfer functions, while the performance remains unchanged in most of the time, but has an increase at 1.00 when using PageRank value directly. Therefore, during the experiments in Table 2, we set α to 0.05 if we use a transfer function for smoothing and set it to 1.00 otherwise. Because when using a transfer function, a lower value of α enables the current graph to enrich the structure information without influencing the relation between phrases. However, when using the original value, the importance of top phrases were exaggerated, so that α was set as 1.00 to "dilute" the effect of top phrases.

6. DISCUSSION

The experiments have shown the performance of ranking the topic importance within chapters, which is useful for students to know the emphasis of upcoming lectures. Moreover, when students prepare for exams, our framework can also guide students according to their learning status. We assume that two students (S_A and S_B) are preparing for the mid-term exam, including 8 chapters. S_A have learned all the content well, while S_B is deficient in "Linear List", "Queue and Stack", "Binary Tree Application" and "Tree and Forest". We take all subtitles as inputs for S_A, so that we can design a overall review plan. While we just take subtitles in those four chapters as inputs for S_B, in order to

Rank	Topics for S_A	Topics for S_B
1	logical structure	sequential list
2	complete binary tree	linear list
3	linear list	binary search tree
4	binary tree structure	binary tree structure
5	binary tree traversal	tree structure

Table 3: The top five topics for S_A and S_B. Each topic is concluded with one phrase.

concentrate on the topics among weak points. The results are shown in Table 3.

$Case_A$ shows that our algorithm suggests topics about "binary tree" as the most important content. In fact, the tree structure is indeed the most important in the first half of the course, for three chapters introduce the foundation, application, and extension of binary tree separately. In $Case_B$, our algorithm puts more emphasis on "linear list". One reliable explanation is that linear list is a fundamental data structure and the instructor frequently mentions it when introducing the implementations of queue, stack, tree structure.

7. CONCLUSION AND LIMITATION

In this paper, we proposed a novel framework to provide guidance for MOOC students before their learning. Our method first generated topics from video subtitles by LDA model. Then, we ranked the importance of phrases based on a particular PageRank method. At last, we smoothed the PageRank value and measured the importance of topics. As the result, we displayed the most important topics of the upcoming chapter. Experiments showed the effectiveness of our algorithm according to three metrics.

Several factors limited the findings of our study. One was the diversity of our dataset, which included only one scientific course. However, it is time-consuming to label the topics with the problems, and the annotations have to be done by domain experts. Another limitation was lack of real personalized guidance. We have considered to further our study by understanding student learning behaviors and including such information into the phrase graph. Nonetheless, the main objective of our study is to introduce such a novel framework that can provide guidance for students at the early stage of their learning process.

8. ACKNOWLEDGEMENT

This paper is partially supported by the National Natural Science Foundation of China (NSFC Grant Nos. 61472006 and 91646202) as well as the National Basic Research Program (973 Program No. 2014CB340405).

9. REFERENCES

[1] A. Agrawal, J. Venkatraman, S. Leonard, and A. Paepcke. Youedu: addressing confusion in mooc discussion forums by recommending instructional video clips. In *Educational Data Mining 2015*, 2015.

[2] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Engaging with massive online courses. In *Proceedings of the 23rd international conference on World wide web*, pages 687–698. ACM, 2014.

[3] S. Basu, A. Wu, B. Hou, and J. DeNero. Problems before solutions: Automated problem clarification at scale. In *Proceedings of the Second ACM Conference on Learning@ Scale*, pages 205–213. ACM, 2015.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[5] A. Bougouin, F. Boudin, and B. Daille. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International Joint Conference on Natural Language Processing (IJCNLP)*, pages 543–551, 2013.

[6] G. G. Conole. Moocs as disruptive technologies: strategies for enhancing the learner experience and quality of moocs. *Revista de Educación a Distancia*, (39), 2015.

[7] G. A. Fowler. An early report card on massive open online courses. *The Wall Street Journal*, 8, 2013.

[8] P. J. Guo, J. Kim, and R. Rubin. How video production affects student engagement: An empirical study of mooc videos. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 41–50. ACM, 2014.

[9] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction (kli) framework: Toward bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 2010.

[10] S. Kolowich. Coursera takes a nuanced view of mooc dropout rates. *The chronicle of higher education*, 2013.

[11] C. E. Kulkarni, M. S. Bernstein, and S. R. Klemmer. Peerstudio: rapid peer feedback emphasizes revision and improves performance. In *Proceedings of the Second ACM Conference on Learning@ Scale*, pages 75–84. ACM, 2015.

[12] N. Matsuda, T. Furukawa, N. Bier, and C. Faloutsos. Machine beats experts: Automatic discovery of skill models for data-driven online course refinement. In *Educational Data Mining 2015*, 2015.

[13] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. In *Conference on Empirical Methods in Natural Language Processing*, 2004.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[15] Z. Pardos, Y. Bergner, D. Seaton, and D. Pritchard. Adapting bayesian knowledge tracing to a massive open online course in edx. In *Educational Data Mining 2013*, 2013.

[16] N. Sonwalkar. The first adaptive mooc: a case study on pedagogy framework and scalable cloud architectureąlpart i. In *MOOCs Forum*, volume 1, pages 22–29, 2013.

[17] Z. Wang, J. Zhu, X. Li, Z. Hu, and M. Zhang. Structured knowledge tracing models for student assessment on coursera. In *Proceedings of the Third ACM Conference on Learning@ Scale*, pages 209–212. ACM, 2016.

[18] D. Yang, M. Wen, I. Howley, R. Kraut, and C. Rose. Exploring the effect of confusion in discussion forums of massive open online courses. In *Proceedings of the Second ACM Conference on Learning@ Scale*, pages 121–130. ACM, 2015.

Grouping Students for Maximizing Learning from Peers

Rakesh Agrawal
Data Insights Laboratories
ragrawal@acm.org

Sharad Nandanwar
Indian Institute of Science
sharadnandanwar@csa.iisc.ernet.in

M. N. Murty
Indian Institute of Science
mnm@csa.iisc.ernet.in

ABSTRACT

We study the problem of partitioning a class of N students into k groups of n students each ($N = k \times n$), such that their learning from peer interactions is maximized. In our formalization of the problem, any student is able to increase his score in the subject the class is studying up to the score of the student who is at p-percentile among his higher ability peers. In contrast, the past work presumed that only students with score below the group mean may increase their score. We give a partitioning algorithm that maximizes total gain summed over all the students for any value of p such that $100/(100-p)$ is integer valued. The time complexity of the proposed algorithm is only $\mathcal{O}(N \log N)$. We also present experimental results using real-life data that show the superiority of the proposed algorithm over current strategies.

1. INTRODUCTION

A basic problem that has challenged educators for a long time is how to group students in a class in order to supplement their learning from the teacher with the learning from peers [6, 11]. Two popular strategies currently in vogue are: i) heterogeneous (also called diversity-based) grouping, and ii) homogeneous (also referred to as stratified or ability-based) grouping [5]. Both have their ardent proponents. The results from the empirical studies on the relative effectiveness of the two are inconclusive and the public opinion has also been mixed [3, 9].

In a major departure from the conventional thinking, a computational perspective was taken to address this problem in [1]. However, the learning model underlying the proposed algorithmic approach postulated that only the below average students are able to increase their ability score [4]. This paper removes this limitation, recognizing that every student can benefit from peer interactions [6, 8].

1.1 Contributions

- We admit a general learning model that specifies that any student is able to increase his ability score up to the level of the student who is at p-percentile amongst his higher ability peers. The value of p is an input parameter, selected by the educator. The model in [8] can be viewed as a special case, with p set to 100.
- For the above learning model, we provide an algorithm for partitioning N students into k groups of n students each ($N = k \times n$) with the goal of maximizing learning gain summed over all the students. We show that the algorithm is optimal for the values taken by p such that $100/(100-p)$ is integer-valued. Thus, it is optimal for $p \in \{99, 98, 95, 90, 80, 75, 66\frac{2}{3}, 50\}$. The time complexity of the algorithm is $\mathcal{O}(N \log N)$.
- We present experimental results using real datasets, showing the superiority of our approach over current strategies.

1.2 Limitations

- Although our learning model has been abstracted from the findings in the education literature, a rigorous empirical validation of the model is future work. The insights gained are nonetheless instructive.
- Teaching others and giving help has been shown to be positively correlated to increase in learning [2]. Incorporating such learning gains for high ability students is future work.

2. RELATED WORK

The question of how to group students to maximize their gain from peer interactions was first addressed from a computational perspective in [1]. The authors proposed two functions to model learning gains. The first maximizes the number of students who improve their ability score [4], while the second incorporates the extent of these improvements. In both the cases, however, only the below average students benefit and the higher ability students have zero gain. The authors showed that the partitioning problem with the goal of maximizing the number of benefiting students is NP-complete, while they left open the question of the complexity class of the problem with the second gain function.

The viewpoint that every student can learn from the higher ability peers is also present in [8]. In their model, every student may increase his ability to a fixed level, which is the ability of the highest ability student, i.e. $p = 100$. This assumption is too rigid and optimistic. In contrast, we admit various levels of gain for different students.

Our problem bears resemblance with the expert-team formation problem, in which the experts are multi-dimensional vectors of skills and the goal is to find a team that can collectively perform a given task requiring certain skills [10]. However, our students are described by 1-dimensional scores, and our objective is not to locate a single team, but to partition the students such that their learning gain is maximized.

Our problem also superficially resembles the classical clustering problem [7]. However, unlike the classical clustering, which aims to maximize the similarity of all the points in a cluster to a cluster center, our problem has no one point in a partition with respect to which the distance of all other points needs to be optimized (see Fig. 1).

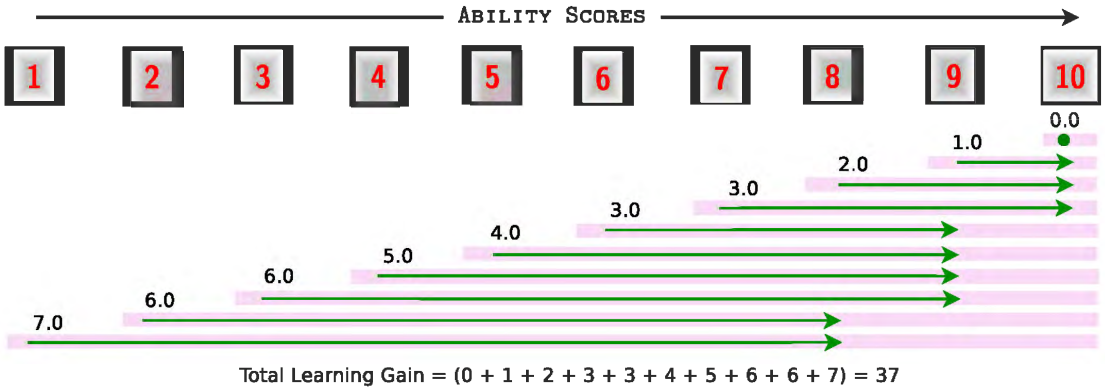


Figure 1: Computation of the potential learning gain for a group of ten students with 75-percentile chosen as the reference point. The i^{th} box contains the score of the i^{th} student. The learning gain for each student is the difference between his score and the score of student at p-percentile amongst his peers having higher score than him. For the first student, the index of the student at 75-percentile amongst his higher ability peers is $(1 + \lceil (10-1) * 75/100 \rceil) = 8$. Since the score of the latter is eight, the gain for the first student is $(8-1) = 7$. For the second student, the index of the student at 75-percentile amongst his higher ability peers is also 8 $(2 + \lceil (10-2) * 75/100 \rceil)$, thus giving him a gain of $(8-2) = 6$, and so on. The gain for the last student is zero, as there is no one above to learn from.

3. PROBLEM STATEMENT

We have a class of N students. Each student i is associated with score $\theta_i \in \mathbb{R}_{\geq 0}$, representing student's ability in the subject the class is studying [4]. For simplicity, scores are assumed to be distinct, so there is a one to one correspondence between the student i and the score θ_i. Students are ordered in the increasing order of scores.

Students are able to increase their score through interactions with peers in the group in accordance with a gain function [12, 13]. The gain from peer learning for a group G is given by a function $\mathcal{L}$. Our objective is to find k groups of n students each $(N = k \times n)$, such that the overall gain for students is maximized. That is, our objective is

$$\max_{\mathcal{G}} \sum_{G \in \mathcal{G}} \mathcal{L}(G). \tag{1}$$

The learning function is of the form

$$\mathcal{L}(G) = \sum_{i=1}^{|G|} \left(R_i^G - \theta_i \right), \tag{2}$$

where R_i^G is the reference score for the G's i^{th} ranked student. The intuition is that each student can increase his score up to the reference score.

3.1 Learning up to p-Percentile

PROBLEM 1 (P-PERCENTILE PARTITIONING PROBLEM). *The gain function in Eq. 2 is given by*

$$\mathcal{L}^p(G) = \sum_{i=1}^{|G|} \left(p_i^G - \theta_i^G \right), \tag{3}$$

where p_i is the score of the student whose score is at the p-percentile position of the scores of the students having higher score than the i^{th} student in G.

For a given set of scores, the p-percentile score is the score below which $p\%$ of scores fall. To find the p-percentile score, the corresponding index is calculated first, which is $\lceil np/100 \rceil$. The value at this index then is the p-percentile score. Thus,

$$p\text{-percentile}(\theta_1, \theta_2, \ldots, \theta_n) = \theta_{\lceil n.p/100 \rceil}. \tag{4}$$

Fig. 1 graphically illustrates the percentile gain function.

4. SOLUTION

THEOREM 1. *For values of p such that $p/(100-p)$ is integer-valued, the p-Percentile Partitioning problem can be solved optimally in $\mathcal{O}(N \log N)$ time.*

We shall prove the theorem constructively by providing an optimal algorithm whose time complexity is $\mathcal{O}(N \log N)$. It is named Percentile Partitions and its pseudo-code is shown in Algorithm 1. The algorithm exploits the special structure of our problem that we elicit next.

We first expand the equation for learning gain w.r.t. p-percentile as given in Eq. 3 into

$$\begin{aligned}\mathcal{L}^P(G) = &\left(\text{p-percentile}(\theta_2^G, \theta_3^G, \ldots, \theta_n^G) - \theta_1^G\right) + \\ &\left(\text{p-percentile}(\theta_3^G, \theta_4^G, \ldots, \theta_n^G) - \theta_2^G\right) + \\ &\ldots + \left(\text{p-percentile}(\theta_n^G) - \theta_{n-1}^G\right).\end{aligned}$$

Using the definition of p-percentile from Eq. 4, the above can be written as

$$\begin{aligned}\mathcal{L}^P(G) = &(\theta^G_{1+\lceil (n-1)p/100 \rceil} - \theta_1^G) + (\theta^G_{2+\lceil (n-2)p/100 \rceil} - \theta_2^G) + \\ &\ldots + (\theta_n^G - \theta_{n-1}^G).\end{aligned}$$

To this we add the term $(\theta_n^G - \theta_n^G)$ corresponding to zero gain of the n^{th} student. Thus, we have

$$\begin{aligned}\mathcal{L}^P(G) = &(\theta^G_{1+\lceil (n-1)p/100 \rceil} - \theta_1^G) + (\theta^G_{2+\lceil (n-2)p/100 \rceil} - \theta_2^G) + \\ &\ldots + (\theta^G_{(n-1)+\lceil p/100 \rceil} - \theta^G_{n-1}) + (\theta_n^G - \theta_n^G).\end{aligned}$$

Collecting the positive and negative terms together, we get

$$\begin{aligned}\mathcal{L}^P(G) = &\left(\theta^G_{1+\lceil (n-1)p/100 \rceil} + \theta^G_{2+\lceil (n-2)p/100 \rceil} + \ldots \right. \\ &\left. + \theta^G_{(n-1)+\lceil p/100 \rceil} + \theta_n\right) \\ &- \left(\theta_1^G + \theta_1^G + \ldots + \theta_{n-1}^G + \theta_n^G\right),\end{aligned}$$

which can be written succinctly as

$$\mathcal{L}^P(G) \;=\; \sum_{i=1}^{n} \theta^G_{i+\lceil (n-i)p/100 \rceil} \;-\; \sum_{i=1}^{n} \theta^G_i. \tag{5}$$

Using this equation, our objective becomes

$$\max_{\mathcal{G}} \sum_{G \in \mathcal{G}} \left(\sum_{i=1}^{n} \theta^G_{i+\lceil (n-i)p/100 \rceil} - \sum_{i=1}^{n} \theta^G_i \right).$$

The second component in the above sum is constant for any given set of ability scores. Therefore, our objective can be simplified to

$$\max_{\mathcal{G}} \sum_{G \in \mathcal{G}} \sum_{i=1}^{n} \theta^G_{i+\lceil (n-i)p/100 \rceil}. \tag{6}$$

LEMMA 1. *Given $p \in [0, 100]$ and an ascending sequence of $\theta_i \in \mathbb{R}_{\geq 0}$, for $(100-p) \nmid 100$, $\sum_{i=1}^{n} \theta_{i+\lceil (n-i)p/100 \rceil}$ is equivalent to $\sum_{i=1}^{n} \gamma_i \cdot \theta_i$, where*

$$\gamma_i = \begin{cases} \frac{100}{100-p}, & \text{if } \left\lceil \frac{np}{100} \right\rceil < i \leq n \\ mod(n, \frac{100}{100-p}), & \text{if } \frac{100}{100-p} \nmid n \text{ and } i = \left\lceil \frac{np}{100} \right\rceil \\ 0, & \text{otherwise.} \end{cases}.$$

PROOF. It is to be noted that a student at index i improves up to the score of student at index $i + \lceil (n-i)p/100 \rceil$. As the student indexes are traversed from the higher-score end to the lower end, with unit decrease in value of i, the quantity $\lceil (n-i)p/100 \rceil$ increments by unity, except for the values of i for which $(n-i)p$ is a multiple of hundred. In the latter case, although there is a decrement in the value of i by one, the value of $\lceil (n-i)p/100 \rceil$ stays the same as that of $\lceil (n-i-1)p/100 \rceil$, causing the index up to which students are improving to decrement by one. It is easy to derive that this process repeats itself after a period of $100/(100-p)$. Further, when n is not a multiple of the above period, there will be $\text{mod}(n, 100/(100-p))$ students who will be improving up to the smallest index value. For the remaining students, as no other student improves up to their score, a γ value of zero is straightforward. □

EXAMPLE 1. *In Fig. 1, we have $n = 10$ and $p = 75$. Thus, in accordance with Lemma 1, we have*

$$\gamma_i = \begin{cases} 4, & \text{if } 8 < i \leq 10 \\ 2, & \text{if } i = 8 \\ 0, & \text{otherwise.} \end{cases}$$

.

The above may also be verified visually from Fig. 1. It is easy to note that the students at 7^{th}, 8^{th}, and 9^{th} index improve up to the score of the 10^{th} student, while the 10^{th} student with zero gain remains at the same score. This makes the score of the 10^{th} student visible four times in the updated scores, leading to the γ value of four. Similarly, the score of the student at 9^{th} index is also visible four times because of students at 3^{rd}, 4^{th}, 5^{th}, and 6^{th} indexes improving up to his score. On the other hand, only students at 1^{st} and 2^{nd} indexes improve up to the score of 8^{th} student. Hence, a γ value of two for the 8^{th} student. No one is improving his score up to the score of any of the students at index below eight. So, the γ values corresponding to them are zero.

Unfortunately, when $(100-p) \nmid 100$, the coefficients γ_i's have complex structure and we defer their study to future work.

Algorithm 1 (Percentile_Partitions) Optimal Partitioning for maximizing Learning Gain - learning up to p-percentile

Input: Distinct descending scores $\{\theta_1, \theta_2, \ldots, \theta_N\}$, Percentile p, Number of groups k, Size of each partition n, $k \times n = N$.
$G_1 = G_2 = \ldots = G_k = \phi$
$m \leftarrow 100/(100-p)$
$q \leftarrow \lfloor n/m \rfloor$
$\hat{q} \leftarrow \lceil n/m \rceil$
if $\text{mod}(n, m) \neq 0$
  $M \leftarrow \{\theta_{kq+1}, \ldots, \theta_{kq+k})\}$
  for $i \in \{1, 2, \ldots, k\}$
    $G_i \leftarrow G_i \bigcup M_i$
  end for
end if
$H1_{global} \leftarrow \{\theta_1, \theta_2, \ldots, \theta_{kq}\}$
$H2_{global} \leftarrow \{\theta_{k\hat{q}+1}, \ldots, \theta_{N-1}, \theta_N\}$
for $i \in \{1, 2, \ldots, k\}$
  $H1_{part} \leftarrow$ randomly sample q scores from $H1_{global}$ without replacement.
  $H2_{part} \leftarrow$ randomly sample $(n - \hat{q})$ scores from $H2_{global}$ without replacement.
  $G_i \leftarrow G_i \bigcup H1_{part} \bigcup H2_{part}$
end for
return $\{G_1, G_2, \ldots, G_k\}$

4.1 Percentile_Partitions

Lemma 1 leads to our optimal partitioning algorithm, which is shown in Algorithm 1. The algorithm first divides the input ability scores into two or three sets depending on whether $\text{mod}(n, 100/(100-p))$ is zero or not respectively. The first set $H1_{global}$ consists of scores that contribute by a factor of $100/(100-p)$ to the learning gain. The second set M if present, consists of scores that contribute by a factor of $\text{mod}(n, 100/(100-p))$. Finally, the third set $H2_{global}$ consists of scores that have zero contribution. These sets correspond to the three different values of the γ coefficients. They are such that $H1_{global} \succcurlyeq M \succcurlyeq H2_{global}$, where $A \succcurlyeq B$ means all elements of set A are greater or equal compared to any element of set B. For each of these sets then, the algorithm creates k equal random partitions. These partitions are then merged to create the final k partitions. The example below illustrates the algorithm.

EXAMPLE 2. *Consider a set of 20 students with ability scores $\{\theta_1, \theta_2, \ldots, \theta_{20}\}$, sorted in the descending order. The set is to be partitioned into four groups, each containing five students. Each student can learn up to the score of the student who is at $66\frac{2}{3}$-percentile of students above.*

For $p = 66\frac{2}{3}$ and $n = 5$, we have $m = 3$, $q = 1$, and $\hat{q} = 2$. The algorithm breaks the scores into three sets:
$H1_{global} = \{\theta_1, \theta_2, \theta_3, \theta_4\}$
$M \quad\quad = \{\theta_5, \theta_6, \theta_7, \theta_8\}$
$H2_{global} = \{\theta_9, \theta_{10}, \theta_{11}, \theta_{12}, \theta_{13}, \theta_{14}, \theta_{15}, \theta_{16}, \theta_{17}, \theta_{18}, \theta_{19}, \theta_{20}\}$

For each set, four equal-sized random partitions are created, which are then merged to create four groups:
$G_1 = \{\theta_3\} \bigcup \{\theta_6\} \bigcup \{\theta_{17}, \theta_{10}, \theta_{15}\}$
$G_2 = \{\theta_1\} \bigcup \{\theta_7\} \bigcup \{\theta_{19}, \theta_{16}, \theta_9\}$
$G_3 = \{\theta_2\} \bigcup \{\theta_5\} \bigcup \{\theta_{13}, \theta_{18}, \theta_{12}\}$
$G_4 = \{\theta_4\} \bigcup \{\theta_8\} \bigcup \{\theta_{14}, \theta_{20}, \theta_{11}\}$

***Note**: There are many equally good ways of partitioning $H1_{global}$, M, and $H2_{global}$. The above is just one of them.*

4.2 Proof of Theorem 1

Clearly, if the input scores were already in the descending order, the time complexity of the Algorithm 1 is $\mathcal{O}(N)$. If the input scores were unsorted, then the extra sorting step would make the complexity $\mathcal{O}(N \log N)$.

The optimality of the algorithm follows from the structure in the values taken by the coefficient γ's. Before proceeding further, we state the following lemma:

LEMMA 2. *For given ordered sets of real numbers, $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$, the quantity $\sum_{a \in A, b \in B} ab$, s.t. each $a \in A$ and $b \in B$ is used exactly once, is maximized if the elements are chosen in a manner such that the product of elements at the same index from A and B is taken.*

Now, according to Lemma 1, γ_i can take only one of the three values and they have ordering amongst them given by $100/(100-p) > mod(n, 100/(100-p)) \geq 0$. The partitions created by the algorithm satisfy, $H1_{global} \succcurlyeq M \succcurlyeq H2_{global}$. Thus, in light of Lemma 2, it is easy to observe that our objective is maximized as the set of students with higher(lower) scores get mapped to highest(lowest) coefficient. Moreover, the random perturbations within $H1_{global}$, M, or $H2_{global}$ do not affect the gain value as all the scores from a set are involved in product with the same γ value.

5. EXPERIMENTS

5.1 Datasets

1. SSC Scores (Normal distribution): Staff Selection Commission - Combined Graduate Level Examination (SSC-CGL) is conducted all across India to recruit employees for various departments of Government of India. The scores of candidates for the 2016 examination, categorized into different regions of the country, are available at ssc.nic.in. The distribution of scores in every region is close to normal. We took the scores from the North Western (SSC-NWR) region that exhibits the largest variance.

2. GATE Scores (Log-Normal distribution): In India, Graduate Aptitude Test in Engineering (GATE) is conducted every year to test the competency of undergraduate students in various engineering disciplines. We took the available scores from year 2016. We experimented with scores from Mech. (GATE-ME), with largest variance.

3. StkXchg UpVotes (Pareto distribution): On the Stack Exchange platform, users can ask and answer questions on various topics. Additionally, they can up-vote or down-vote a question. The number of up-votes a user receives is an indicative measure of his level of expertise. Pareto distribution fitted the data for the active users having at least one up-vote. The Stack Exchange data dump is available from archive.org/details/stackexchange. We take data for Stack Overflow that ehibits lowest skew in distribution.

5.2 Algorithms

In addition to Percentile_Partitions, we consider two algorithms that correspond to the strategies currently prevalent in practice: Stratified and Random.

1. Stratified: This algorithm puts in each group those students who exhibit similar ability. This grouping represents the practice of homogeneous or ability-based grouping.

2. Random: Students are assigned to groups randomly. This method corresponds to the practice of heterogeneous or diversity-based grouping.

5.3 Set Up

We conducted our experiments setting the number of students, N, to 1024. We varied the number of groups, k, over {2,4,8,..., 512}, and the reference percentile point p over $\{50, 66\frac{2}{3}, 75, 80, 90, 95, 98, 99\}$. Thus, for each dataset, we randomly sample 1024 scores and generate the groups for different combinations of k and p values. In order to have tight confidence intervals, we repeat this exercise 30 times each and report average learning gain.

For the groups generated by Percentile_Partitions, we compute learning gain using Eq. 3. When applying Stratified or Random to a dataset, we generate groups only once but compute gain using the appropriate parameter value for p.

We also study the group structures generated by different algorithms. By the structure of a group, we mean the distribution of scores in the group. Although we run each algorithm 30 times, we only show the structure of the group generated by the first run.

5.4 Results

Fig. 2 shows the learning gain as the reference percentile value, p, is varied for different algorithms on various datasets. We show the plots for three values for the number of groups, $k \in \{128, 32, 8\}$ (and the corresponding group sizes, $n \in \{8, 32, 128\}$). Fig. 3 shows the learning gain as the number of groups, k, is varied. We show the plots for two percentile values, $p \in \{75, 90\}$. Fig. 4 shows the group structures generated by different algorithms. We show the structures for groups of size, $n = 8$, and for the reference percentile, $p = 75$. We alert the reader that different scales have been used for Y-axis in Figs. 2-3 and a logarithmic scale has been employed for X-axis in Fig. 3 for the sake of clarity.

We see that the overall behavior of different algorithms remains similar across different group sizes and reference percentile values. Clearly, Percentile_Partitions consistently outperforms the other algorithms that corroborates its theoretical optimality. The following additional observations are noteworthy:

- With increasing value of p, total learning gain increases super linearly (Fig. 2). It is because the extent of learning gain for each student increases. The gain plateaus for small groups because beyond some percentile value, all students improve up to the same highest ability student. Then, it does not matter whether the reference percentile is at 90 or 95.
- The advantage of Percentile_Partitions over Random is more pronounced when the number of students in a group is in a more realistic range of 32 or less (Fig. 3). When the number of groups is small and each group is large, Percentile_Partitions assigns very many students randomly

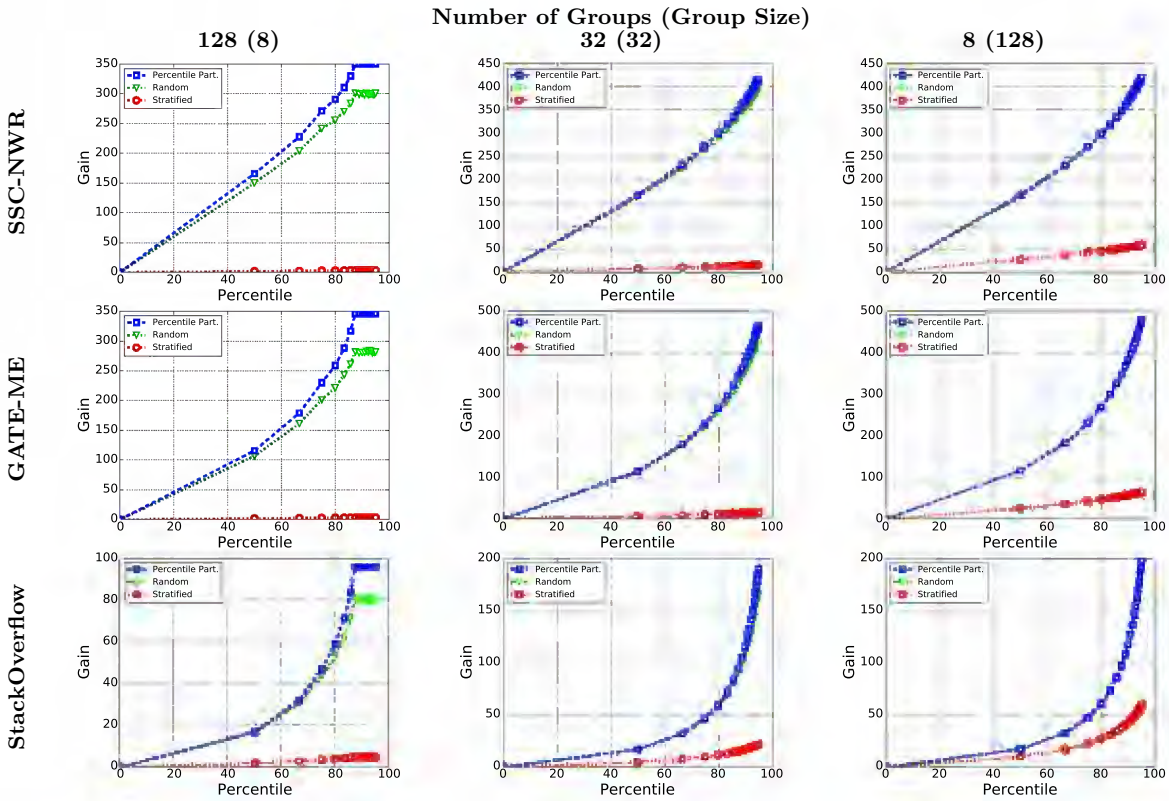


Figure 2: Learning Gain as the reference percentile point, p, is varied.

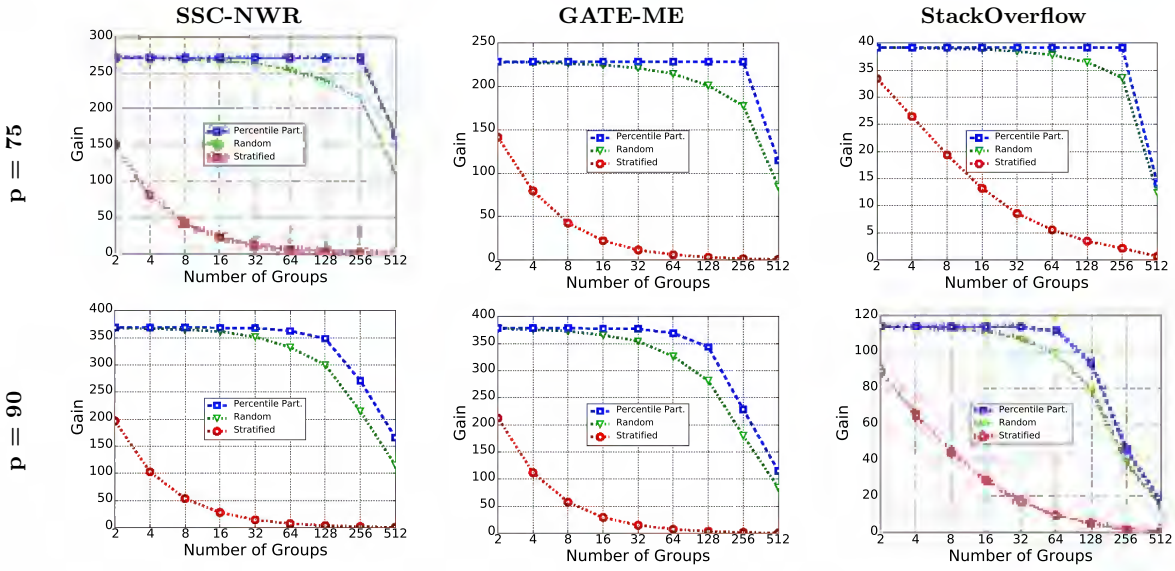


Figure 3: Learning Gain as the number of groups, k, is varied.

and therefore the group structure and gain produced by it become similar to that of Random.

- The learning gain is worst with the stratified strategy. Fig. 4 shows that this strategy produces groups in which the students have similar scores. Therefore, the improvements from peer interactions are small. Fig. 4 also shows that the p-percentile value of every group produced by Percentile Partitions is higher than the global p-percentile value of all the undivided scores. However, this pattern is not true for Random. Some groups generated by Random have p-percentile to the extreme right of global p-percentile. The scores in between the two p-percentiles in such groups do not contribute to the total gain. But then some other groups end up having smaller scores above p-percentile that leads to smaller additions to the total gain. Hence, the superior performance of Percentile Partitions.

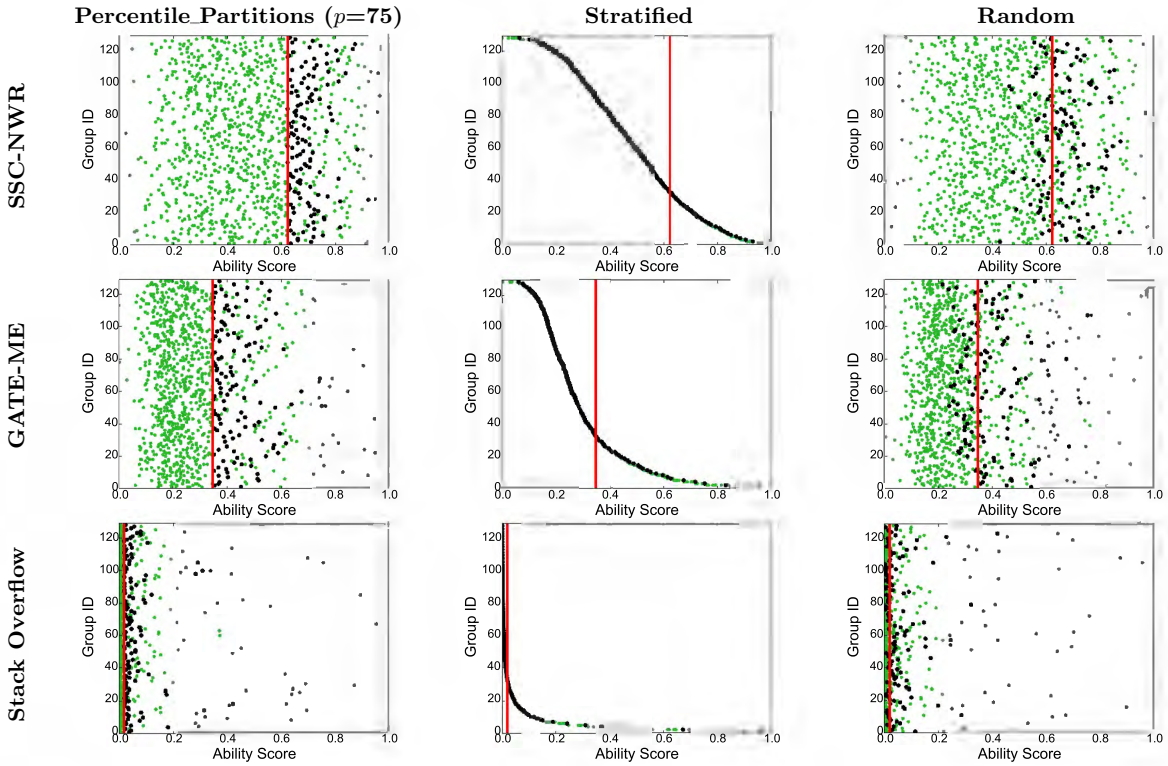


Figure 4: Group structure generated by different algorithms for groups of size 8. Each row in the plots corresponds to a particular group and there is a dot for each ability score in that group. The p-percentile score for each group is plotted in black. The vertical red line shows the global p-percentile score. The groups are numbered according to the order in which they are generated. Only for Percentile Partitions, the p-percentile score for every group is higher than the global p-percentile value.

6. SUMMARY

We investigated the important educational data mining problem of how to group students in a class to maximize their learning gains from peer interactions. We worked with a general learning gain function in which every student is able to increase his ability score up to the score of the student who is at p-percentile amongst his higher ability peers. We gave an algorithm which is provably optimal for maximizing learning gain, the value of p is such that $100/(100-p)$ is integer valued. We also studied the performance characteristics of the proposed algorithm using real-life datasets that corroborated the theoretical analysis and showed its superiority over the current approaches. Surprisingly, the time complexity of optimally grouping N students using our algorithm is only $\mathcal{O}(N \log N)$.

Acknowledgment Rakesh Agrawal's visit to the Indian Institute of Science, where bulk of the present work was done, was funded through the Rukmini Gopalakrishnachar Visiting Chair Professorship. His work was also partially funded by EPFL - School of Computer and Communication Sciences, Data Intensive Applications and Systems Laboratory.

7. REFERENCES

[1] Agrawal, R., Golshan, B., and Terzi, E. Grouping students in educational settings. In *20th ACM SIGKDD international conference on Knowledge discovery and data mining* (2014), pp. 1017–1026.

[2] Bargh, J. A., and Schul, Y. On the cognitive benefits of teaching. *Journal of Educational Psychology 72*, 5 (1980).

[3] Boaler, J., Wiliam, D., and Brown, M. Students' experiences of ability grouping - disaffection, polarisation and the construction of failure. *British educational research journal 26*, 5 (2000), 631–648.

[4] Crocker, L., and Algina, J. *Introduction to classical and modern test theory*. ERIC, 1986.

[5] Esposito, D. Homogeneous and heterogeneous ability grouping: Principal findings and implications for evaluating and designing more effective educational environments. *Review of Educational Research 43*, 2 (1973), 163–179.

[6] Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., and Wenderoth, M. P. Active learning increases student performance in science, engineering, and mathematics. *Proceedings National Academy of Sciences 111*, 23 (2014), 8410–8415.

[7] Jain, A. K. Data clustering: 50 years beyond k-means. *Pattern recognition letters 31*, 8 (2010), 651–666.

[8] Kosheleva, O. Diversity is the optimal education strategy: A mathematical proof. *International Journal of Innovative Management, Information & Production 4*, 1 (2013), 1–8.

[9] Kulik, J. A. An analysis of the research on ability grouping: Historical and contemporary perspectives. Tech. Rep. NRC-G/T-9204, National Research Center on the Gifted and Talented, Storrs, CT, February 1992.

[10] Lappas, T., Liu, K., and Terzi, E. Finding a team of experts in social networks. In *15th ACM SIGKDD international conference on Knowledge discovery and data mining* (2009), pp. 467–476.

[11] Slavin, R. Cooperative learning. *Learning and Cognition in Education* (2011), 160–166.

[12] Vygotsky, L. S. *Mind in society: The development of higher psychological processes*. Harvard Press, 1980.

[13] Webb, N. M. Peer interaction and learning in small groups. *International journal of Educational research 13*, 1 (1989).

Assessing the Dialogic Properties of Classroom Discourse: Proportion Models for Imbalanced Classes

Andrew M. Olney
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
aolney@memphis.edu

Borhan Samei
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
bsamei@memphis.edu

Patrick J. Donnelly
Department of Computer Science
California State University
Chico, CA 95929
pjdonnelly@csuchico.edu

Sidney K. D'Mello
Departments of Psychology & Computer Science
Notre Dame University
Notre Dame, IN 46556
sdmello@nd.edu

ABSTRACT

Automatic assessment of dialogic properties of classroom discourse would benefit several widespread classroom observation protocols. However, in classrooms with low incidences of dialogic discourse, assessment can be highly biased against detecting dialogic properties. In this paper, we present an approach to addressing this imbalanced class problem. Rather than perform classifications at the utterance level, we aggregate feature vectors to classify proportions of dialogic properties at the class-session level and achieve a moderate correlation with actual proportions, $r(130) = .50,\ p < .001,\ CI_{95}[.36, .61]$. We show that this approach outperforms aggregating utterance level classifications, $r(130) = .27,\ p = .001,\ CI_{95}[.11, .43]$, is stable for both low and high dialogic classrooms, and is stable across both automatic speech recognition and human transcripts.

Keywords

dialogic instruction, questions, authenticity, machine learning, imbalanced classes

1. INTRODUCTION

Classroom observation for measuring teaching effectiveness is currently used in 47 states [1]. Simply stated, classroom observation involves a trained evaluator watching how a class is taught and using a rubric to score the teacher's performance. The widespread use of classroom observation is based on previous research which indicates that instructional quality has a greater impact on student achievement than class size, teacher experience, or teacher graduate education [16]. Beyond such research findings, classroom observation is also driven by the teacher accountability era coinciding with the passage of the federal No Child Left Behind Act, which mandated annual testing of students by all states. In this highly politicized environment, classroom observation is increasingly being used to determine teacher's salary and tenure.

Curiously, given the high stakes associated with classroom observation, the majority of research linking instructional quality to student achievement over the past several decades has been correlational only. However there has been an increasing interest in randomized controlled trials. One recent randomized trial is the multi-year Measures of Effective Teaching (MET), which tracked approximately 3,000 teachers in seven states [4]. In year 1, MET researchers built predictive models of teaching effectiveness, and in year 2, teachers were randomly assigned to new classrooms to test the predictive models from year 1. Major MET findings were that teaching effectiveness measured via classroom observation protocols correlated with achievement gains and that question asking behavior was a key component of variability in teaching quality [11].

Although instructional quality is linked to achievement, the current practice of assessing instructional quality through classroom observation is logistically complex and expensive, requiring observer rubrics, observer training, and continuous assessment to maintain a pool of qualified observers [2]. To address these practical challenges, our work has focused on the automated assessment of classroom discourse, with a particular emphasis on measuring dialogic questions in classrooms. Our approach is to automate an existing, fine grained classroom observation protocol that focuses on dialogic questions, known as the Classroom Language Assessment System[1] [13]. Unlike the classroom observation protocols used in the MET study, in which an observer makes rubric-based judgments approximately every 10 minutes, CLASS uses fine-grained coding at the question level, creating suitably detailed labeled data for machine learning purposes.

[1]CLASS denotes the CLASS created by Nystrand and colleagues, as opposed to the CLASS used in the MET study.

The dialogic instruction measured by CLASS is characterized by open-ended discussion and the exchange of ideas (cf. [3]), which in turn are characterized by questions that truly seek information (authentic questions) and which incorporate ideas from the student (questions with uptake). For example, "How did you feel by the end of the story?" is an authentic question because there is no pre-scripted response, and a follow-on question "Why do you think that is?" has uptake because "that" refers to the student's previous reply. As is clear in these examples, dialogic properties are contextualized by the discourse such that the antecedents and consequents of the question shape whether a question is authentic or has uptake. Previous research using CLASS has shown that authenticity and uptake are significant predictors of student achievement [10, 9, 14].

Our project, which we call CLASS 5, seeks to fully automate classroom observations under the CLASS protocol. In our work, we have used archival data collected in previous CLASS projects, containing human transcripts of dialogic questions, as well as new data using automatic speech recognition (ASR) of teacher speech. Models built with archival human transcript data are as effective at classifying authenticity and uptake as humans on isolated questions [18]. However, as we began to analyze the new CLASS 5 data, we realized that there were two serious limitations undermining our existing models. First, the archival data used in previous work [18, 17] contained only transcripts of questions, and even these did not represent all questions but a subset of questions that were *instructional*, and so excluded rhetorical questions, procedural questions, and discourse management questions [13]. In the archival data, approximately 50% of the questions were coded as authentic questions. In contrast, the new CLASS 5 data included all questions and non-questions, i.e. all utterances, from which authentic questions must be detected. Secondly, in the CLASS 5 data, the base rates for dialogic properties were dramatically lower than in previous samples. For example, authentic questions in our new data collection constituted about 30% of instructional questions compared to approximately 50% of instructional questions in the archival data; moreover, authentic questions in our new data constituted only about 3% of all utterances. Therefore to be robust in detecting dialogic properties across samples, our models must be able to deal adequately with imbalanced classes.

The so-called "class imbalance problem" is well known in the data mining community, and has been proposed as one of data mining's top 10 challenging problems [20]. The essence of the problem is that a classifier can maximize accuracy by always selecting the majority class and that this strategy, typically considered as a baseline for performance, becomes increasingly hard to beat as the majority class distribution approaches 100%. A review of the class imbalance problem describes three major approaches for addressing it [8]. First, algorithmic approaches may be used to bias learning towards the minority class. Secondly, preprocessing methods may change the class distribution before learning occurs, either by undersampling the majority class or oversampling the minority class. Thirdly, cost-sensitive approaches may be used to assign higher costs, or weights, to minority class errors, such that the learning algorithm tries to minimize the total cost.

In this paper, we present another method for addressing the class imbalance problem, which is to transform the problem into a different problem that is easier to handle. Specifically, we explore the consequences of shifting from classifiers that classify utterances as authentic questions to classifiers that classify the proportion of authentic questions in a class session. As will be shown in the remainder of the paper, this problem transformation outperforms aggregating utterance level classifications, is stable for both low and high dialogic classrooms, and is stable across both automatic speech recognition and human transcripts.

2. METHOD

2.1 Data sets

CLASS 5 data. New data for the CLASS 5 project were collected between January 2014 and May 2016 at seven schools in rural Wisconsin. Observations for 132 class sessions taught by 14 different teachers were manually coded using the CLASS system, and audio was simultaneously recorded. Both teacher and school identifiers were preserved with the data. Given the logistical constraints of individual microphones for each student, the recording instrumentation instead focused on high quality teacher audio suitable for ASR that was recorded using a wireless microphone headset. Classroom audio, which included both teacher and student speech, was recorded from a stationary boundary microphone, and was not of sufficient quality to be used for ASR; however, it is useful for marking when students speak. The teacher audio was later automatically segmented into utterances and then submitted to a speech recognition service [6]. Thus this dataset differs from the archival data (see below) in that the transcripts are provided by ASR with its accompanying errors, only teacher speech is transcribed, and the transcripts contain all utterances rather than just instructional questions. The data contained 45,044 utterances, of which 1282 were authentic questions (3% of utterances; 30% of instructional questions) and 290 were questions with uptake (.01% of utterances; .07% of instructional questions). Authenticity and uptake are even more highly related in this data set than in the archival data since only 5 questions have uptake without authenticity. Given the small number of observations of uptake and the finding that virtually all questions with uptake are also authentic, we primarily focused on detecting authenticity.

Archival data. The archival data was collected during the Partnership for Literacy Study (Partnership), a study of professional development, instruction, and literacy outcomes in middle school English and language arts classrooms. The Partnership collected data from 7th- and 8th-grade English and language arts teachers in Wisconsin and New York from 2001 to 2003. Over that two-year period, 119 classes in 21 schools were observed twice in the fall and twice in the spring. Teacher identifiers were not embedded in the CLASS data files, and out of 119 teachers only 70 could be unequivocally matched to data files. However, school identifiers were directly embedded in data files. Classroom observations for Partnership were also conducted using the CLASS annotation system [13]. During this process instructional questions were transcribed, and the transcriptions were mostly accurate but not verbatim. Reliability studies using CLASS indicate that raters agree on question properties approximately 80% of the time, with observation-level inter-rater

correlations averaging approximately .95 [14]. After removing questions with partially incomplete annotations, 25,711 instructional questions remained for use in our analyses, of which 12,862 were authentic questions (50%) and 5,489 were questions with uptake (22%). Authenticity and uptake were highly related: only 593 (2%) questions had uptake without authenticity.

2.2 Features

In early work, we established that word and part-of-speech features that are useful for classifying types of questions [15] were also useful for predicting dialogic question properties like authenticity and uptake [18, 17]. In the present work we have extended these 36 predictive features to include features obtained through syntactic and discourse parsing [12, 19]. At the word level, these new features include 45 part-of-speech tags as well as named entity type, which subdivides real world objects described by proper nouns into 13 classes including PERSON, LOCATION, and DATE. At the sentence level, the features include 47 syntactic dependencies like subject, agent, direct object, or indirect object. And at the discourse level, the features include 18 discourse relations including contrast, elaboration, and topic-change, as well as features for joint, nucleus, and satellite elementary discourse units. Because the discourse parse returns a tree of elementary discourse units, the discourse features were mapped to the sentence level by summing the discourse relations, satellite, joint, and nucleus features that occur in each elementary discourse unit composing the sentence. Anaphora resolution was converted into four features including the number of coreference chains in an utterance extending into future sentences, the sum of those chain's lengths, and the same features in the backwards direction. In other words, the anaphora features capture how well a sentence was connected to other sentences in both directions. While all features were encoded at the sentence/utterance level (i.e. a count of the feature in the utterance), the 36 question features used in previous work were additionally encoded as occurring at either the first token or after the first token. For example, if a *definition keyword* feature occurred in the first token, then that would be recorded as a single count in the corresponding overall feature and the first token feature, but not in the corresponding after the first token feature. Additionally, the named entity PERSON feature was encoded with first token and last token variants based on the observation that questions addressed to students typically use the name at the beginning or end of an utterance if at all. With the positional variants, there were 242 linguistic features in our models that span word, sentence, and discourse levels.

To generate these features we used the CLU processor, which contains syntactic and discourse parsers [19]. Because discourse parsing requires a discourse context, utterances for each classroom observation were grouped into separate files before parsing. The parsers were configured with a maximum sentence length of 120 words, which was empirically determined by observing the lengths of a subsample of utterances. Parses for each class-level file were converted into utterance level features and aggregated into a 242-dimension feature vector where the value at each position was the frequency count of a particular feature in that utterance. Models built at the question level for archival data or utterance level for new data used these 242-dimension feature vectors. Models built at the class-session level used these features but summed them over all questions (Partnership) or utterances (CLASS 5) in a given class. Models at the class-session level additionally added the means and standard deviations of these summed feature vectors, for a total of 726 features.

2.3 Model training

Cross validation. We used cross validation such that a given teacher would not appear in both the training and testing folds, in order to study generalizability to new teachers. For the CLASS 5 data, this was achieved using leave-one-teacher-out cross validation. For the archival Partnership data, the mapping between teachers and data files was incomplete and so the mapping between schools and data files was used instead. This leave-one-school-out cross validation makes the assumption that a teacher did not transfer between schools during the study (a likely assumption) and in a sense is even more conservative than leave-one-teacher-out validation because it controls for similarities shared by teachers at the same school. Ideally the same cross validation technique would be used for both data sets, but for CLASS 5 data there aren't enough schools (2) and for the Partnership data the teacher identifiers are incomplete.

Models. Different models were used depending on the nature of the task and the class imbalance. For question-level authenticity prediction in the archival Partnership data, where classes are balanced, a J48 decision tree was used. J48 models were chosen because of their previous performance on this task and data set [18]. For utterance-level authenticity prediction in the new CLASS 5 data, where classes are highly imbalanced, SMOTEBoost was selected [5]. SMOTEBoost combines oversampling of the minority class by synthesizing new exemplars (SMOTE) with boosting, which builds a serial ensemble of models such that each successive model increases the weight, or focus, to instances misclassified in the previous model. SMOTEBoost applies SMOTE in each of these successive models in order to improve accuracy over the minority class, and evidence suggests it is one of the best all-purpose algorithms for imbalanced problems, though not necessarily the fastest [8]. Several other algorithms were evaluated on this task, including k-nearest neighbors, random forests, various cost-sensitive classifiers, and various ensembles, but SMOTEBoost had the best utterance-level performance. For class-level authenticity prediction (for both Partnership and CLASS 5 data), M5P model trees, which are decision trees with regression functions at the leaves [7], were used to predict the proportion of authentic questions in the class period. As a comparison to the class-level models, we aggregated over the question- and utterance-level classifications to calculate a proportion score at the class level.

3. RESULTS & DISCUSSION

3.1 Proportion models for imbalanced data

Our first comparison was between class-session level proportion models and aggregated utterance level classifications for the new CLASS 5 data where authenticity was very rare. A M5P model trained to predict the proportion of authentic questions per class made predictions that had a significant correlation with the actual proportions, $r(130) = .50,\ p < .001,\ CI_{95}[.36, .61]$. A SMOTEBoost

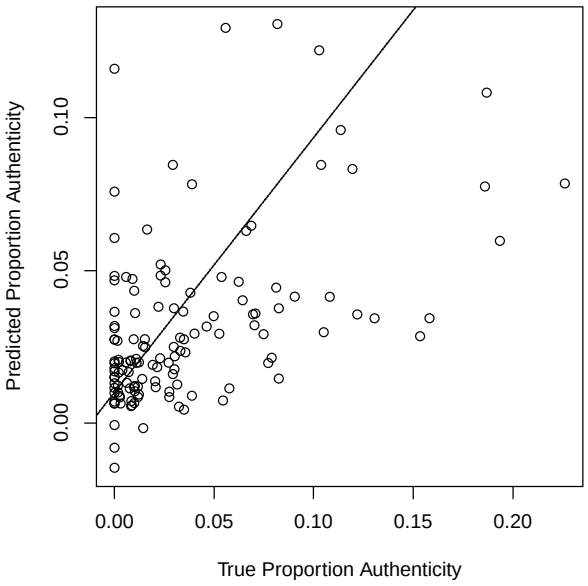


Figure 1: M5P session-level proportion predictions on the CLASS 5 data set.

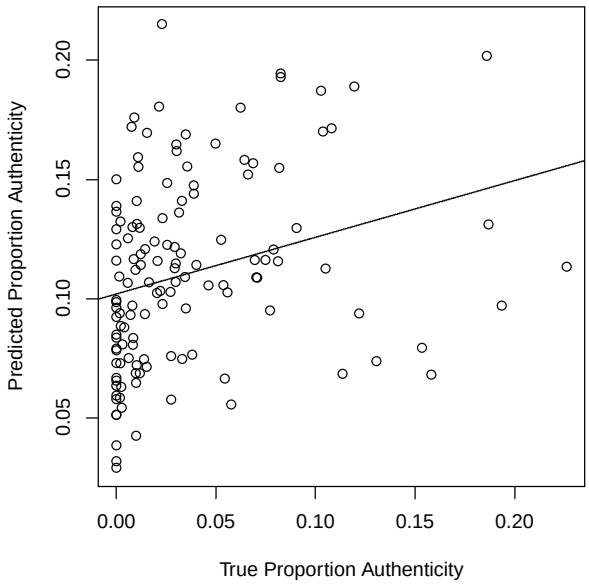


Figure 2: SMOTEBoost utterance-level predictions aggregated to session-level proportions on the CLASS 5 data set.

model trained to predict the authenticity of utterances and whose predictions were aggregated to obtain class-session level proportions made predictions that had a significant size correlation with actual proportions, $r(130) = .27,\ p = .001$, $CI_{95}[.11, .43]$. However, these two correlations were significantly different, $t(258) = 2.42,\ p = .017$. These results suggest that class-session level proportion predictions are more accurate than aggregating predictions from utterance level models.

Scatterplots of the actual vs. predicted proportion of authentic questions in the new CLASS 5 data are shown in Figures 1 and 2. Perhaps the major difference between these two scatterplots is the relationship between predicted and authentic proportions for values near zero. For the aggregated utterance-level predictions generated by SMOTEBoost, the scatterplot in Figure 2 shows a large vertical column of predictions above zero, indicating that for values near zero the classifier is overestimating the true occurrence of authentic questions. Conversely in Figure 1, predictions at zero are more tightly clustered.

Based on these results, it appears that session-level proportion models like M5P are more forgiving of the imbalanced classes than are utterance-level models like SMOTEBoost. There are two plausible explanations for why this might be. First, the session-level models are predicting a continuous number between 0 and 1 rather than making crisp binary judgments as in the case for the utterance-level models. Continuous predictions more closely match the model's internal probability, as opposed to a binary judgment where the binary prediction is the same irrespective of how far the model's internal probability is from the threshold, so long as it is on the same side of the threshold. Secondly, utterance-level models do not take advantage of the probability of a previous utterance's authenticity in determining the current utterance's authenticity, whereas the session-level models are accumulating all of this weak evidence before rending a proportion authenticity prediction. Based on this reasoning, an additional comparison of interest would be to take the utterance-level prediction probabilities and aggregate over them instead of the binary classifications. Unfortunately in the case of SMOTEBoost, these probabilities are within 10^{-6} of zero and one, so the results are no different than aggregating over class predictions.

3.2 Proportion model stability

To demonstrate model stability we undertook two comparisons. First, predictions of a M5P model for the Partnership data trained to predict the proportion of authentic questions per class session were significantly correlated with the actual proportions, $r(426) = .42,\ p < .001,\ CI_{95}[.34, .50]$. This correlation is remarkably similar to the 0.5 correlation obtained for the new CLASS 5 data. The similarity in correlations is particularly noteworthy given the differences between data sets: for CLASS 5, the classifier is operating over ASR transcribed utterances where authentic questions are 3% of the total data, but for the Partnership data, the classifier is operating over human transcribed instructional questions where authentic questions are 50% of the total data.

Secondly, a J48 model for the Partnership data trained to predict the authenticity of utterances and whose predictions were aggregated to class-session level proportions made predictions that were correlated with actual proportions, $r(426) = .44, p < .001,\ CI_{95}[.36, .51]$. These two correlations were not significantly different, $t(870) = .37,\ p = .71$. Scatterplots of the actual vs. predicted proportion authentic questions in the Partnership data in Figures 3 and 4 further

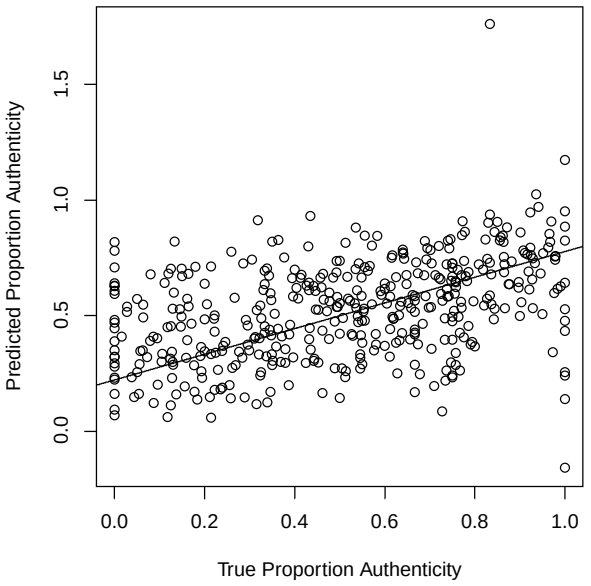


Figure 3: M5P session-level proportion predictions on the Partnership data set.

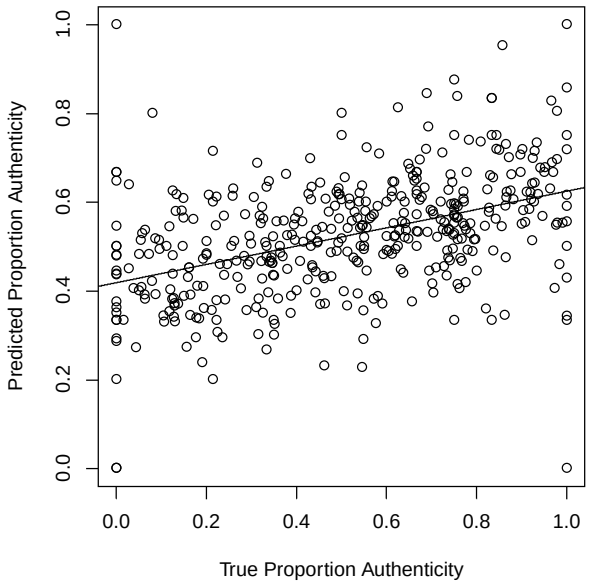


Figure 4: J48 utterance-level predictions aggregated to session-level proportions on the Partnership data set.

illustrate the similarities of these predictions. The equivalence between utterance- and session-level models for the Partnership data (shown in in Figures 3 and 4) and lack of equivalence between utterance- and session-level models for the new CLASS 5 data (shown in Figures 1 and 2) serves to further illustrate the enhancement to predictive stability that comes from using session-level models for this task. When the classes are relatively balanced, as in the case of the Partnership data, there is no difference between aggregating utterance-level predictions and session-level predictions. However, when the classes are imbalanced, as in the case of the new CLASS 5 data, the differences are significant and favor the session-level model.

4. DISCUSSION

We have presented and validated a method for assessing classroom instructional quality based on authentic questions that is effective even when such questions are rare. Our approach transforms the problem of utterance-level authentic question classification into the problem of session-level regression predicting the proportion of authentic questions. This problem transformation outperforms aggregating utterance-level classifications when classes are imbalanced, is stable for both low and high dialogic classrooms, and is stable across both automatic speech recognition and human transcripts. As such it is more appropriate for use in assessing classroom instructional quality across a wide range of dialogic discourse, complementing previous work that has investigated model generalization in different discourse communities [17]. Because question asking behavior of this type is a common component of the major classroom observation protocols in use today (e.g., those used in the MET study [11]), this research may potentially be used to help automate various protocols in addition to the target protocol here, CLASS.

Because many major classroom observation protocols call for judgments of quality approximately every 10 minutes, session-level proportion predictions are not too dissimilar from current practice. A useful point for future research would be to obtain data coded with these protocols in addition to the speech data we used, subdivide the data into 10 minute bins, and then calculate accuracy. On the other hand, the CLASS protocol is much more fine grained, and the current approach sacrifices the utterance-level resolution CLASS specifies for robustness. From a teacher professional development perspective, fine grained annotations are more useful because they can be replayed to the teacher to highlight particularly effective portions of the class. Our session-level approach in its present form appears to be less useful for professional development.

An avenue for future work would be to combine session-level and utterance-level models. For example, a session-level model could first be applied to the data, generating a session-level prediction variable, and then that variable could be used as a feature in an utterance-level model. Presumably this would be used by the model as an intercept to adjust the baseline probability of authenticity for all utterances in that session. Of course the session- and utterance-level processes could also be jointly modeled, e.g. using a hierarchical Bayesian approach.

Finally, we raise the question of why authentic questions were rarer in our new CLASS 5 data collected from 2014-2016 compared to the archival Partnership data collected from 2001-2003. The question is whether the low rate of authentic questions in our new sample is something that can reasonably be expected to reoccur, or whether it is the product of a relative small homogeneous sample. Indeed we find that some of the first studies with CLASS found levels

of authenticity between 10% and 30% [14], suggesting that the rate of authentic questions in our new sample is in the normal range. The fact that rates as low as 10% have been observed serve as a warning and challenge to future research. In our new CLASS 5 data, authenticity rates of 30% for instructional questions translated to 3% of utterances being authentic. Presumably a 10% authenticity rate for instructional questions would mean that only 1% of utterances are authentic.

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. We also thank Marci Glaus, Xiaoyi Sun, and Brooke Ward of the University of Wisconsin-Madison for their help with the collection and annotation of the classroom data.

6. REFERENCES

[1] American Institutes for Research. Databases on state teacher and principal evaluation policies, 2016.

[2] J. Archer, S. Cantrell, S. L. Holtzman, J. N. Joe, C. M. Tocci, and J. Wood. *Better Feedback for Better Teaching: A Practical Guide to Improving Classroom Observations*. Jossey-Bass, 2016.

[3] M. M. Bakhtin. *The dialogic imagination: Four essays*. University of Texas Press, 1981.

[4] S. Cantrell and T. J. Kane. Ensuring fair and reliable measures of effective teaching: Culminating findings from the MET project's three-year study. resreport, Bill & Melinda Gates Foundation, 2013.

[5] N. V. Chawla. Data mining for imbalanced datasets: An overview. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 853–867. Springer US, Boston, MA, 2005.

[6] 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*, ICMI '15, pages 557–566, New York, NY, USA, 2015. ACM.

[7] E. Frank, Y. Wang, S. Inglis, G. Holmes, and I. H. Witten. Using model trees for classification. *Machine Learning*, 32(1):63–76, 1998.

[8] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2012.

[9] A. Gamoran and S. Kelly. Tracking, instruction, and unequal literacy in secondary school English. In R. Dreeben and M. T. Hallinan, editors, *Stability and change in American education: Structure, process, and outcomes*, pages 109–126. Eliot Werner Publications Incorporated, Clinton Corners, NY, 2003.

[10] A. Gamoran and M. Nystrand. Background and instructional effects on achievement in eighth-grade English and social studies. *Journal of Research on Adolescence*, 1(3):277–300, 1991.

[11] T. J. Kane and D. O. Staiger. Gathering feedback for teaching: Combining high-quality observations with student surveys and achievement gains. resreport, Bill & Melinda Gates Foundation, 2012.

[12] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[13] M. Nystrand. *CLASS (Classroom language assessment system) 2.0: A Windows laptop computer system for the in-class analysis of classroom discourse.*, 1988.

[14] M. Nystrand, editor. *Opening Dialogue: Understanding the Dynamics of Language and Learning in the English Classroom*. Language and Literacy Series. Teachers College Press, New York, 1997.

[15] A. M. Olney, M. Louwerse, E. Mathews, J. Marineau, H. Hite-Mitchell, and A. C. Graesser. Utterance classification in AutoTutor. In *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*, pages 1–8, Philadelphia, 2003. Association for Computational Linguistics.

[16] S. G. Rivkin, E. A. Hanushek, and J. F. Kain. Teachers, schools, and academic achievement. *Econometrica*, 73(2):417–458, 2005.

[17] B. Samei, A. M. Olney, S. Kelly, M. Nystrand, S. D'Mello, N. Blanchard, and A. Graesser. Modeling classroom discourse: Do models that predict dialogic instruction properties generalize across populations? 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, , and M. Desmarais, editors, *Proceedings of the 8th International Conference on Educational Data Mining*, pages 444–447. International Educational Data Mining Society, 2015.

[18] B. Samei, A. M. Olney, S. Kelly, M. Nystrand, S. D'Mello, N. Blanchard, X. Sun, M. Glaus, and A. Graesser. Domain independent assessment of dialogic properties of classroom discourse. In J. Stamper, Z. Pardos, M. Mavrikis, and B. McLaren, editors, *Proceedings of the 7th International Conference on Educational Data Mining*, pages 233–236, 2014.

[19] M. Surdeanu, T. Hicks, and M. A. Valenzuela-Escarcega. Two practical rhetorical structure theory parsers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 1–5, Denver, Colorado, June 2015. Association for Computational Linguistics.

[20] Q. Yang and X. Wu. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(4):597–604, 2006.

When and who at risk? Call back at these critical points

Yuntao Li
Department of Machine Intelligence
Key Laboratory of Machine Perception(MOE)
Peking University
li-yuntao@qq.com

Chengzhen Fu
Department of Machine Intelligence
Key Laboratory of Machine Perception(MOE)
Peking University
wallaceapril@163.com

Yan Zhang
Department of Machine Intelligence
Key Laboratory of Machine Perception(MOE)
Peking University
zhy@cis.pku.edu.cn

ABSTRACT

Since MOOC is suffering high dropout rate, researchers try to explore the reasons and mitigate it. Focusing on this task, we employ a composite model to infer behaviors of learners in the coming weeks based on his/her history log of learning activities, including interaction with video lectures, participation in discussion forum, and performance of assignments, etc.

The prediction accuracy of our proposed model outperforms related methods. Besides, we try combining the model with suggested interventions, such as sending reminder emails to at-risk learners. Future work, which is currently underway, will evaluate its influence on mitigating dropout rate.

Keywords

MOOC; dropout; Stacked Sparse Autoencoder; RNN

1. INTRODUCTION

Recently, online education, for which landmark concept is MOOCs (Massive Open Online Courses), has become a new global craze, bringing several MOOC platforms including EdX, Coursera, and Udacity, etc. Due to the freedom of time and place learning at MOOCs, a large scale of learners has been benefit from this new form of online learning. A typical course of MOOC lasts for 6-12 weeks, with learners of diverse backgrounds and major field. Besides, MOOC learners may have different intentions and motivations, causing their extents and leave for various reasons.

Despite the increasing popularity of MOOCs, the extremely low rate of completion has been considered from the beginning. Drop-out is concerned as one of the most critical problem of MOOCs. Drop-out indicates situations that a student registers a course, watches course materials, or even attends the quizzes, but eventually quits without attending the final test. It has been researched that an average completion rate of MOOCs comes as low as 7 percent, ranging from 0.8 percent in Princeton's (History of the World since 1300), to 19.2 percent in the "Functional Programming Principles in Scala" course [7]. MOOC platforms are facing a concerning issue due to a high learners' dropout rate.

Thus, identifying at-risk learners by predicting their dropout probability thus becomes timely important, given that early prediction can help instructors provide proper support to those learners to retain their learning interests aiming at guaranteeing them a regular process of study without doing a crash job or even dropout. Addressing this task, we focus on predicting learners' state for the next consecutive two weeks. We particularly formulate this issue as a multi-classification problem, and develop a Stacked Sparse Autoencoder (SSAE)+Softmax model to solve it. Essentially, our model has several advantages. First, it incorporates multiple features based on characterizing learners' weekly engagements on the MOOC platform. Second, it discovers correlations between observed explanatory features. The new compressed feature representation transformed by SSAE performs better than the previous one, based on the input of classifiers. Third, the model considers both the current and previous states to estimate the next states, which makes it more flexible to model students' dynamics.

By training a model to identify at-risk students, we can apply this model on online MOOC platforms, enabling it to calculate students' at-risk-rate regularly and send emails to them automatically. Hopefully some of these at-risk students will continue their learning.

We make contributions in this paper as follows:

1. We employ different composite models that incorporate multiple features to infer behavior in the coming weeks based on weekly history of learning data. The model is an end-to-end neural network model, which means it can be trained as a whole. Our results indicate that model of SSAE+Softmax performs best and achieve higher AUC score consistently, which is superior to the baseline SVM model.

2. We try combining the model with suggested interventions such as sending reminder emails to at-risk learners. Though we do not conduct real experiments of sending emails, the paper proposes a preliminary framework of applying experimental results to determining to whom reminder emails should be sent and when to send.

3. We explore to what extent each single feature can influence dropout probability and try to cluster dropout learners by employing k-means clustering algorithm, proving that features extracted from course engagements are effective indicators of which class a low-performing learner belongs to separated by their pattern of behaviors. Future work will shade light into the relationships between behavior patterns of learners and reasons why they quit the course.

The rest of this paper is organized as follows. Section 2 describes the related work. Section 3 presents the description of the dataset and features derived from the dataset. Section 4 introduces our model in detail. Experimental results and discussion are presented in Section 5 and 6. Finally, Section 7 concludes our work in this paper.

2. RELATED WORKS

Mitigating MOOC dropout rate is essential for boosting the values of MOOCs, thus the mechanisms that can predict student dropout become increasingly important.

Some exploratory analysis suggests that student behavior in the discussion forum helps predict attrition. Yang et al. [6] present a foundation for research investigating the social factors that affect dropout along the way during participation in MOOCs. To operationalize these factors, they define metrics related to posting behavior (thread starter, post length, content length) and social positioning (posts & replies) within the resulting reply network. Similarly, some researchers (Ramesh et al. [8]) explore other aspects of discussion forum such as viewing posts, sentiment. This perspective provides a potentially valuable source of insight for design of MOOCs that may be more conducive to social engagement that promotes commitment and therefore lower attrition. It is restrictive in application because it mainly lowers attrition of learners who drops out mainly because of hard interpersonal connection foundation online.

Many researchers aim at modeling learning behaviors over duration of weeks. Their pursuit is to extract significant features by parsing the clickstream file where each line represents a web request. These effective features include lecture interaction features, forum interaction features, assignment features [1–4,11], which capture the activity level of learners.

In terms of applied models, Kloft et al. [5] explore support vector machines (SVM) to predict the state of learners in the later phases of a course. Balakrishnan et al. [2] quantize the feature space into a discrete number of observable states that are integral to a Discrete Single Stream HMM. Fei et al. [9] propose recurrent neural network (RNN) model with long short-term memory (LSTM) cells.

3. DATA SET AND FEATURE SET

3.1 Dataset

The learner activity log data came from a publicly held data mining competition called KDD CUP 2015. It includes 79186 learners, each of whom enrolled in at least one course of the whole set of 39 courses. In total, the clickstream data includes 8,157,277 log records and the longest lifetime of enrollment is 6 weeks. Most of the data is user activity log data and course structure data.

3.2 Feature set

As stated above, our goal is to estimate the probability that a student stops engaging with a course for the next two weeks, given her/his learning activities up to the current time step.

The dropout probabilities are closely related to learners' engagements to courses, which are mainly characterized by design of forum, lecture and assessment features. To express the time-varying behaviors of learners, we extract 17 typical features of each week t for each learner i, denoted as vector $x_i^{(i)} \in R^1 7$, as presented in Table 1. It can be noticed that, features we selected are vital but highly correlated with each other, and we will introduce a model to cancel this redundancy.

Feature	Description
f1-f3	Number of posts in discussions, videos watched, problems attempted in week t respectively
f4-f6	Total number of discussions made, videos watched, problems attempted by week t
f7-f9	Average number of discussions, videos, problems attempted per week by week t
f10-f12	Average number of discussions, videos, problems attempted per session in week t
f13	Sum of number of another activities (navigate, access, page close, wiki) in week t
f14	Total number of activities in week t
f15	Total number of active days in week t
f16	Total number of time consumption in week t
f17	Total numbers of sessions in week t

Table 1: List of features derived for week t

3.2.1 *Interactions with forums*

A MOOC forum provides a platform to facilitate the communication between learners and lecturers. The more actively the learners interact with their partners, the more a learner feels she/he belongs in the course learning and the more likely she/he is to complete the learning tasks. Some features, such as viewing a post, receiving a reply, following a thread and up-voting, are strong indicators of engagement and sense of community [6, 7].

3.2.2 *Interactions with lectures*

Because the lecture videos are the most important learning resource for the learning participants, the video playing should be investigated, as done by other researchers. Among these works, Kim et al. [1] explored some click actions when watching videos. These behaviors can be classified into six types: skipping, zooming, playing, replaying, pausing, and quitting.

3.2.3 *Interactions with assignments*

It is reasonable to hypothesize that an active and engaged student would monitor their assignment a few times every week because material is released and due on a weekly basis. When monitoring this week by week, we can roughly estimate how far up-to-date a student is with a course. It is acknowledged that if a learner falls behind too much, it is hard to catch up and thus determination to complete is lost [2].

Furthermore, we observe from the user activity log data whether the learners are active in session, as the data contain multiple records in quick succession. We define the elapsed time of two separate sessions as 45 minutes. If the gap between a learner's two consecutive operation is more than 45 minutes, we assume that the learner quit and logged in again.

Consequently, for current week t, we obtain a sequence of $(x_1^{(i)}, x_2^{(i)}, ..., x_t^{(i)})$ for each learner i across t weeks and the corresponding sequence of dropout labels $(y_1^{(i)}, y_2^{((i))}, ..., y_{t-1}^{(i)})$. If there are activities associated with student i in the coming week, the dropout label in week t is assigned as $y_t(i) = 0$, otherwise, $y_t(i) = 1$. Notably, all features should be centered and normalized to unit standard deviation (mean of 0 and variance of 1).

4. OUR MODEL

4.1 Feature Extractor: Stacked Sparse Autoencoder (SSAE)

Now suppose that we have extracted weekly features from user activity log record, we employ a model named Stacked Sparse Autoencoder (SSAE) to discover high level representation of input features and correlations among them. In this part, we aim to produce a better feature representation that can show patterns of behavior for learners.

Autoencoder neural networks are a serial of models which can re-represent features by encoding them into a high level representation using a set of parameters and decode it back to its original values using another set of parameters. A sparse autoencoder neural network consists of an input layer, a hidden layer and an output layer, whose size of hidden layer is greater than its input layer. The network structure is presented in Figure 1.

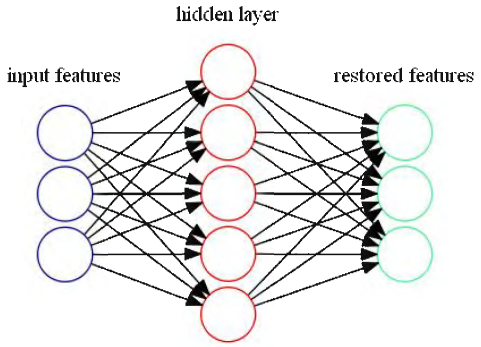


Figure 1: Network Structure

Formally, let the vector of input layer be the features of learner i extracted from weekly history of learning behavior features. We train the network to minimize the divergence between the input layer and the output layer, i.e., $h_{W,b}(x_t^{(i)}) = x_t^{(i)}$. After the model goes into convergence, which means it achieves a minimal difference between input features and output values, the hidden layer learns a new representation of the input. The numbers and dimensions of the hidden layer controls the complexity of the network and requires parameter values tuning to determine its optimal value. Notably, the new features that the hidden layer represents will be as the input of a classifier. To train this autoencoder network, we apply back-propagation algorithm to minimize overall cost function as follows:

$$J_{\text{sparse}}(W, b) = J(W, b) + \beta \sum W \cdot W$$

Where J (W, b) is calculated by two parts: an average sum-of-squares error and penalty term that helps prevent over fitting. $\sum W \cdot W$ means a sum of every element in matrix which is the element wised multiple of W. β represents weight of the sparsity penalty term.

Here we do not introduce the details; computational details can be found in [10].

In order to generate more general (higher-level-presented) features, we use a method called stacked to enrich capacity of our model. We train an autoencoder first and use its features as the input and output of another autoencoder. Thus we get a more abstract representation of original features which can be more suitable for describing learners' inner condition.

Compared with other methods like PCA, the neural network based SSAE is more strong. For most cases, relations between meta features are complex and can not be represented by simple functions like linear functions, thus traditional methods are not able to separate them well. However, neural networks have the ability to fit any function as long as it is given enough capacity(e.g. enough depth of layers of amount of cells), which ensures it to project meta features in an independent orthogonal linear space.

4.2 Sequenced feature combiner: RNN

A RNN (or Recurrent Neural Network) is a class of artificial neural networks dealing with sequence data. It takes sequenced data step by step, and generates an output according to all previous inputs on every step. A basic RNN with one hidden layer is shown in Figure 2.

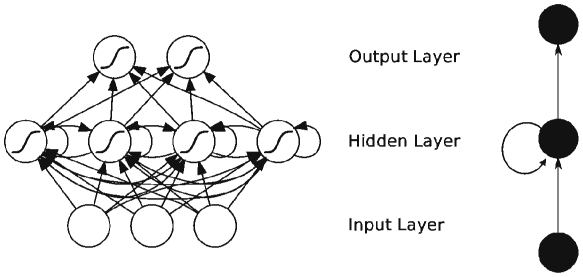


Figure 2: Basic RNN Structure

Formally, RNN is a function , where h is the hidden status (memory) of hidden units, and D is the size of input vector and L is the size of the output vector. The memory h changes every time while giving new inputs at each step.

The input vector of RNN is the high level representation generated by SSAE introduced in part 2. We aim to get a

good feature representation, which can contain all learners' event histories within a fixed-length vector, to make prediction and classify dropout learners by his/her reason.

For a simple RNN, it has parameters $(W_h, U_h, W_y, b_h, b_y)$, where W_h controls what to absorb to memory from input features, and U_t determines what to remember and what to forget from the last memory status, and W_y sets the output value, and b_h and b_y are biases who make a global offset to both hidden status and output value.

The computational formula of this kind of RNN is shown below:

$$h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h)$$
$$y_t = \sigma_y(W_y h_t + b_y)$$

where x_t and y_t represents input features and output vector at time t, and h_t is the memory hold by RNN. Here, σ_h and σ_y can be the same or different activation functions. Typical choices of activation functions are the sigmoid function and tanh function. Particularly, we choose tanh as activation function for both of the formulas. We will apply tanh in this paper as it typically yields to faster training (and sometimes also to better local minima). The operation tanh is calculated as follows:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

We do not apply an LSTM used by other researchers [8] because of some reasons. An LSTM is a special kind of RNN who has the ability of forgetting, which means it can determine what to remember and when to forget its memory while getting new inputs, however, a simple RNN can only remember all its inputs. We think that, for a sequence no longer than six, forgetting should not be accepted. Besides, simple RNN requires less calculated quantities which makes it more suitable for a large scale online service.

4.3 Classifier

4.3.1 Support Vector Machine(SVM)

Some prior work mentioned in the related work inspires us to employ SVM to predict the learning state in the next consecutive two weeks. The SVM computes an affine-linear prediction function based on maximizing the margin of positive and negative examples:

$$(w, b) := argmin_{w,b} \frac{1}{2}||w||^2 + C\Sigma_{i=1}^{n} max(0, 1 - y_i(< w, x > + b))$$

After extracting features, we try to predict by using SVM and compare with results from Softmax. As there is distinct difference between dropout users and non-dropout users, therefore, we use the method of random sampling to confine the amount of these users into a comparatively small one. With this done, the model we gain will not cause overfitting to either classification.

With learning feature of current week obtained in 'Feature set' Section as input, we apply SVM to predict whether to drop out at the end of this week. Three Kernel Functions: linear, rbf and mlp are tried, and the prediction accuracy is estimated via 5-folds cross validation.

4.3.2 Softmax Regression

In the softmax regression setting, we are interested in multiclass classification (as opposed to only binary classification). It is expected to classify learners into three cases, which can be represented as $\{(0,0),(0,1),(1,1)\}$, where 1 means dropout, and the first number depends on whether to drop out after one week, the latter indicates results after two weeks. In this case, the label set can take on 3 different values, letting the predicted outcome for i-th learner $\in \{1,2,3\}$.

We aim to estimate the probability of the class label taking on each of the 3 different possible values of each learner. Thus, our hypothesis will output a 3-dimensional vector (whose elements sum to 1) giving us our estimated 3 probabilities. Concretely, our hypothesis takes the form:

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)}=1|x^{(i)};\theta) \\ p(y^{(i)}=2|x^{(i)};\theta) \\ p(y^{(i)}=3|x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{3} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ e^{\theta_3^T x^{(i)}} \end{bmatrix}$$

Where $\theta_1, \theta_2, \theta_3 \in R^n$ represent model parameters of softmax, and $\sum_{j=1}^{3} e^{\theta_j^T x^{(i)}}$ generalizes the probability distribution, leading to the sum of all the probability is 1.

5. EXPERIMENTS

5.1 AUC Score

We can observe from the KDD cup's label set that the labels are displayed with 79% positives and 21% negatives. Due to class imbalance phenomenon, accuracy is not a good metric. Instead, Area under receiver operating characteristic curve (ROC AUC) is the main metric we use to do parameter tuning and model selection. Furthermore, AUC measures how likely a classifier can correctly discriminate between positive and negative samples.

	Week 1	Week 2	Week 3	Week 4	Week5
SSAE+ Softmax	0.924	0.895	0.887	0.803	0.754
SSAE+ SVM	0.894	0.867	0.849	0.784	0.729
SVM	0.831	0.826	0.817	0.749	0.698

Table 2: AUC comparison of SSAE+Softmax, SSAE+SVM, SVM

Table 2 presents the average AUC scores across weeks by applying two different classifiers (Softmax, SVM). The results indicate that the models that employ SSAE to discover correlations among initial features extracted from dataset, such as SSAE + Softmax, SSAE + SVM, are more competitive. They are superior to the baseline SVM model and achieve higher AUC score consistently. For instance, for the first week, the AUC score of SSAE+SVM is 0.894, which is 7.58% improvement relative to that of SVM.

Specifically, we can observe that our proposed model SSAE + Softmax outperforms the other models across different weeks. The observation implies that Softmax performs consistently better than SVM in terms of classifying a learner's previous states and predicting whether he will drop out.

More notably, the AUC score decreases with increasing lifetime of the course. We infer that there might be more un-

certainties related with dropout behavior that our model could not discover only from weekly history records. External forces such as lack of free time may result in more complex patterns of behavior. For instance, a learner may leave suddenly at week 4, while all statistical features of the previous three weeks strongly indicate he is not inclined to drop out.

5.2 Confusion Matrix

In this two class classification problem, the confusion matrix is a matrix with 4 entries, true positive(TP), false negative(FN), false positive(FP), and true negative(TN).

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = TruePositiveRate = \frac{TP}{TP + FN}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

The comparisons of metric mentioned above are presented in Table 3. Model of SSAE+Softmax outperforms the other models consistently, proving good implement of the prediction task. It is convincing that the results across weeks lay a foundation to identify patterns of behavior and suggest interventions for inactive learners.

Model	Precision	Recall	F1 score
SSAE+Softmax	0.891	0.942	0.916
SSAE+SVM	0.873	0.907	0.890
SVM	0.854	0.887	0.870

Table 3: Performance comparison of SSAE+Softmax, SSAE+SVM, SVM

6. DISCUSSION

Experimental results of a real-world dataset demonstrate that dropout probability is consistently predicable across weeks for different students. The next step in applying the newly proposed model (SSAE+Softmax) to MOOC platforms aims to mitigate dropout rate by suggesting interventions, such as sending reminder emails, with the goal of informing at-risk learners to retain interests.

Email is a very cheap medium to reach learners and create awareness quickly. Our proposed model will contribute to determining to whom an email should be sent and when to send. Identifying at-risk learners precisely avoids bombarding active learners with unnecessary emails and at the same time informs them in time to call back as many of them as possible.

Here we only present a preliminary framework for sending reminder emails. Specifically, at the end of week t, first, we extract weekly feature vectors for t weeks and employ SSAE+Softmax to predict future states y_t and y_{t+1}. Then, we determine a candidate set of potential at-risk learners who satisfy y_t=1 and $y_{t+1} = 1$ where y_t means status of the next week. Finally, we observe her/his behavior in the coming week $t + 1$ for every selected learner. If the 'at risk' state is confirmed ($y_t = 1$), the platform will send reminder emails at the end of week $t + 1$ immediately.

Although the experiments presented in this paper are limited to KDD Cup, we plan to augment our model and evaluate the effectiveness of sending reminder emails in a real MOOC platform established by our university. Future work applying this model is currently underway and the idea for sending emails will be improved step by step.

With features observed as stated in Section 3, we finish the analysis of predicting dropout based on model mentioned in Section 4. After gauging the goodness of model performance, it is persuadable that we have the ability of predicting and diagnosing dropout. In the following part, we analyze how each feature could influence final dropout probability by conducting sensitivity analysis, and try to cluster dropout learners to figure out their patterns of behavior by applying k-means algorithm.

In order to make data comparable, we separate user events by different courses and take the course with the most students (which is also the one with the most accomplished students) as our studying example. First, we try to find out standard learner behaviors of those who accomplish the course with a good quality. We simply take all non-dropout students' event logs and take an average on each of the features, and regard this as a medial requirement for finish this course. Next, we change each of the features step by step and make prediction using our neural networks with fixed parameters, and then we get three outputs representing probabilities of dropout in one or two weeks, or not dropout. We evaluate a score ranging from 0 to 1 to evaluate quality of these features.

Algorithm 1 Univariate analysis of feature_i

procedure UNIVARIATEANALYSIS($model, input\_features$)
  $standard \leftarrow average(input\_feature)$
  for $rate \in (0.5 \ldots 1.5)$ **do**
    $features \leftarrow standard$
    $features_i \leftarrow rate \times feature_i$;
    EvaluateDropoutRate(model, features);
  end for
end procedure

In Algorithm 1, "input-features" are features of those complete the courses, and "model" is the model we introduced above using SSAE, RNN and Softmax to predict a dropout rate, which is regraded as a score ranging from 0 to 1.

Notably, these features representing learning behaviors are classified into two categories: those related to course materials directly (e.g., watching videos, browsing wiki) and those not (e.g., navigate, page_close). We test some features to show how they influence a learner's dropout probability, as presented in Figure 3.

When times of watching video is 60 percent the amount of the standard statistic, the dropout probability increases sharply from 0.12 to 0.875. In this case, the dropout probability for feature page_close increases from 0.52 to 0.774, less significantly. It implies that, metrics closely related to course materials matter more than the others. Compared to indirect activities, times of direct engagements with course

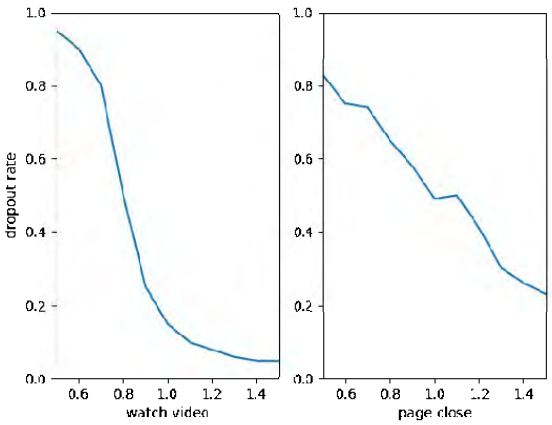


Figure 3: Sensitivity Analyses

materials are highly relevant to probability of accomplishing the course.

We then try to cluster dropout learners by employing k-means clustering algorithm, in which we set $k = 10$. Features extracted in Section 3 are effective indicators of which pattern of behavior a low-performing learner belongs to. We map any feature vector to one of the 10 clusters. There are two clusters whose number of low-performing learners are apparently larger than the others.

Inactive learners belonging to one cluster mentioned above preform worse with increasing lifetime of engagements. By monitoring their learning behavior in terms of lecture video, discussion and assignments, we find the numbers decrease week by week significantly. It can be inferred that they are putting less and less effort into learning as the course continues, which is a great indicator of failing to keep up with the pace of the course.

Inactive learners belonging to another cluster display a complex pattern of behavior. For instance, they leave the course for one or two weeks and then come back to learn. At the beginning, these learners display a high level of perseverance and self-discipline. Almost all the statistics demonstrate that they have regular patterns of studying, which can be confirmed by low dropout probability computed by our model. However, they behaved poorly in the coming weeks. Specifically, for some learners, the number of video watched, discussion made, and problems attempted all reach 0 suddenly. After some weeks, these learners come back to learn. Meanwhile, all learning data reaches the highest in comparison with previous weeks. Finally, they don't take exams and drop out. It may be inferred that such learners are "trying but not succeeding", due to the limit of time allowance (maybe other external forces).

In the future, to extend our model, we will send those learners predicted to leave the course a survey to find out why they are disengaging. We will shade light into the relationships between behavior patterns of learners and reasons why they quit the course.

7. CONCLUSIONS

In this paper, we propose different composite models that incorporate multiple features to infer behavior for the next two weeks based on features extracted from weekly history of learning data. The SSAE+Softmax model achieves a higher AUC score consistently, being superior to the baseline SVM model. Besides, application of the model including an automated email reminder system is under construction.

8. ACKNOWLEDGEMENT

This work is supported by NSFC under Grant No. 61532001 and 61370054, and MOE-RCOE under Grant No. 2016ZD201.

9. REFERENCES

[1] Kim, J., Guo, P. J., Seaton, D. T., Mitros, P., Gajos, K. Z., & Miller, R. C. (2014, March). Understanding in-video dropouts and interaction peaks inonline lecture videos. In Proceedings of the first ACM conference on Learning@ scale conference (pp. 31-40). ACM.

[2] Balakrishnan, G., & Coetzee, D. (2013). Predicting student retention in massive open online courses using hidden markov models. Electrical Engineering and Computer Sciences University of California at Berkeley.

[3] Halawa S, Greene D, & Mitchell J. Dropout prediction in MOOCs using learner activity features[J]. Experiences and best practices in and around MOOCs, 2014, 7.

[4] He, J., Bailey, J., Rubinstein, B. I., & Zhang, R. (2015, January). Identifying At-Risk Students in Massive Open Online Courses. In AAAI (pp. 1749-1755).

[5] Kloft, M., Stiehler, F., Zheng, Z., & Pinkwart, N. (2014, October). Predicting MOOC dropout over weeks using machine learning methods. In Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs (pp. 60-65).

[6] Yang, D., Sinha, T., Adamson, D., & RosÃl', C. P. (2013, December). Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In Proceedings of the 2013 NIPS Data-driven education workshop (Vol. 11, p. 14).

[7] Rachelle Peterson. 2013. Why Do Students Drop Out of MOOCs? Article. (13 November 2013.). https://www.nas.org/articles/why_do_students_drop_out_of_moocs

[8] Ramesh, A., Goldwasser, D., Huang, B., DaumÃl' III, H., & Getoor, L. (2014, July). Learning latent engagement patterns of students in online courses. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (pp. 1272-1278). AAAI Press.

[9] Fei, M., Yeung, & D. Y. (2015, November). Temporal Models for Predicting Student Dropout in Massive Open Online Courses. In 2015 IEEE International Conference on Data Mining Workshop (ICDMW) (pp. 256-263). IEEE.

[10] Ng, A. (2011). Sparse autoencoder. CS294A Lecture notes, 72, 1-19.

[11] Henrie, C. R., Halverson, L. R., & Graham, C. R. (2015). Measuring student engagement in technology-mediated learning. Elsevier Science Ltd.

Characterizing Collaboration in the Pair Program Tracing and Debugging Eye-Tracking Experiment: A Preliminary Analysis

Maureen M. Villamor
Ateneo de Davao University, Quezon City Philippines
University of Southeastern Philippines, Davao City, Philippines
maui@usep.edu.ph

Ma. Mercedes T. Rodrigo
Ateneo de Davao University
Quezon City
mrodrigo@ateneo.edu

ABSTRACT
This paper characterized the extent of collaboration of pairs of novice programmers as they traced and debugged fragments of code using cross-recurrence quantification analysis (CRQA). This was a preliminary analysis that specifically aimed to compare and assess the collaboration of pairs consisting of two individuals who may have different or same level of prior knowledge given a task. We performed a CRQA to build cross-recurrence plots using eye tracking data and computed for the CRQA metrics, such as recurrence rate (RR), determinism (DET), average diagonal length (L), longest diagonal length (LMAX), entropy (ENTR), and laminarity (LAM) using the CRP toolbox for MATLAB. Results showed that low prior knowledge pairs (BL) collaborated better compared to high prior knowledge pairs (BH) and mixed prior knowledge (M) pairs because of its high RR and DET implying that they had more recurrent fixations and matching scanpaths. However, the BL pairs' high ENTR and LAM could mean that they seemed to have more difficulty in understanding and debugging the programs. All pairs regardless of category had more or less exerted the same level of attunement when asked to debug the programs as evident in their L values. The mixed pairs seemed to have struggled with eye coordination the most as it had the most incidences of low LMAX.

Keywords
Eye-tracking, Collaboration, Cross-recurrence quantification

1. INTRODUCTION
Eye gaze plays an essential role in social interaction processes. In Computer-Supported Collaborative Learning (CSCL), eye-tracking had been used in previous works to study joint attention in collaborative learning situations [9][16]. Two eye-trackers, for instance, can be synchronized for studying the gaze of two persons collaborating in order to solve a problem and for understanding how gaze and speech are coupled [11-13].

The use of gaze coupling was first proposed in [11] to study conversation coordination. In this study, they defined gaze coupling as episodes when participants are looking at the same target. Their results showed that the coupling of eye gaze between collaborating partners may be an indicator of quality interaction and better comprehension. In the domain of pair programming, Pietinen et al. [10] suggested that gaze closeness could reflect tightness of collaboration. More prior studies [1][11-13] have shown that the coupling of eye gaze between collaborating partners may be an indicator of quality interaction and better comprehension and that joint attention, and more generally, synchronization between individuals is essential for an effective collaboration.

Cross-recurrence quantification analysis or CRQA, introduced in [18], is an extension of Recurrence Quantification Analysis (RQA) [7] that is used to quantify how frequently two systems exhibit similar patterns of change or movement in time. It takes two different trajectories of the same information as input and tests between all points of the first trajectory with all points of the second trajectory forming a cross-recurrence plot (CRP). The CRP permits visualization and quantification of recurrent state patterns between two time series. Analysis using CRP's has been proposed as a generalized method to unveil the interlocking of two interacting people [2]. It has been used to analyze the coordination of gaze patterns between individuals and has been used to determine how closely two collaborators' gaze follow each other. In the scientific literature, a cross-recurrence gaze plot is considered as the standard way of representing social eye-tracking data [16].

CRQA was used in [11], which provided the first quantification of gaze coordination in their monologue data to analyze the relation between eye movements of the speaker and the listener. The analysis revealed that the coupling between speaker and listener eye-movements predicted how well the listener understood what was said. They extended their findings in their succeeding studies [12-13] and results revealed that eye movement coupling found in monologue indeed extends to dialogues.

In the context of pair programming, Jermann et al. [5] used synchronized eye-trackers to assess how programmers collaboratively worked on a segment of code, and they also contrasted a "good" and a "bad" pair using cross-recurrence plots. Results showed that high gaze recurrence seems to be typical of a "good" pair where the flow of interaction is smooth and where partners sustain each other's understanding. A dual eye-tracking study was also conducted that demonstrated the effect of sharing selection among collaborators in a remote pair-programming scenario [4]. They used gaze cross-recurrence analysis to measure the coupling of the programmers' focus of attention. Their

findings showed that pairs who used text selection to perform collaborative references have high levels of gaze cross-recurrence.

This paper aimed to use CRQA to characterize collaboration of pairs of novice programmers in the act of tracing fragments of code and debugging. Specifically, this was a preliminary study that attempted to answer the following research question: Using CRQA, what characterizes collaboration of pairs consisting of (a) both high prior knowledge students, (b) both low prior knowledge students, and (c) high- and low-prior knowledge students?

Although the use of CRQA as an approach to assess collaboration between participants in a pair programming eye-tracking experiment is not an entirely novel approach, the main contribution of this study was the inclusion of the composition of the pairs in terms of expertise levels. Previous studies did not characterize the pairs based on prior knowledge in programming or level of expertise.

2. METHODS

2.1 Participants

The study was conducted in two private universities in the Philippines. Students who had taken the college-level fundamental programming course were recruited to participate in this study. Since the study is not finished yet and is still on-going, we recruited only 16 pairs of participants as of writing of this paper.

2.2 Structure of the Study

A screening questionnaire was distributed to student volunteers, to determine their eligibility to take part in this study (e.g. no cataracts, no implants, etc.), and they were required to undergo an eye-tracking calibration test. Participants who passed both screenings were given consent letters to fill up and sign. They were then asked to take a written program comprehension test (20 minutes) to determine their level of programming knowledge and skills. The actual eye-tracking experiment followed which was designed for 60 minutes at the maximum. Two Gazepoint eye-trackers were used to collect the pairs' eye-tracking data. The pairs were shown 12 programs with known bugs and were asked to mark the location of the bugs with an oval. There was no need to correct the errors.

A slide sorter program with "Previous", "Reset", "Finish" and "Next" buttons was created to display the program specifications followed by the buggy programs. The participants were free to click any of the buttons as they liked and were free to navigate the slides. No scrolling was needed. When the participant finds a bug, he/she clicks on the location of the bug and the software then draws an oval to mark it. Figure 1 is an excerpt from a specific slide in the slide sorter program showing the ovals.

The pairs were told to work with their partner on the problems and should collaborate using a chat program. All communications with their partner was via chat. The participants were seated together in the same room but were spaced far enough to ensure that all communication with their partners was via chat only. After the actual eye-tracking experiment, the pairs were asked to fill up a post-test questionnaire privately to assess how well they knew each other, how well they thought they collaborated, and how they felt about their partner. This study limits its analysis to the results of the programming comprehension test and the eye gaze data.

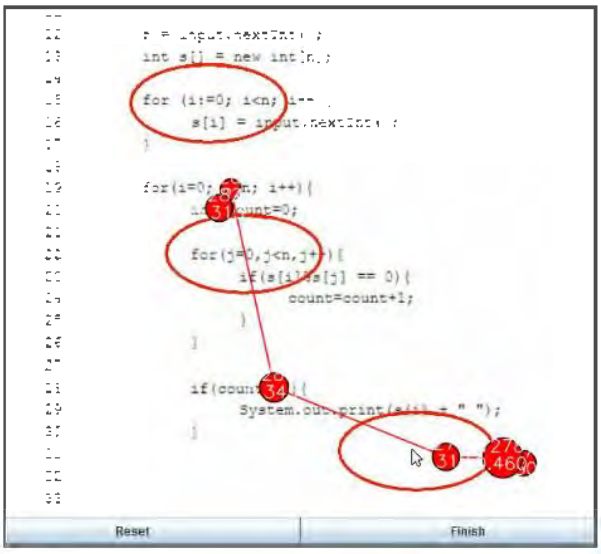


Figure 1. An excerpt from the slide sorter program showing the ovals after marking.

2.3 Constructing a Cross-Recurrence Plot

To conduct a cross-recurrence analysis, an N x N matrix called cross-recurrence plot is built, which is essentially a representation of the time coupling between two time series. The horizontal axis represents time for the first collaborator (C1) and the vertical axis represents time for the second collaborator (C2). Given two fixation sequences of the collaborators, f_i and g_i, i – 1… N, we define the cross- recurrence as r_{ij} = 1 if d (f_i, g_j) ≤ ρ, and 0, otherwise [7].

Recurrence occurs when two fixations from different sequences land within a given radius ρ of each other, where d is some distance metric (e.g., Euclidean distance). Cross-recurrence points are represented as a black point (pixel) in the plot (see Figure 2). For a pixel to be colored, the distance between the fixations of the two collaborators has to be lower than a given threshold. If two collaborators uninterruptedly looked at two different spots on the screen for the entire interaction, the resulting CRP would be completely blank (white space in Figure 2). On the contrary, if the two collaborators looked at the same spot on the screen continuously, the plot would show only a dark line on the diagonal. Points exactly on the diagonal of the plot correspond to synchronous recurrence, such as, collaborators look at the same target at exactly the same time. Points above the diagonal correspond to fixations of *C2* that happen after *C1* has fixated the element. Points below the diagonal correspond to *C2*'s gaze leading *C1*'s. Asymmetries above and below the diagonal line could therefore be indicative of leading and following behaviors.

2.4 CRQA Metrics

CRQA defines several measures that can be assessed along the diagonal and vertical dimensions. For the diagonal dimension, we have: recurrence rate, determinism, average and longest length of diagonal structures, entropy, and diagonal recurrence profile. For the vertical dimension, we have: laminarity and trapping time. The definitions that follow are taken from [7].

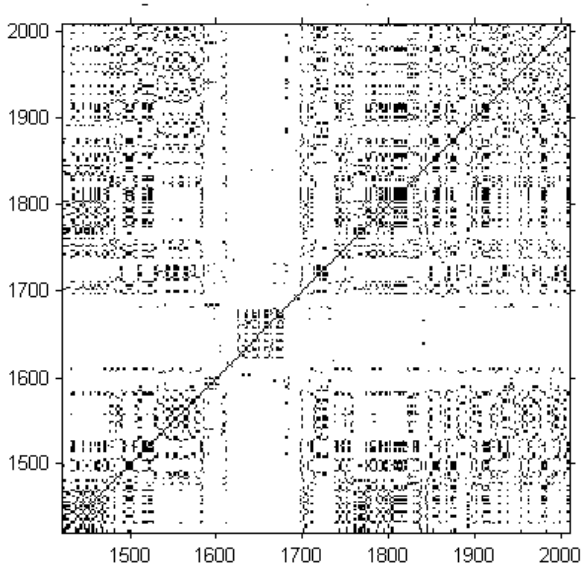


Figure 2. Example of a Cross-Recurrence Plot

Cross-Recurrence Rate (RR) represents the "raw" amount of similarities between the trajectories of two systems, which refers to the degree to which they tend to visit similar state. In eye-tracking data, this represents the percentage of cross-recurrent fixations. The more closely coupled the two systems are, in terms of sharing the same paths, the more recurrences will be formed along the diagonal lines. Hence, a high density of recurrence points in a diagonal results in a high value of RR.

Determinism (DET) is the proportion of recurrence points forming long diagonal structures of all recurrence points. Relative to eye-tracking data, this refers to the percentage of identical scanpath segments of a given minimal length in the two scanpaths.

The average diagonal length (L) reports the duration that both systems stay attuned. High coincidences of both systems increase the length of these diagonals. High values of DET and L represent a long time span of the occurrence of similar dynamics in both trajectories.

The longest diagonal length (LMAX) on a recurrence plot denotes the longest uninterrupted period of time that both systems are in concurrence, which can be seen as an indicator of stability of the coordination.

Entropy (ENTR) measures the complexity of the attunement between systems. In eye-tracking, this represents the complexity of the relation between scanpaths of the two eye-movement data. ENTR is low if the diagonal lines tend to all have the same length, signifying that the attunement is regular; otherwise, ENTR is high if the attunement is complex.

Using the diagonal recurrence profile (DiagProfile) offers the possibility of observing the direction of the coordination, that is, if there is an asymmetry with one interlocutor leading the other.

Vertical structures in a CRP quantify the tendency of the trajectories to stay in the same region. The laminarity (LAM) of the interaction refers to the percentage of recurrence points forming vertical lines, whereas trapping time (TT) represents the average time two trajectories stay in the same region.

2.4 Data Preparation and Measures

Results of the written program comprehension test, post-test and the number of bugs identified were recorded. The program comprehension results were used to categorize the students as having high or low prior knowledge. A student was considered to have high prior knowledge if his/her program comprehension score was equal to or greater than the median score. Otherwise, the student has low prior knowledge.

The fixation data was cleaned first by removing fixations less than 100 milliseconds [8]. The number of fixations per slide that contained the actual program were segregated and saved on separate files. Hence, each participant has at most 12 fixation files. Fixation alignment was performed in case of uneven number of fixations per program file. Fixation files with sequences less than 20 were discarded because it usually returned a NaN value when the CRQA was performed using the CRP toolbox for MATLAB [7].

Given 16 pairs and 12 programs, there should have been 16×12 = 192 cases, but we only had 179 cases for the analysis since some pairs did not finish all 12 programs and some fixations sequences were discarded. A cross-recurrence plot was then constructed for each pair for every program, and the cross-recurrence analysis was performed to get the RR, DET, LMAX, L, ENTR, and LAM.

The challenge of using CRQA is finding optimal parameters for *delay*, *embed*, and *radius* [7]. An optimal delay can be identified when mutual information drops and starts to level off. The embedding dimension can be determined using false nearest neighbors and checking when there is no information gain in adding more dimensions. For this experimental data, however, no further embedding was done [3]. With an embedding dimension of one, delay was also set equal to one since no points were time delayed [17]. For this experimental data, the radius, which is the threshold that determines if two fixation points are recurrent, was set to 5% of the maximal phase space diameter [15] to avoid subjective biases when looking at recurrent patterns.

3. RESULTS AND DISCUSSION

Of the 16 pairs, there were three (3) both high prior knowledge pairs, five (5) both low prior knowledge pairs, and eight (8) mixed prior knowledge pairs. The remainder of the text will refer to these categories as BH, BL, and M respectively. The CRQA metrics per program according to these relationships were averaged separately to get the aggregated CRQA metrics.

The aggregated results were examined to find differences among the categories, which entailed looking at incidences of high and low values of the CRQA metrics. A value was considered high if it was equal to or greater than the mean plus one standard deviation and low if it was equal to or less than the mean minus one standard deviation. Table 1 shows the descriptive values of all aggregated CRQA metrics per program. No further statistical measures were performed since there were not too many pairs to consider and this was only for hypothesis generation purposes.

Findings showed that the BH pairs only had incidences of low to average RR's and BL pairs only had incidences of average to high RR's. The M pairs had a mix of high, low, and average RR's. See Table 1 for high and low RR. Figure 3 shows the boxplots of RR in these categories.

Table 1. Descriptive values of the CRQA metric per program

CRQA Metric	Mean	SD	Min	Max	Low <=	High >=
RR	0.13	0.05	0.06	0.27	0.08	0.17
DET	0.42	0.09	0.25	0.67	0.33	0.51
L	3.50	1.31	1.94	7.12	2.19	4.81
LMAX	39.59	22.45	9.33	82.57	17.14	62.05
ENTR	0.76	0.20	0.44	1.34	0.57	0.96
LAM	0.50	0.13	0.26	0.78	0.38	0.64

This could possibly mean that the BL pairs collaborated better than BH and M pairs due to its incidences of higher recurrent fixations. However, it could also mean that the high RR's found in BL pairs was because of the BL pairs' greater number of fixation points, implying that the BL pairs had spent more time comprehending the program flow and finding the errors in the program. More time spent could have resulted to more chances of having more recurrent fixations. BH and M pairs exhibited the same degree of collaboration based on their comparable average RR's with M only slightly higher than BH. It can also be noted that the high RR's observed in all categories were all found in the middle programs, possibly indicating that the middle programs required more concentration compared to other programs.

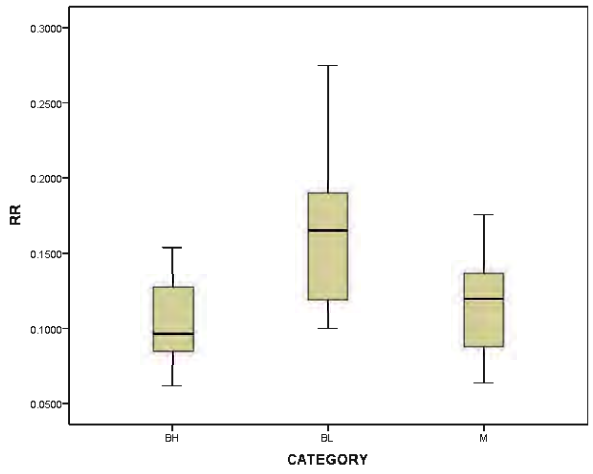


Figure 3. Boxplots of RR in All Catgories

The BL pairs only had average to high DET values, whereas BH and M pairs both only had low to average DET values. See Table 1 for high and low DET values. Figure 4 shows the boxplots of DET in all categories. The greater number of high DET values found in BL pairs could possibly mean that the BL pairs had shared more identical scanpaths compared to BH and M pairs. Also, since the BL pairs had more occurrences of high RR's and seemed to have spent longer durations in the task; this might have resulted to more matching scanpaths compared to BH and M pairs. As with RR, BH and M pairs' average DET were nearly the same, indicating the same degree of collaboration as assessed through their percentage of identical scanpaths.

Upon examination of their L values, results showed the BL pairs neither had high nor low L values. All but two of their L values were below the mean. The M pairs had few occurrences of high L values whereas BH pairs had one incidence each of high and low L values. Hence, a large majority of their L values were average. See Table 1 for high and low L values. Figure 5 shows the boxplots of the L values in all categories. These results implied that all of the pairs regardless of their expertise level or prior knowledge had more or less concentrated and exerted the same level of attunement on the given task. However, the M pairs possibly exhibited frequent longer durations where the pairs stay attuned compared to BH and BL pairs. BL pairs, on the other hand, had exhibited frequent shorter durations of attunement.

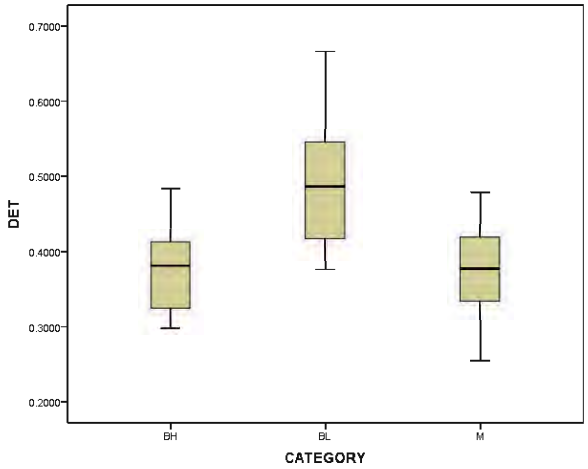


Figure 4. Boxplots of DET in All Catgories

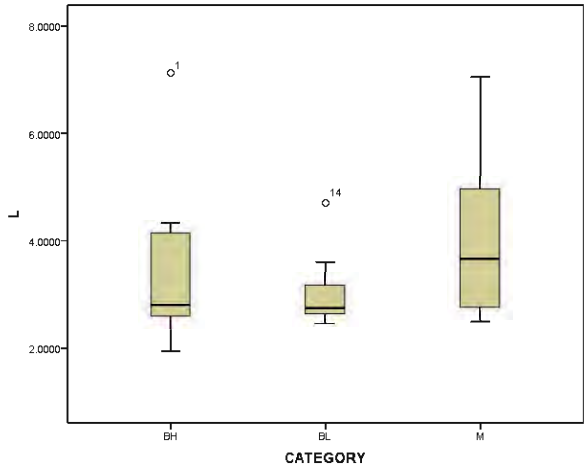


Figure 5. Boxplots of L in All Catgories

As for LMAX, BL pairs seemed to have exhibited better stability in terms of eye coordination particularly in the middle programs since they had more occurrences of high LMAX values. M pairs seemed to have struggled with eye coordination the most because of more incidences of low LMAX values. However, the average LMAX values of BH and M pairs were comparable, possibly indicating that the BH pairs' eye coordination stability was almost the same as M pairs. See Table 1 for high and low LMAX values. Figure 6 shows the boxplots of LMAX in all categories.

The same pattern in DET can also be observed in ENTR in terms of the incidences of high and low ENTR. The BL pairs had average to high ENTR values, whereas both BH and M pairs only had low to average ENTR, with M pairs having more low ENTR values than the BH pairs. See Table 1 for high and low ENTR values. Figure 7 shows the boxplots of ENTR in all categories. These findings imply that the BL pairs seemed to have more complex scanpaths in looking for bugs compared to BH and M pairs particularly in the middle programs. The BH pairs had the least complicated and, hence, more predictable scanpaths but their average ENTR was comparable to M pairs' average ENTR indicating that their scanpaths when looking for bugs were almost identical.

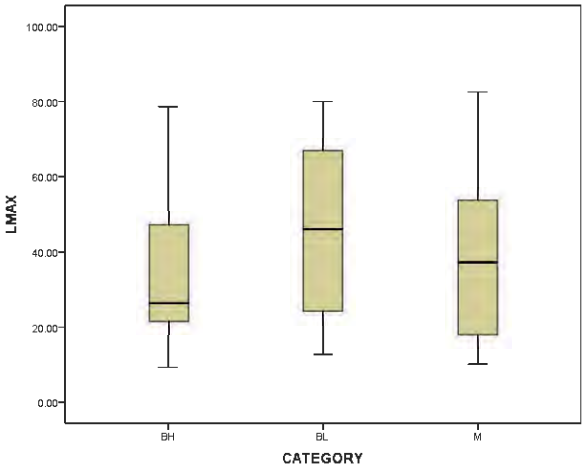


Figure 6. Boxplots of LMAX in All Catgories

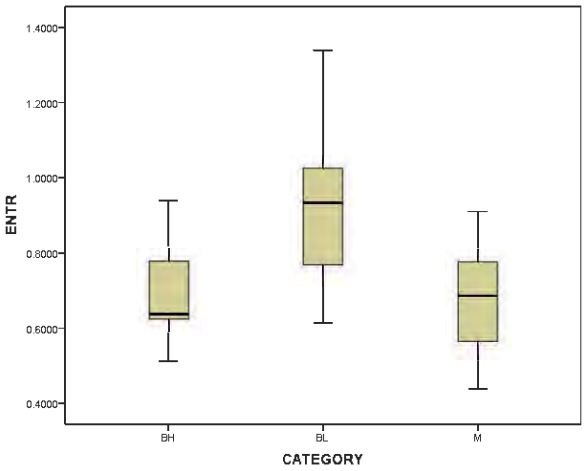


Figure 7. Boxplots of ENTR in All Catgories

As with DET and ENTR, the BL pairs only had average to high LAM values, whereas both BH and M pairs only had low to average LAM values. See Table 1 for high and low LAM values and Figure 8 for the boxplots. This could imply that the BL pairs seemed to have encountered more problems in understanding the program and, hence, tended to spend more time in certain regions of the code. BH and M pairs, on the other hand, seemed to have struggled less in understanding and debugging the programs. .

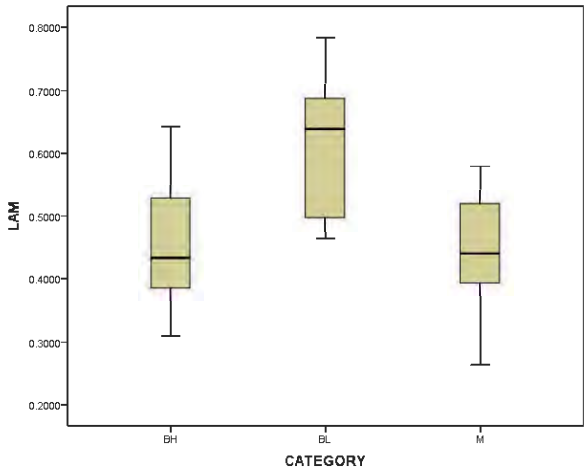


Figure 8. Boxplots of LAM in All Catgories

We also examined the number of slide switches between the program specification and the buggy program. We observed that the BL pairs had the least average number of slide switches among the pairs, but with the highest LAM values. This could mean that BL pairs tended to spend more time focusing on the actual program finding for bugs and switched less frequently between the program specification and the buggy program compared to other categories. BH and M pairs had higher frsequency of slide switches but with the lowest LAM values. BH and M pairs probably switched between slides more frequently because they just read the program specification to quickly check and recheck what the program does and were fast in terms of inspecting what was wrong in the actual program. BL pairs probably did not mind the program specification too much and just focused on the actual program locating bugs for the most part of the task.

Overall, it can be noted that for all the pairs, more evidences of collaboration and concentration happened in the middle part of the task. Perhaps, all the pairs perceived the middle programs the most difficult to debug.

4. SUMMARY AND CONCLUSION

The goal of this paper was to characterize the collaboration between pairs of novice programmers in the act of tracing and debugging a program in an attempt to understand the collaborative relationship of two individuals on a given task. Their collaboration was assessed through their CRQA metric results.

Findings showed that BL pairs are characterized with high RR, high DET, high ENTR and high LAM. Their high RR and DET signify that BL pairs are inclined to collaborate with their peers more compared to BH and M pairs. However, their high ENTR may signify complicated scanpaths in looking for bugs and their high LAM imply tendencies to stay in same regions of the code, which implies further that they frequently have difficulties in understanding and debugging the programs.

All pairs regardless of category tend to exhibit the same level of attunement in debugging as evident in their L values. The M pairs, however, are characterized as having more incidences of LMAX values, which could mean that they tend to struggle with

eye coordination the most. Overall, BH and M pairs are comparable in terms of collaboration as assessed through their CRQA results. We hypothesized, therefore, that the presence of a participant with high prior knowledge in M pairs may have contributed to the similarity between BH and M pairs

5. ACKNOWLEDGMENTS

The authors would like to thank Ateneo de Davao University and Ateneo de Naga University for allowing us to conduct the eye-tracking experiment. Many thanks also to Japheth Duane Samaco, Joanna Feliz Cortez, and Joshua Martinez for facilitating the data collection. We would like to thank also Dr. Norbert Marwan for giving us the permission to use the CRP toolbox for MATLAB. Lastly, thank you to Private Education Assistance Committee of the Fund for Assistance to Private Education for the grant entitled "Analysis of Novice Programmer Tracing and Debugging Skills using Eye Tracking Data."

6. REFERENCES

[1] Cherubini, M., Nüssli, M.A. and Dillenbourg, P., 2008, March. Deixis and gaze in collaborative work at a distance (over a shared map): a computational model to detect misunderstandings. In *Proceedings of the 2008 symposium on Eye tracking research & applications* (pp. 173-180). ACM.

[2] Dale, R., Warlaumont, A.S. and Richardson, D.C., 2011. Nominal cross recurrence as a generalized lag sequential analysis for behavioral streams. *International Journal of Bifurcation and Chaos*, *21*(04), pp.1153-1161.

[3] Iwanski, J.S. and Bradley, E., 1998. Recurrence plots of experimental data: To embed or not to embed?. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *8*(4), pp.861-871.

[4] Jermann, P. and Nüssli, M.A., 2012, February. Effects of sharing text selections on gaze cross-recurrence and interaction quality in a pair programming task. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work* (pp. 1125-1134). ACM.

[5] Jermann, P., Mullins, D., Nüssli, M.A. and Dillenbourg, P., 2011. Collaborative gaze footprints: Correlates of interaction quality. In *Connecting Computer-Supported Collaborative Learning to Policy and Practice: CSCL2011 Conference Proceedings.* (Vol. 1, No. EPFL-CONF-170043, pp. 184-191). International Society of the Learning Sciences.

[6] Jermann, P., Nüssli, M.A. and Li, W., 2010, September. Using dual eye-tracking to unveil coordination and expertise in collaborative Tetris. In *Proceedings of the 24th BCS Interaction Specialist Group Conference* (pp. 36-44). British Computer Society.

[7] Marwan, N. and Kurths, J., 2002. Nonlinear analysis of bivariate data with cross recurrence plots. *Physics Letters A*, *302*(5), pp.299-307.

[8] Matos, R., 2010. Designing eye tracking experiments to measure human behavior. *Merriënboer, JJG van, and Sweller, J.(2005). Cognitive load theory and complex learning: Recent developments and future directions. Educational Psychology Review*, *17*.

[9] Pietinen, S., Bednarik, R. and Tukiainen, M., 2009. An exploration of shared visual attention in collaborative programming. In *21st Annual Psychology of Programming Interest Group Conference, PPIG*.

[10] Pietinen, S., Bednarik, R., Glotova, T., Tenhunen, V. and Tukiainen, M., 2008, March. A method to study visual attention aspects of collaboration: eye-tracking pair programmers simultaneously. In *Proceedings of the 2008 symposium on Eye tracking research & applications* (pp. 39-42). ACM.

[11] Richardson, D.C. and Dale, R., 2005. Looking to understand: The coupling between speakers' and listeners' eye movements and its relationship to discourse comprehension. *Cognitive science*, *29*(6), pp.1045-1060.

[12] Richardson, D.C. and Dale, R., 2006. Grounding dialogue: eye movements reveal the coordination of attention during conversation and the effects of common ground. In *Proceedings of the 28th Annual Cognitive Science Society Conference*.

[13] Richardson, D.C., Dale, R. and Kirkham, N.Z., 2007. The art of conversation is coordination common ground and the coupling of eye movements during dialogue. *Psychological science*, *18*(5), pp.407-413.

[15] Schinkel, S., Dimigen, O. and Marwan, N., 2008. Selection of recurrence threshold for signal detection. *The European Physical Journal-Special Topics*, *164*(1), pp.45-53.

[16] Schneider, B., Abu-El-Haija, S., Reesman, J. and Pea, R., 2013, April. Toward collaboration sensing: applying network analysis techniques to collaborative eye-tracking data. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge* (pp. 107-111). ACM.

[17] Webber Jr, C.L. and Zbilut, J.P., 2005. Recurrence quantification analysis of nonlinear dynamical systems. *Tutorials in contemporary nonlinear methods for the behavioral sciences*, pp.26-94.

[18] Zbilut, J.P., Giuliani, A. and Webber, C.L., 1998. Detecting deterministic signals in exceptionally noisy environments using cross-recurrence quantification. *Physics Letters A*, *246*(1), pp.122-128.

Linking Language to Math Success in an On-Line Course

Scott Crossley
Georgia State University
Atlanta, GA 30303
scrossley@gsu.edu

Tiffany Barnes
Collin Lynch
North Carolina State University
Raleigh, NC 27606
tmbarnes, cflynch@ncsu.edu

Danielle S. McNamara
Arizona State University
Tempe, AZ, 85287
dsmcnama@asu.edu

ABSTRACT

This study takes a novel approach toward understanding success in a math course by examining the linguistic features and affect of students' language production within a blended (with both on-line and traditional face to face instruction) undergraduate course (n=158) on discrete mathematics. Three linear effects models were compared: (a) a baseline linear model including non-linguistic fixed effects, (b) a model including only linguistic factors, (c) a model including both linguistic and non-linguistic effects. The best model (c) explained 16% of the variance of final course scores, revealing significant effects for one non-linguistic feature (days on the system) and two linguistic features (*Number of dependents per prepositional object nominal* and *Sentence linking connectives*). One non-linguistic factor (*Is a peer* tutor) and two linguistic variables (*Words related to* self and *Words related to tool* use) demonstrated marginal significance. The findings indicate that language proficiency is strongly linked to math performance such that more complex syntactic structures and fewer explicit cohesion devices equate to higher course performance. The linguistic model also indicated that less self-centered students and students using words related to tool use were more successful. In addition, the results indicate that students that are more active in on-line discussion forums are more likely to be successful.

Keywords

NLP, math, student success, on-line learning

1. INTRODUCTION

Cognitive skills are crucial for student success in the math classroom. While research has primarily focused on skills that strongly overlap with math knowledge including spatial attention and quantitative ability [1], cognitive skills supporting math success such as language ability remain under-researched. At the same time, a number of researchers have argued that language skills are a prerequisite for transferring cognitive operations between math and language domains and that lower language skills can present critical obstacles in math reasoning.

Prior research has examined links between language skills and math success to examine the premise that students with greater language abilities are better able to engage with math concepts and problems. This research is based on the notion that success in the math classroom can be partially explained through language skills that allow students to constructively participate in math discussions as well as to quantitatively engage with math problems [2]. Similarly, math literacy is thought to be not just knowledge of numbers and symbols, but also knowledge of language to understand the discourse of math (i.e., the words surrounding the numbers and symbols) [3].

Despite research that links language skills to math success in the classroom, a major methodological problem in previous studies is the reliance on correlational analyses among standardized tests of math and linguistic knowledge. For instance, several studies have looked at the links between tests of language proficiency (e.g., syntax, knowledge, verbal ability, and phonological skills) and success on tests of math knowledge (e.g. algebraic notation, procedural arithmetic, and arithmetic word problems [4, 5]). Other studies have compared success on standardized math tests between native speakers of English and second language speakers of English with lower linguistic ability [6, 7]. While a few studies have focused on the perceived linguistic complexity of math problems in standardized tests [8, 9], the majority of studies have not analyzed the actual language produced by students and the relationship between language complexity and success on math assessments (see [10] for an exception).

This study builds on the work of Crossley et al. [10] and examines links between the complexity of language produced by students in on-line question/answer forum in a blended math class and their success in the course. To do so, we examine students' forum posts within the on-line tools used in the class for a number of linguistic features related to text cohesion, lexical sophistication, syntactic complexity, and sentiment derived from natural language processing (NLP) tools. The goal of this study is to examine the extent to which the linguistic features produced by students are predictive of their final scores in a blended discrete mathematics course. In addition to the linguistic features, we also examined a number of non-linguistic factors that are potentially predictive of math success including: whether the student was a peer tutor, class section (of two sections), and on-line forum behavior including: how many times they viewed posts, how many posts they made, how many questions they asked, how many answers they provided, and how many days they visited the on-line class forum.

1.1 Language and Math Relationships

Prior studies have investigated the connections between language proficiency and math skills in native speakers (NS) of English. These studies generally demonstrate strong links between math ability and language ability. For instance, Macgregor and Price [5] found that students who scored high on an algebra test also scored well on language tests. A follow-up study using a more difficult algebra test found a stronger relationship between algebraic notation and language ability. Similarly, Vukovic and Lesaux [4] reported links between language and math skills, but that the language skills differed in their degree of relation with math

knowledge. For example, general verbal ability was indirectly related through symbolic number skills while phonological skills were directly related to arithmetic knowledge. Other research has focused on the indirect links between math and language skills. For example, Hernandez [11] analyzed students' scores from the reading and math sections of a standardized test and found significant positive correlations between reading ability and math achievement. These findings led Hernandez to recommend that students' reading skills and strategy training should be factored into math instruction in order to increase effectiveness, especially for poor readers. However, not all studies have found strong links between math knowledge and language skills. For instance, LeFevre et al. [1] reported that linguistic skills were related to number naming, that quantitative abilities were related to processing numerical magnitudes, and that spatial attention was related to a variety of numerical and math tests. However, non-linguistic features such as quantitative abilities and spatial attention were stronger predictors of math ability.

In terms of language production, only one study, to our knowledge, has examined the links between the language produced by students and their success in the math classroom. Crossley et al. [10] examined linguistic and non-linguistic features of elementary student discourse while students were engaged in collaborative problem solving within an on-line math tutoring system. Student speech was transcribed and NLP tools were used to extract linguistic information related to text cohesion and lexical sophistication. They examined links between the linguistic features and pretest and posttest math performance scores as well as links with a number of non-linguistic factors including gender, age, grade, school, and content focus (procedural versus conceptual). Their results indicated that non-linguistic factors are not predictive of math scores but that linguistic features related to cohesion, affect, and lexical proficiency explained around 30% of the variance in students' math scores. Specifically, higher scoring students produced more cohesive texts that were more linguistically sophisticated.

1.2 Current Study

A number of studies have demonstrated strong links between students' linguistic knowledge, affect, and their success in math. Studies examining these links have traditionally relied on correlational analyses between linguistic knowledge tests and standardized math tests [1, 3, 4]. In this study, we take a novel approach and examine the linguistic features and affect of students' language production in a blended math class with both on-line and traditional face to face instruction. To derive our variables of interest, we analyzed the linguistic and affective features produced by the students in their forum postings using a number of NLP tools. These tools extract information related to text cohesion, lexical sophistication, syntactic complexity, and sentiment. In contrast to most prior studies (see [10] for an exception), our interest is not on linguistic performance as measured by standardized tests, but on linguistic performance as a function of language production as found in students' forum posts.

Our criterion variables are students' final score in the semester-long blended math class. In addition to examining relations between linguistic features of student language production and math scores, we also examined a number of non-linguistic factors including: whether the student was a peer tutor; how many times they viewed posts in the on-line forum; how many posts they made in the on-line forum; how many answers they provided in the on-line forum; how many questions they asked in the on-line forum; how many days they visited the on-line forum; and class section (there were two sections). Thus, in this study, we addressed two research questions:

1. Are non-linguistic factors significant predictors of math performance in a blended math class?
2. Are linguistic factors related to lexical sophistication, cohesion, syntactic complexity, and affect significant predictors of math performance in a blended math class?

2. METHOD

2.1 The Blended Math Class: Discrete Math

Discrete Mathematics is an undergraduate math course offered by the computer science department at North Carolina State University. Students in the course are provided instruction on the mathematical tools and abstractions that are integral to a general CS education, including logic, truth tables, set theory, graphs, counting, induction, recursion, and functions. Students majoring in CS must complete the course with a grade of C or better in order to remain in their degree program. The course includes 10 homework assignments, 5 lab assignments, 3 midterms, and a final exam.

The discrete math course studied is a *blended course*. In addition to the standard lecture and office hours, students are supported by a range of on-line tools. These include a Piazza question/answer forum, on-line homework assignments through WebAssign, and two labs that are Intelligent Tutoring Systems for logic and probability. Our focus in this analysis is the Piazza data. Piazza is a standard question-answering forum. Students, teaching assistants (TAs), and instructors are allowed to post questions or topic prompts as well as general polls. The members of the class may then respond to these posts with replies and sub-replies. They may also choose to recommend both posts and replies as being particularly informative but clicking on a "good question" or "good answer" button. Question responses are classified in Piazza. The instructors and TAs may post an official "instructor response". If that is done, then these are flagged separately from student replies. Individuals may edit their replies over time in response to users' comments. While Piazza may be configured to permit anonymous posting by students, this function was turned off by default in this course. In addition to the basic thread structure, Piazza requires that posts be categorized by topic and it keeps a running list of threads and supports basic search to help students locate relevant information.

We study data from the Fall 2013 semester of this course. During that semester, the class was divided into two sections with two primary instructors, five teaching assistants, and 250 students. In addition to the instructor and official graduate TAs, the course was supported by a set of peer tutors. These are high-performing students in the course who are given extra credit for acting as mentors. During the Fall 2013 semester, 32 students volunteered to act as peer tutors and roughly 1/3 of them completed the required 10 hours to receive extra credit.

For the purposes of our analysis, we collected Piazza data recording the students' interactions once the course was complete. This data included how many times students viewed posts in the Piazza forum, how many posts students made in the Piazza forum, how many answers students provided in the Piazza forum, how many questions students asked in the Piazza forum, and how many days students visited the Piazza forum.

2.2 Forum Posts

We selected forum posts because they provide students with a platform to exchange ideas, discuss lectures, ask questions about the course, and seek technical help, all of which lead to the production of language in a natural setting. Such natural language can provide researchers with a window into individual student motivation, linguistics skills, writing strategies, and affective states. This information can in turn be used to develop models to improve students' learning experiences [12].

Students in the course were given access to the Piazza forum at the start of the class. Students were encouraged to use Piazza (not email) for course communications by posting their questions to the forum outside of class, and answering questions posed by their peers. The TAs and peer tutors were required to check the forum regularly with the goal of ensuring an average response time of 15 minutes per post, and that no single question would "go stale" by being left for more than 2 hours without a reply. In addition to basic question/reply Piazza interactions, the instructor and TAs posted regular announcements and general comments to the forum, making it the primary vehicle for non-lecture communication in the course.

Student posts were retrieved from a Piazza database that was extracted at the end of the course. The student posts were segmented out to eliminate duplicate content as well as unnecessary markup. Of the 250 students in the course, 169 made posts on the forum. For the 169 students who made a forum post, we aggregated each of their posts such that each post became a paragraph in a text file. We selected only those students who produced at least 50 words in their aggregated posts (n = 158). We selected a cut off of 50 words in order to have sufficient linguistic information to reliably assess the student's language using NLP tools.

2.3 Natural Language Processing Tools

We used several NLP tools to assess the linguistic features in the aggregated posts of sufficient length. These included the Tool for the Automatic Analysis of Lexical Sophistication (TAALES) [13], the Tool for the Automatic Analysis of Cohesion (TAACO) [14], the Tool for the Automatic Analysis of Syntactic Sophistication and Complexity (TAASSC) [15], and the SEntiment ANalysis and Cognition Engine (SEANCE) [16]. The selected tools reported on language features related to lexical sophistication, text cohesion, and sentiment analysis respectively. The tools are discussed in greater detail below.

2.3.1 TAALES

TAALES incorporates about 150 indices related to basic lexical information (e.g., the number of tokens and types), lexical frequency, lexical range, psycholinguistic word information (e.g., concreteness, meaningfulness), and academic language for both single word and multi-word units (e.g., bigrams and trigrams).

2.3.2 TAACO

TAACO incorporates over 150 classic and recently developed indices related to text cohesion. For a number of indices, the tool incorporates a part of speech (POS) tagger and synonym sets from the WordNet lexical database [17]. TAACO provides linguistic counts for both sentence and paragraph markers of cohesion and incorporates WordNet synonym sets. Specifically, TAACO calculates type token ratio (TTR) indices, sentence overlap indices that assess local cohesion, paragraph overlap indices that assess global cohesion, and a variety of connective indices such as logical connectives (e.g., *moreover, nevertheless*) and sentence linking connectives (e.g., *nonetheless, therefore, however*).

2.3.3 TAASSC

TAASSC measures large and fined grained clausal and phrasal indices of syntactic complexity and usage-based frequency/contingency indices of syntactic sophistication. TAASSC includes 14 indices measured by Lu's [18] Syntactic Complexity Analyzer (SCA), 31 fine-grained indices or clausal complexity, 132 fine-grained indices of phrasal complexity, and 190 usage-based indices of syntactic sophistication. The SCA measures are classic measures of syntax based on t-unit analyses [19]. The fine-grained clausal indices calculate the average number of particular structures per clause and dependents per clause. The fine-grained phrasal indices measure 7 noun phrase types and 10 phrasal dependent types. The syntactic sophistication indices are grounded in usage-based theories of language acquisition [Ellis, 2002] and measure the frequency, type token ratio, attested items, and association strengths for verb-argument constructions (VACs) in a text.

2.3.4 SEANCE

SEANCE is a sentiment analysis tool that relies on a number of pre-existing sentiment, social positioning, and cognition dictionaries. SEANCE contains a number of pre-developed word vectors developed to measure sentiment, cognition, and social order. These vectors are taken from freely available source databases. For many of these vectors, SEANCE also provides a negation feature (i.e., a contextual valence shifter) that ignores positive terms that are negated (e.g., not happy). SEANCE also includes a part of speech (POS) tagger.

2.4 Statistical Analysis

We calculated linear models to assess the degree to which linguistic features in the students' language output along with other fixed effects (e.g., question/note posted, questions answered, site visits) were predictive of students' final math scores. Prior to linear model analysis, we first checked that the linguistic variables were normally distributed. We also controlled for multicollinearity between all the linguistic and non-linguistic variables ($r \geq .900$) such that if two or more variables were highly collinear, all but one of the variables was removed from the analysis. We used R [21] for our statistical analysis. Final model selection and interpretation was based on t and p values for fixed effects and visual inspection of residuals distribution for non-standardized variables. To obtain a measure of effect sizes, we computed correlations between fitted and observed values, resulting in an overall R^2 value for the fixed factors. We developed and compared three models: (a) a baseline linear model including non-linguistic fixed effects, (b) a model including only linguistic factors, (c) a model including both linguistic and non-linguistic effects. We compared the strength of each model using Analyses of Variance (ANOVAs) to examine which models were most predictive.

3. RESULTS

3.1 Non-linguistic Linear Model

A linear model considering of all non-linguistic fixed effects revealed significant effects for whether the student was a tutor or not and number of days spent on the Piazza forum. Table 1 displays the coefficients, standard error, t values, and p values for each of the significant non-linguistic fixed effects. The overall model was significant, $F(3, 154) = 6.116$, $p < .001$, $r = .326$, $R^2 = .107$. Inspection of residuals suggested the model was not

influenced by homoscedasticity. The non-linguistic variables explained around 11% of the variance of the math scores and indicated that students who acted as peer tutors and visited the system more often received higher overall grades in the class.

Table 1. Non-linguistic model for predicting math scores

Fixed Effect	Coefficient	Std. Error	*t*
(Intercept)	83.988	1.484	56.603***
Is a peer tutor	5.410	1.995	2.712**
Is not a peer tutor	3.340	2.090	1.598
Days on system	0.038	0.012	3.116**

Note * $p < .050$, ** $p < .010$, **p < .001

3.2 Linguistic Linear Model

A linear model including linguistic fixed effects revealed significant effects for a number of features related to reference self, syntactic complexity, reference to tools, and cohesion. Table 2 displays the coefficients, standard error, *t* values, and *p* values for each of the linguistic fixed effects. The overall model was significant, $F(4, 153) = 9.456$, $p < .001$, $r = .360$, $R^2 = .130$. Inspection of residuals suggested the model was not influenced by homoscedasticity. The linguistic variables explained around 13% of the variance of the math scores and indicated that students who referred to themselves less often, used more complex syntax, referred to words related to the use of tools, and used fewer sentence linking terms received higher final grades in the course. An ANOVA comparison between the non-linguistic model and the linguistic found a significant difference between the models, ($F = 8.120$, $p < .001$), indicating that linguistic features contributed to a better model fit than non-linguistic features.

Table 2. Linguistic model for predicting math scores

Fixed Effect	Coefficient	Std. Error	*t*
(Intercept)	91.089	3.795	24.002***
Words related to self	-67.146	26.024	-2.580*
Number of dependents per prepositional object nominal	6.800	2.478	2.744**
Words related to tools	144.097	62.658	2.300*
Sentence linking connectives	-77.055	33.947	-2.27*

Note * $p < .050$, ** $p < .010$, **p < .001

3.3 Full Linear Model

A linear model considering non-linguistic and linguistic fixed effects revealed significant effects for one of the non-linguistic features (days on the system) and two of the linguistic features (*Number of dependents per prepositional object nominal* and *Sentence linking connectives*). One non-linguistic factor (*Is a peer* tutor) and two linguistic variables (*Words related to* self and *Words related to tool* use) demonstrated marginal significance. Table 3 displays the coefficients, standard error, *t* values, and *p* values for each of the fixed effects. The overall model was significant, $F(7, 150) = 9.295$, $p < .001$, $r = .399$, $R^2 = .159$. Inspection of residuals suggested that the model was not influenced by homoscedasticity. The non-linguistic and linguistic variables explained around 16% of the variance of the math scores and followed the same trends as reported in the first two models. An ANOVA comparison between the full model and the linguistic model found a significant difference between the models, ($F = 2.790$, $p < .050$), indicating that a combination of non-linguistic and linguistic features contributed to a better model fit than linguistic features alone.

Table 3. Full model for predicting math scores

Fixed Effect	Coefficient	Std. Error	*t*
(Intercept)	86.564	4.065	21.296***
Is a peer tutor	3.840	1.974	1.946
Is not a peer tutor	1.516	2.065	0.734
Days on system	0.028	0.012	2.273*
Words related to self	-44.990	26.876	-1.674
Number of dependents per prepositional object nominal	6.156	2.455	2.507*
Words related to tools	120.451	62.545	1.926
Sentence linking connectives	-72.463	33.644	-2.154*

Note * $p < .050$, ** $p < .010$, **p < .001

4. DISCUSSION AND CONCLUSION

Previous research has indicated that language skills are related to math success. Much of this research examined links between standardized tests of language proficiency and success on tests of math knowledge [4, 5] while other research has compared native English speakers to second language speakers of English in terms of success on standardized math tests [6, 7]. In general, these studies have yielded positive relationships between language skills and math success. However, the majority of these studies did not examine links between the language produced by students and math success. A notable exception to this is Crossley et al.'s [10] study that used NLP tools to examine links between language used in an third grade math classroom and success on math assessments. This study reported that linguistic features related to cohesion, affect, and lexical proficiency explained around 30% of the variance in the math scores.

In this study, we take a similar approach to Crossley et al. [10] and use NLP tools to extract a number of linguistic and sentiment features from forum posts found in a blended discrete math undergraduate course. We found that a number of non-linguistic and linguistic features were strong predictors of math success. For instance, peer tutors and students who spent more time on the Piazza forums were more likely to be successful in the class. Linguistically, students who used fewer words related to self, more syntactically complex sentences, more words related to tool use, and fewer connectives were also more successful in the class. The non-linguistic model explained about 11% of the variance in the math scores while the linguistic model explained about 13% of the variance. A model that included both non-linguistic and linguistic variables explained about 16% of the variance in the math scores.

The variance explained by our model was lower than that reported in Crossley et al [10]. However, unlike Crossley et al., our participants were not elementary level students and they were not involved in collaborative discourse. Rather, our participants were college students and the language samples used in this study came from on-line forum posts as compared to natural conversation between students in a classroom. These differences likely explain the disparities reported between the two studies. For instance, in the current study we found a negative correlation between a cohesion index (*sentence linking connectives*) and math scores. This may be the result of linguistic development in which young children develop text cohesion using explicit markers of cohesion while college students use complex syntax to develop cohesive text [22, 23]. This distinction likely indicates that the strong positive correlation between syntactic complexity and math success reported in this study indicates that more skilled writers have greater success in the math classroom.

This study also found that a number of different indices than those reported by Crossley et al. were predictive of math success. These included words related to self, which was negatively associated with math success, and words related to tool use, which was positively associated with math success. The finding for words related to self should likely be interpreted in terms of self-centered behavior such that students who were more self-centered were likely to be less successful in the math class. This may be a result of the collaborative nature of the Piazza forum in which students were encouraged to work together to answer questions and solve problems. In terms of words related to tool use, the findings likely indicate that more successful students used terms that were more strongly related to the domain such as *computer, equipment, file, machine, mechanism,* and *paper*. However, it is notable that neither the use of words related to self or to the use of tools were a significant predictor in the full model that included both linguistic and non-linguistic variables.

In terms of non-linguistic features, this analysis demonstrated that two non-linguistic factors were important indicators of math success: *peer tutoring* and *days on Piazza*. The findings indicate that those students who volunteered to peer tutor were more successful in the class. In addition, those students who spent a greater number of days on the Piazza forum were more successful suggesting that engagement in the class discussion forum led to greater success. However, only the number of days spent on the Piazza forum was a significant predictor in the full model.

The findings from this study have practical implications for understanding math performance in a blended math class at the university level. Specifically, the findings provide additional support that language proficiency is strongly linked to math performance such that more complex syntactic structures and fewer explicit cohesion devices equate to higher course performance. The linguistic model also indicated that less self-centered students and students using words related to tool use were more successful. In addition, the results indicate that students who are more active in on-line discussion forums are more likely to be successful. The study also provides a contrast to early research [10] in that differences are reported between age levels (elementary and college level students) and learning environments (collaborative discussions and forum posts). Future studies can build on these results by continuing to examine language features and math success in a number of different student populations and learning environments.

5. ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation (DRL- 1418378). Ideas expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

6. REFERENCES

[1] LeFevre J. A., Fast L., Skwarchuk S.L., Smith-Chant B.L., Bisanz J., Kamawar D., Penner-Wilger M. 2010. Pathways to math: Longitudinal predictors of performance. *Child Development 81*(6): 1753–1767.

[2] Vukovic, RK, Lesaux NK (2013) The relationship between linguistic skills and arithmetic knowledge. Learning and Individual Differences *23*: 87-91.

[3] Adams TL (2003) Reading math: More than words can say. The Reading Teacher 56(8): 786–795.

[4] Vukovic, RK, Lesaux NK (2013) The relationship between linguistic skills and arithmetic knowledge. Learning and Individual Differences *23*: 87-91.

[5] MacGregor M, Price E (1999) An exploration of aspects of language proficiency and algebra learning. Journal for Research in Math Education 449–467. doi: 10.2307/749709

[6] Alt M, Arizmendi GD, Beal CR (2014) The relationship between math and language: Academic implications for children with specific language impairment and English language learners. Language, Speech, and Hearing Services in Schools 45(3): 220–233. doi:10.1044/2014_LSHSS-13-0003

[7] Hampden-Thompson G, Mulligan G, Kinukawa A, Halle T (2008) Math Achievement of Language-Minority Students During the Elementary Years. Washington, DC: U.S. Department of Education, National Center for Education Statistics.

[8] Martiniello M (2008) Language and the performance of English-language learners in math word problems. Harvard Educational Review 78(2): 333–368.

[9] Martiniello M (2009) Linguistic complexity, schematic representations, and differential item functioning for English language learners in math tests. Educational Assessment 14(3–4): 160–179. doi:10.1080/10627190903422906

[10] Crossley, SA, Liu R, McNamara D (2017). Predicting Math Performance Using NLP Tools. Proceedings of the 7th International Learning Analytics and Knowledge (LAK) Conference. New York, NY: ACM.

[11] Hernandez F (2013) The Relationship Between Reading and Math Achievement of Middle School Students as Measured by the Texas Assessment of Knowledge and Skills (Doctoral dissertation).

[12] Ramesh, A., Goldwasser, D., Huang, B., Daume, H., & Getoor, L. (2014). Understanding MOOC Discussion Forums using Seeded LDA. In *9th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 28–33). Baltimore, MA: ACL.

[13] Kyle, K., & Crossley, S. A. (2015). Automatically assessing lexical sophistication: Indices, tools, findings, and application. *TESOL Quarterly, 49*(4), 757-786. doi:10.1002/tesq.194

[14] Crossley, S. A., Kyle, K., & McNamara, D. S. (in press). The tool for the automatic analysis of text cohesion (TAACO): Automatic assessment of local, global, and text cohesion. *Behavior Research Methods*.

[15] Kyle, K., & Crossley, S. A. (in press). Assessing syntactic sophistication in L2 writing: A usage-based approach. *Language Testing.*

[16] Crossley, S. A., Kyle, K., & McNamara, D. S. (in press). Sentiment Analysis and Social Cognition Engine (SEANCE): An Automatic Tool for Sentiment, Social Cognition, and Social Order Analysis. *Behavior Research Methods*.

[17] Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM, 38*(11), 39–41.

[18] Lu, X. (2010). Automatic analysis of syntactic complexity in second language writing. *International Journal of Corpus Linguistics*, *15*(4), 474–496.

[19] Ortega, L. (2003). Syntactic complexity measures and their relationship to L2 proficiency: A research synthesis of college-level L2 writing. *Applied Linguistics*, *24*(4), 492–518.

[20] Ellis, N. C. (2002). Frequency effects in language processing. *Studies in Second Language Acquisition*, *24*(2), 143–188.

[21] R Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. 2013: ISBN 3-900051-07-0.

[22] Crossley, S. A., Weston, J., McLain Sullivan, S. T., & McNamara, D. S. (2011). The development of writing proficiency as a function of grade level: A linguistic analysis. Written Communication, 28(3), 282-311.

[23] Haswell, R. H. (2000). Documenting improvement in college writing: A longitudinal approach. Written Communication, 17, 307-352.

Task and Timing: Separating Procedural and Tactical Knowledge in Student Models

Joshua Cook* , Collin F. Lynch, Andrew G. Hicks, & Behrooz Mostafavi
Department of Computer Science, North Carolina State University, Raleigh, NC, U.S.A.
jacook7, cflynch, aghicks3, & bzmostaf@ncsu.edu

ABSTRACT

BKT and other classical student models are designed for binary environments where actions are either correct or incorrect. These models face limitations in open-ended and data-driven environments where actions may be correct but non-ideal or where there may even be degrees of error. In this paper we present BKT-SR and RKT-SR: extensions of the existing BKT model that distinguish knowing *how* to apply a skill from knowing *when*. We compare their relative performance to that of classical BKT and PFA on data collected from Deep Thought, an open-ended propositional logic tutor. We develop basic performance curves for student outcomes to help us visually compare models predictions to data. We also introduce a new approach for finding a probability distribution of actions in ranked, multiple option environments with RKT and RKT-SR. Our results show that knowing when to use skills is more important than how in these open-ended contexts. In fact, including the *how* components may hurt performance if implemented naively. Furthermore we show that ranked models outperform binary-based models even under restrictive assumptions.

Keywords

Student-Modeling, Data-Driven Tutoring, Open-Ended Tutors, BKT, PFA, Interaction Networks, RKT, RKTSR, BKTSR

1. INTRODUCTION

Bayesian Knowledge Tracing (BKT) and other existing learner models, such as Performance Factors Analysis (PFA), are about *right* and *wrong* but for many realistic problem-solving situations students are not choosing just correct or incorrect actions. They are choosing from among a range of potential actions some of which may be optimal or substantively better than others. Thus the classical models are out of sync with the performance criteria by which the students are being judged. It also means that the models, by design, conflate two distinct skills: knowing *how* to apply a skill (procedural knowledge), and knowing *when* to apply a skill (tactical knowledge). In classical BKT we base performance on the validity of an individual action not on its optimality. Thus students receive points for correctly applying sub-optimal skills.

In this paper we present an extension to BKT, BKT-SR, which separates tactical knowledge (recognition of optimal skills) from procedural knowledge (correct skill application). This model is designed for use in open-ended and data-driven tutorial domains where students are expected to learn not just how to apply individual skills but how to recognize the sequence of skill applications that make up an optimal solution. We also present a refinement of the existing probability calculations for ranked options, and apply these in two new models: RKT and RKT-SR. This refinement leads to an improvement in the accuracy of the models over existing methods.

Additionally, in order to investigate which component of BKT-SR is most important, we tested the individual components (how, when, and some slight variations) on student data. Our data is drawn from an open ended propositional logic tutor called Deep Thought that is designed for use in discrete mathematics and philosophy. We compare the differing models on our data set to demonstrate that knowing when to apply a skill is separable from knowing how.

2. EXISTING MODELS

BKT and PFA are two of the most successful student modeling approaches. Both are binary action models that predict whether a student will take actions that are correct or incorrect at any given time given their level of understanding and other parameters. In prior head-to-head comparisons the two have performed similarly [5].

BKT is a simple two state Hidden-Markov Model (HMM) [3]. It is based upon five assumptions. Each skill is independent and has two states: learned, L, and not learned. Each problem depends on exactly one skill, and answers are either correct or incorrect. Students can learn, but cannot forget. After an opportunity to apply a skill, there is a constant probability to transition, T, from unlearned to learned. Students who know a skill will answer a problem correctly unless they slip, S, and students who don't know a skill answer incorrectly, unless they guess, G.

The parameters of BKT are: L0, the initial probability of knowing a skill. T, the probability of transitioning from unlearned to learned. G, the probability of answering a question correctly when a skill is not learned. S, the probability of answering a question incorrectly when a skill is learned.

*Corresponding author

Let L_i be the probability of knowing a skill at step i. Then the probability of answering a problem correctly is calculated as:

$$P(\text{Correct}) = L_i \cdot (1-S) + (1-L_i) \cdot G$$

To update L, we first apply Bayes theorem, then apply the transition probability. The reinforcement process has two steps:

$$B_i(\text{Answer}) = \begin{cases} \frac{L_i \cdot (1-S)}{L_i \cdot (1-S) + (1-L_i) \cdot G} & \text{Answer is correct} \\ \frac{L_i \cdot S}{L_i \cdot S + (1-L_i) \cdot (1-G)} & \text{Answer is incorrect} \end{cases}$$

$$L_{i+1} = B_i(\text{Answer}) + T \cdot (1 - B_i(\text{Answer}))$$

BKT is time tested, easily interpreted and implemented, but fitting BKT parameters is difficult. One difficulty lies in avoiding degenerate parameters: parameters that cause BKT to behave counter to its' physical interpretation. We avoid degenerate models using brute force grid search [5].

PFA, by contrast, is a logistic regression model based upon the skill difficulty(β), number of successes (γ), and number of failures(ρ) [11]. PFA has many upsides, not the least of which is that it can be fit efficiently with general regression calculations.

3. INTERACTION NETWORKS

The above models were designed for classical binary problems. Most realistic problems however are more open-ended. Problems are defined by a goal state and a set of given information that problem solvers may apply a range of rules to achieve their goal. Rather than each action being correct or incorrect some actions are correct in a given solution context and there many be many ways to solve a problem or many actions to take at a given time with some being more efficient than others. The structure of these open-ended solutions can be efficiently represented in a data structure called an interaction network. Interaction Networks are directed graphs representing a solution space where each node is a partial solution state and each edge is a rule application [4]. Individual solutions are represented as paths in the network from the start state to a goal state. An Interaction Network is the aggregation of all the student solutions for a problem where each edge is weighted by the number of students who followed it.

3.1 Value Iteration

Value iteration is an algorithm for identifying the optimal policy (π) for use in a Markov Decision Process (MDP) [1]. The core of the algorithm depends upon an update function that estimates the current value of a state ($V_{i+1}(s)$) based upon a set reward (R), the current values of the neighboring states ($V_i(u_e)$), a discount factor or cost for taking each action (γ), and the probability of taking an action ($P(e)$). In these experiments we use a constant reward function and a discount factor. Goal states were assigned a constant value, and the probability of a given action ($P(e)$) transitioning from state s to s' was estimated based upon the number of times that it was taken relative to the total number of steps out of s.

For the purposes of our study we defined two forms of the value function. The *optimistic* function assumes that students will take the best possible action in a given state and thus the best possible route to a goal. The *conservative* function, by contrast, assumes that they will follow the general probability distribution of the dataset. Thus:

Conservative: $V_{i+1}(s) = R + \gamma \cdot \sum_{e \in E_s} P(e) \cdot V_i(u_e)$

Optimistic: $V_{i+1}(s) = \max_{e \in E_s} R + \gamma \cdot P(e) \cdot V_i(u_e)$

The former approach was used in the Hint Factory system which uses interaction networks to generate data-driven hints [15], while the latter is equivalent to a single option MDP [16]. Any iteration that maximizes over contracting functions like these is, by definition, a contraction mapping [7]. Thus both forms will converge over time to a stable value.

4. OUR EXTENSIONS

We built several different extensions to the existing BKT model that are designed to take advantage of extra information in the interaction network to separate *tactical knowledge* (when to apply a skill) from *procedural knowledge* (how to apply a skill).

4.1 BKT-SR (BKT Skill Recognition)

BKT Skill Recognition (BKT-SR) is a semi-binary model that predicts students' behavior on a binary basis but reinforces on a more complex paired. In it we maintain two separate BKT models for each skill, one tracks procedural knowledge BKT_{How}, and the the other tracks tactical knowledge BKT_{When}. BKT-SR assumes that the ideal skill will be used only if the student correctly recognizes how to apply it, *and* knows that it is ideal.

The probability of answering a question correctly is the probability given by BKT_{How} multiplied by that given by BKT_{When}. The difference between the two models lies in their reinforcement. BKT_{How} reinforces the skill component of the action used, positively if it was used correctly. BKT_{When} reinforces skill component of both the action used AND the ideal action, positively if they are the same, negatively otherwise.

4.2 RKT (Ranked Knowledge Tracing)

Our environment is not binary, there are almost always several 'correct' options of ranked quality for each state. We therefore introduce the ranked models, RKT and RKT-SR. These models introduce a technique to give a probability distribution over a set of ranked options from simpler single skill models. The underlying model and reinforcement technique of RKT and RKT-SR is similar to BKT however it can be replaced by other comparable models so long as the reinforcement process is modified appropriately. This approach gives us a rigorous way to aggregate simple learner model predictions into a valid probability distribution over all actions. Conceptually, RKT tries the best option, if that fails it tries the second best, if that fails it tries the third and so on, wrapping back to the first.

Let x be our current model state and let $\alpha_i(x)$ be the probability that a student can use the skill required for option i given state x. Assuming the that the n skill options for a problem are given in order, the probability of using the i^{th} action is

$$p_i(x) = \frac{\alpha_i(x) \prod_{j=1}^{i-1} (1 - \alpha_j(x))}{1 - \prod_{j=1}^{n} (1 - \alpha_j(x))}$$

RKT's underlying model uses a simple two state Hidden-Markov Model (HMM) for each skill. State x is a vector of knowledge confidence. While $\alpha_i(x)$ is defined by taking the i^{th} component as L, and then calculating the probability as in standard BKT. RKT's update function is inspired by Bayes' theorem but differs slightly as our probability function is not linear. An exact, naïve imple-

mentation of an HMM would require that we sum over every combination of skill knowledge, which is prohibitively expensive.

To illustrate the update algorithm, suppose skill k is applied in state x, and that x_j is the probability of knowing skill j, and u_j is x with the j^{th} skill set to 1. We then calculate the new value for skill j, y_j, as:

$$y_j = \frac{p_k(u_j) \cdot x_j}{p_k(x)}$$

After each update we apply our transition function only to the ideal skill model. This function is applied in the same way as in BKT. Here p_i is convex in each argument, so our update will keep L between 0 and 1. Further, it will increase L iff knowing L will increase the chance of the given action. Thus the update is consistent and in the appropriate direction.

4.3 RKT-SR (RKT Skill Recognition)

Like BKT-SR, RKT-SR tries to separate procedural and tactical knowledge using two parallel RKTs, one for *how* and one for *when*. Like RKT, for state x, let $\alpha_i(x)$ denote confidence of being able to apply the skill used in option i, and $\beta_i(x)$ denote confidence of being able to identify when to use skill of option i.

In the RKT-SR approach we model the student's process as first noticing a set of options (how skill). Then, of the noticed options, they rank them (when skill). And finally they select the highest rank action to the best of their ability. Thus the probability of doing action i is:

$$p_i(x) = \sum_{\{i\} \in S \subseteq [n]} \frac{1}{1 - \prod_{j \in [n]} (1 - \alpha_j(x))} \prod_{j \in S} \alpha_j(x) \prod_{j \in [n] \setminus S} (1 - \alpha_j(x)) \cdot \frac{\beta_i(x) \prod_{j < i, j \in S} (1 - \beta_j(x))}{1 - \prod_{j \in S} (1 - \beta_j(x))}$$

This simplifies to:

$$p_i(x) = \frac{\alpha_i(x)\beta_i(x)}{1 - \prod_{k \in [n]} (1 - \alpha_k(x))} \sum_{j=0}^{\infty} (1 - \beta_i(x))^j \cdot \prod_{k=1}^{i-1} (\alpha_k(x)(1 - \beta_k(x))^{j+1} + 1 - \alpha_k(x)) \cdot \prod_{k=i+1}^{n} (\alpha_k(x)(1 - \beta_k(x))^j + 1 - \alpha_k(x))$$

Assuming that each β is bounded away from 1 and 0, we can approximate the infinite sum by taking a fixed number of terms, then normalizing it. For the sake of efficiency, we limit the number of terms to 3. We believe that RKT-SR has a convex probability function like RKT. Thus we update it similarly, with how and when updated independently.

Note that setting all $\alpha_i = 1$ in this model yields RKT, as does setting all $\beta_i = 1$. Thus RKT does not necessarily claim that either tactical or procedural knowledge is more important, since modelling either one with the assumption that the other is trivial yields the same model.

5. DATA SET

For this analysis we collected data from two semesters of an undergraduate Discrete Mathematics course at NCSU where Deep Thought is used. This dataset includes 4 class sections, 205 students, 2322 problem attempts, and 28640 individual steps. Unfortunately the data includes several cases where individual events were not logged such as the student entering or exiting the program, and cases where events were logged out of order due to network issues. While we cleaned these up as much as possible, we still include 913 errors in our data that we could detect but could not fix. While this missing data may contain important information, the average student only had a few such errors, even though 148 of the students had some kind of error in their logs.

In open-ended tutors like Deep Thought, problem-solving errors (i.e. incorrect applications) are often treated in one of two ways. Either the system records the mistake but leaves it onscreen and does not permit it to hinder forward progress. Or the system forces the student to fix or retract it immediately. In effect this forces the user to always step back to their prior state before moving on. Deep Thought adopts this latter approach. Consequently it is possible to ignore user mistakes in our dataset or to recognize them explicitly. With that in mind we tested our models with two different interaction networks. One network ignored self-loops, thus ignoring mistakes, and the other included them.

For each state, we ranked the set of derived statements to obtain a canonical order. Thus the states are dependant only on what was derived, not how or when it was derived.

5.1 Deep Thought

Deep thought is an intelligent tutoring system for propositional logic. Deep thought has been continually improved with hints [15], worked examples [10], and proficiency profiling [9]. The system's assessments have been verified against student test scores [8]. Deep Thought uses a GUI to guide students through 6 problem levels with increasing difficulty. Problems in Deep Thought are presented as a set of logical assumptions, and a statement which the student must to derive from them by applying axioms of propositional logic.

6. METHODOLOGY

We first generated the networks using all of the student data. This ensured that all actions taken by the students were included in the graph thus simplifying our analysis. This was not expected to bias in favor of any model. For the modeling step we only calculated the error and reinforced the models based upon steps with multiple correct options.

We used InVis to produce the graphs and perform the value iteration [12]. We fixed the value of our goal states at 100, used a negative immediate reward for each action of -1, and a discount factor of 0.9. Every other state started with a value of 0.

When measuring error, we focus on the cases where the system predicts that that a student will take the ideal action. We use a running average as our baseline. For the present we are more interested in the relative performance of our models than their absolute performance against chance.

In many states there are two distinct ideal actions that lead to different states with the same value. In this case, we want to know if a student completed either one. To get the appropriate probability of an ideal action we calculate the individual probabilities of the two ideal actions and, assuming they are independent, we then return the probability that either one is

performed. This approach works for simpler models like BKT and PFA which return per-action probabilities. However it may be unfairly penalizing RKT and RKT-SR, who return a complete probability distribution.

We tested our models using 10 fold cross validation. Each model was fit using an exhaustive grid search minimizing RMSE. Final metrics were found by calculating the RMSE and AUC for each fold, and then averaging them.

6.1 Applying Binary Models

BKT and PFA are not designed to handle non-ideal solutions, thus their models do not tell us how to reinforce them in this case. For each skill, we can reinforce the underlying knowledge component of the skill positively (*reward*), or negatively (*punishment*). Thus each model is seen as a black box, where we "select" skills to reinforce, and reward or punish it appropriately. In this context we can reinforce the skills that the student actually performed as well as the ideal skills, which they may not. Here we tested four different versions of BKT which differ in what skills are selected for punishment and which are selected for reward.

Stock-BKT: This focuses solely on the students' demonstrated skills, ignoring idealness. It selects the skill used and rewards it if the action is correct. **ActualSkill-BKT:** This focuses on the students demonstrated skills, but with only the best possible action considered correct. It selects the skill used and rewards it if it is ideal. **IdealApp-BKT:** Focuses on whether or not the student knows which action is ideal and penalizes them for anything else. Selects the ideal skill and rewards if it was used ideally. The model makes no change if they performed a correct, but non-ideal use of the skill, and it punishes otherwise. **IdealActual-BKT:** Attempts to model both using a joint probability distribution. Thus it explicitly conflates knowing when to do something and knowing how and then sets a standard of correctness consistent with that. Selects both the ideal and the applied skills. If the ideal skill is used it is rewarded, otherwise both are punished.

We chose to reinforce PFA and the running average using the same selection model as in ActualSkill-BKT. For reference, BKT-SR is equivalent to IdealActual-BKT times Stock-BKT, reinforced independently.

6.2 Plotting Performance

In order to quickly check for skill acquisition, we developed a visualization technique. For each student, we look at the opportunities that they had to apply a skill ideally, and whether they actually used it. We then plotted these frequencies for all students on a single scatter plot.

Specifically, for each student x, and for each skill k, we make vector k^x, where the length of k^x is the number of times when skill k was ideal, with k_i^x 1 if the student used the ideal option the ith time k was ideal, 0 otherwise. Let $n_x(i)$ be the set of skills that were ideal at least i times. Define v^x as

$$v_i^x = \frac{\sum_{k \in n_x(i)} k_i^x}{|n_x(i)|}$$

Then we just plot each v^x together on a scatter graph. For comparison purposes we simulated data using BKT and plotted it using this technique. In it, you can see a clear trend. This trend is not clearly visible in our real data set. While some tweaking of the parameters in the simulated data show slower learning, they still show learning. Even graphs with errors look almost identical to the ones shown irrespective of value iteration algorithm. Thus this technique, while interesting, is ill-suited to detect learning in this domain.

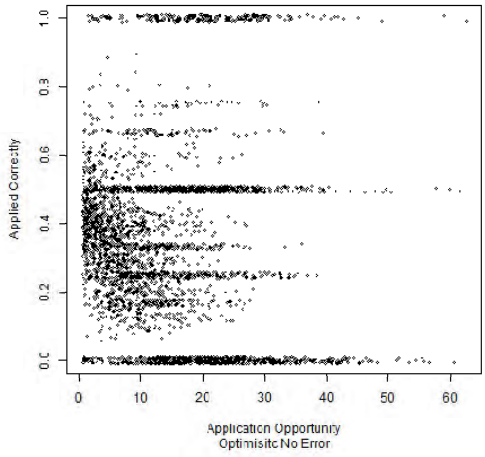


Figure 1: Real Data Performance

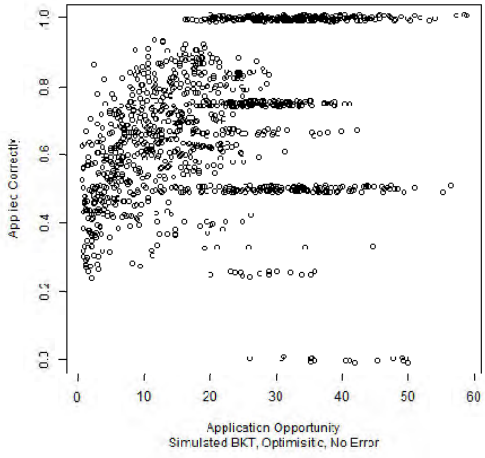


Figure 2: Simulated Performance

6.3 Model Fitting

We fit our parameters using exhaustive grid search. Grid search often performs favorably with other fitting methods like EM [14]. We define our grid by specifying the upper bound, the lower bound, and the number of equal length steps between them for each parameter. We chose the parameter bounds so that no fit would be degenerate [17]. BKT-SR used the same parameters to fit both the when and how subskills, but fits them independently to save time. Similarly for RKT-SR.

We chose the resolution for our grid search model in these cases to guarantee a similar amount of time per search, around 2

Table 1: Model Fitting Results

Model	Optimistic				Conservative			
	No Err		Err		No Err		Err	
	RMSE	AUC	RMSE	AUC	RMSE	AUC	RMSE	AUC
Average	0.451457	0.696120	0.438547	0.690875	0.465104	0.674632	0.446898	0.667558
PFA	0.454968	0.690093	0.442861	0.681035	0.469697	0.661166	0.451412	0.660922
Stock-BKT	0.493906	0.664096	0.489387	0.647382	0.492487	0.663387	0.495561	0.633865
ActualSkill-BKT	0.458204	0.676102	0.446208	0.671619	0.471135	0.656281	0.454614	0.646841
IdealApp-BKT	0.452686	0.699546	0.438583	0.709654	0.465627	0.681597	0.448043	0.686899
IdealActual-BKT	*0.449347*	0.697695	*0.436518*	0.704180	*0.462758*	0.682124	*0.444161*	0.684025
BKT-SR	0.452071	0.691284	0.469820	0.650012	0.465264	0.671495	0.479585	0.628389
RKT	0.450763	*0.737032*	0.437331	*0.724183*	0.464668	*0.709409*	0.447027	*0.704591*
RKT-SR	**0.440841**	**0.739516**	**0.432296**	**0.729586**	**0.455561**	**0.715869**	**0.438965**	**0.713305**

Table 2: KT Fitting Parameters

	BKT				BKTSR				RKT				RKTSR			
	L0	T	G	S	L0	T	G	S	L0	T	G	S	L0	T	G	S
Min	0.1	0.02	0.04	0.02	0.2	0.03	0.04	0.03	0.2	0.06	0.07	0.06	0.3	0.06	0.08	0.1
Steps	5	5	5	5	4	4	4	4	3	4	4	3	2	3	3	2
Max	0.9	0.30	0.40	0.30	0.8	0.30	0.40	0.30	0.8	0.30	0.40	0.30	0.7	0.30	0.40	0.25

Table 3: Baseline Fitting Parameters

	Running Avg		PFA		
	Prior Avg	Start	β	γ	ρ
Min	0.00	1	-2.4	0.05	-1.25
Steps	21	21	9	9	9
Max	1.00	101	2.4	1.25	-0.05

hours, save for RKT-SR, which takes about 5 times as long as RKT to run, and takes 10 times as long to fit using our grid search. We determined that lowering the resolution any more would make fitting ineffective. We expect that the real running time could be greatly improved through code tweaks and by using a more efficient implementation language.

7. RESULTS

The results of the optimistic and conservative value iteration are largely equivalent, with every model predicting a little better on the optimistic value iteration, including the running average. This is likely because the optimistic value iteration favors the most frequently used options more than conservative value iteration.

Stock-BKT, the standard *how* BKT, performed worse then any other model across the board. This implies that tactical knowledge is more important then procedural knowledge in this domain. Surprisingly, removing all error observations does not change the performance of Stock-BKT relative to the other models.

ActualSkill-BKT does slightly worse then a running average, as does PFA, but IdealApp-BKT, which reinforces the ideal skill alone, performs better, trading blows with the running average. This suggests that using the wrong skill is more an indication that the right skill is not known, rather than that the used skill is unknown. Ultimately it appears that they are more important together, this is supported by the fact that IdealActual-BKT outperforms both the other models and the running average.

BKT-SR does not perform as well as its *when* sub-component, IdealActual-BKT. In fact, when we include errors in our data set, BKT-SR does significantly worse. The fact that including errors did not help Stock-BKT or BKT-SR was a surprise. This seems to suggest that failing to use a skill correctly does not always stem from not knowing that skill. We suggest that this is actually just noise from random guesses. When looking at individual records, we find that this is consistent with what we have seen in the logs. There we find long stretches where students solve problems in order followed by bursts of failed skill applications. Thus the extra noise in the *how* component of BKT-SR hurts the model.

But, if we compare the more informed models, RKT and RKT-SR, we get a better picture. RKT-SR is the best performing model across the board with RKT second in terms of AUC, and IdealActual-BKT second in RMSE. RKT and RKT-SR incorporate more then just the ideal option, their predictions incorporate all of the other skills into the probabilities. Thus in BKT terms, the guess and slip are not constant, and they depend upon the other options and upon how good the student is with them. In line with this, RKT and RKT-SR reinforces every applicable skill, not just a few.

Both RKT and RKT-SR assume that the options are ordered, the conceptual difference is that RKT does not distinguish between procedural and tactical knowledge. That is enough to outdo all our other models (except RKT-SR) in terms of AUC. Unlike our simpler models, incorporating both how and when information further improves performance, as RKT-SR outperforms RKT. So *when* and *how* are both different and useful concepts, but separating them takes a little more effort then BKT-SR.

8. CONCLUSIONS & FUTURE WORK

Open-ended tutoring systems are designed to teach students not only how to apply a skill but when to do so. Classical student modeling approaches, however, have focused entirely on procedural knowledge and generally ignore tactical information. In practice it is often difficult to assess whether or not students

are gaining this tactical knowledge and prior studies have either assumed it or have been content to conflate the two.

In this paper we address this lack of information in two ways. First we sought to visually inspect improvements in tactical knowledge. We found that for real student data there is no clear or statistically significant indication of improvement. We therefore opted to develop novel student models that incorporate this information and then to assess their performance on real student data.

In future work we plan to enhance the structure of both our experimental and baseline models. Since this project started, there have been a number of interesting extensions to BKT, such as adding forgetting, and latent student abilities [6]. We did not implement these extensions, but they should be directly applicable to this context, as well as to RKT and RKT-SR.

Additionally, Deep thought originally implemented interaction networks for the purposes of hint generation [15]. Later improvements saw worked examples incorporated into it [10]. This significantly effected student behaviour. Since none of our models integrate contextual data, we restricted our data to the students that saw no worked examples. In future, we may modify the update for the model to incorporate the worked examples. This integration of contextual information has been done before [13], but in this case it is probably more accurate to apply a transition probability.

Many interactive tutors have solutions that can be expressed as an interaction network and thus can be used with these methods. These include Andes [18], and Pyrenees [2]. We will seek to generalize these results by testing them on datasets collected from these tools.

RKT and RKT-SR are new models which make strong assumptions. In future work we will reevaluate the behavior of these models and the underlying assumptions that they make. RKT, for example, assumes that quality is ranked, but removing that assumption could change the model significantly.

RKT gives a valid probability distribution over all options, but we have only tested its accuracy in predicting whether the ideal action is used. We did not test whether or not it was accurate at predicting which of the other actions would be used. This is believed to be an advantage of RKT, but we have not verified that.

9. ACKNOWLEDGMENTS

This research was supported in part by the Provost's Professional Experience Program (PEP) at North Carolina State University.

10. REFERENCES

[1] R. Bellman. A Markovian Decision Process. *Journal of Mathematics and Mechanics*, 6, 1957.

[2] M. Chi and K. VanLehn. Meta-cognitive strategy instruction in intelligent tutoring systems: How, when, and why. *Educational Technology & Society*, pages 25–147, 2010.

[3] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4:253–278, 1995.

[4] M. Eagle, M. Johnson, and T. Barnes. Interaction networks: Generating high level hints based on network community clustering. In *Proceedings of the fifth international conference on educational data mining*, pages 164–167, 2012.

[5] Y. Gong, J. E. Beck, and N. T. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting. In *Intelligent Tutoring Systems: 10th International Conference*, pages 35–44, 2010.

[6] M. M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 94–101, 2016.

[7] J. E. Marsden and M. J. Hoffman. *Elementary Classical Analysis*. W.H. Freeman and Company, 1993.

[8] B. Mostafavi and T. Barnes. Exploring the impact of data-driven tutoring methods on students' demonstrative knowledge in logic problem solving. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 460–465, 2016.

[9] B. Mostafavi, Z. Liu, and T. Barnes. Data-driven proficiency profiling. In *Proceedings of the 8th International Conference on Educational Data Mining*, pages 335–341, 2015.

[10] B. Mostafavi, G. Zhou, C. Lynch, M. Chi, and T. Barnes. Data-driven worked examples improve retention and completion in a logic tutor. In *17th International Conference on Artificial Intelligence in Education*, pages 726–729, 2015.

[11] Philip I. Pavlik Jr., H. Cen, and K. R. Koedinger. Performance factors analysis âĂŞ a new alternative to knowledge tracing. In *Proceedings of the 2009 conference on Artificial Intelligence in Education*, pages 531–538, 2009.

[12] V. Sheshadri, C. Lynch, and T. Barnes. Invis: An edm tool for graphical rendering and analysis of student interaction data. In *EDM 2014 (G-EDM 2014: Workshop on Graph-based Educational Data Mining)*, pages 65–69, 2014.

[13] 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 knowledge tracing. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, pages 406–415, 2008.

[14] 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. In *Proceedings of the 19th international conference on User modeling*, pages 13–24, 2011.

[15] J. C. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental evaluation of automatic hint generation for a logic tutor. In *Artificial Intelligence in Education*, pages 345–352, 2011.

[16] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

[17] B. Van De Sande. Properties of the bayesian knowledge tracing model. *Journal of Educational Data Mining*, 5(2):253–278, 2013.

[18] 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. *International Journal of Artificial Intelligence in Education*, pages 147–204, 2005.

Evaluation of a Data-driven Feedback Algorithm for Open-ended Programming

Thomas Price
North Carolina State Univ.
Raleigh, NC, USA
twprice@ncsu.edu

Rui Zhi
North Carolina State Univ.
Raleigh, NC, USA
rzhi@ncsu.edu

Tiffany Barnes
North Carolina State Univ.
Raleigh, NC, USA
tmbarnes@ncsu.edu

ABSTRACT

In this paper we present a novel, data-driven algorithm for generating feedback for students on open-ended programming problems. The feedback goes beyond next-step hints, annotating a student's whole program with suggested edits, including code that should be moved or reordered. We also build on existing work to design a methodology for evaluating this feedback in comparison to human tutor feedback, using a dataset of real student help requests. Our results suggest that our algorithm is capable of reproducing ideal human tutor edits almost as frequently as another human tutor. However, our algorithm also suggests many edits that are not supported by human tutors, indicating the need for better feedback selection.

1. INTRODUCTION AND BACKGROUND

A hallmark of Intelligent Tutoring Systems (ITSs) is their ability to support learners with adaptive feedback as they work on problem solving tasks. In the domain of open-ended computer programming, much research has addressed how this feedback can be generated automatically using reference solutions [11] or data-driven methods [5, 6, 9]. However, existing techniques (including our own work [6]) have two notable limitations: the type of feedback they can provide and the methods with which they are evaluated.

Existing work has focused almost exclusively on generating next-step hints, suggesting how a student can proceed if they get stuck. Next-step hints make sense in the context of a structured problem-solving task, with well-defined, discrete steps, but they may not always be appropriate in an open-ended programming context. Students may request help for other reasons, such as to verify that code they have written is correct, or to help find a bug in code that does not produce correct output. A more comprehensive feedback generation algorithm is needed to address these concerns. In this work, we present SourceCheck, a novel feedback generation algorithm that builds on existing work to check over a student's whole program, suggesting useful edits throughout.

While extensive effort has been put into the generation of feedback for programming, efforts to evaluate the quality of this feedback are still underdeveloped. Most existing evaluations are either technical evaluations that focus on how often hints can be generated and theoretical hint quality (e.g. [6, 9, 11]) or small classroom studies that use case studies (e.g. [7]). Ideally, we would employ controlled studies to evaluate the impact of feedback on students' course outcomes, as was done by Stamper et al. in their evaluation of data-driven hints in the Deep Thought logic tutor [10]. However, recent work suggests that programming hints can vary widely in quality and that low-quality hints may deter students from later asking for help when they need it [8]. A meaningful first step would therefore be to better understand and evaluate the quality of the feedback we generate. Piech et al. [5] suggest evaluating automatically generated hints for programming by comparing them to "gold standard," expert-authored hints. We build on this method to evaluate our feedback algorithm, comparing it to human-authored feedback.

Our initial results show that SourceCheck's feedback has good overlap with that from human tutors. However, SourceCheck also produces much more feedback than human tutors, and much of this feedback is not represented in human tutor feedback. This suggests that SourceCheck has good potential but that more work is needed to select targeted feedback from potential suggestions.

2. FEEDBACK GENERATION

At a high level, SourceCheck works on a simple premise. To generate feedback for a student on a given assignment, we use a two-step process. First, in the *Solution Matching* step, we look at previously submitted, correct student solutions for that assignment and select the one that best matches that student's code. Then, in the *Edit Inference* step, we extract the edits that separate the student's code from the correct solution and present these as feedback. This idea dates back to the original Hint Factory [1] and was successfully implemented by Rivers and Koedinger for programming hints [9]. Rather than changing the fundamentals of this idea, we present techniques for improving both steps of the process. These improvements center on the understanding that students' solutions are diverse and often include much correct code that does not directly match a known solution because of small changes in structure. SourceCheck attempts to make use of this code, and can suggest *moving* code in addition to inserting and deleting it.

SourceCheck takes as input a set of complete, correct prior student solutions for an assignment and a snapshot of code from a new student requesting a hint. As in previous work, we represent both as an abstract syntax tree (AST), a directed, rooted tree where each node is labeled to represent 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. To each AST we apply simple canonicalization to reduce syntactic complexity while preserving semantic meaning, as described in [6]. SourceCheck outputs a set of edits, (insertions, deletions, moves and reorders) that can be used to annotate the student's code with feedback. While this feedback can include next steps hints in the form of insertions, it also highlights potential errors and provides reassurance that unannotated code is likely correct.

2.1 Solution Matching

Most hint generation algorithms for programming select a goal solution by finding the "closest" solution to the student's current code, determined by some distance metric. Researchers have used string edit distance [9] and approximations of tree edit distance [11], though more complex metrics have been proposed [4]. The problem with edit distances, however, is that they heavily penalize differences in the position of code fragments [3, 4]. For example, swapping the order of two independent subroutines in a program does not affect its semantic meaning, but this movement is treated as a large set of deletions and insertions by edit distance algorithms.

Mokbel et al. suggest addressing this by fragmenting each AST into subgraphs, pairing similar subgraphs from the two ASTs, and computing their distance independently [3]. We build on this idea, along with our previous work decomposing ASTs using root paths [6], to produce a distance metric designed specifically for code. The *root path* of a node n in an AST is the sequence of node labels on the path from the root of the AST to n. Multiple nodes in an AST will have the same root path if they and each of their respective ancestors have matching labels, such as two calls to the same function in the same block of code.

Given ASTs A and B, consisting of nodes $\{a_1, \ldots, a_{|A|}\}$ and $\{b_1, \ldots, b_{|B|}\}$ respectively, SourceCheck produces a matching, $M = \{[a_i, b_j], \ldots\}$, pairing nodes from A to nodes from B, and a cost C for the mapping. Nodes can only appear in one pair, and some nodes may be left unmatched. First, we iterate over each root path in A, from shortest to longest path. For a given root path r, let A_r and B_r be the set of nodes in A and B respectively with root path r. Let us define $c(n)$ as the child-sequence of n, or the sequence of node labels of the immediate children of n. For each pair of nodes $a_{ri} \in A_r$ and $b_{rj} \in B_r$, we compute the pairwise distance between their child-sequences, $d(c(a_{ri}), c(b_{rj}))$. This is used to match nodes with the same root path and similar children.

For the distance function d, we could use a string edit distance, such as Levenshtein distance, since AST child-sequences are just sequences of node labels. However, SourceCheck is designed to match incomplete student code (A) to complete solutions (B), so for d we use a "progress" function that measures how much of $c(a_{ri})$ represents progress towards $c(b_{rj})$. Our progress function is similar to an edit distance, but it is intentionally asymmetrical and penalizes deletions (student's code not found in the solution) much more than insertions (solution code not yet found in the student's code). Additionally, our progress function identifies insertion/deletion pairs with the same label and treats these as a "reorder", which has a much lower cost, distinguishing between code that should be deleted and code that is out of order.

SourceCheck calculates the pairwise distances for all nodes in A_r and B_r and then uses the Hungarian algorithm to select the set of pairs of minimum total cost, which it adds to the mapping, M. This cost is added to C. This procedure is performed for each root path in A to determine the total mapping and cost. To select a target solution T for a student's current code S, SourceCheck simply finds the solution with the minimum mapping cost. The result is a target solution that maximizes the number of nodes in the student's code which can be reasonably mapped to nodes in the target solution.

2.2 Edit Inference

Once a target solution T has been identified for a student's code S, SourceCheck identifies a set of edits that can bring the student closer to this solution. In previous work, this is accomplished by selecting the top-level applicable edit [9] or following edits from previous students [6]. Instead, we use the mapping M between the student's AST and the target solution to calculate a more precise set of edits between S and T. These edits take the form of Moves and Reorders, along with traditional Insertions and Deletions, determined as follows:

Deletions: First, all nodes $s \in S$ without a pair in M are marked for deletion; however, these nodes may be reused in the final step of the algorithm.

Moves: Next, we consider all pairs $[s_i, t_i] \in M$. Let $P(n)$ denote the parent of the node n in its AST. If $[P(s_i), P(t_i)] \notin M$, this means s_i is under the wrong parent node, so we mark s_i to be moved under p, where $[p, P(t_i)] \in M$, at an index corresponding to that of t_i. If no such p exists, this means that the appropriate parent has not yet been added to S. We still mark s_i for movement, but we cannot specify a destination.

Reorders: Next, we ensure that the children of s_i are in the correct order. We do this by identifying the set of matching child pairs $[c_s, c_t] \in M$ such that $P(c_s) = s_i$ and $P(c_t) = t_i$. For each node c_s, if the node's index among its siblings is different than that of its pair, c_t, we mark it for reordering.

Insertions: Any node $t \in T$ which has no pair in S is marked for insertion. If $P(t)$ has a pair in S, this pair is used as a parent. If $P(t)$ has no pair in S, we do not yet have a place to insert t. We still mark t for insertion, since it may be useful in the next step.

Combining Insertions and Deletions: If a node is deleted in one place and a node with the same label is inserted in another, this may actually represent a Move or Reorder. We identify pairs of Deletions and Insertions with the same label and replace these with an appropriate Move or Reorder. This is a key feature of SourceCheck that encourages a student to use existing code, rather than deleting and re-inserting it.

Using the mapping M, SourceCheck is able to infer more semantically meaningful edits, such as Moves and Reorders,

which convey more information than their component insertions and deletions would alone. The Deletions that remain indicate likely errors to the students, and the Moves and Reorders suggest areas in need of editing. Any node not marked with an edit has been "checked" and likely represents correct code.

3. METHODS

Our evaluation focuses on measuring the quality and appropriateness of SourceCheck's feedback by comparing it to human tutor feedback. This is in contrast to previous technical evaluations [6, 9] that used theoretical measures of hint quality and availability. Instead, we extend the work of Piech et al., who assessed feedback quality by comparing hint policies with "gold standard," human-authored, expert hints on small, constrained programming problems (4-6 lines of code for an ideal solution) [5]. However, for the more complex problems we investigate here (about twice as many lines of code), we argue that it is not realistic to define a single best "gold standard" hint for a given code snapshot. There may be many useful ways a tutor can advise a student, so it is more reasonable to measure the similarity of human and algorithmic feedback, rather than whether they match exactly. We build on the "gold standard" method to compare the feedback of human tutors and SourceCheck in a more nuanced way. We focus on the following research questions:

RQ1: How well does SourceCheck's feedback agree with ideal human tutor feedback?

RQ2: How does the agreement between SourceCheck and a human tutor compare to the agreement between human tutors?

We evaluated SourceCheck in the context of an introductory computing course for non-majors, consisting of 51 students, held at a research university during the Spring 2017 semester. During the first half of the course, undergraduate teaching assistants (TAs) facilitated Snap*!* programming labs derived from the Beauty and Joy of Computing (BJC) AP Computer Science Principles curriculum [2] (available at bjc.edc.org). The course includes three in-lab programming assignments, completed with TA help available, interleaved with three homework assignments, completed independently. Students programmed using iSnap[1] [7], an extension of the block-based, novice programming environment Snap*!* [2]. iSnap supports students working on open-ended assignments with data-driven, on-demand hints [6].

We selected one homework assignment (Squiral – SQ) and one in-lab assignment (The Guessing Game – GG) for analysis. In SQ, students draw a square-shaped spiral using loops, variables and a custom block (function), and a typical solution is around 10 lines of code. In GG, students create a simple game in which the player must guess a random number using loops, variables, conditionals and user input, and a typical solution is around 13 lines of code. We built a dataset of student hint requests on GG and SQ to serve as authentic scenarios for evaluating SourceCheck. We sampled up to two hint requests from each student. Where possible we sampled one request from the first half of their working time and one from the second half to avoid overly similar samples. We also ensured that at least 30 seconds and one code edit occurred between sampled hint requests. We sampled hints from 14 and 15 students on SQ and GG for a total of 22 and 29 hints respectively, 51 altogether.

[1]Demo and datasets available at http://go.ncsu.edu/isnap

3.1 Human Feedback Generation

For each hint request, we extracted a snapshot of the student's code at the time of the request. Importantly, these snapshots represent code for which real students requested help, making them an ideal sample on which to evaluate SourceCheck. We did so using a post hoc Wizard-of-Oz-style experiment. The first two authors, who were familiar with the assignments and context, acted as human tutors and manually generated feedback for each selected snapshot. The two tutors were graduate students in Computer Science who were domain experts but not teaching experts, making them similar to most course TAs for advanced computing courses.

When generating feedback, the human tutors attempted to offer pedagogically useful feedback, but they were limited to communicating their feedback using the edits defined in Section 2.2. These edits also had to be independent, meaning no edit could be dependent on the student following another suggested edit (e.g. suggesting inserting both a for-loop and the body of the loop). Tutors crafted these edits with the understanding that the edits would be (theoretically) presented to students without further explanation or any guarantee of further feedback requests. These limitations forced the human tutors to generate feedback that could be provided through the same user interface that SourceCheck would use, as in a Wizard-of-Oz experiment, allowing us to directly compare human and algorithmic feedback. Tutors generated their feedback based on the student's current code at the time of the hint request, using previous snapshots of the student's code for context. However, tutors did not have access to a student's code *after* the hint request or the student's final solution. While the two tutors generated feedback independently, they first practiced on a dataset with the same assignments from another semester and compared results to ensure a consistent understanding of the feedback guidelines. The tutors generated feedback in a two-phase process:

Phase I: Tutors identified the edit(s) they would recommend to best support the student's current goal and promote learning. The edit(s) should convey a single idea.

Phase II: Tutors envisioned a correct solution that most closely matched the student's current code and identified *all* edits that would bring the student closer to this solution.

Phase I allows us to measure how well the algorithm reproduces ideal, targeted tutor feedback, addressing RQ1. In Phase II, tutors generate a large set of all applicable edits, just as SourceCheck does, allowing us to directly compare algorithmic and human feedback, addressing RQ2.

4. ANALYSIS AND RESULTS

To quantify the overlap between two sets of feedback for a given snapshot, we define feedback generation as the process of labeling each node of an AST with an edit (Delete, Reorder, Move or nothing) and generating a set of Insertions

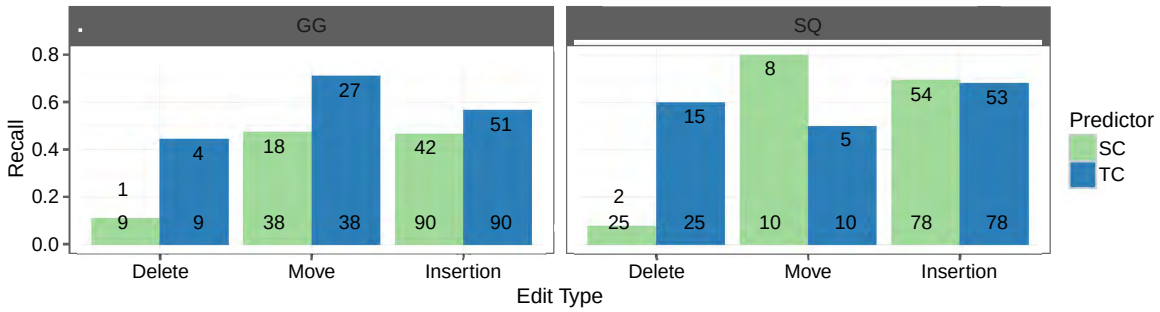


Figure 1: The percentage of tutor Phase I edits predicted by SourceCheck (SC) and TC (the recall) for each edit type on the GG and SQ assignments. Bars are labeled with the total number of Phase I edits of each type (bottom) and the number of correctly labeled edits (top).

(each consisting of a type of node to insert and an index in the AST at which to insert it). Under this definition, we can treat SourceCheck as a classifier and evaluate its ability to predict the feedback provided by tutors. We measure classification success for each type of edit separately, treating it as a binary classification task. For Deletions and Moves, we consider each node of each snapshot in our dataset to be a classification instance, where both the tutors and SourceCheck have labeled the node. Successful classification occurs when SourceCheck produces the same label as the tutor. Each Insertion provided by either the tutors or SourceCheck for a given snapshot is also considered a classification instance, where both the tutors and SourceCheck have either included or not included that Insertion. Since Reorders were rarely suggested by SourceCheck and were never suggested by tutors, we exclude them from analysis.

Treating feedback generation as a classification task allows us to address RQ1 and evaluate the extent to which SourceCheck agrees with (predicts) the feedback of human tutors. The results of this evaluation would be difficult to interpret without a baseline for comparison. Therefore, we also define a "Tutor Classifier" (TC), which predicts feedback from Tutor 1 using the feedback collected in Phase II from Tutor 2, and vice versa. Since tutors generate a full set of applicable edits in Phase II, just like SourceCheck, we can directly compare the SourceCheck and TC classifiers. This allows us to address RQ2, comparing the agreement of human and algorithmic feedback with that of two humans. We would not generally expect an algorithm to predict human tutor feedback better than it would be predicted by another tutor, so TC provides a high performance target.

4.1 Results

We first look at predicting the targeted feedback that tutors provided in Phase I. Figure 1 shows the percentage of the tutor edits that were also generated by SourceCheck and TC, or the recall of both predictors. We did not observe large differences between prediction success for edits generated by the two human tutors, so we report their results in aggregate. While Deletions were fairly rare, SourceCheck performs quite poorly at predicting them on both assignments. However, SourceCheck predicts 46% and 47% of tutor Moves and Insertions respectively on GG, and 69% and 80% of Moves and Insertions for SQ, where it even outperforms TC. Totalling all edits, SourceCheck had a recall of 0.45 and 0.57 on GG and SQ respectively, while TC achieved 0.59 and 0.65.

An important limitation of recall is that it only considers how many of the tutor edits were successfully predicted, and not how many "guesses" (suggested edits), it took to do so. To understand how much of SourceCheck's feedback agrees with tutor feedback, we must compare it against all tutor edits collected in Phase II. Figure 2 shows the recall (top) for SourceCheck and TC over all Phase II edits, as well as the precision (bottom), or the percentage of SourceCheck and TC edits that agreed with human tutor edits. We see very similar trends for recall across Phases I and II, implying that both SourceCheck and TC predict "ideal" (Phase I) and "possible" (Phase II) edits at similar rates. Totalling all Phase II edits, SourceCheck had a recall of 0.41 and 0.41 on GG and SQ respectively, while TC achieved 0.57 and 0.54.

However, SourceCheck's precision is much lower, particularly for GG, where SourceCheck suggests more of every type of edit, for a total of over 50% more suggested edits. SourceCheck generated on average 10.7 and 6.4 edits per snapshot on GG and SQ respectively, compared to 6.3 and 5.2 edits per snapshot for the tutors' Phase II edits. Despite this low precision, SourceCheck is not simply suggesting edits everywhere in the code and getting a few correct by chance. It correctly suggests no edit for 1092/1238 (88%) of GG AST nodes where the tutors also did not suggest an edit in Phase II and for 662/703 (94%) of SQ nodes. Totalling all edits, SourceCheck had a precision of 0.27 and 0.38 on GG and SQ respectively, while TC achieved 0.57 and 0.54[2].

4.2 A Closer Look

We manually investigated edits on which the human tutors and SourceCheck disagreed, and in this section we present some common causes of disagreement:

Variables: We noticed that many disagreements were over variable assignments and references. For example, most of

[2]Note that because with TC, Tutor 1 predicts Tutor 2 and vice versa, the precision and recall of TC in Phase II will be the same, and this value indicates percent agreement.

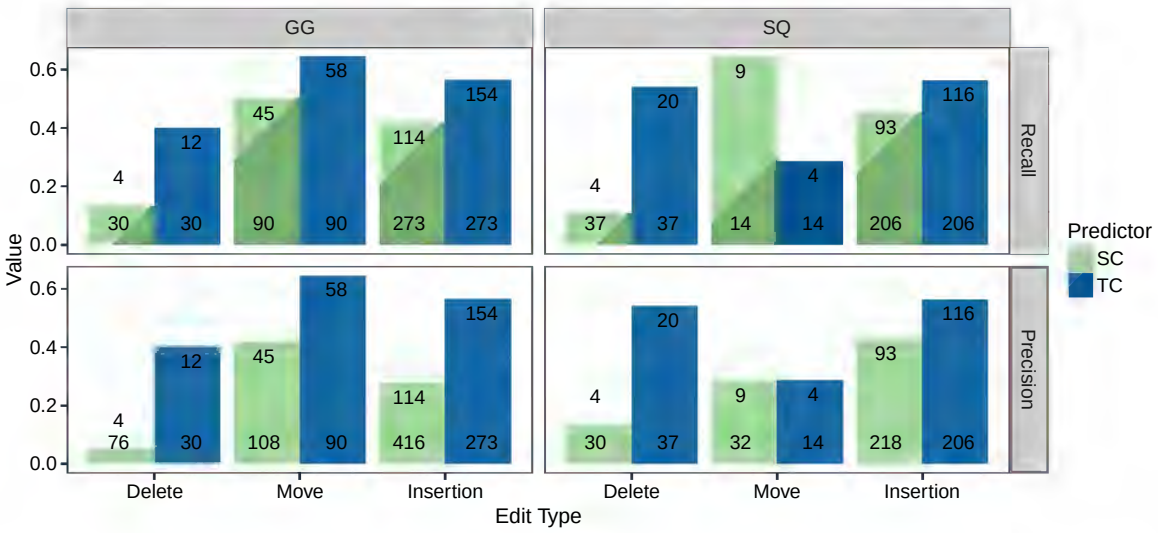


Figure 2: The recall for Phase II edits (top), as well the the percentage of SourceCheck (SC) and TC edits that agreed with a human tutor (the precision, bottom).

the Phase I deletions that SourceCheck failed to predict were instances of a tutor deleting a variable, such when a student used the wrong variable in an expression. This is largely due to SourceCheck's canonicalization process [6], which currently gives all variables the same label, making them indistinguishable. This simplification makes code matching easier, but clearly a more robust solution is needed.

Supporting Unusual Code: Many times, a tutor suggested an edit, such as deleting an unneeded control structure, that would lead the student away from a potentially confusing program state. In many of these cases, SourceCheck found *some solution* which used this unusual code *correctly* and instead suggested how the student could do the same. We view this behavior as a design choice, rather than a flaw per se, but it is worth investigating when this behavior would lead to its intended effect of supporting unconventional solutions, and when it would lead to confusion.

Code Variability: The assignments we analyzed were complex enough to allow the student to make a number of small design choices, such how to reset the sprite and canvas before drawing the "Squiral" in the SQ assignment. Often, the tutor and the target solution chosen by SourceCheck made different, correct suggestions. This also occurred between human tutors, emphasizing that disagreement with the tutors does not always indicate poor feedback.

Human Traits: Sometimes the human tutors were able to infer information from natural language in a student's code that influenced their feedback in a way that would not be possible for SourceCheck. For example, the name of a variable might imply how it is intended to be used (e.g. "randomNumber"). This sometimes led to very different edits from SourceCheck and the human tutors. On the other hand, humans are also capable of making careless errors, and our tutors sometimes simply forgot to suggest a small, useful edit in Phase II, which SourceCheck remembered.

5. DISCUSSION

RQ1: *How well does SourceCheck's feedback agree with ideal human tutor feedback?* SourceCheck agrees with approximately half of ideal tutor feedback provided in Phase I, almost as much as another human tutor, with SourceCheck achieving a recall 76% and 88% as high as TC on GG and SQ respectively. This does not necessarily mean that SourceCheck's feedback is almost *as good* as a tutor's. It is possible that when SourceCheck's feedback diverges from a tutor's, it does so in a less useful way than when another tutor does so; however, this is difficult to investigate without some direct measure of hint quality (e.g. [8]). For now, we can say that these results suggest good potential for data-driven feedback generation, in that ideal tutor feedback is frequently contained in the set of edits generated by SourceCheck.

RQ2: *How does the agreement between SourceCheck and a human tutor compare to the agreement between human tutors?* Our results for RQ2 are mixed. In Phase II, SourceCheck was 72-76% as likely to agree with a given tutor's edit as another tutor was on GG and SQ (as measured by recall). However, a given tutor was only 47-70% as likely to agree with SourceCheck's edit as with another tutor's edit (as measured by precision). This is largely because SourceCheck generated more total edits than the tutors did, especially on GG. This lack of precision seems to be the largest difference between SourceCheck and human tutors. Even if SourceCheck can produce quality feedback, the benefit to the student might be lost if it is hidden among less useful suggestions. Additionally, recent work suggests that students seek less help after receiving poor quality hints [8]. A critical direction for future research will be how to *select* feedback once a set of possible edits has been generated.

It is also worth noting that our two human tutors had relatively low agreement. Comparing all suggested Phase II edits, we see that they have a 54% and 57% agreement on

the GG and SQ assignments respectively. In fact, tutors only agreed completely on 8 out of 22 SQ snapshots (36%) and 7 out of 29 GG snapshots (24%) in Phase I. This suggests the assignments we studied truly are open-ended, since tutors often disagreed on the best path forward, though we cannot make any strong claims using our human data because it was generated by the authors. This supports our choice to measure agreement using the similarity of edits, rather than using a single, best "gold standard" hint, as was done by Piech et al. on simpler assignments [5].

6. CONCLUSION

In this work, we have presented SourceCheck, an algorithm for automatically generating data-driven feedback for students working on open-ended programming problems. SourceCheck builds on existing methods [3, 9] to improve the processes of selecting a target solution from a set of correct solutions and inferring edits to get the student to that solution. It does so with a code-specific matching function and more semantically meaningful suggested edits: Moves and Reorders. We have also presented a method for evaluating automatically generated feedback by comparing it to feedback generated by human tutors playing the same role. We extend existing methods [5] by using a dataset of real student help requests to ensure authenticity and by formulating the problem as a prediction task, allowing us to compare the similarity among an algorithm and multiple human tutors. This allows us to envision the high standard of an algorithm as similar to human tutors are they are to each other. We show that SourceCheck approaches this target in some ways and falls well short in others.

Based on our results, iSnap has been updated to include SourceCheck feedback, and we envision a number of practical application for the algorithm. In busy classrooms, large-scale MOOCs and informal learning settings, instructors are often absent or unavailable. The on-demand feedback provided by SourceCheck can keep students going when they get stuck and would otherwise give up. SourceCheck could also be used to identify potential struggling students in real-time, based on their distance to a known solution. Both SourceCheck and our evaluation methodology were designed to scale to the larger, more complex programming problems found in real classrooms. This will require SourceCheck to support a greater diversity of student code, which will require a larger dataset of correct solutions for matching.

This work also has clear limitations. We only used two tutors to generate human feedback, and the authors who served as tutors were not pedagogical experts and had limited teaching experience. While their experience is on par with many graduate computing TAs, results may be different with experienced teachers. Additionally, despite efforts at objectivity, the tutors' familiarity with each other and with SourceCheck may have biased their feedback. Our work is also limited by the small sample of assignments and hint requests we investigated, especially given that our results were quite different for GG and SQ. Finally, the methods presented here do not lend themselves to traditional statistical testing, making it difficult to make claims about true differences in recall and precision. Our methods only speak to the relative similarity of algorithmic and human tutor feedback, but this does not directly assess feedback quality.

This work opens many avenues for future work. Our results suggest a number of ways SourceCheck could be improved, such as a method for selecting which of the generated edits are most useful to show the student. Future work could also explore how to expand data-driven ITS feedback for programming beyond edit-based hints, towards richer descriptions or explanations. Our results also raise questions about the consistency of human feedback on open-ended programming problems, and future work should determine how much agreement can be expected among human tutor feedback. Lastly, the methods presented here can be used to evaluate, compare and benchmark other feedback generation techniques, giving researchers a better understanding of their strengths and weaknesses.

7. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under grant 1623470.

8. REFERENCES

[1] T. Barnes and J. Stamper. Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data. In *Proc. of Int. Conf. on Intelligent Tutoring Systems*, pages 373–382, 2008.

[2] D. Garcia, B. Harvey, and T. Barnes. The Beauty and Joy of Computing. *ACM Inroads*, 6(4):71–79, 2015.

[3] B. Mokbel, S. Gross, B. Paaßen, N. Pinkwart, and B. Hammer. Domain-independent proximity measures in intelligent tutoring systems. In *Proc. of Int. Conf. on Educational Data Mining*, 2013.

[4] B. Paaßen, B. Mokbel, and B. Hammer. Adaptive Structure Metrics for Automated Feedback Provision in Java Programming. In *European Symp. on Artificial Neural Networks, Computational Intelligence and Machine Learning*, page 312, 2015.

[5] C. Piech, M. Sahami, J. Huang, and L. Guibas. Autonomously Generating Hints by Inferring Problem Solving Policies. In *Proc. of ACM Conf. on Learning @ Scale*, pages 1–10, 2015.

[6] T. W. Price, Y. Dong, and T. Barnes. Generating Data-driven Hints for Open-ended Programming. In *Proc. of Int. Conf. on Educational Data Mining*, 2016.

[7] T. W. Price, Y. Dong, and D. Lipovac. iSnap: Towards Intelligent Tutoring in Novice Programming Environments. In *Proc. of ACM Technical Symp. on Computer Science Education*, 2017.

[8] T. W. Price, R. Zhi, and T. Barnes. Hint Generation Under Uncertainty: The Effect of Hint Quality on Help-Seeking Behavior. In *Proc. of Int. Conf. on Artificial Intelligence in Education*, 2017.

[9] 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*, 16(1), 2015.

[10] J. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental Evaluation of Automatic Hint Generation for a Logic Tutor. *Int. J. of Artificial Intelligence in Education*, 22(1):3–17, 2013.

[11] K. Zimmerman and C. R. Rupakheti. An Automated Framework for Recommending Program Elements to Novices. In *Proc. of Int. Conf. on Automated Software Engineering*, 2015.

Making the Grade: How Learner Engagement Changes After Passing a Course

David Lang
Stanford University
dnlang@stanford.edu

Alex Kindel
Princeton University
akindel@princeton.edu

Ben Domingue
Stanford University
bdomingu@stanford.edu

Andreas Paepcke
Stanford University
paepcke@stanford.edu

ABSTRACT

Understanding how individuals interact with a course after receiving a passing grade could have important implications for course design. If individuals become disengaged after passing a class, then this may raise questions about optimal ordering of content, course difficulty, and grade transparency. Using a person-fixed effects model, we analyze how individuals who obtained passing grades subsequently behaved within a course. These learners were less likely to complete videos and more likely to watch videos faster after receiving notice of a passing grade in the class. These learners were also less likely to reattempt items they initially got wrong.

Keywords

Video-interactions; grading schemes; learning analytics; MOOCs;

1. INTRODUCTION

Grades are a key component of online courses. However, there is a great deal of heterogeneity in the downstream effects of grading and grading schemes. For instance, female students who received an 'A' in their introductory economics courses were substantially more likely to major in the subject than individuals who received a B but had similar scores in the class [1]. On the other end of the spectrum, research suggests that pass-fail grading schemes may be beneficial in terms of student stress in high-stakes environments [2]. Other work suggests that the presence of pass-fail grading discourages student performance[3].

MOOCs offer a unique opportunity to understand how grading affects within-course behavior. First, clickstream data can document subtle changes in behavior that are reasonable proxies for engagement and effort (e.g. video consumption, video interactions, multiple attempts on items). Second, compared to traditional courses, grading in MOOCs is much more salient and immediate. Grades are recomputed instantaneously, and solutions are presented after every single problem.

Understanding how individuals interact with a course after receiving a passing grade could have important implications for course design. If individuals disengage after passing a class, then it may make sense to structure a course such that final grades are not revealed until all problems have been attempted. Alternatively, if individuals exert more effort in a class after reaching passing status, then perhaps courses should be designed with gamification/scaffolding in mind such that a learner is continually working for a new certificate/badge.

2. DATA

The dataset used in this analysis was an introductory course in Statistical Learning administered multiple times via Stanford's Lagunita Platform. 55,000 individuals enrolled in the class. Of that population, 11,301 individuals interacted with both course videos and with assignments related to the course at least once. Of these individuals, 2,485 achieved certification.

The course includes 77 videos. The cumulative length of these videos is 15.3 hours. We used the clickstream created by learners who viewed the course via the Lagunita platform. Clickstream events are generated each time a video is loaded, finished, played or paused, fast forwarded or rewound. Other clickstream activities include changes to the media player's playback speed to one of six settings (0.5X, .75X, 1.0X, 1.25X,1.5X, and 2.0X). These activities were aggregated on a user-video level. In total, there were 126,799 learner-video observations.

2.1 Course Items

The course assignments consisted of 103 multiple-choice, short-responses, and fill-in-the-blank items. Learners who answered at least 50% of all items correctly received a certificate. Individuals who obtained a score of 90% or more received a certificate of distinction. We limited the dataset to include only individuals who attempted at least a simple majority of items.

3. ANALYSIS

In this course, learners are keenly aware of the grading cutpoints. The distribution of learners' scores show substantial jumps in density at just above 50% and just above 90% (red lines), as seen in Figure 1. In an educational context, such jumps usually indicate a bias on the part of graders to give students with marginal scores the benefit of the doubt [4]. In this instance, though, all exams are graded electronically, and this type of manipulation by a grader is not possible. Instead, this heaping likely reflects a subset of learners who are extremely motivated by the certificate, and cease attempts after obtaining it. In this case, we identified that approximately 5% of students stopped attempting items shortly after they hit the 50% threshold. Formal evaluation via the McCrary Density[1] tests rejects continuity of the density function

[1] The McCrary Density Test estimates the continuity of exam scores at the cutoff using local linear regression. If the left and

at the cutoff scores with a t-statistic of 7.1 and 5.1 at the 50% percent and 90% percent thresholds [5].

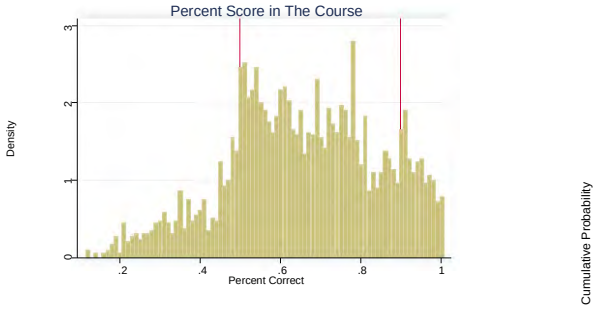


Figure 1 Histogram of Course Scores

Given how pronounced and precise this heaping was, we examined the grade-reporting interface. If a user clicks on a link to their progress, a report is generated with a user's score on each exam, as well as their overall status with the course, indicating whether they have currently passed the course. Figure 2 depicts a mock-up of these reports. There are several noteworthy features of this reporting format. First, these grading thresholds are very clearly identified by their shading. Light grey depicts the region that is considered passing (>50%) and dark grey depicts the region that is considered passing with distinction (>90%). A learner's grade is communicated by their total score bar (right most column). If this bar is at 50% or more, they will be able to observe the top of the total score bar in the light grey region, indicating that they passed. If the total score bar is in the dark grey region, this indicates the learner has earned a certificate of distinction. On top of these features, the total score is computed and displayed in percentages terms, making the learner's grade relative to the passing threshold eminently clear. In this artificial example, the learner obtained a 100% on every item but stopped almost immediately after obtaining a passing grade in the course.

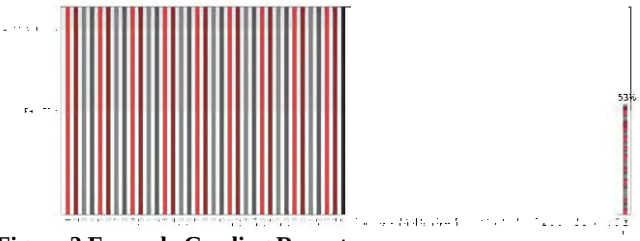


Figure 2 Example Grading Report

This reporting format could help explain the popularity of grade checking behavior in the course. Ninety-eight percent of learners checked their grades at least once, and the median user checked their grade 32 times during the course of the class.

3.1 When Passing Occurs

There is considerable variation in when a learner passes a course. Our identification strategy leverages within-learner variation before and after they became aware they passed the course. Figure 3 shows that of the 2,485 learners who passed the course, the median individual tends to do so within the first 70 items. This leaves almost a third of the course and its items to serve as a behavioral contrast. We also exploit variation of when students become aware they have passed the course. Approximately 70% of individuals, checked their grade on the day that they passed a course. Others realized this information at a later date.

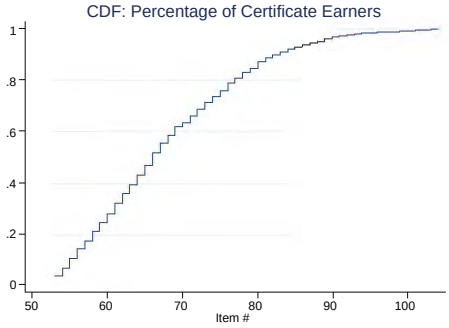


Figure 3 Item on which an Individual Obtains a Certificate

3.2 Impact of Passing on Engagement

We estimate user engagement by analyzing video interactions before and after a learner receives notification that they have passed the course via person fixed-effect regression. The specification is below:

$$UserBehavior_{ij} = B_1 PassNotification_{ij} + \Gamma_i + e_{ij}$$

The Γ_i denotes the person-fixed-effect and the user behavior/pass notification refers to the ith person's performance on their jth video. Outcomes include playback speed, fast forwarding, and video completion. For the purposes of this analysis, we define video completion as a student completing 90% of a video. This threshold was chosen as these videos often contain summaries, production details, and end titles in the last minute or so of content.

Our first analysis suggests that individuals sped up after passing a course. The first column of Table 1 corresponds to a univariate regression model of playback speed on pass notification. The second column corresponds to a regression model of playback speed on pass notification and a person-fixed-effect. The third column also includes a time trend that accounts for how many days a student has been enrolled in a course at the time of their video interactions. After accounting for person-fixed effects, our preferred regression model (Column 2) finds individuals speed up on average about 1%. Given that playback speed has six discrete speeds (0.5X, 0.75X, 1.0X, 1.25X, 1.5X, 2.0X) this speed-up reflects a subset of learners adjusting their playback speed on a subset of videos that they interacted with rather than a gradual shift across all videos. Depending on how early a learner obtained a passing grade for the course, this speedup represents as much as a 10-minute reduction on time spent watching videos over the remainder of the course. In terms of effect size, this increase corresponds to roughly an increase of .05 of a standard deviation.

right hand-side estimates produce substantially different estimates, it would suggest manipulation or selection into one of the two groups.

Table 1 Effect of Pass Notification on Playback Speed

	(1) Univariate	(2) Person Effects	(3) Time Trend
Pass Notice	0.0184***	0.0107***	0.00607**
	(3.68)	(5.17)	(3.04)
Log Days			0.00412***
			(5.29)
Constant	1.080***	1.082***	1.070***
	(298.23)	(2318.28)	(438.87)
Observations	126799	126799	126799
Adjusted R^2	0.002	0.776	0.776

t statistics in parentheses
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Other video behaviors suggest that individuals may be less engaged in the course after receiving certification. Modeling the effect of receiving a passing grade on fast forwarding behavior suggests that passing notification is associated with a 4-5% percentage point reduction in fast forwarding and a 3-4% percentage point reduction in rewinding. A decrease in fast forwarding behavior may be seen as a form of increased engagement by some. However, it should be noted that fast forwarding and rewinding are symmetric actions (The concordance within video between rewinding and fast forwarding is 73%).

When answering a question on an assignment, a very common learner strategy is to review prior material. If a user is searching a video for a particular statement or graph, a learner is unlikely to skip to exactly the right point in time. Even if they were, learners may like to check the immediately preceding and following slides for context or clarifying information. In these cases, one would expect to see both fast forwarding and rewinding. Most of the reduction in rewinding and fast forwarding seems to come from cases like these. In terms of total effect size, these reductions correspond to a .10 reduction in fast forwarding and a .06 reduction in rewinding.

Table 2 Effect of Pass Notification on Fast Forwarding (Top) and Rewinding (Bottom)

	(1) Univariate	(2) Person Effects	(3) Time Trend
Pass Notice	-0.0430***	-0.0419***	-0.0496***
	(-7.59)	(-11.20)	(-12.03)
Log Days			0.00682***
			(4.07)
Constant	0.259***	0.259***	0.239***
	(64.27)	(307.34)	(48.20)
Observations	126799	126799	126799
Adjusted R^2	0.002	0.180	0.181

	(1) Univariate	(2) Person Effects	(3) Time Trend
Pass Notice	-0.0258***	-0.0303***	-0.0418***
	(-4.08)	(-7.05)	(-9.10)
Log Days			0.0102***
			(5.73)
Constant	0.393***	0.394***	0.364***
	(78.88)	(407.29)	(68.39)
Observations	126799	126799	126799
Adjusted R^2	0.000	0.200	0.201

t statistics in parentheses
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

We also note the percentage of videos that are completed decreases after pass notification. Here we find that individuals are less likely to complete videos after passing a course by approximately five percentage points. This corresponds to approximately .15 of a standard deviation.

Table 3 Effect of Pass Notification on Video Completion

	(1) Univariate	(2) Person Effects	(3) Time Trend
Pass Notice	-0.0226***	-0.0498***	-0.0425***
	(-5.12)	(-14.69)	(-11.99)
Log Days			-0.00646***
			(-4.45)
Constant	0.858***	0.864***	0.882***
	(303.85)	(1130.74)	(200.95)
Observations	126799	126799	126799
Adjusted R^2	0.001	0.182	0.182

t statistics in parentheses
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Finally, we examine the number of attempts individuals made to answer items. We find that individuals who passed the course were subsequently less likely to make multiple attempts on incorrect items.[2] Before passing, there were an average of 1.11 attempts. After passing, this declined to 1.07 attempts. This corresponds to an effect size of approximately of .07.

Table 4 Effect of Pass Notification on Item Attempts

	(1) Univariate	(2) Person Effects	(3) Time Trend
Pass Notice	-0.0361***	-0.0394***	-0.0316***
	(-7.60)	(-6.98)	(-4.73)
Log Days			-6.702*
			(-2.41)
Constant	1.114***	1.114***	67.54*
	(427.81)	(1888.66)	(2.45)
Observations	113562	113562	113562
Adjusted R^2	0.001	0.203	0.203

t statistics in parentheses
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

3.3 Limitations to Analysis

This study was conducted on a single MOOC. It should also be noted that this MOOC was a terminal course. This course was not part of a broader sequence and its content was not necessary for other courses available within the platform. A such, our findings that users disengaged in course material after passing the course may not generalize.

4. DISCUSSION

On balance, our findings suggest that passing notification discourages subsequent engagement for at least a subset of users. We see increases in playback speed and less video completion.

These findings are consistent with evidence from the educational psychology and behavioral economics literature, which has suggested that receipt of a certificate or badges can discourage intrinsic motivation in individuals [6][7]. Earlier work in MOOCs

[2] Observations differ in this specification because it is based on person-item level data rather than person-video level data.

also found that individuals who obtain certificates in courses actually skipped nearly a quarter of a course's video content [8]. We have documented several learner behaviors that are relevant to the design of MOOCs, and likely the design of online teaching more generally.

With respect to grading schema, there is substantial evidence that individuals act in a more engaged manner before passing a course than after they have received a notification of passing. We also see this strategic behavior in that there are substantially more students just above the passing threshold than just below it.

One policy implication of these findings is how quickly learners should be notified about their overall success in a course. Currently many courses notify learners instantaneously, daily, or on a near weekly basis when these events occur. For courses with a well-defined end date, it may make sense to not notify users of their final grades until the course is completed.

A second consideration is how transparent instructors should be in terms of grading. Learners could not manipulate their grades as easily if they did not know the exact threshold for passing. Using language that describes *approximate* cutpoints may discourage learners from conflating certification and completion while allowing for more rigorous causal inference.

Lastly there is the question of course structure, if individuals put forth less effort after passing a class, then perhaps a more traditional instructional environment of weekly assignments with a summative final project or exam may yield more total learning.

5. FUTURE STEPS

We found that notification of a passing grade decreased subsequent effort in the *same* course. An equally intriguing question is how individuals who are enrolled in multiple classes behave after this notification. If these individuals are solely interested in accumulation of credentials or certificates, presumably we would see effort shift to courses where learners have yet to obtain certificates.

ACKNOWLEDGMENTS

The research reported here was supported in part by the Institute of Education Sciences, U.S. Department of Education, through Grant (#R305B140009). We also acknowledge the support of Stanford University's CAROL (Center for Advanced Research through Online Learning), the Spencer Foundation, and CEPA (Center for Education and Policy Analysis).

6. REFERENCES

[1] A. Owen, "Grades, gender, and encouragement: A regression discontinuity analysis," *J. Econ. Educ.*, 2010.

[2] D. Rohe, P. Barrier, M. Clark, and D. Cook, "The benefits of pass-fail grading on stress, mood, and group cohesion in medical students," *Mayo Clin.*, 2006.

[3] R. Gold, A. Reilly, and R. Silberman, "Academic achievement declines under pass-fail grading," *J.*, 1971.

[4] T. Dee, B. Jacob, and J. Rockoff, "Rules and discretion in the evaluation of students and schools: The case of the New York regents examinations," *Sch. Res. Pap.*, 2011.

[5] J. McCrary, "Manipulation of the running variable in the regression discontinuity design: A density test," *J. Econom.*, 2008.

[6] M. Lepper, D. Greene, and R. Nisbett, "Undermining children's intrinsic interest with extrinsic reward: A test of the" overjustification" hypothesis.," *J. Personal.*, 1973.

[7] M. Hanus and J. Fox, "Assessing the efects of gamification in the classroom: A longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic," *Comput. Educ.*, 2015.

[8] P. Guo and K. Reinecke, "Demographic differences in how students navigate through MOOCs," *Proc. first ACM Conf.*, 2014.

Using a Single Model Trained across Multiple Experiments to Improve the Detection of Treatment Effects

Thanaporn Patikorn, Douglas Selent, Neil T. Heffernan, Joseph E. Beck, Jian Zou
100 Institute Rd.
Worcester, MA 01609
{tpatikorn, dselent, nth, josephbeck, jzou} @wpi.edu

ABSTRACT

In this work, we describe a new statistical method to improve the detection of treatment effects in interventions. We call our method TAME (Trained Across Multiple Experiments). TAME takes advantage of multiple experiments with similar designs to create a single model. We use this model to predict the outcome of the dependent variable in unseen experiments. We use the predictive accuracy of the model on the conditions of the experiment to determine if the treatment had a statistically significant effect. We validated the effectiveness of our model using a large-scale simulation study, where we showed that our model can detect treatment effects with 10% more statistical power than an ANOVA in certain settings. We also applied our model to real data collected from the ASSISTments online learning platform and showed that the treatment effects detected by our model were comparable to the effects detected by the ANOVA.

Keywords

Intervention Effectiveness; Randomized Controlled Experiments; Meta-Analysis; ANOVA; Treatment Effect; TAME;

1. INTRODUCTION

The goal of this paper is to develop a method that can more effectively detect treatment effects in randomized controlled experiments that are run inside online tutoring systems. Common methods for analyzing these experiments include existing statistical tests such as a T-Test, regression, and an Analysis of Variance (ANOVA). Although these analysis methods are typically used, there are disadvantages that must be considered.

Grossman et al discuss several disadvantages of randomized controlled experiments [4]. One disadvantage is having a small sample size compared to the number of variables and it is unlikely that there will be an equal balance of variables in the control and treatment groups of the experiment. Another disadvantage is that a single study may not be able to infer the overall treatment effect on the entire population. The treatment may have different effects on different subpopulations, experiments settings may be different, and there may also be several different dependent measures to consider. There also may be a large number of experiments where the reported effects are false due to Type I error.

We hope to ameliorate several of these issues by using a technique that combines data from several randomized controlled experiments in order to build a model to estimate the difference between conditions in experiments. Advantages of combining data from multiple experiments include increasing the sample size, and also reducing the variance for better confidence estimates [1].

Two major questions to consider when pooling experiments are discussed in [1]. The first question is, "Which experiments should be combined for analysis?", and is considered "the most serious methodological limitation" [3]. Experiments should be combined if they have similar research questions, populations, experiment settings, intervention components, implementation, and dependent measures. In our paper we select experiments with the same dependent measures and study design format (A/B).

The second question is how to combine experiments once they are chosen for inclusion. One method, called *lumping*, combines all the data into a single data set, ignoring the differences among the experiments. Another method called *pooling*, combines experiments into a single data set but adjusts for differences in experiments [1]. In our case, we have experiments that can have very different effect sizes. We applied the pooling method, but instead of applying standard meta-analysis techniques, we trained a linear model to predict the outcome measures.

Our goal is to use our method called TAME (Trained Across Multiple Experiments) to more effectively detect treatment effects. We use data from multiple experiments to increase the power of the model, and to utilize linear regression to model subject outcomes for treatment effect detection. We hope that TAME would also reduce the bias of meta-analyses in efforts to improve the reliability of statistical results.

The data we use comes from a data set previously collected and synthesized from twenty-two randomized controlled experiments run inside the ASSISTments online tutoring system [5]. These experiments were proposed by internal and external researchers on a large variety of topics. The student population consists of mostly middle-school students ranging from grades 6-8. All experiments had a single control group and a single experiment group (A/B study design) with at least 50 students in each group. A total of 102,252 problems were attempted by 8,297 students across 22 different experiments.

We conducted a large-scale simulation experiment to compare the accuracy of TAME to the accuracy of an ANOVA under different experiment settings. To determine how well each method performed we looked at the chance of detecting an effect when there really is one (true positive) and the chance of not detecting an effect when there really is not one (true negative). This is conversely related to Type I and Type II errors. Our research questions are 1) Does TAME perform better than the ANOVA method? 2) Under what circumstances do TAME perform better?

2. TAME Model

TAME borrows the idea of meta-analysis, where many experiments are used to report on generalized effects. The main concept of TAME is to first model the outcome measure in the absence of the condition assignment. Any other factors can still be used in the creation of the model. To do this, one must use data outside of the experiment of interest (the "test" experiment) to ensure that the model does not overfit to the test experiment. By training a model on a collection of similar experiments, it is less likely that the model will overfit to any given experiment. For the rest of this paper, we will refer to a group of similar experiments as an experiment group.

For each experiment in an experiment group, we first train a linear model on all of the other experiments in the same group, using all factors in the data set except the condition assignments in the experiments. Note that the model used does not have to be a linear model and other types of models will work as well. Once a model is trained, it is applied to estimate the dependent measure of the test experiment. Then, we compute the residual value for each subject in the test experiment, which is the actual outcome measure minus the modeled outcome measure. Assuming that all other factors that may affect the outcome measures are accounted for in the model, the only cause of the residual values must be the condition assignments and noise. A two-tailed unpaired T-test is performed on the residual values of the samples from the control group and the treatment group in the test experiment to determine if there is a significant treatment effect. If the T-test reports that there are significant differences, we claim that the effect of the intervention was statistically significant.

The sign of the residual matters for our usage of the model, which is contrary to most modeling approaches, where the absolute or squared residuals are analyzed. If the residual is positive, it means that the student overperformed the model due to some factors that the model does not account for. Those factors positively affect the student outcome measure and could be attributed to helpful interventions. If the residual is negative, it means that the student underperformed the model, which may be caused by harmful interventions. We believe the reason that our method will result in a better estimate of treatment effects is because training on all experiments except for one, without knowing the conditions of the experiment, will generate a less biased model than an ANOVA, which operates on a single experiment and includes the condition of the experiment while training the model.

3. SIMULATION EXPERIMENT

Simulated data are often used in the EDM community as well as other research areas to validate models , such as [7]. One advantage of using simulated data is that the ground truth values are known, which make it possible to compare the learned values to the true values. Another advantage of using simulated data is that it gives us the ability to control for and test any combinations of parameters. To evaluate the effectiveness of our model, we ran a large scale simulation experiment to compare the accuracy of treatment effects detected by TAME to the accuracy of treatment effects detected by an ANOVA. For both methods, we used a between-subject ANOVA (type III SS) to compare the main effects of the condition variable on our dependent measure using all other factors as fixed factors. We looked at the percent of treatment effects correctly detected (true positive, $p<0.05$) and incorrectly detected (false positive). Our simulation data was generated using Java code and the models were trained and evaluated using R.

Table 1. Parameters, value ranges, and an example of a setting

Parameter	Possible Parameter Values	Example Setting
Expr. in a Group	2, 4, 6, 8, 10, 12, 14, 16, 18, 20	2
Expr. with Diff.	[0, n], n = number of expr. in group	1
Effect sizes	0.05, 0.1, 0.15, 0.2, 0.4, 0.6, 0.8, 1	1.0
Samples	20, 40, 60, 80, 100, 200	20
Factors	0, 1, 2, 3, 4	1
Values per Factor	2, 3, 4	3

3.1. Data Generation

The parameters we experimented with and their possible values are summarized in the first and second column of Table 1, while the third column shows an instantiation of values for an example experiment setting. Ten trials of experimental data were generated for all combinations of parameters resulting in over ten million trials generated.

Experiments in a Group: This parameter represents the number of experiments in a group. We chose to sample groups in the range of [2, 20] experiments in increments of two because we believe this is a realistic number of experiments that could be analyzed together. Several recent meta-analysis papers publish data with the number of studies ranging from 12 - 217 [2, 5, 9]. It is also reasonable to have this many experiments with a similar designs, which can be analyzed together. Our analysis of real data includes a dataset consisting of 22 experiments reported in [5].

Experiments with Differences: This parameter is number of experiments where there is a difference in the outcome measure between the control and treatment group. This value ranges from having no experiments in group with differences to having all the experiments within a group with differences. All experiments that have a difference between the control group and the treatment group all have equal effect sizes.

Samples: This parameter is for the number of samples assigned into a given experiment. In the context of the EDM community, the number samples is equivalent to the number of students that have participated in an experiment. We chose to simulate data for a number of students in the range of {20, 40, 60, 80, 100, and 200} because we believe this range consists of values for a typical number of students expected to participate in most experiments.

Factors: The number of factors for all experiments within an experiment group. The condition of the experiment is considered a special factor and is not grouped with the other factors. All factors are categorical variables. Factors are used to represent features of the student such as gender or levels of prior knowledge, which have been shown to improve predictive modeling [8]. We add features to the generated data to more accurately simulate a real-world scenario. We assume the features do not correlate with the intervention, and therefore do not have interaction effects.

Values per Factor: This parameter represents the number of categorical values that all factors can subsume. For example a factor with two values could represent the gender of a student or a factor with several values could represent the prior knowledge of the student discretized into several bins.

Effect Size: The effect size measured with Cohen's D. Both smaller ranges of differences and larger ranges of differences were tested for both practical and theoretical contexts. In practice many experiments report small effect sizes; therefore we test in the range of [0.05, 0.2] in increments of 0.05 to simulate what would

Table 2. A concrete example of simulated data

Row Number	Experiment Number	Sample Number	Condition	Condition Value	Factor 1	Factor 1 Value	Base Outcome Value	Final Outcome Value
1	1	1	A	0	A	0.4	0	0.4
2	1	2	B	1	B	0.1	0.1	1.2
3	2	2	B	N/A	C	-0.7	0.05	-0.65
4	2	3	A	N/A	A	0.38	-0.7	-0.42

happen in a likely scenario. We also use values from [0.2, 1.0] in increments of 0.2 for larger differences to observe what would happen in a best-case scenario with a large difference in means.

Table 2 shows an example of what the data generated under the example setting in Table 1 looks like. The first column in Table 2 shows what experiment each sample belongs to. In this example there are only two experiments. Each experiment in this example has twenty samples each, however only two samples are shown for both experiments in Table 2. The sample column represents a unique sample number for each experiment. In the context of an experiment, the sample number represents the student. The condition column represents what condition the sample is assigned into. The condition is uniformly and randomly chosen between either "A", or "B", where "A" represents the control group and "B" represents the treatment group. Each condition has a value associated with it, which is equivalent to the effect of the treatment. Table 2 shows that in this example, the intervention has an effect size of 1.0 standard deviation. Therefore the condition value is set to 1.0 where the condition is "B" (treatment), and the condition value is set to 0 where the condition is "A" (control).

Each factor in the experiment has a column for the categorical value of that factor and a value for how that factor value affects the dependent measure of the experiment. Since there is only one factor in this experiment setting, there is only a single factor column ("Factor 1") shown in Table 2. This column can hold three values ("A", "B", or "C"), because the number of values per factor is set to three in this experiment setting. Each factor value is generated randomly and uniformly for each sample. The value for how the factor effects the dependent measure is randomly generated from a standard normal distribution ($\mu = 0$, $\sigma = 1.0$) with Gaussian noise added to the value for each sample for a more realistic simulation. The noise is generated from a normal distribution with the mean centered at the randomly generated value for the factor with a standard deviation of 0.25. In Table 2, this can be seen by looking at rows 1, and 4, which are assigned to factor "A", where all the values for this factor are close to 0.4. In this example the randomly generated effect of factor "A" is 0.4 with noise added for each sample. In the context of educational data mining, certain features of the student can have effects on learning gains which may vary slightly for each student.

The base outcome value is a random number chosen from a normal distribution ($\mu = 0$, $\sigma = 1$). This number represents how a random sample performs. The final column represents the dependent measure in experiments. This value is the sum of the base outcome values, all feature values, and the condition value. For example, row 2 has a condition value of 1, a factor value of 0.1 and a base outcome value of 0.1. Therefore the final outcome value is 1 + 0.1 + 0.1 = 1.2. This representation may be thought of as the average learning gains a student has when comparing their pretest score to their posttest scores. We do not have an explicit dependent measure and will refer to it in the general context.

4. SIMULATION RESULT

To analyze our results we calculated the mean true positive rate and false positive rate at the experiment group level. Each experiment group consisted of a varying number of experiments, with ten trials each. Each trial had a ground truth value where there was either a difference in conditions or there was not a difference in conditions. The ground truth value on whether or not an experiment had differences in conditions is represented in the "experiments with differences" variable described in section 3.1. If a model correctly detected significant differences ($p<0.05$) between conditions it was counted as a true positive. Similarly, if a model incorrectly detected significant differences it was counted as a false positive. An average of the true positive counts and false positive counts for all experiments and trials was used to equally weight each experiment group. Some random data samples generated errors in analysis. If an error occurred for any trial the entire experiment group was removed from analysis to ensure the analysis would be as unbiased as possible. There were 79,200 simulated experiment groups, of which 58 were removed, resulting in 77,842 experiment groups analyzed. The data from the results of the simulation experiment and the code used can be found here. https://sites.google.com/site/tamemethod/

Since there was little change in the false positive rate (Type I error) regardless of method or factors, we exclude it from further analysis. All sets of parameters had a Type I error of roughly 5%, which is the threshold we used to determine if a model detected significant differences. Our analysis focuses on the true positive rates (statistical power) of each method. We ran a repeated measure ANOVA to compare the main effects of the parameters (see data section) on the statistical power of our method to the statistical power of an ANOVA. Out of 70,742 simulated experiments, TAME has an average power of 0.376 (SD = 0.357), which is slightly better than the ANOVA which had an average power of 0.366 (SD = 0.353). This power may seem low, however many experiments in the learning science community do in fact have low power due to the combination of low sample sizes and low effect sizes.

Table 4 shows the results of a repeated measures ANOVA, which determined that the average power of TAME was significantly better than the ANOVA ($F(1, 70{,}713) = 804.144$, $p < 0.001$). We discuss the effect of each parameter in the following sections. We discuss the overall effect each parameter has on both methods and compare the effects between each method.

4.1. Experiments in a Group

There is no general effect of the number of experiments in a group. This is because this variable will only matter for our method which takes advantage of a larger number of experiments in a group when training a model. An ANOVA trains and tests on experiments individually; therefore the number of experiments in group has no effect on the power of the ANOVA. Since the number of experiments has no effect on the power of the ANOVA, it is less likely to see an overall effect considering both TAME and the ANOVA.

Table 3. Tests of Between-Subject Effects

Source	Type III Sum of Squares	df	Mean Square	F	Significance	Partial Eta Squared
Intercept	3285.103	1	3285.103	115891.122	< 0.001	0.621
effect size	13753.480	7	1964.783	69313.168	< 0.001	0.873
factors	37.834	4	9.459	333.678	< 0.001	0.019
values per factor	1.196	2	0.598	21.096	< 0.001	0.001
samples	2013.213	5	402.643	14204.334	< 0.001	0.501
experiments	0.163	9	0.018	0.638	0.765	0
percent of exp. with diff.	0	1	0	0.011	0.917	0
Error	2004.463	70713	0.028			

Table 4. Test of Within-Subjects Effects

Source	Type III Sum of Squares	df	Mean Square	F	Significance	Partial Eta Squared
method	0.576	1	0.576	804.144	< 0.001	0.011
method * effect size	2.948	7	0.421	587.633	< 0.001	0.055
method * factors	2.205	4	0.551	769.046	< 0.001	0.042
method * values per factor	0.671	2	0.335	467.906	< 0.001	0.013
method * samples	1.173	5	0.235	327.340	< 0.001	0.023
method * experiments	0.050	9	0.006	7.709	< 0.001	0.001
method * percent of exp. with diff.	0.002	1	0.002	2.468	0.116	0
error(method)	50.683	70713	0.001			

There is also no overall noticeable difference between TAME and an ANOVA for different number of experiments in a group. Table 3 shows that the number of experiments in a group has a significant effect on power ($F(9,70713) = 7.71$, $p<0.001$) with a partial eta squared = 0.001. Although the difference between the two methods is statistically significant, the effect size is insignificant.

Although there is no overall difference in method type for varying the number of experiments in a group, the number of experiments has a major impact in the case where there are a large number of factors and a small number of samples with a high effect size. Figure 1 shows that for a subset of experiments, as the number of experiments in a group increases, the difference in power between the two methods increases. TAME has a power of 0.27 compared to a power of 0.22 for the ANOVA with two experiments in a group and TAME has a power of 0.35 compared to a power of 0.25 for the ANOVA with ten experiments in a group.

4.2. Number of Factors

More factors introduce more noise in the data, making it harder to detect treatment effects. Table 3 shows that the number of factors has a significant effect on power ($F(4,70713) = 333.67$, $p<0.001$) with a partial eta squared = .019. Figure 2 shows that as the number of factors increases, the power of TAME decreases less than the power of ANOVA. This decrease leads to a difference in power between the two methods based on the number of factors. The number of factors is statistically significant ($F(4,70713) = 769.046$, $p<0.001$) with a partial eta squared of 0.042. We believe this is because TAME accounts for noises better than ANOVA by using more data that is available to TAME.

4.3. Number of Samples

In general, more samples lead to a better estimate of the true means and more power. Table 3 shows that the number of samples has a significant effect on power ($F(5,70713) = 14204.334$, $p<0.001$) with a partial eta squared = 0.5. As the number of samples increases, both methods perform equally well. This result is expected.

Table 4 shows that TAME performs better slightly than the ANOVA when there are a fewer number of samples, since the ANOVA is not an optimal method in this situation. The number of samples is a statistically significant factor when comparing the power differences between the two methods ($F(5,70713) = 327.340$, $p<0.001$) with a partial eta squared of 0.023.

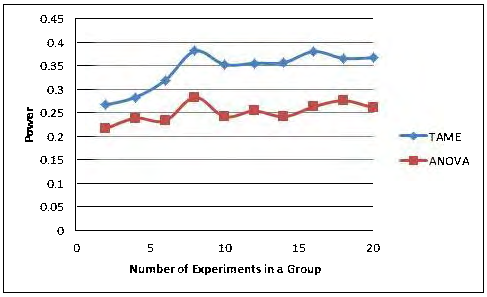


Figure 1. The power as the number of experiments in a group increases for experiment groups with 20 samples, four factors, and a treatment effect size of 0.8 and 1.0.

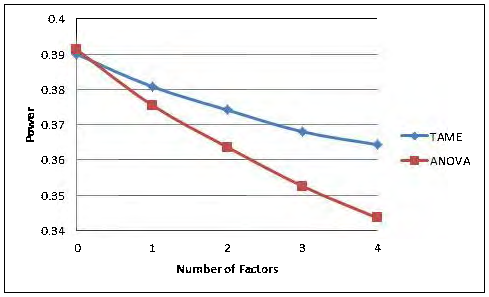


Figure 2. The statistical power of TAME and ANOVA by the number of factors used to train the models

4.4. Effect Size

A larger treatment effect is easier to detect and therefore has a positive impact on power. Table 3 shows that the size of the treatment effect has a significant effect on power ($F(7,70713) = 69313.168$, $p<0.001$) with a partial eta squared = 0.873. As the size of the effect increases so does the power.

Table 4 shows TAME performs slightly better than the regular ANOVA as the treatment effect increases. The effect size is a statistically significant factor when comparing power differences between TAME and ANOVA, ($F(7,70713) = 587.633$, $p<0.001$) with a partial eta squared of 0.055.

5. REAL DATA RESULT

We applied both TAME and the ANOVA method on a data set composed of twenty-two randomized controlled experiments run inside the ASSISTments online learning platform to compare the two method on real data [6]. Every experiment in the group is a Skill Builder consisting of one control group and one treatment group. A Skill Builder is "an assignment type that consists of a large number of similar problems, where students must answer a specified number of problems (usually three) correctly in a row on the same day in order to finish the assignment." [6]. We applied both TAME and an ANOVA on students in the studies, with the following factors as training factors: Prior Percent Correct, Guessed Gender, Prior Percent Completion, Z Scored Mastery Speed, Prior Homework Percent Completion, Z Scored HW Mastery Speed. For dependent measure, we use logarithm with base ten of the Mastery Speed, which is the number of problems a student took to answer three problems correctly in a row [9]. We use the logarithm of Mastery Speed to reduce the effect of outliers.

Table 6 shows that our method can be applied to detect significant different between conditions of a real data set. Since the size of each experiment in the data set is greater than 100, the result of simulation study suggests that TAME is as good at detecting significant differences as ANOVA. Both TAME and ANOVA detected significant differences between conditions of the same experiments (2, 3, 4, 10, and 22). This result further supports our claim that TAME is a good alternative to ANOVA, if not better.

We further investigated the reliability of TAME and ANOVA. For each experiment, we trained a model using all of the data from the other twenty-one experiments. We then used this model to predict the performance on the data in the test experiment. We experimented with a different sample size of (10, 20, 30, 40, 50, 60, 70, 80, 90, and 100) to predict in the test experiment. The evaluation of each model was an average of running the model 1,000 times, with a different random set of data points in the test experiment each time. This methodology does not invalidate our analysis since TAME was designed to utilize all data from outside of the target experiment, such as data from experiments in the past, and such data are not affected by the sample size of the target experiment. We chose to report on the results of two of the experiments in Table 5 and Table 7.; experiment 3, which was the experiment that we found the strongest treatment effect for, and experiment 6, which was one of the experiments that we did not find a significant treatment.

Table 5. The probability and the confident interval of detecting the treatment effect on the resampled data set ($p < 0.05$) on experiment 3

Experiment 3	Probability of Detecting Treatment Effect ($p<0.05$)		Size of Adjusted Wald Confidence Interval	
sample size	TAME	ANOVA	TAME	ANOVA
10	0.2280	0.0650	0.0260	0.0154
20	0.4410	0.3360	0.0307	0.0292
30	0.5810	0.5590	0.0305	0.0307
40	0.7070	0.7050	0.0282	0.0282
50	0.7970	0.8020	0.0249	0.0247
60	0.8580	0.8530	0.0217	0.0220
70	0.9170	0.9180	0.0172	0.0171
80	0.9420	0.9580	0.0147	0.0127
90	0.9660	0.9610	0.0115	0.0122
100	0.9710	0.9810	0.0107	0.0088

Table 6. Summary statistics and significance for the real dataset

	Mastery Speed Control and Experiment Group	Mastery Speed Control Group	Mastery Speed Experiment Group	TAME Sig.	ANOVA Sig.	ANOVA Partial Eta Squared
1	$\mu = 0.80$, $n = 468$, $\sigma = 0.21$	$\mu = 0.79$, $n = 256$, $\sigma = 0.20$	$\mu = 0.82$, $n = 212$, $\sigma = 0.22$	0.208	0.222	0.003
2	$\mu = 0.78$, $n = 672$, $\sigma = 0.24$	$\mu = 0.76$, $n = 324$, $\sigma = 0.21$	$\mu = 0.80$, $n = 348$, $\sigma = 0.26$	0.014	0.013	0.009
3	$\mu = 1.16$, $n = 240$, $\sigma = 0.12$	$\mu = 1.12$, $n = 123$, $\sigma = 0.11$	$\mu = 1.21$, $n = 117$, $\sigma = 0.11$	0.000	0.000	0.162
4	$\mu = 1.12$, $n = 540$, $\sigma = 0.21$	$\mu = 1.10$, $n = 298$, $\sigma = 0.19$	$\mu = 1.16$, $n = 242$, $\sigma = 0.22$	0.001	0.001	0.020
5	$\mu = 0.67$, $n = 1303$, $\sigma = 0.25$	$\mu = 0.67$, $n = 667$, $\sigma = 0.25$	$\mu = 0.67$, $n = 636$, $\sigma = 0.24$	0.503	0.389	0.001
6	$\mu = 0.63$, $n = 337$, $\sigma = 0.17$	$\mu = 0.62$, $n = 165$, $\sigma = 0.18$	$\mu = 0.63$, $n = 172$, $\sigma = 0.16$	0.634	0.737	0.000
7	$\mu = 0.65$, $n = 365$, $\sigma = 0.16$	$\mu = 0.65$, $n = 202$, $\sigma = 0.16$	$\mu = 0.65$, $n = 163$, $\sigma = 0.16$	0.489	0.562	0.001
8	$\mu = 0.59$, $n = 455$, $\sigma = 0.17$	$\mu = 0.59$, $n = 223$, $\sigma = 0.18$	$\mu = 0.59$, $n = 232$, $\sigma = 0.16$	0.542	0.571	0.001
9	$\mu = 0.91$, $n = 119$, $\sigma = 0.16$	$\mu = 0.93$, $n = 52$, $\sigma = 0.18$	$\mu = 0.90$, $n = 67$, $\sigma = 0.14$	0.460	0.478	0.005
10	$\mu = 1.09$, $n = 432$, $\sigma = 0.20$	$\mu = 1.07$, $n = 212$, $\sigma = 0.18$	$\mu = 1.11$, $n = 220$, $\sigma = 0.22$	0.037	0.045	0.010
11	$\mu = 0.95$, $n = 171$, $\sigma = 0.20$	$\mu = 0.96$, $n = 84$, $\sigma = 0.21$	$\mu = 0.93$, $n = 87$, $\sigma = 0.19$	0.297	0.225	0.009
12	$\mu = 0.90$, $n = 122$, $\sigma = 0.18$	$\mu = 0.92$, $n = 60$, $\sigma = 0.19$	$\mu = 0.88$, $n = 62$, $\sigma = 0.16$	0.302	0.389	0.007
13	$\mu = 1.11$, $n = 148$, $\sigma = 0.24$	$\mu = 1.08$, $n = 70$, $\sigma = 0.28$	$\mu = 1.12$, $n = 78$, $\sigma = 0.21$	0.320	0.395	0.005
14	$\mu = 0.83$, $n = 174$, $\sigma = 0.16$	$\mu = 0.84$, $n = 99$, $\sigma = 0.17$	$\mu = 0.82$, $n = 75$, $\sigma = 0.14$	0.159	0.216	0.009
15	$\mu = 0.93$, $n = 240$, $\sigma = 0.19$	$\mu = 0.94$, $n = 124$, $\sigma = 0.19$	$\mu = 0.91$, $n = 116$, $\sigma = 0.19$	0.159	0.177	0.008
16	$\mu = 0.98$, $n = 121$, $\sigma = 0.17$	$\mu = 0.97$, $n = 63$, $\sigma = 0.14$	$\mu = 1.00$, $n = 58$, $\sigma = 0.19$	0.159	0.324	0.009
17	$\mu = 0.94$, $n = 226$, $\sigma = 0.18$	$\mu = 0.95$, $n = 120$, $\sigma = 0.17$	$\mu = 0.94$, $n = 106$, $\sigma = 0.20$	0.529	0.342	0.004
18	$\mu = 0.70$, $n = 264$, $\sigma = 0.13$	$\mu = 0.71$, $n = 126$, $\sigma = 0.14$	$\mu = 0.70$, $n = 138$, $\sigma = 0.13$	0.455	0.226	0.006
19	$\mu = 1.02$, $n = 218$, $\sigma = 0.19$	$\mu = 1.02$, $n = 105$, $\sigma = 0.17$	$\mu = 1.02$, $n = 113$, $\sigma = 0.20$	0.844	0.994	0.000
20	$\mu = 0.81$, $n = 825$, $\sigma = 0.18$	$\mu = 0.81$, $n = 409$, $\sigma = 0.19$	$\mu = 0.81$, $n = 416$, $\sigma = 0.17$	0.887	0.926	0.000
21	$\mu = 0.84$, $n = 291$, $\sigma = 0.15$	$\mu = 0.85$, $n = 140$, $\sigma = 0.15$	$\mu = 0.84$, $n = 151$, $\sigma = 0.15$	0.855	0.892	0.000
22	$\mu = 0.78$, $n = 213$, $\sigma = 0.16$	$\mu = 0.80$, $n = 111$, $\sigma = 0.16$	$\mu = 0.75$, $n = 102$, $\sigma = 0.15$	0.020	0.018	0.027

Table 7. The probability and the confident interval of detecting the treatment effect on the resampled data set (p < 0.05) on experiment 6

experiment 6	probability of detecting treatment effect (p<0.05)		Size of Adjusted Wald Confidence Interval	
sample size	TAME	ANOVA	TAME	ANOVA
10	0.0560	0.0340	0.0144	0.0115
20	0.0500	0.0450	0.0137	0.0131
30	0.0510	0.0660	0.0138	0.0155
40	0.0470	0.0650	0.0133	0.0154
50	0.0560	0.0580	0.0144	0.0147
60	0.0720	0.0780	0.0162	0.0167
70	0.0540	0.0540	0.0142	0.0142
80	0.0490	0.0530	0.0136	0.0141
90	0.0620	0.0670	0.0151	0.0156
100	0.0520	0.0600	0.0139	0.0149

Table 5 shows that for the experiment with the strongest treatment effect (experiment 3), TAME is able to detect the treatment effect better than ANOVA, especially when the sample size <= 40. This result agrees with the result of our simulation study. When the treatment effect is not present (experiment 6), the false positive rate of both TAME and ANOVA are around 5% as shown in Table 7. This result is to be expected from using a p-value threshold of 0.05.

6. CONTRIBUTIONS

This paper makes three contributions. The first contribution of this paper is TAME, a more robust and more effective method of detecting treatment effects that can analyze several experiments simultaneously. Since the TAME model is not built specifically for any particular experiment, it allows the same model to generalize to experiments unseen by the model, including future experiments. To our knowledge, this is the first method that detects treatment effects on multiple experiments individually and simultaneously.

The second contribution this paper makes is that the results from a large-scale simulation experiment showed that TAME is better at detecting treatment effects compared to an ANOVA by more than ten percent in the case where there is a large effect, fewer samples, more factors, and with more experiments. This simulation experiment validated our proposed method and also showed that TAME has slightly better statistical power than an ANOVA and never performs worse. TAME can quickly detect large differences, such as when the treatment is harmful. It is important to detect harmful interventions as soon as possible to ensure that students are exposed to the least amount of negative effects.

The third contribution this paper makes is taking our validated method and applying it to real data collected from twenty-two randomized controlled experiments run in the ASSISTments online learning platform. On this data set, TAME and ANOVA are in agreement on significant differences between conditions. This result allows the associated researchers to further investigate the interventions and their effects, allowing them to better understand how students learn and, eventually, develop better tools and interventions for students.

6.1. Future Work and Conclusions

This work is a first step in building a model that can be used across interventions to estimate effect sizes. As such, there are many future directions to explore. A possible future work involves equally weighting the experiments our model uses. It is rare for all experiments to all have the same number of samples. Currently our model gives more weight experiments with more samples. This may lead to a small number of experiments accounting for a large amount of the weight when training a model. For future work the weighting of experiments and the effect can be investigated.

In conclusion, we have created a single model that generalizes across experiments. We have shown how it can be applied to multiple, unseen, experiments in order to evaluate their efficacy. This approach is in contrast to creating separate models for each intervention we are evaluating. This model is able to detect the effect of each intervention relative to other interventions and provide a set of features that might affect and interact with interventions. In addition, the same trained model can be applied to investigate future interventions. We evaluated the effectiveness of our model in a simulation study, which shows that our model can detect significant differences 10% more than an ANOVA in certain cases. We then applied our model to real data and found that three out of twenty-two interventions are significantly different from the control conditions.

7. ACKNOWLEDGEMENTS

We thank multiple 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.

8. REFERENCES

[1] Bangdiwala, S. I., Bhargava, A., O'Connor, D. P., Robinson, T. N., Michie, S., Murray, D. M., & Pratt, C. A. (2016). Statistical methodologies to pool across multiple intervention studies. *Translational Behavioral Medicine*, 1-8.

[2] D'Mello, S.K. (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

[3] DeMets, D. L. (1987). Methods for combining randomized clinical trials: strengths and limitations. *Statistics in medicine*, *6*(3), 341-348.

[4] Grossman, J., & Mackenzie, F. J. (2005). The randomized controlled trial: gold standard, or merely standard?. *Perspectives in biology and medicine*, *48*(4), 516-534.

[5] Patall, E.A., Cooper, H., & Robinson, J.C. (2008). The Effects of Choice on Intrinsic Motivation and Related Outcomes: A Meta-Analysis of Research Findings. *Psychology Bulletin. 134* (2), pp 270-300.

[6] Selent, D., Patikorn, T., & Heffernan, N. (2016). ASSISTments Dataset from Multiple Randomized Controlled Experiments. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale* (pp. 181-184). ACM.

[7] Sinharay, S. (2010). How often do subscores have added value? Results from operational and simulated data. Journal of Educational Measurement, 47(2), 150-174.

[8] Walkington, C. A. (2013). Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes. Journal of Educational Psychology,105(4), 932.

[9] Xiong, X., Li, S., & Beck, J. E. (2013). Will You Get It Right Next Week: Predict Delayed Performance in Enhanced ITS Mastery Cycle. In FLAIRS Conference.

Data-Mining Textual Responses to Uncover Misconception Patterns

Joshua J. Michalenko
Rice University
jjm7@rice.edu

Andrew S. Lan
Princeton University
andrew.lan@princeton.edu

Andrew E. Waters
Rice University
aew2@rice.edu

Phillip J. Grimaldi
Rice University
phillip.grimaldi@rice.edu

Richard G. Baraniuk
Rice University
richb@rice.edu

ABSTRACT

An important, yet largely unstudied problem in student data analysis is to detect *misconceptions* from students' responses to *open-response* questions. Misconception detection enables instructors to deliver more targeted feedback on the misconceptions exhibited by many students in their class, thus improving the quality of instruction. In this paper, we propose a new natural language processing-based framework to detect the common misconceptions among students' textual responses to short-answer questions. We propose a probabilistic model for students' textual responses involving misconceptions and experimentally validate it on a real-world student-response dataset. Experimental results show that our proposed framework excels at classifying whether a response exhibits one or more misconceptions. More importantly, it can also automatically detect the common misconceptions exhibited across responses from multiple students to multiple questions; this property is especially important at large scale, since instructors will no longer need to manually specify all possible misconceptions that students might exhibit.

Keywords

Learning analytics, Markov chain Monte Carlo, misconception detection, natural language processing

1. INTRODUCTION

The rapid developments of large-scale learning platforms (e.g., MOOCs (edx.org, coursera.org) and OpenStax Tutor (openstaxtutor.org)) have enabled not only access to high-quality learning resources to a large number of students, but also the collection of student data at very large scale. The scale of this data presents a great opportunity to revolutionize education by using machine learning algorithms to *automatically* deliver personalized analytics and feedback to students and instructors in order to improve the quality of teaching and learning.

1.1 Detecting misconceptions from data

The predominant form of student data, their *responses* to assessment questions, contains rich information on their knowledge. Analyzing why a student answers a question incorrectly is of crucial importance to deliver timely and effective feedback. Among the possible causes for a student to answer a question incorrectly, exhibiting one or more *misconceptions* is critical, since upon detection of a misconception, it is very important to provide targeted feedback to a student to correct their misconception in a timely manner. Examples of using misconceptions to improve teaching include incorporating misconceptions to design better distractors for multiple-choice questions [10], implementing a dialogue-based tutor to detect misconceptions and provide corresponding feedback to help students self-practice, preparing prospective instructors by examining the causes of common misconceptions among students [19], and incorporating misconceptions into item response theory (IRT) for learning analytics [18].

The conventional way of leveraging misconceptions is to rely on a set of pre-defined misconceptions provided by domain experts [10, 19]. However, this approach is not scalable, since it requires a large amount of human effort and is domain-specific. With the large scale of student data at our disposal, a more scalable approach is to automatically detect misconceptions from data.

Recently, researchers have developed approaches for data-driven misconception detection; most of these approaches analyze students' response to *multiple-choice* questions. Examples of these approaches include detecting misconceptions in multiple-choice mathematics questions and modeling students' progress in correcting them [9] via the additive factor model [3], and clustering students' responses across a number of multiple-choice physics questions [20]. However, multiple-choice questions have been shown to be inferior to open-response questions in terms of pedagogical value [8]. Indeed, students' responses to open-response questions can offer deeper insights into their knowledge state.

To date, detecting misconceptions from students' responses to open-response questions has largely remained an unexplored problem. A few recent developments work exclusively with *structured* responses, e.g., sketches [17], short mathematical expressions [11], group discussions in a chemistry class [16], and algebra with simple syntax [4].

1.2 Contributions

In this paper, we propose a natural language processing framework that detects students' common misconceptions from their *textual* responses to open-response, short-answer questions. This problem is very difficult, since the responses are, in general, *unstructured*.

Our proposed framework consists of the following steps. First,

we transform students' textual responses to a number of short-answer questions into low-dimensional textual feature vectors using several well-known word-vector embeddings. These tools include the popular Word2Vec embedding [12], the GLOVE embedding [15], and an embedding based on the long-short term memory (LSTM) neural network [6]. We then propose a new statistical model that jointly models both the transformed response textual feature vectors and expert labels on whether a response exhibits one or more misconceptions; these labels identify only *whether or not* a response exhibits one or more misconceptions but not *which* misconception it exhibits.

Our model uses a series of latent variables: the feature vectors corresponding to the correct response to each question, the feature vectors corresponding to each misconception, the tendency of each student to exhibit each misconception, and the confusion level of each question on each misconception. We develop a Markov chain Monte Carlo (MCMC) algorithm for parameter inference under the proposed statistical model. We experimentally validate the proposed framework on a real-world educational dataset collected from high school classes on AP biology.

Our experimental results show that the proposed framework excels at classifying whether a response exhibits one or more misconceptions compared to standard classification algorithms and significantly outperforms a baseline random forest classifier. We also compare the prediction performance across all three embeddings. More importantly, we show examples of common misconceptions detected from our dataset and discuss how this information can be used to deliver targeted feedback to help students correct their misconceptions.

2. DATASET AND PRE-PROCESSING

In this section, we first detail our short-answer response dataset, and then detail our pre-processing approach to convert responses into vectors using word-to-vector embeddings.

2.1 Dataset

Our dataset consists of students' textual responses to short-answer questions in high school classes on AP Biology administered on OpenStax Tutor [14]. Every response was labeled by an expert grader as to whether it exhibited one or more misconceptions. A total of $N = 386$ students each responded to a subset of a total of $Q = 1668$ questions; each response was manually labeled by one or multiple expert graders, resulting in a total of $\sim 60,000$ labeled responses. Since there is no clear rubric defining what is a misconception, graders might not necessarily agree on what label to assign to each response. Therefore, we trim the dataset to only keep responses that are labeled by multiple graders and they also assigned the same label, resulting in $13,099$ responses. We also further trim the dataset by filtering out students who respond to less than 5 questions and questions with less than 5 responses in every dataset. This subset contains $6,152$ responses.

The questions in our dataset are drawn from the OpenStax AP biology textbook; we divide the full dataset into smaller subsets corresponding to each of the first four units [13], since different units correspond to entirely different sub-areas in biology. These units cover the following topics: Unit 1—The Chemistry of Life, Chapters 1-3, Unit 2—The Cell, Chapters 4-10, Unit 3—Genetics, Chapters 11-17, and Unit 4—Evolutionary Processes, Chapters 18-20. To summarize, we show the dimensions of the subsets of the data corresponding to each unit in Table 1. Since not every student was assigned to every question, the dataset is sparsely populated; Table 1 also shows the portion of responses that are observed in the trimmed data subsets, denoted as "sparsity".

	N	Q	Sparsity (%)
Unit 1	47	77	0.280
Unit 2	101	104	0.243
Unit 3	73	91	0.236
Unit 4	43	75	0.315

Table 1: Dataset statistics.

2.2 Response embeddings

We first perform a pre-processing step by transforming each textual student response into a corresponding real-valued vector via three different word-vector embeddings. Our first embedding uses the Word2Vec embedding [12] trained on the OpenStax Biology textbook (an approach also mentioned in [2]), to learn embeddings that put more emphasis on the technical vocabulary specific to each subject. We create the feature vector for each response by mapping each individual word in the response to its corresponding feature vector, and then adding them together. Concretely, denote $\mathbf{x}_{i,j} = \{w_1, w_2, ..., w_{T_{i,j}}\}$ as the collection of words in the textual response of student j to question i, where $T_{i,j}$ denotes the total number of words in this response (excluding common stopwords). We then map each word w_t to its corresponding D-dimensional feature vector $r(w_t) \in \mathbb{R}^D$ using the trained Word2Vec model. We use $D = 10$ for the Word2Vec embedding. We then compute the student response feature vector as $\mathbf{f}_{i,j} = \sum_{t=1}^{T_{i,j}} r(w_t)$.

Our second word-vector embedding is a pre-trained GLOVE embedding with $D = 25$ [15]. The GLOVE embedding is very similar to the Word2Vec embedding, with the main difference being that it takes corpus-level word co-occurrence statistics into account. Moreover, the quality of the GLOVE embedding for common words is likely higher since it is pre-trained on a huge corpus (comparing to only the OpenStax Biology textbook for Word2Vec).

Both the Word2Vec embedding and the GLOVE embedding do not take word ordering into account, and for misconception classification, this drawback can lead to problems. For example, responses "If X then Y" and "If Y then X" may have completely different meanings depending on the context, where it's possible for one to exhibit a common misconception while the other one does not. Using the Word2Vec and GLOVE embeddings, these responses will be embedded to the same feature vector $\mathbf{f}_{i,j}$, making them indistinguishable from each other. Therefore, our third word-vector embedding is based on the long short-term memory (LSTM) neural network, which is a recurrent neural network that excels at capturing long-term dependencies in sequential data. Therefore, it can take word ordering into account, a feature that we believe is critical for misconception detection. We implement a 2-layer LSTM network with 10 hidden units and train it on the OpenStax Biology textbook. For each student

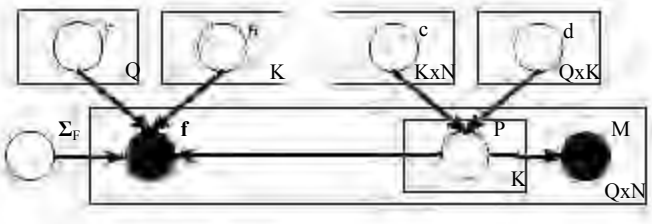


Figure 1: Visualization of the statistical model. Black nodes denote observed data; white nodes denote latent variables to be inferred.

response, we use the text as character-by-character inputs to the LSTM network and use the last layer's hidden unit activation values (stacked in a $D = 10$ dimensional vector) as its textual feature $\mathbf{f}_{i,j}$.

3. STATISTICAL MODEL

We now detail our statistical model; its graphical model is visualized in Figure 1. Concretely, let there be a total of N students, Q questions, and K misconceptions. Let $M_{i,j} \in \{0, 1\}$ denote the binary-valued misconception label on the response of student j to question i provided by an expert grader, with $j \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, Q\}$, where 1 represents the presence of (one or more) misconceptions, and 0 represents no misconceptions.

We transform the raw text of student j's response to question i into a D-dimensional real-valued feature vector, denoted by $\mathbf{f}_{i,j} \in \mathbb{R}^D$, via a pre-processing step (detailed in the previous section). Let $\Omega \subseteq \{1, \ldots, Q\} \times \{1, \ldots, N\}$ denote the subset of student responses that are labeled, since every student only responds to a subset of the questions.

We denote the *tendency* of student j to exhibit misconception k, with $k \in \{1, \ldots, K\}$ as $c_{k,j} \in \mathbb{R}$, and the *confusion level* of question i on misconception k, as $d_{i,k} \in \mathbb{R}$. Then, let $P_{i,j,k} \in \{0, 1\}$ denote the binary-valued latent variable that represents whether student j exhibits misconception k in their response to question i, with 1 denoting that the misconception is present and 0 otherwise. We model $P_{i,j,k}$ as a Bernoulli random variable

$$p(P_{i,j,k} = 1) = \Phi(c_{k,j} + d_{i,k}), \quad (i, j) \in \Omega, \tag{1}$$

where $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(t; 0, 1)\mathrm{d}t$ denotes the inverse probit link function (the cumulative distribution function of the standard normal random variable). Given $P_{i,j,k}\ \forall k$, we model the observed misconception label $M_{i,j}$ as

$$M_{i,j} = \begin{cases} 0 & \text{if } P_{i,j,k} = 0\ \forall k, \\ 1 & \text{otherwise,} \end{cases} \qquad (i, j) \in \Omega. \tag{2}$$

In words, a response is labeled as having a misconception if one or more misconceptions is present (given by the latent misconception exhibition variables $P_{i,j,k}$). Given $P_{i,j,k}\ \forall k$, the textual response feature vector that corresponds to student j's response to question i, $\mathbf{f}_{i,j}$, is modeled as

$$\mathbf{f}_{i,j} \sim \mathcal{N}(\boldsymbol{\gamma}_i + \sum_k P_{i,j,k}\boldsymbol{\theta}_k, \boldsymbol{\Sigma}_F), \quad \forall (i, j) \in \Omega, \tag{3}$$

where $\boldsymbol{\gamma}_i$ denotes the feature vector that corresponds to the correct response to question i, $\boldsymbol{\theta}_k$ denotes the feature vector that corresponds to misconception k, and $\boldsymbol{\Sigma}_F$ denotes the covariance matrix of the multivariate normal distribution characterizing the feature vectors. In other words, the feature vector of each response is a *mixture* of the feature vectors corresponding to the correct response to the question and each misconception the student exhibits. In the next section, we develop an MCMC inference algorithm to infer the values of the latent variables $\boldsymbol{\gamma}_i$, $\boldsymbol{\theta}_k$, $\boldsymbol{\Sigma}_F$, $P_{i,j,k}$, $c_{k,j}$, and $d_{i,k}$, given observed data $\mathbf{f}_{i,j}$ and $M_{i,j}$.

4. PARAMETER INFERENCE

We use a Gibbs sampling algorithm [5] for parameter inference under the proposed statistical model. The prior distributions of the latent variables are listed as follows:

$$\boldsymbol{\gamma}_i \sim \mathcal{N}(\boldsymbol{\mu}_\gamma, \boldsymbol{\Sigma}_\gamma), \boldsymbol{\theta}_k \sim \mathcal{N}(\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta), \boldsymbol{\Sigma}_F \sim IW(h_F, \mathbf{V}_F),$$
$$c_{k,j} \sim \mathcal{N}(\mu_c, \sigma_c^2), d_{i,k} \sim \mathcal{N}(\mu_d, \sigma_d^2),$$

where $IW(\cdot)$ denotes the inverse-Wishart distribution and $\boldsymbol{\mu}_\gamma$, $\boldsymbol{\Sigma}_\gamma$, $\boldsymbol{\mu}_\theta$, $\boldsymbol{\Sigma}_\theta$, h_F, $\mathbf{V}_F$, μ_c, σ_c^2, μ_d, and σ_d^2 are hyperparameters.

We start by randomly initializing the values of the latent variables $\boldsymbol{\gamma}_i$, $\boldsymbol{\theta}_k$, $\boldsymbol{\Sigma}_F$, $P_{i,j,k}$, $c_{k,j}$, $d_{i,k}$, a_j, and μ_i by sampling from their prior distributions. Then, in each iteration of our Gibbs sampling algorithm, we iteratively sample the value of each random variable from its full conditional posterior distribution. Specifically, in each iteration, we perform the following steps:

a) Sample $P_{i,j,k}$: We first sample the latent misconception indicator variable $P_{i,j,k}$ from its posterior distribution as

$$P_{i,j,k} = \begin{cases} 0 & \text{if } M_{i,j} = 0, \\ 1 & \text{if } M_{i,j} = 1 \text{ and } P_{i,j,k'} = 0\ \forall\ k' \neq k, \\ \frac{r}{r+1} & \text{if } M_{i,j} = 1 \text{ and } \exists\ k' \neq k \text{ s.t. } P_{i,j,k'}{=}1, \end{cases}$$

where

$$r = \frac{p(\mathbf{f}_{i,j}|\boldsymbol{\gamma}_i, \boldsymbol{\theta}_k, \forall k, \boldsymbol{\Sigma}_F, P_{i,j,k'\neq k}, P_{i,j,k} = 1)}{p(\mathbf{f}_{i,j}|\boldsymbol{\gamma}_i, \boldsymbol{\theta}_k, \forall k, \boldsymbol{\Sigma}_F, P_{i,j,k'\neq k}, P_{i,j,k} = 0)} \cdot \frac{p(P_{i,j,k} = 1|c_{k,j}, d_{i,k})}{p(P_{i,j,k} = 0|c_{k,j}, d_{i,k})}.$$

Terms in these expressions are given by (1) and (3).

b) Sample $\boldsymbol{\gamma}_i$: We then sample the feature vector that corresponds to the correct response to each question, $\boldsymbol{\gamma}_i$, from its posterior distribution as $\boldsymbol{\gamma}_i \sim \mathcal{N}(\boldsymbol{\mu}_{\gamma_i}, \boldsymbol{\Sigma}_{\gamma_i})$ where

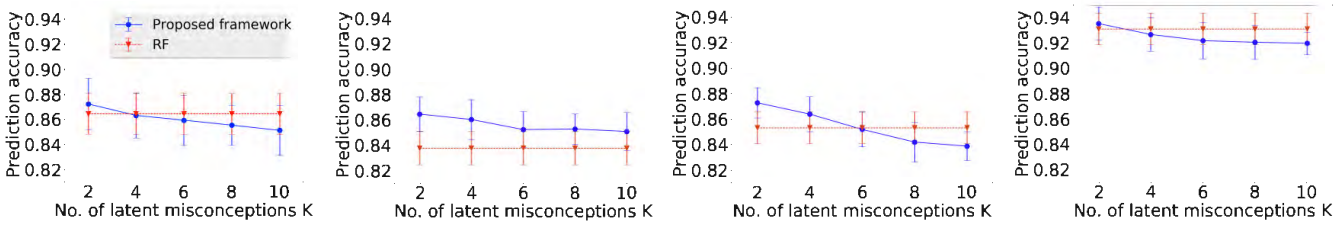
$$\boldsymbol{\mu}_{\gamma_i} = \boldsymbol{\Sigma}_{\gamma_i}\left(\boldsymbol{\Sigma}_\gamma^{-1}\boldsymbol{\mu}_\gamma + \boldsymbol{\Sigma}_F^{-1}\sum_{j:(i,j)\in\Omega}(\mathbf{f}_{i,j} - \sum_k P_{i,j,k}\boldsymbol{\theta}_k)\right),$$
$$\boldsymbol{\Sigma}_{\gamma_i} = (\boldsymbol{\Sigma}_\gamma^{-1} + n_i\boldsymbol{\Sigma}_F^{-1})^{-1},$$

where $n_i = \sum_j I\left((i, j) \in \Omega\right)$.

c) Sample $\boldsymbol{\theta}_k$: We then sample the feature vector that corresponds to each misconception, $\boldsymbol{\theta}_k$, from its posterior distribution as $\boldsymbol{\theta}_k \sim \mathcal{N}(\boldsymbol{\mu}_{\theta_k}, \boldsymbol{\Sigma}_{\theta_k})$ where

$$\boldsymbol{\mu}_{\theta_k} = \boldsymbol{\Sigma}_{\theta_k}\left(\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta + \boldsymbol{\Sigma}_F^{-1}\sum_{i,j:P_{i,j,k}=1}(\mathbf{f}_{i,j} - \boldsymbol{\gamma}_i - \sum_{k'\neq k}P_{i,j,k'}\boldsymbol{\theta}_{k'})\right),$$
$$\boldsymbol{\Sigma}_{\theta_k} = (\boldsymbol{\Sigma}_\theta^{-1} + n_k\boldsymbol{\Sigma}_F^{-1})^{-1},$$

where $n_k = \sum_{i,j} I\left(P_{i,j,k} = 1\right)$.



(a) Unit 1 (b) Unit 2 (c) Unit 3 (d) Unit 4

Figure 2: Comparison of the prediction performance of the proposed model against RF on our AP Biology dataset using the ACC metric as the number of latent misconceptions K varies, with the LSTM embedding.

d) Sample $\mathbf{\Sigma}_F$: We then sample the covariance matrix $\mathbf{\Sigma}_F$ from its posterior distribution as

$$\mathbf{\Sigma}_F \sim IW\left(h_F + n, \mathbf{V}_F + \mathbf{M}\right),$$

where n=$\sum_{i,j} I\left((i,j) \in \Omega\right)$ and $\mathbf{M} = \sum_{i,j:(i,j)\in\Omega}(\mathbf{f}_{i,j} - \boldsymbol{\gamma}_i - \sum_k P_{i,j,k}\boldsymbol{\theta}_k)(\mathbf{f}_{i,j} - \boldsymbol{\gamma}_i - \sum_k P_{i,j,k}\boldsymbol{\theta}_k)^T$.

e) Sample $c_{k,j}$ and $d_{i,k}$: In order to sample $c_{k,j}$ and $d_{i,k}$, we first sample the value of the auxiliary variable $z_{i,j,k}$ (following the standard approach proposed in [1]) as

$$z_{i,j,k} \sim \mathcal{N}^{\pm}(c_{k,j} + d_{i,k}, 1), \forall (i,j) \in \Omega,$$

where $\mathcal{N}^{\pm}(\cdot)$ denotes the truncated normal random distribution truncated to the positive side when $P_{i,j,k} = 1$ and negative side when $P_{i,j,k} = 0$. We then sample $c_{k,j}$ from its posterior distribution as

$$c_{k,j} \sim \mathcal{N}(\mu_{c_{k,j}}, \sigma^2_{c_{k,j}}),$$

where $n_j = \sum_i I\left((i,j) \in \Omega\right)$, $\sigma^2_{c_{k,j}} = 1/(1/\sigma_c^2 + n_j)$, and $\mu_{c_{k,j}}$=$\sigma^2_{c_{k,j}}\big(\mu_c/\sigma_c^2 + \sum_{i:(i,j)\in\Omega}(z_{i,j,k} - d_{i,k})\big)$. We then sample $d_{i,k}$ from its posterior distribution as

$$d_{i,k} \sim \mathcal{N}(\mu_{d_{i,k}}, \sigma^2_{d_{i,k}}),$$

where $\sigma^2_{d_{i,k}}$=$1/(1/\sigma_d^2 + n_i)$, and $\mu_{d_{i,k}} = \sigma^2_{d_{i,k}}(\mu_d/\sigma_d^2 + \sum_{j:(i,j)\in\Omega}(z_{i,j,k} - c_{k,j}))$.

We run the iterations detailed above for a number of T total iterations with a certain burn-in period, and use the samples of each latent variable to approximate their posterior distributions.

Parameter inference under our model suffers from the label-switching issue that is common in mixture models [5], meaning that the mixture components might be permuted between iterations. We employ a post-processing step to resolve this issue. We first calculate the augmented data likelihood at each iteration, (indexed by ℓ) we then identify the iteration $\ell_{\max}$ with the largest augmented data likelihood, and permute the variables $\boldsymbol{\theta}_k^{\ell}$, $c_{k,j}^{\ell}$, and $d_{i,k}^{\ell}$ that best match the variables $\boldsymbol{\theta}_k^{\ell_{\max}}$, $c_{k,j}^{\ell_{\max}}$, and $d_{i,k}^{\ell_{\max}}$. After this post-processing step, we can simply calculate the posterior means of each one of these sets of variables by taking averages of their values across non burn-in iterations.

5. EXPERIMENTS

We experimentally validate the efficacy of the proposed framework using our AP Biology class dataset. We first compare the proposed framework against a baseline random forest (RF) classifier that classifies whether a student response exhibits one or more misconceptions. We then show common misconceptions detected in our datasets and discuss how the proposed framework can use this information to deliver meaningful targeted feedback to students that helps them correct their misconceptions.

5.1 Experimental setup

We run our experiments with $K \in \{2, 4, 6, 8, 10\}$ latent misconceptions with hyperparameters $\boldsymbol{\mu}_\gamma = \boldsymbol{\mu}_\theta = \mathbf{0}_D$, $\mathbf{\Sigma}_\gamma = \mathbf{\Sigma}_\gamma = \mathbf{V}_F = \mathbf{I}_D$, $h_F = 10$, $\mu_c = \mu_d = 0$, and $\sigma_c^2 = \sigma_d^2 = 1$, for a total of $T = 500$ iterations with the first 250 iterations as burn-in. We compare the proposed framework against a baseline random forest (RF) classifier[1] using the textual response feature vectors $\mathbf{f}_{i,j}$ to classify the binary-valued misconception label $M_{i,j}$, with 200 decision trees.

We randomly partition each dataset into 5 folds and use 4 folds as the training set and the other fold as the test set. We then train the proposed framework and RF on the training set and evaluate their performance on the test set, using two metrics: i) prediction accuracy (ACC), i.e., the portion of correct predictions, and ii) area under curve (AUC), i.e., the area under the receiver operating characteristic (ROC) curve of the resulting binary classifier [7]. Both metrics take values in $[0, 1]$, with larger values corresponding to better prediction performance. We repeat our experiments for 20 random partitions of the folds.

For the proposed framework, the predictive probability that a response with its feature vector $\mathbf{f}_{i,j}$ exhibits a misconception, i.e., the probability that at least one of the K latent misconception exhibition state variables take the value of 1, is given by $1 - \widehat{p}_{i,j}$, where

$$\begin{aligned}\widehat{p}_{i,j} &= p(M_{i,j} = 0 \,|\, \mathbf{f}_{i,j}, \boldsymbol{\gamma}_i, \mathbf{\Sigma}_F, \boldsymbol{\theta}_k, \forall k, c_{k,j}, d_{i,k}) \\ &= \frac{p(\mathbf{f}_{i,j}|\boldsymbol{\theta}_k, P_{i,j,k} = 0, \forall k) \prod_k p(P_{i,j,k} = 0|c_{k,j}, d_{i,k})}{\sum_{P_{i,j,k},\forall k}(p(\mathbf{f}_{i,j}\,|\,\boldsymbol{\theta}_k, P_{i,j,k}\forall k) \prod_k p(P_{i,j,k}|c_{k,j}, d_{i,k}))},\end{aligned}$$

where in the last expression we omitted the conditional dependency of $\mathbf{f}_{i,j}$ on $\boldsymbol{\gamma}_i$ and $\mathbf{\Sigma}_F$ due to spatial constraints. For RF, the predictive probability is given by the fraction of decision trees that classifies $M_{i,j} = 1$ given $\mathbf{f}_{i,j}$.

5.2 Results and discussions

The number of latent misconceptions K is an important parameter controlling the granularity of the misconceptions that

[1]The RF classifier achieves the best performance among a number of off-the-shelf baseline classifiers, e.g., logistic regression, support vector machines, etc. Therefore, we do not compare it against other baseline classifiers.

	Unit 1		Unit 2		Unit 3		Unit 4	
	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Proposed framework	**0.789±0.014**	**0.762±0.027**	**0.774±0.015**	**0.758±0.023**	**0.779±0.019**	**0.752±0.020**	**0.887±0.011**	**0.774±0.029**
RF	0.762±0.019	0.645±0.025	0.735±0.011	0.676±0.014	0.758±0.017	0.630±0.024	0.873±0.009	0.604±0.034
Proposed framework	0.867±0.014	**0.762±0.048**	**0.870±0.010**	**0.821±0.024**	**0.893±0.017**	**0.794±0.039**	**0.953±0.015**	**0.892±0.047**
RF	**0.876±0.014**	0.697±0.022	0.859±0.013	0.771±0.040	0.883±0.008	0.616±0.043	0.948±0.019	0.731±0.006
Proposed framework	**0.873±0.042**	**0.772±0.093**	**0.865±0.025**	**0.829±0.044**	**0.873±0.027**	**0.792±0.061**	**0.936±0.032**	**0.832±0.094**
RF	0.865±0.035	0.711±0.086	0.838±0.028	0.722±0.043	0.854±0.028	0.697±0.057	0.931±0.025	0.709±0.105

Table 2: Performance comparison on misconception label classification of a textual response in terms of the prediction accuracy (ACC) and area under the receiver operating characteristic curve (AUC) of the proposed framework against a random forest (RF) classifier, using the AP Biology dataset and the Word2Vec (top), GLOVE (middle), and LSTM (bottom) embeddings.

we aim to detect. Figure 2 shows the comparison between the proposed framework using different values of K and RF using the ACC metric with the LSTM embedding. We see an obvious trend that, as K increases, the prediction performance decreases. The likely cause of this trend is that the proposed framework tends to overfit as the number of latent misconceptions grows very large since some of our datasets do not contain very rich misconception types. Moreover, the number of common misconceptions varies across different units, with Unit 2 likely containing more misconception types than Units 1 and 4.

We then compare the performance of the proposed framework against RF on misconception label classification in Table 2 using $K = 2$ and all three embeddings. The proposed framework significantly outperforms RF (1–4% using the ACC metric and 4-18% using the AUC metric) on almost all 4 data subsets using every embedding. The only case where the proposed framework does not outperform RF is on Unit 1 using the GLOVE embedding. We postulate that the reason for this result is that this unit is about chemistry and has a lot of responses with more chemical molecular expressions than words; therefore, the proposed framework does not have enough textual information to exhibit its advantages (grouping responses that share the same misconceptions into clusters) over the RF classifier.

Both the proposed framework and RF perform much better using the GLOVE and LSTM embeddings than the Word2Vec embedding. This result is likely due to the fact that these embeddings are more advanced than the Word2Vec embedding: the GLOVE embedding considers additional word co-occurrence statistics than the Word2Vec embedding, is trained on a much larger corpus, and has a higher dimension $D = 25$, while the LSTM embedding is the only embedding that takes word ordering into account. Moreover, both algorithms perform best on Unit 4, which is likely due to two reasons: i) the Unit 4 subset has a larger portion of its responses labeled, and ii) Unit 4 is about evolution, which results in responses that are much longer and thus contains richer textual information.

5.3 Uncovering common misconceptions

We emphasize that, in addition to the proposed framework's significant improvement over RF in terms of misconception label classification, it features great interpretability since it identifies common misconceptions from data. As an illustrative example, the following responses from multiple students across two questions are identified to exhibit the same misconception in the Unit 4 subset using the Word2Vec embedding:

> *Question 1*: People who breed domesticated animals try to avoid inbreeding even though most domesticated animals are indiscriminate. Evaluate why this is a good practice.
> *Correct Response*: A breeder would not allow close relatives to mate, because inbreeding can bring together deleterious recessive mutations that can cause abnormalities and susceptibility to disease.
> **Student Response 1**: Inbreeding can cause a rise in unfavorable or detrimental traits such as genes that cause individuals to be prone to disease or have unfavorable mutations.
> **Student Response 2**: Interbreeding can lead to harmful mutations.

> *Question 2*: When closely related individuals mate with each other, or inbreed, the offspring are often not as fit as the offspring of two unrelated individuals. Why?
> *Correct Response*: Inbreeding can bring together rare, deleterious mutations that lead to harmful phenotypes.
> **Student Response 3**: Leads to more homozygous recessive genes thus leading to mutation or disease.
> **Student Response 4**: When related individuals mate it can lead to harmful mutations.

Although these responses are from different students to different questions, they exhibit one common misconception, that inbreeding leads to harmful mutations. Once this misconception is identified, course instructors can deliver the targeted feedback that inbreeding only brings together harmful mutations, leading to issues like abnormalities, rather than directly leading to harmful mutations.

Moreover, the proposed framework can automatically discover common misconceptions that students exhibit without input from domain experts, especially when the number of students and questions are very large. Specifically, in the example above, we are able to detect such a common misconception that 4 responses exhibit by analyzing the 1016 responses in the AP Biology Unit 4 dataset; however, it would not likely be detected if the number of responses was smaller and fewer students exhibited the misconception. This feature makes it an attractive data-driven aid to domain experts in designing educational content to address student misconceptions.

We show another example that the proposed framework can automatically group student responses to the same group

according to the misconceptions they exhibit. The example shows two detected common misconceptions among students' responses to a single question in the Unit 2 subset using the LSTM embedding:

> *Question*: What is the primary energy source for cells?
> *Correct response*: Glucose.
> **Student responses with misconception 1:**
> a) sunlight b) sum c) The sun d) he sun?
> **Student responses with misconception 2:**
> a) ATP b) adenosine triphosphate
> c) ATPPPPPPPPPPPPP d) atp mitochondria

We see that the proposed framework has successfully identified two common misconception groups, with incorrect responses that list "sun" and "ATP" as the primary energy source for cells. Note that the LSTM embedding enables the framework to assign the full and abbreviated form of the same entity ("adenosine triphosphate" and "ATP") into the same misconception cluster, without employing any preprocessing on the raw textual response data. The likely reason for this result is that our LSTM embedding is trained on a character-by-character level on the OpenStax Biology textbook, where these terms appear together frequently, thus enabling the LSTM to transform them into similar vectors. This observation highlights the importance of using good, information-preserving word-vector embeddings for the proposed framework to maximize its capability of detecting common misconceptions.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a natural language processing-based framework for detecting and classifying common misconceptions in students' textual responses. Our proposed framework first transforms their textual responses into low-dimensional feature vectors using three existing word-vector embedding techniques, and then estimates the feature vectors characterizing each misconception, among other latent variables, using a proposed mixture model that leverages information provided by expert human graders. Our experiments on a real-world educational dataset consisting of students' textual responses to short-answer questions showed that the proposed framework excels at classifying whether a response exhibits one or more misconceptions. Our proposed framework is also able to group responses with the same misconceptions into clusters, enabling the data-driven discovery of common misconceptions without input from domain experts. Possible avenues of future work include i) automatically generate the appropriate feedback to correct each misconception, ii) leverage additional information, such as the text of the correct response to each question, to further improve the performance on predicting misconception labels, iii) explore the relationship between the dimension of the word-vector embeddings and prediction performance, and iv) develop embeddings for other types of responses, e.g., mathematical expressions and chemical equations.

7. REFERENCES

[1] J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *J. Am. Stat. Assoc.*, 88(422):669–679, June 1993.

[2] S. Bhatnagar, M. Desmarais, N. Lasry, and E. S. Charles. Text classification of student self-explanations in college physics questions. In *Proc. 9th Intl. Conf. Educ. Data Min.*, pages 571–572, July 2016.

[3] 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. Intell. Tutoring Syst.*, pages 164–175, June 2006.

[4] M. Elmadani, M. Mathews, A. Mitrovic, G. Biswas, L. H. Wong, and T. Hirashima. Data-driven misconception discovery in constraint-based intelligent tutoring systems. In *Proc. 20th Int. Conf. Comput. in Educ.*, pages 1–8, Nov. 2012.

[5] A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin. *Bayesian Data Analysis*. CRC press, 2013.

[6] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1–32, Nov. 1997.

[7] H. Jin and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Trans. Knowl. Data Eng.*, 17(3):299–310, Mar. 2005.

[8] S. Kang, K. McDermott, and H. Roediger III. Test format and corrective feedback modify the effect of testing on long-term retention. *Eur. J. Cogn. Psychol.*, 19(4-5):528–558, July 2007.

[9] R. Liu, R. Patel, and K. R. Koedinger. Modeling common misconceptions in learning process data. In *Proc. 6th Intl. Conf. on Learn. Analyt. & Knowl.*, pages 369–377, Apr. 2016.

[10] J. K. Maass and P. I. Pavlik Jr. Modeling the influence of format and depth during effortful retrieval practice. In *Proc. 9th Intl. Conf. Educ. Data Min.*, pages 143–149, July 2016.

[11] T. McTavish and J. Larusson. Discovering and describing types of mathematical errors. In *Proc. 7th Intl. Conf. Educ. Data Min.*, pages 353–354, July 2014.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, Sep. 2013.

[13] OpenStax Biology. https://openstax.org/details/biology, 2016.

[14] OpenStax Tutor. https://openstaxtutor.org/, 2016.

[15] J. Pennington, R. Socher, and C. D. Manning. GloVe: Global vectors for word representation. In *Proc. ACM SIGDAT Conf. Emp. Method. Nat. Lang. Process.*, pages 1532–1543, Oct. 2014.

[16] H. J. Schmidt. Students' misconceptions—Looking for a pattern. *Sci. Educ.*, 81(2):123–135, Apr. 1997.

[17] A. Smith, E. N. Wiebe, B. W. Mott, and J. C. Lester. SketchMiner: Mining learner-generated science drawings with topological abstraction. In *Proc. 7th Intl. Conf. Educ. Data Min.*, pages 288–291, July 2014.

[18] K. K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *J. Educ. Meas.*, 20(4):345–354, Dec. 1983.

[19] D. Tirosh. Enhancing prospective teachers' knowledge of children's conceptions: The case of division of fractions. *J. Res. Math. Educ.*, 31(1):5–25, Jan. 2000.

[20] G. Zheng, S. Kim, Y. Tan, and A. Galyardt. Soft clustering of physics misconceptions using a mixed membership model. In *Proc. 9th Intl. Conf. Educ. Data Min.*, pages 658–659, July 2016.

Automated Assessment for Scientific Explanations in On-line Science Inquiry

Haiying Li
Rutgers University
10 Seminary Place
New Brunswick, NJ 08901
1(848)932-0868
haiying.li@gse.rutgers.edu

Janice Gobert
Rutgers University
10 Seminary Place
New Brunswick, NJ 08901
1(848)932-0867
janice.gobert@gse.rutgers.edu

Rachel Dickler
Rutgers University
10 Seminary Place
New Brunswick, NJ 08901
1(848)932-0869
rachel.dickler@gse.rutgers.edu

ABSTRACT
Scientific explanations, which include a claim, evidence, and reasoning (CER), are frequently used to measure students' deep conceptual understandings of science. In this study, we developed an automated scoring approach for the CER that students constructed as a part of virtual inquiry (e.g., formulating questions, analyzing data, and warranting claims) in an intelligent tutoring system (ITS), called Inq-ITS. Results showed that the automated scoring of CER was strongly correlated with human scores when validated using independent sets of data from both the same inquiry task/question, as well as when using data from a different inquiry task/question. These findings imply that automated CER is a very promising approach to reliably and efficiently score scientific explanations in open response format for both small- and large-scale assessments. It also provides Inq-ITS with the capability to assess the full complement of inquiry practices described by NGSS.

Keywords

automated assessment, scientific explanation, claim, evidence, reasoning

1. INTRODUCTION
The implementation of the Next Generation Science Standards (NGSS) has led to a need for assessments that are able to capture students' competencies at science inquiry practices [21]. Open-response tasks have been used in assessments for science inquiry because they can elicit students' communication skills, conceptual understandings, and ability to reason from evidence due to the measurement constraints of traditional multiple-choice items [10]. Rubrics for scoring students' explanations have been developed according to frameworks, such as Toulmin's [27] model of argumentation [8,16]. A modified version of Toulmin's model consists of three components: claim (an assertion about an investigated question), evidence (data or observations that support the assertion, i.e., the claim), and reasoning (articulating how the evidence supports the claim and how scientific principles explain the relationship between the data and claim).

Previous studies have developed rubrics to assess the accuracy of claim, evidence, and reasoning (CER) in students' scientific explanations. Gotwals and Songer [8] applied a rubric following the CER framework in order to score middle school students' explanations in an ecological science assessment. The rubric scoring for each component of CER was on a scale from 0 to 2 according to the accuracy and depth students' responses. McNeill et al. [16] scored students' responses to explanation prompts for middle school chemistry with a rubric that also followed the CER format using a 0 to 2 scale. These general rubrics for open response items provide some insight regarding the argumentation skill level of students, which can be valuable for guiding teachers' instruction and feedback. Open response items, however, can be time consuming and costly to score [28]; they can be inaccurately scored due to human factors such as rater fatigue [19], and rubrics can be interpreted and used differently by different raters [1]. One way to resolve these issues is through the use of automated scoring techniques [30].

Automated scoring techniques also permit automated feedback to students *as* they write scientific explanations or immediately following their writing tasks, when students have the opportunity to revise their writing. Automated, real-time feedback has been found to: significantly reduce the time between response submission and feedback relative to human scorers [15] and be as, if not more, effective than feedback presented by teachers [3]. While automated scoring presents an efficient and accurate means for promoting student learning gains, no studies, to date, have developed techniques for online, automated scoring of scientific explanations according to CER.

The current paper presents a new automated scoring approach to CER using the techniques of both natural language processing and machine learning. The approach addresses accuracy as well as important structural components of explanations as identified in the CER framework. The approach was validated using correlations between human scores and automated scores for scientific explanations produced in the Inq-ITS learning environment. Automated scoring of CER will: dramatically reduce time and expense, improve the efficiency and accuracy of CER scores, allow for instantaneous feedback, and make individualized instruction from teachers and/or automated scaffolding possible. Furthermore, scoring these data is critical because our data show that many students who have acquired a deep understanding of science content and inquiry practices, cannot articulate in words what they have learned. Conversely, some students are able to simply parrot what they have heard/read when doing written CER tasks, but do not actually understand the science content or practices [4, 11].

1.1 Automated Open Response
Automated scoring techniques have been developed to assess students' open responses in computer-assisted assessments and learning environments for science. Techniques include natural language-processing (NLP), such as regular expressions [12], to determine whether students' scientific explanations include key conceptual phrases [3, 13, 14]. The specific techniques and rubrics

used for automated scoring of science open response items vary across programs as described below.

The Summarization Integrated Development Environment (SIDE) uses a combination of NLP techniques and machine learning algorithms to score scientific explanations for the inclusion of biology concept knowledge [9, 20]. This system yielded correlations between human-scored and computer-scored responses ranging from 0.79 to 0.87 depending on the sample of participants. Disagreement was attributed to differences in linguistic tendencies across samples [9]. A later study by Nehm et al. [20] on the same system found that agreement between human and computer scoring was strong (i.e. $k > .81$). The SIDE program may be a valuable tool in scoring student scientific explanations [20], but is limited to identifying the presence of concepts within responses, and as such is not useful at scoring students' competencies at generating claims, evidence, and reasoning, which are critical to NGSS inquiry practices.

Another program that has been used to autoscore scientific explanations is the SPSS Text Analysis (SPSSTA) program [29], which uses language-processing procedures to identify terms and note patterns within texts [25]. A study by Weston et al. [29] applied SPSSTA to score undergraduate responses to biology explanation prompts. The agreement between human-coded responses and the SPSSTA for different levels of an analytic rubric ranged from a kappa of 0.67 to 0.88. The SPSSTA program relates to SIDE in terms of its potential to identify important concepts, but is unable to automatically produce machine learning algorithms from a trained data set [9], and this is limited in utility.

EvoGrader automatically scores constructed explanations using machine-learning algorithms [17]. A study compared EvoGrader scores to human scores based on the identification of nine key evolution concepts and strong agreement was found, as indicated by kappas above 0.85 for all concepts except one ($k = 0.71$) [17]. The EvoGrader automated assessment system was able to produce human-like scoring of key evolutionary concepts, but would need retraining in order to be generalized to other domains.

The c-Rater program scores scientific explanations based on the presence of central concepts using natural language processing [13]. A study by Liu et al. [13] compared human and c-Rater scores for four energy open response questions and found moderate agreement with Pearson correlations ranging from 0.67 to 0.72. While c-Rater was able to capture the presence of concepts, the program did not perform highly enough to be recommended for use as a sole scorer. Liu et al. [14] examined the agreement between human scorers and c-rater-ML, which is an autoscoring program that uses support vector regressions, a machine learning technique. Kappas across eight science explanation items ranged from 0.62 to 0.90, indicating good to very good agreement between human raters and c-rater-ML on a 5-point rubric for connecting key ideas [14]. The high agreement on certain explanation items demonstrated the potential for c-rater-ML to be used as a sole scorer, but, as noted by the authors, sensitivity to variations in phrasing of central concepts needed to be improved.

Automated scoring programs for scientific explanations exemplify the potential for accurate and efficient scoring of open responses in terms of the presence of scientific concepts, but do not provide opportunity for scoring more fine-grained components of explanations. That is, auto-scoring techniques have yet to address argumentative components of explanations that are central to science inquiry, namely students' competencies at generating claims, evidence for claims, and articulating the link between the two using reasoning, which are required by NGSS. Auto-scoring specific sub-components of responses, as we have done in our work, enables automated scaffolds that can, in turn, target specific areas of student difficulty. The rubrics for CER in previous studies broadly categorized responses into incorrect, partially correct, or fully correct, but failed to break down CER into finer-grained sub-skills or sub-components. As a result, previous rubrics have been unable to pinpoint exactly why students are having difficulties constructing explanations. In the present study, we developed a fine-grained rubric modified from McNeill et al. [16].

1.2 Description of Inq-ITS

Inq-ITS is a web-based intelligent tutoring system for Physical, Life, and Earth science that automatically assesses scientific inquiry practices at the middle school level in real time within interactive microworld simulations [5, 24]. Within each microworld, inquiry practices proposed in the NGSS for middle school are assessed including: question asking/hypothesizing, collecting data, analyzing data, warranting claims, and communicating findings using a CER framework.

Automated scoring has been implemented within Inq-ITS with patented algorithms [5] to measure sub-skills of each inquiry practice based on actions recorded in log files [7, 23]. Automated scoring of sub-skills in Inq-ITS required building detectors based on data-mined algorithms that captured variations of complex behaviors, such as designing controlled experiments [7, 24]. In order to build detectors, human raters used text-replay tagging to identify key behavioral features and train models that determined the presence of particular sub-skills [23]. The additional implementation of Bayesian Knowledge Tracing and Knowledge Engineering has enabled real-time, automated feedback that scaffolds students as they engage in inquiry practices in Inq-ITS [7] and has been found to result in significant inquiry learning gains for students [18, 22]. Sao Pedro and his colleagues [22, 24] found that students who had no experience with designing controlled experiments and testing stated hypotheses were able to acquire these skills after receiving scaffolded feedback from Inq-ITS's pedagogical agent, Rex. Moussavi, Gobert, and Sao Pedro [18] found that students who received scaffolds on data interpretation skills in one science topic of Inq-ITS were better able to apply those skills in a new science topic.

While automated scoring and feedback has been successfully applied to student actions in Inq-ITS, automated scoring has yet to be developed for written explanations. The automated scoring approach presented in this paper allows for automatic scoring of students' written scientific explanations in Inq-ITS, as well as lays the groundwork for the development of specific, automated feedback for open response items.

2. METHOD

2.1 Participants and Materials

Participants were 293 middle school students from 18 classes in six public middle schools who completed the Inq-ITS density virtual lab. The Density Virtual Lab contained three activities aimed to foster understanding about density of a liquid when using: different shapes of a container (narrow, square, and wide), different types of liquid (water, oil, and alcohol), and different amounts of liquid (quarter, half, and full). This study validated the automated scoring for the scientific explanations that students constructed in the first two activities: shape-density ($N = 293$) and type-density ($N = 268$) after a series of scientific investigations. The type-density data set was used to train and test the model with the method of 10-fold cross-validation. The shape-density data set was used to further test

the model to examine how well the model performed when it was generalized to an independent data set.

2.2 Rubrics and Inter-Rater Reliabilities

Scientific explanations in Inq-ITS consisted of three components: claim, evidence, and reasoning (CER). As previously stated, other rubrics have been unable to pinpoint exactly why students are having difficulties when constructing explanations. In the present study, we developed a fine-grained rubric modified from McNeill et al. [16], described as follows.

Claim was graded by four sub-skills: independent variable (IV), IV relationship (IVR; the conditions that students changed in the controlled target IV), dependent variable (DV), and DV relationship (DVR; the effect of IV on DV). For example, a good claim that a student wrote in the type-density activity was: I found out when you change the *type* (IV) of the liquid from *water* to *oil* (IVR), the *density* (DV) will *decrease* (DVR). IV and DV were graded with binary scores: 1 for presence of the sub-skill and 0 for the absence. IVR was classified into four levels: (1) correct answers in which students reported two controlled conditions of the target IV, (2) general answers in which students stated IVR using general expressions rather than specifically stating the conditions of change (e.g., I found that the *change (IVR) of type of liquid (IV) changes (DVR) the density (DV)*, (3) partial answers in which students only reported one controlled target condition (e.g., The density of *water* is the largest), and (4) incorrect answers. Therefore, correct IVR was given 1 point; general IVR, 0.8 points; partially correct IVR, 0.5 points; and incorrect IVR, 0 points. DVR in the type-density activity was scored according to three levels: correct (1 point), general (0.8 as shown in IVR example), and incorrect. DVR in the shape-density activity was scored dichotomously, correct (1 point) versus incorrect (0 points). The DVR (shape of the container) did not affect the DV (density), so responses were either correct or incorrect and no general expressions were involved.

Evidence was scored by two sub-skills: sufficiency and appropriateness [6]. Sufficiency was a measure of whether students provided sufficient evidence. If two controlled target conditions were stated, then 2 points were given. Mentioning only one controlled target condition was insufficient and was given 1 point. Using general expressions was given 0.5. Not mentioning any controlled target condition was incorrect and was given 0 points. Appropriateness was a measure of whether students provided appropriate data, such as the data of mass, volume, and density, as displayed in students' data tables in Inq-ITS. This sub-skill was consistent with the sufficiency of evidence, but focused on the data. Here is an example of a good answer in the shape-density activity: No matter what the container shapes are, narrow or wide, and the *mass of oil was 212.5* (data of mass) while the *volume was 250* (data of volume). *The density resulted in 0.85 g/ml* (data of density). If students specified the data of density, they were given 1 point for DVR in appropriate evidence; otherwise, 0 points. If they reported both the data of mass and volume, they were given 1 point. If they only reported the data of either mass or volume, they were given 0.5 points. If they did not report any data of mass or volume, they were given 0 points.

Reasoning was measured by three sub-skills: theory, connection between data and the claim, and data that supports or refutes the claim. Theory referred to whether students stated a scientific principle related to density, here being: the properties of a substance (based on the type of liquid) affect the density, not the shape of the container. Four categories were classified: (1) complete theory for 2 points (e.g., When looking at the data chart, it is noticeable that *the mass and volume don't change so the density doesn't change*.), (2) partial but closer to complete for 1 point (e.g., only mentioning two of three properties), (3) partial but closer to none for 0.5 points (e.g., only mentioning one property), and (4) incorrect or no theories for 0 points (e.g., no property was mentioned). Connection between data and claim referred to whether students specified that their data supports or refutes their claim. If they did, 1 point was given (e.g., *My evidence supports my claim*...). If they only partially stated the connection, 0.5 points were given (e.g., *It will support my claim*…) because the student did not specify whether the data or evidence supported the claim. If there were no expressions specified, 0 points were given. Data in the reasoning task were similar to the claim task with one main difference. In scoring reasoning data, mentioning *either* IV or IVR was accepted as correct (1 points) and mentioning only one condition of change was considered partially correct (0.5 points).

Two expert raters scored students' CER according to the fine-grained rubric. The interrater-reliabilities by Cronbach's α were .993, .994, .938 and the intraclass correlations were .986, .988, .882 for claim, evidence, and reasoning, respectively, higher than human agreement in prior studies (e.g., [14]). Disagreements were discussed until agreement was reached and agreed upon scores were used for analyses.

2.3 Automated Scoring

The target sub-skills were extracted using regular expressions (RegEX) based on the rubrics used by human raters in section 2.2. RegEX is a natural language processing technique that often applies algorithms to search for specific phrases or phrases that are semantically equivalent to a target concept [26]. In ITSs, RegEX has been used to accurately identify the presence of target concepts in students' responses [12]. Table 1 displays some examples of the RegEX that we used to extract features. RegEXs were generated based on semantically similar phrases that corresponded to a particular concept noted in the rubric.

Table 1. Examples of RegEX in the Shape-Density Activity.

CER	Sub-Skill		RegEX
Claim (0~4)	IV		shape
	IVR		(narrow.*square| (square.*wide)
	DV		density
	DVR		(^((?!n[o']t|doesn(')?t.)* (same|constant))
Evidence (0~4)	Sufficient		Same as IVR
	IVR		((mass.*volume).*250)| ((volume*mass).*250)
	DVR		1|85|78
Reasoning (0~6)	Theory		((mass.*volume).*density)
	Connection		(data|evidence).*(support|prove|indicate| |show|refute).*(claim|hypothesis|theory))
	Data	IV/IVR	shape|(narrow.*(wide|square))
		DV	Same as DV claim
		DVR	Same as DVR claim

If the sub-skill was binary, RegEX was used to detect the presence or absence of the content with Python programming language. If the sub-skill contained more than two levels, RegEx was used to detect the presence or absence of the sub-skill with a higher score first, and then with a lower score. Each sub-skill at each level was assigned to a binary score, 1 for the presence and 0 for absence of the sub-skill. If the sub-skill had more than two scales, each scale was assigned to a binary score first and then transformed into the true scores. Take IVR in claim as an example (e.g., *I found that the change of the container shape does not change the density*). RegEX

matched two conditions first and assigned this category a score of 0 because the two specific shapes were not mentioned. Then RegEX matched general expressions and found the target expression, *change of the container shape*, so 1 was assigned to the general expression category and matching stopped when the target content was found. In the analysis, this claim IVR was given a score of 0.8 points.

In this study, we used an if-then algorithm to search for a particular word or phrase, as is done in AutoTutor [12]. Take the IV (e.g., shape of the container) in the claim as an example. First, RegEX "shap" was generated to match the word "shape." Second, this RegEX was used to search a written claim. Third, if there was the word "shape" in the claim, then IV was present and scored as "1". If no word "shape" existed in the claim, then IV was considered absent and scored as "0". Moreover, before searching the target work, the misspelt target words were corrected to avoid a decrease in agreement [14]. If-then algorithms enhanced the performance especially for the more complex sub-skills, such as IVR, by matching the higher-level features first and then filtering down to the lower-level features. The modification of RegEx and algorithms typically took about 10 iterations for complicated sub-skills, such as theory, IVR, but fewer iterations for simple sub-skills, such as IV and DV. Each iteration took about 1-30 minutes, depending on the complexity of the sub-skills.

2.4 Statistical Analyses

Linear regression analyses were conducted using M5-prime method to assess whether sub-skills were predictive of human scores of CER. We used two methods to validate the model. The first method was 10-fold cross-validation. The second method was to further validate the model with an independent data set in a different inquiry, shape-density activity. If the model yields good performance with similar statistics as the cross-validation analyses, our confidence in model stability is increased and the model could be generalized to different Inq-ITS activities. We used the Pearson correlations as previous studies [14] did to evaluate automated scores and followed the same rules for describing their magnitude [2]: none (0.00–0.09), small (0.10–0.30), moderate (0.31–0.50), and large (0.51–1.00).

3. RESULTS

3.1 Performance of Automated Scores

A linear regression analysis for automated claim scoring with 10-fold cross-validation yielded a significant model in the type-density activity, $r = .97$, $p < .001$. The four sub-skills of claims were combined to account for 94% of the variance in the human claim scores, with correlation coefficients (β) of 1.02, 1.04, 1.07, 0.86 ($p < .001$) for IV, IVR, DV, and DVR, respectively. When this model was validated in the shape-density data set, it was also significantly correlated with human scores, $r = .94$, $p < .001$, which explained 88% of the variance in the human scores.

The same procedures were applied to the automated evidence scores. The cross-validation analysis showed a significant model, $r = .97$, $p < .001$, with three sub-skills accounting for 94% of the variance in the human evidence scores, with βs of 0.99, 0.87, and 0.90 ($p < .001$) for sufficiency, appropriateness IVR, and DVR, respectively. When this model was validated in the shape-density data set, the automated scores were also almost perfectly correlated with human scores, $r = .97$, $p < .001$, which explained 94% of the variance in the human evidence scores.

Finally, the same analysis was conducted for automated reasoning scores. The cross-validation analysis indicated a significant model, $r = .84$, $p < .001$, with five sub-skills accounting for 71% of the variance in the human reasoning scores, with βs of 0.21, 0.94, 0.85, 1.09, and 0.96 ($p < .001$) for theory, connection between data and claim, data of IV/IVR, DV, and DVR, respectively. When this model was validated in the shape-density data set, the automated scores were highly correlated with human reasoning scores, $r = .85$, $p < .001$, which explained 72% of the variance in the human reasoning scores.

These findings imply that the automated CER scores could best capture human CER scores in the independent sets of data from both the same inquiry task/question and data from a different inquiry task/question ($r = .84$~$.97$, larger than threshold of .50) [2]. These findings imply that the automated methods with the sub-skills of CER are a promising approach to automatically score scientific explanations respective of CER in science inquiry. This automated method with regular expressions and if-then algorithms enables automated scoring to be generalized to different inquiry activities without additional training and testing of the model, and yields satisfactory performance.

3.2 Analyses of Errors

Across three components of scientific explanations, automated claim and evidence scores almost perfectly predicted human claim and evidence scores when validated using independent sets of data from both the same inquiry task/question, as well as when using data from a different inquiry task/question. Reasoning showed a very good correlation between automated scores and human scores in both data sets, but this correlation was relatively low as compared to claim and evidence. This section, therefore, analyzes the errors of reasoning in the type-density data set. Table 2 displays the confusion matrix of automated rating and human rating for reasoning, which explicitly demonstrated a discrepancy for disagreement in scores between humans and automated scores. Results showed a high discrepancy for scores 2 – 4. Specifically, when the human score was 2, only 40% were given a score of 2 by automated methods. Almost half of the remaining responses were given 1 and the other half were given 3 points or more. Similarly, when the human score was 3, only 44% was scored 3 by automated methods. More than 30% was scored 2 and about another 30% was scored 4 – 5. It is the same for the human score of 4. Less than 40% of responses were scored 4 by automated methods, while more than half was scored 3 by automated methods.

Table 2. Confusion Matrix for Reasoning.

Scores	**Automated (Column)**							
Human (Row)	0	1	2	3	4	5	6	*N*
0	**19**	5	1					25
1		**23**	7					30
2	1	**13**	**18**	9	1	2	1	45
3		6	**34**	**47**	6	**11**	2	106
4			5	**27**	**20**		1	53
5			1	1	1	**2**	1	6
6						1	**2**	3
N	20	47	66	84	28	16	7	268

Note. 0–6 are the total reasoning scores rated by humans and automated methods based on the analytic rubrics.

This relatively lower agreement may have been largely due to inaccuracy that was caused by simple regular expressions. As constructed reasoning responses involve more complex causal relationships and different levels of sub-skills, the simple regular expressions may not completely cover all alternative expressions in students' responses. To examine which sub-skill showed high discrepancy between human rating and automated rating, we compared the agreement for the five sub-skills of reasoning

between automated scores and human scores. Results showed very high agreement for the first four sub-skills: 85% for theory, 85% for connection, 92% for data IV/IVR, and 95% for data DV, whereas the agreement for data DVR was only 46%. The confusion-matrix analyses for data DVR revealed that the automated scores used the binary score for this sub-skill (i.e., incorrect versus correct), whereas the humans rated DVR on the four levels mentioned in section 2.3. Binary scoring for DVR in the reasoning of the type-density activity was used to remain consistent with the scoring used in the shape-density activity. In the shape-density activity, there were no partially correct or general answers. Only correct answers (i.e. "density of liquid *is the same"* or "density of liquid *doesn't change*") or incorrect answers were considered. In the type-density activity, responses for DVR included correct answers (i.e. "density of the liquid *decreases from water to oil*"), general answers (i.e. "density of liquid *changes* due to the change of liquid"), partial answers (i.e. "density of *water* is largest"), and incorrect answers. With the rule of least effort, we did not change the algorithms from one activity to another to satisfy the multiple categories of students' responses accounted for by humans. Thus, a large disagreement arose due to the binary scoring used by the automated method versus the four level scoring used by humans.

Even though the criteria that humans and automated methods used to score DVR in reasoning were different, automated scores still yielded pretty good performance. The performance can be improved if the automated method scores reasoning using the same criteria as humans. A future study may explore whether the consistency in DVR between automated and human rating would improve the performance of reasoning scores overall.

4. DISCUSSION

These findings demonstrate that using regular expressions to match key sub-skills of CER with if-then algorithms is a very promising approach to effective and efficient automated scoring of open response scientific explanations. This assertion can be confirmed based on two key factors. First, the automated methods showed very good correlations with human scores for CER in the independent sets of data with the 10-fold cross-validation analyses in the same inquiry task/question as well as in a different inquiry task/question. Previous studies on automated scoring of constructed response items showed that good correlations between automated scores and human scores ranged from .60 to .91 (e.g., [14]). In our study, automated scores for claim and evidence reached .97 in the cross-validation analyses in the same inquiry task/question. When transferred to a different inquiry task/question, results remain .97 for evidence and .94 for claim. These results greatly exceed the current state of research on automated scoring of scientific explanations, as they are almost perfectly correlated with human scoring of claim and evidence scores. Even for reasoning using evidence, a more complex task, results were good as well, ranging from .84 to .85. One explanation for the slightly lower performance of automated reasoning scores is that the agreement between humans was lower relative to agreement for claim and evidence (.88 versus .99) due to the complexity of the reasoning task. Another explanation is that the regular expressions and algorithms applied across different tasks were the same. If we modify regular expressions to satisfy each activity, the performance of automated scoring for reasoning will likely increase.

Second, the sub-skill features that were extracted by regular expressions along with if-then algorithms not only consistently predicted human scores, but also were simple to implement. A central factor to the success of this method was that experts were able to generate accurate regular expressions to identify sub-skills of explanations in science inquiry. More specifically, experts knew how to identify the sub-skills of CER, how to develop a fine-grained rubric to guide human and machine scoring, and how to generate nearly-complete regular expressions to capture as many alternative expressions as possible in students' responses. The use of appropriate regular expressions was key to the success of our automated scores. Regular expressions were easier and quicker to generate for simple sub-skills such as IV, IVR, and DV for claim and data in evidence. For more complex sub-skills, such as DVR and theory, more time was needed to develop sets of alternative expressions. However, once the algorithms yield good performance, only a slight modification is needed for different activities. Compared to manual scoring, the time and effort that was spent on the development of automated scoring was worthwhile. Another key to the success of our automated scoring method was the development of the fine-grained rubric. Our rubric was finalized over many iterations. When we used more general rubrics, the inter-rater reliabilities for reasoning were very low ($r = .50$). With the fine-grained rubric, the reliabilities increased to .88. The high agreement between human coders guaranteed the possibility of high agreement between human scores and machine scores.

The success of automated scoring for open responses in science inquiry will greatly contribute to science education by making possible immediate individualized feedback on students' explanations, as well as adaptive instruction and scaffolding. The implementation of automated scoring in computer-assisted learning and assessment systems will provide students with instant feedback on their constructed CER, which will allow students to immediately know their strengths and weaknesses with regard to scientific explanations. Teachers could then use the explicit feedback from automated scoring to adapt instruction based on what students need. In addition, the automated scoring of CER in science inquiry will advance the development of computer-assisted systems for inquiry, such as Inq-ITS. Inq-ITS has used automated scoring to implement immediate feedback and scaffolding for inquiry skills involved in "doing" science, such as formulating a question/hypothesis, collecting data, analyzing data, and warranting claims. Automated scoring could also be used to align students' "doing" science skills with their science "writing" skills. The alignment of sub-skills involved in "doing" with "writing" during inquiry will allow for comparison of students' conceptual knowledge with their ability to communicate such knowledge. Thus, this automated scoring approach truly advances science education by meeting the comprehensive assessment criteria that NGSS [21] demands: science assessments that include both students' understandings of core ideas, their skills at conducting inquiry, as well as their skills at effectively articulating what they know by generating a claim and evidence for that claim, and articulating their reasoning linking their claim to their evidence.

Even though the automated methods for scientific CER demonstrated good performance, there is one limitation that needs to be addressed in future studies. Namely, regular expressions for reasoning may be modified to adapt to each task/question to align with criteria used by humans. In doing so, the accuracy may be improved.

5. ACKNOWLEDGMENTS

The research reported here was supported by Institute of Education Sciences (R305A120778) to Janice Gobert. Any opinions are those of authors and do not reflect the views of these funding agencies, cooperating institutions, or other individuals.

6. REFERENCES

[1] Bejar, I. I. 2012. Rater cognition: implications for validity. *Educ. Meas.* 31 (Sept. 2012), 2-9.

[2] Cohen, J. 1968. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychol. Bull*. 70 (Oct. 1968), 213-220.

[3] Gerard, L. F., Ryoo, K., McElhaney, K. W., Liu, O. L., Rafferty, A. N., and Linn, M. C. 2016. Automated guidance for student inquiry. *J. Educ. Psychol.* 108 (Jan. 2016), 60-81.

[4] Gobert, J. D. 2016. Op-Ed: Educational data mining can be leveraged to improve assessment of science skills. (May 2016). Retrieved from US News & World Report: https://www.usnews.com/news/articles/2016-05-13/op-ed-educational-data-mining-can-enhance-science-education

[5] Gobert, J. D., Baker, R. S., and Sao Pedro, M. A. 2014. Inquiry skills tutoring system. (Jan. 2014). US Patent no. 9,373,082, Filed Feb. 1st., 2013, Issued Jan. 29th., 2014.

[6] Gobert, J. D., Pallant, A. R., and Daniels, J. T. 2010. Unpacking inquiry skills from content knowledge in geoscience: a research and development study with implications for assessment design. *Int. J. Learn. Technol.* 5 (June. 2010), 310-334.

[7] Gobert, J. D., Sao Pedro, M. A., Baker, R. S., Toto, E., and Montalvo, O. 2012. Leveraging educational data mining for real-time performance assessment of scientific inquiry skills within microworlds. *J. Educ. Data Min.* 4, 111-143.

[8] Gotwals, A. W., and Songer, N. B. 2010. Reasoning up and down a food chain: using an assessment framework to investigate students' middle knowledge. *Sci. Educ.* 94 (Oct. 2010), 259-281. DOI= http://dx.doi.org/10.1002/sce.20368.

[9] Ha, M., Nehm, R. H., Urban-Lurain, M., and Merrill, J. E. 2011. Applying computerized-scoring models of written biological explanations across courses and colleges: prospects and limitations. *CBE-Life Sci. Educ.* 10 (Sept. 2011), 379-393.

[10] Lee, H. S., Liu, O. L., and Linn, M. C. 2011. Validating measurement of knowledge integration in science using multiple-choice and explanation items. *Appl. Meas. Educ.* 24 (Mar. 2011), 115-136.

[11] Li, H., Gobert, J., and Dickler, R. 2017. Dusting off the messy middle: Assessing students' inquiry skills through doing and writing. In *Artificial Intelligence in Education: Lecture Notes in Computer Science* (Wuhan, China, June 25-28, 2017). AIED '17. Spinger, China.

[12] Li, H., Shubeck, K., and Graesser, A. C. 2016. *Using Technology in Language Assessment.* Bloomsbury Academic, London, UK.

[13] Liu, O. L., Brew, C., Blackmore, J., Gerard, L., Madhok, J., and Linn, M. C. 2014. Automated scoring of constructed-response science items: prospects and obstacles. *Educ. Meas.* 33 (Mar. 2014), 19-28.

[14] Liu, O. L., Rios, J. A., Heilman, M., Gerard, L., and Linn, M. C. 2016. Validation of automated scoring of science assessments. *J. Res. Sci. Teach.* 53 (Jan. 2016), 215-233.

[15] Matthews, K., Janicki, T., He, L., and Patterson, L. 2012. Implementation of an automated grading system with an adaptive learning component to affect student feedback and response time. *J. Inform. Syst. Educ.* 23 (Apr. 2012), 71-83.

[16] McNeill, K., Lizotte, D.J., Krajcik, J., and Marx, R.W. 2006. Supporting students' construction of scientific explanations by fading scaffolds in instructional materials. *J. Learn. Sci.* 15 (Nov. 2006), 153-191.

[17] Moharreri, K., Ha, M., and Nehm, R. H. 2014. EvoGrader: an online formative assessment tool for automatically evaluating written evolutionary explanations. *Evol.* 7 (Aug. 2014), 15.

[18] Moussavi, R., Gobert, J., and Sao Pedro, M. 2016. The effect of scaffolding on the immediate transfer of students' data interpretation skills within science topics. In *Proceedings of the 12th International Conference of the Learning Sciences.* (Singapore, June 20-24, 2016). ICLS '16. Scopus, Ipswich, MA, 1002-1005.

[19] Myford, C., and Wolfe, E. 2009. Monitoring rater performance over time: a framework for detecting differential accuracy and differential scale category use. *J. Educ. Meas.* 46 (Dec. 2009), 371-389.

[20] Nehm, R. H., Ha, M., and Mayfield, E. 2011. Transforming biology assessment with machine learning: automated scoring of written evolutionary explanations. *J. Sci. Educ. Technol.* 21 (Feb. 2012), 183-196.

[21] Next Generation Science Standards (NGSS) Lead States. 2013. *Next Generation Science Standards: For States, by States*. The National Academies Press, Washington, DC.

[22] Sao Pedro, M. 2013. Real-time assessment, prediction, and scaffolding of middle school students' data collection skills within physical science simulations. Ph.D. Dissertation. Worcester Polytechnic Institute, Worcester, MA.

[23] Sao Pedro, M. A., Baker, R. S., Gobert, J. D., Montalvo, O., and Nakama, A. 2013. Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill. *User Model. User-Adap.* 23 (Mar. 2013), 1-39.

[24] Sao Pedro, M. A., Gobert, J. D., & Baker, R. S. 2014. The impacts of automatic scaffolding on students' acquisition of data collection inquiry skills. Roundtable presentation at *2014 American Educational Research Association Annual Meeting* (Philadelphia, Pennsylvania, April 03-07, 2014).

[25] SPSS Inc. 2006. *SPSS Text Analysis for Surveys 2.0 User's Guide.* SPSS Inc, Chicago, IL.

[26] Thompson, K. 1968. Regular expression search algorithm. *Commun. ACM.* 11 (June. 1968), 419–422.

[27] Toulmin, S. 1958. *The Uses of Argument.* Cambridge University Press, Cambridge, MA.

[28] Wainer, H., and Thissen, D. 1993. Combining multiple-choice and constructed-response test scores: toward a Marxist theory of test construction. *Appl. Meas. Educ.* 6 (Apr. 1993), 103-118.

[29] Weston, M., Parker, J., and Urban-Lurain, M. 2013. Comparing formative feedback reports: human and automated text analysis of constructed response questions in biology. In *Annual Conference of the National Association on Research in Science Teaching* (Rio Grande, Puerto Rico, April 06-09, 2013). NARST '13.

[30] Williamson, D. M., Xi, X., and Breyer, F. J. 2012. A framework for evaluation and use of automated scoring. *Educ. Meas.* 31 (Mar. 2012), 2-13.

Can Typical Behaviors Identified in MOOCs be Discovered in Other Courses?

Truong-Sinh An
Beuth Hochschule für Technik Berlin
Luxemburger Str. 10
13353 Berlin, Germany
truong-sinh.an@beuth-hochschule.de

Christopher Krauss
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
christopher.krauss@fokus.fraunhofer.de

Agathe Merceron
Beuth Hochschule für Technik Berlin
Luxemburger Str. 10
13353 Berlin, Germany
agathe.merceron@beuth-hochschule.de

ABSTRACT

The emergence of Massive Open Online Courses (MOOCs) has enabled new research to analyze typical behaviors of learners. In this paper, we investigate whether this research is generalizable to other courses that are backed by a learning management system (LMS) as MOOCs are. Building on methods developed by others, we characterize individual learning behaviors in different ways taking into account specificities of the LMS we use. We then apply clustering techniques to uncover typical behaviors in two university courses. One course, JavaFX, teaching about the software programming framework, has been offered as a supplementary online course to students enrolled in an online degree. Enrolling in this course was voluntary and students did not earn any credit towards their degree; in this sense, the JavaFX course bears similarities to a MOOC though it is neither *massive* nor *open to everybody*. The other course is a classical face-to-face course on Advanced Web Technologies (AWT) backed by our LMS; students earn a degree when they pass the final exam. It turns out that the different characterizations of individual learning behaviors are consistent for the JavaFX course and uncover typical behaviors reminiscent of those found by others in MOOCs, while they aren't as applicable to the AWT course. However, typical behaviors found in the AWT course give insights on styles that lead to better marks.

Keywords

MOOC, Typical behaviors, X-means clustering

1. INTRODUCTION

The emergence of MOOCs with the general observation of their low completion rates has triggered new research to analyze typical behaviors of learners in MOOCs and brought forth evidence for various engagement/disengagement patterns such as *completing*, *auditing*, *disengaging* and *sampling, as* proposed by Kizilcec et al. [1]. In their paper, Kidzínski et al. [2] write that categorization schemes as found in [1] and others "remain robust in terms of generalizability within the MOOC's context, but they are hard to generalize outside of it". In this paper, we tackle that claim. We investigate whether this research can offer interesting insights to other courses that are backed by a learning management system (LMS), even though analyzed courses are not necessarily massive nor open, and even not completely online.

We investigate two courses presented with the Learning Companion App (LCA) [3]. The LCA is a LMS designed in the first place for vocational training. Compared to other LMSs common in higher education like Moodle, the Coursera-platform or edX, LCA has two salient features to encourage self-reflection and support efficient learning. The first feature concerns the learning objectives that need to be associated with each learning object (LO) in the course. All the learning objectives of one chapter are displayed for rating at the beginning and at the end of any chapter. A learner can assess how much s/he knows each learning objective. These self-assessments encourage learners to reflect on their previous knowledge, and on how much they know after learning the chapter. The second feature is a recommendation engine that suggests learners what to learn next [4]. Learners are free to consult these recommendations. Comprehensive user interactions are stored as xAPI statements [5]. The LCA is independent of any topic and any institution and, therefore, can be used in other contexts and for other courses.

The two courses considered in this study, JavaFX and Advanced Web Technologies (AWT) have taken place in the context of higher education. The JavaFX course has been offered as an optional online course to students enrolled in an online degree in computer science. These students learned to program graphical interfaces with the older framework Swing instead of the newer framework JavaFX. By taking part in this course, students did not earn any mark for their studies, they only increase their knowledge of the topic. The AWT course targeted master computer science students. It was a classical face-to-face course taught with the support of the LCA in winter semester 2016/17. When enrolled in this course, students usually had the aim of passing the final exam and earn the corresponding credits for their master degree.

In this study, we follow and adapt the approach of [1, 6] and explore several different ways of qualifying individual learning behaviors as similar. It turns out that for the JavaFX course, these different ways are consistent and uncover two to three typical learning behaviors reminding those exhibited by Kizilcec et al. [1]. For the AWT course, only one way of qualifying behaviors turns out to be sensible. The uncovered typical learning behaviors

from both courses match those exhibited in [1, 6] and give insight on styles that lead to better marks.

This paper is organized as follows. Related works are discussed in Section 2. Specificities of courses in our LMS, the Learning Companion App, are presented in Section 3. Subsequently, different ways of characterizing individual learning behaviors are explained and typical learning behaviors found in both courses are presented and discussed. Conclusion and future works are given in Section 5.

2. RELATED WORK

Kizilcec et al. [1] investigated learners' engagement in courses from Coursera which consist of weekly videos and assessments, and proposed four typical engagement / disengagement patterns that they call

- *Completing*: "learners who completed the majority of the assessments offered in the class",
- *Auditing*: "learners who did assessments infrequently if at all and engaged instead by watching video lectures",
- *Disengaging*: "learners who did assessments at the beginning of the course but then have a marked decrease in engagement (their engagement patterns look like Completing at the beginning of the course but then the student either disappears from the course entirely or sparsely watches video lectures)" and
- *Sampling:* "learners who watched video lectures for only one or two assessment periods".

These categories have been identified in three courses; however, their proportions differ in each course. To discover these categories, they have first characterized a student by a tuple giving her/his status each week: "*on track [T]* (did the assessment on time), *behind [B]* (turned in the assessment late), *auditing [A]* (didn't do the assessment but engaged by watching a video or doing a quiz), or *out [O]* (didn't participate in the course in that week)" [1].

In an attempt to replicate the work of [1], Ferguson and Clow [6] suggest that the methodology used to uncover typical learning behaviors in a MOOC's context does not necessarily generalize to another MOOC adopting different elements of pedagogy and learning design. Since the courses analyzed in [6] follow a social constructivist pedagogy, Ferguson and Clow adapt the methodology of [1]. They consider also participation in discussions and end up with 10 values to characterize the weekly status of a student, instead of the four values T, B, A and O introduced in [1]. They have identified the following typical learning behaviors: *Samplers* ("Learners in this cluster visited, but only briefly", similar to sampling above), *Strong Starters* ("these learners completed the first assessment of the course, but then dropped out"), *Returners* ("these learners completed the assessment in the first week, returned to do so again in the second week, and then dropped out"), *Mid-way Dropouts* ("these learners completed three or four assessments, but dropped out about half-way through the course"), *Nearly There* ("these learners consistently completed assessments, but then dropped out just before the end of the course"), *Late Completers* ("this cluster includes learners who completed the final assessment, and submitted most of the other, but were either late or omitted some") and *Keen Completers* ("this cluster consists of learners who completed the course diligently, engaging actively throughout" similar to completing above). The two approaches in [1, 6] share the same principle of selecting a priori features that are sensible to describe a student's individual engagement, and then use K-means clustering to discover typical learning behaviors.

Gelman et al. [7] adopt a different, more bottom-up approach to discover typical behaviors: they use a set of 21 features that they can extract week by week from the log data and adapt non-negative matrix factorization to obtain weekly behaviors that are supported by a combination of those features. This approach is attractive because it does not need a careful selection of features to characterize the behavior of a student; instead, the algorithm selects and combines features from the set it receives as input. A difficulty lies in the interpretation and the practical use of the discovered behaviors. While an *auditing* behavior "learners who did assessments infrequently if at all and engaged instead by watching video lectures" [1] is easy to derive, it is less clear what a weekly *deep* behavior "the associated students must have spent a long time on a single resource" [7] means for educators.

In this paper, we adopt the first approach and adapt it to our context, taking inspiration from the work in [6].

3. COURSES IN THE LCA

The Learning Companion App (LCA) is a whole infrastructure that can be thought of as LMS equipped with a repository for learning objects, a recommendation engine and a learning analytics module. It is at the same time an App in responsive design which is the entry point for students to access courses, learning objects (LOs) and lecture schedule as well as to get recommendations for the next best contents to be learnt; furthermore, it triggers the tracking of all relevant user interactions [8].

In LCA, each learning object at the lowest level is paired with its metadata that includes at least one learning objective, a typical learning time and its prerequisites. A learning object can be a piece of text, a video, an exercise (similar to an exercise of an assessment in a MOOC), an animation, even a downloadable document and so on. Learning objects are bundled into learning units and a course is essentially a sequence of learning units. The learning objectives of a learning unit are the union of the learning objectives of its learning objects. A learning unit is rendered in the LCA as an "accordion" GUI element with a specific sequential structure. The top item of the accordion view that can be opened is the list of the learning objectives of that unit. Learners can rate each learning objective and so indicate how much they know already on that topic, from 1 "know nothing" to 5 "expert". We call this list *self-assessments*. This item is followed by the sequence of the LOs of that unit. The user can interact with the learning objects by clicking on the title in the accordion view whereupon the requested content is presented. The user is only shown one learning object at any time so that s/he can concentrate fully on this content. Following the sequence of LOs, the next item in the accordion view is again the list of learning objectives. By rating them, a student can reflect on how much s/he knows after learning the unit. The next item in the accordion view allows students to provide feedback on the typical learning time for that unit (from 1, "way too little time" to 5, "way too much") and give comments. The last item in the accordion view opens a discussion thread on that unit. Apart from its sequence of learning units, a course contains a schedule which specifies dates for the start and end of the course, as well as when each learning unit should be learned.

All users' interactions are stored using the xAPI specification [5] in the open-source learning record store called Learning Locker[1]. The accordion view allows inferring how long any item of the view is opened. Typical mined data include number of clicks on all items of the accordion view (self-assessments, LOs, feedback, discussion threads), time an item is open, answers and performance in exercises, ratings of pre- and post-study self-assessments, feedback, messages of discussion threads. Note that a student can access any LO directly by clicking on the recommendations. For this study, this does not change the kind of interactions that are stored.

The two courses discussed in this paper make all the learning material available from the start of the course to encourage self-pacing and self-organization of students. Furthermore, the time schedule of the courses is indicative only, in the sense that there is no penalty if someone does not follow the schedule. Finally, in both of them, students did not post in the discussion threads; they only wrote (few) comments in the feedback area. However, the two courses differ significantly in their didactical organization and contents.

The JavaFX online course, available for a period of 11 weeks, offers an introduction into the FX-Framework for the development of platform independent Java applications and targets bachelor computer science students. This course is suggested as an optional online course to students enrolled in an online computer science bachelor course. By taking part in this course, students do not earn any mark for their studies, only knowledge for themselves.

It comprises three learning units. Each learning unit has about five learning objectives and contains about fifteen to thirty LOs (units are not of equal length). About half of the LOs are texts to explain concepts and example programs, and half are exercises (single/multiple choice, cloze tests and so on). The last LO before the self-assessment of the learning objectives is a comprehensive programming task; students can send their program per email and obtain a manually commented evaluation. Based on the educational discussion on MOOCs, Daniel [9] pointed out that "students seek not merely access, but access to success". However, success can be different for each student. Driven by this consideration, a specific LO has been added to this course allowing each student to rate her/his motivation on a scale from 0 (do nothing) to 100 (engage thoroughly with everything offered). 51 students enrolled in this course; however, there were 23 no-shows (defined in [10] as "people register but never login to the course while it is active"). Only the remaining 28 students are considered for the analysis in this paper. The 28 users generated 3624 xAPI statements in total during the course.

Advanced Web Technologies (AWT) targets master computer science students. Technical experts teach in 12 weekly presence lectures diverse topics that are of interest for future web developers – from web technology basics, such as HTML, over media delivery and content protection, to personalization through recommender systems, and the Internet of Things. The lectures are mostly held with slides created in PowerPoint showing definitions, specifications, and source code, animations for concepts and videos for practical examples. The about 1000 presented slides are converted to digital learning objects, one slide being a single LO, and grouped into 105 learning units for the representation in the LCA – with videos, animations and additional multiple-choice questions at the end of the learning units. Moreover, as some students still want to learn with a printed version of the slides, the last LO of the accordion view is a downloadable PDF file containing all the slides of the unit.

142 students enrolled for AWT in winter semester 2016/17; however, there were 43 no-shows. Only the remaining 99 students are considered for the analysis in this paper. Especially in the first weeks before the official registration deadline, students frequently change their mind regarding participating in specific courses – which might explain the high loss ratio of the participants. At the end of the course, students can earn credits by completing an one-hour exam consisting of 50 multiple choice questions and 5 bonus questions. Exactly 75 students completed the final exam (even two who did not used the LCA) and the average mark was 1.90 (only one student failed the exam; note that the best mark is 1.0 and the worst possible mark is 5.0). The 99 users generated 92825 xAPI statements in total during the course.

In contrast to the courses offered by [1], [6] and the JavaFX course, the primary goal for students of AWT is to pass the final exam. AWT does not offer any intermediate assessment. Students access online material, first and foremost, for the wrap-up of face-to-face lectures and for exam preparation.

4. METHODOLOGY AND RESULTS

In our context, there are multiple sensible ways to compare students in their learning behaviors. Because this time schedule is purely indicative for students and all the materials are available from the start of the course, we compared students on how they have interacted with the course independently of time. In this paper, we investigate four such ways.

Clicks only: In this way, we consider only click counts per learning object. A student is represented by a vector that represents how many times s/he has clicked each element of the whole course. In this way, two students are similar if they access almost the same learning objectives, learning objects, feedback, and motivation (for JavaFX only as AWT does not have this feature) a similar number of times.

Elapsed time: In this way, elapsed time spend on that learning object replaces click count. A student is represented by a vector that has the size of all learning objects of the course. The learning objectives, feedback, and motivation are not considered because the time spend is not tracked individually for these features. Two students are similar if they spend a similar overall time on the same learning objects (texts, videos, exercises, etc.). The overall time is the sum of the elapsed times in each visit.

Assessment scores: In this way, we consider performance on all assessments, including programming tasks of the JavaFX course. A student is represented by a vector that has the size of all assessments; values are ratings given in all self-assessments, marks earned in all exercises, rating given in feedback and motivation (AWT does not have the motivation feature). The final exam for AWT is not considered. Two students are similar if they achieved similar scores on all assessments.

Elapsed time and assessment scores: In this way, we consider a combination of the latter two: elapsed time on what students look at (texts, videos and so on) and scores on what students answer (self-assessments, exercises and so on). Two students are similar if they spend a similar overall time on similar learning objects

[1] Learning Locker. See: https://learninglocker.net/

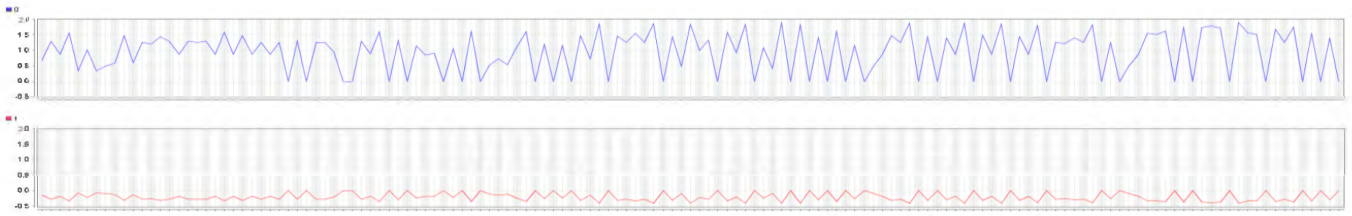


Figure 1: Plot of the centroids of the 2 clusters returned by X-means in the JavaFX course. The x-axis represents all the elements of the course (learning objectives, learning objects etc.); the y-axis gives the average normalized number of clicks per element.

such as texts, videos, slides and so on (that are not exercises) and achieve similar scores on all assessments.

We used RapidMiner[2] and applied the X-means clustering algorithm with Euclidean distance. X-means finds an optimal number of clusters and is known to find fewer clusters than K-means [11]. Due to the size of the vector representing each student (in the way *Clicks only* a student is represented by a vector with 143 values in the JavaFX course) and the small data sets, clustering is challenging. Furthermore, in RapidMiner, X-means is implemented in such a way that it will always find a minimum of two clusters, even if the data is uniform. To validate that the data does cluster naturally, we applied also K-means and checked for the drop in the curve plotting K against the sum of squared errors (which corresponds to the *average within distance* of RapidMiner). Values of clicked counts and elapsed time have been normalized. Assessment values like marks in exercises or self-assessments are already stored as scaled values.

4.1 JavaFX

X-means returns exactly the same two clusters for three approaches: *Clicks only*, *Elapsed time* and *Elapsed time and assessment scores* (the results of the fourth approach are described later on). Figure 1 shows a visualization of these two clusters for the *Clicks only* way; it lists all elements of the course on the X-axis and shows the corresponding normalized number of clicks of the clusters' centers on each element. The first cluster (cluster 1, the blue line in the upper diagram of Figure 1) consists of 5 students who engage with many elements such as self-assessments, learning objects and also interact with the automatically generated features like feedback. When sorting the students according to the number of distinct elements they have accessed in the course, these 5 students come on top. On average, students in this cluster have accessed 72 distinct course elements. If the elements are restricted to the exercises only, as they best match assessments in MOOCs, these 5 students remain on top: except for one, who performed 15 exercises, they have performed 25 to 30 exercises out of 34. The other 23 students in the course, represented in the second cluster (cluster 2, red line and bottom diagram of Figure 1) accessed the learning objects less often and did very few self-assessments. On average, students in this group have accessed 10 distinct elements of the course and solved exercises infrequently, if at all four times or less. Transferred to the categories in [1], we find that these two patterns of engagement are reminiscent of *completing* and *auditing* but without any reference to time. In [1] it is clear that *completing* students have solved assessments week by week because assessments are available in the course week by week only. In our course, *completing* students could have solved exercises regularly, or all during a few weeks only, depending on their own time-management.

The K-means algorithm finds an optimal set of 4 clusters; see the upper elbow-curve of Figure 2 with the drop when k is 4. One cluster matches exactly cluster 2 found with X-means, while the cluster with 5 students is split into 3 clusters. This finding shows that data naturally clusters; however, the two clusters returned by X-means are more interpretable.

X-means returns three clusters when using *Assessment scores*. Cluster 1 with the pattern *completing* is also found here. Cluster 2 above is now split into two clusters: one with 18 students and cluster 3 with 5 students. What distinguishes these 5 students from the remaining 18 students is that they answered self-assessments and engaged with exercises mostly from the first unit of the course, hardly from the follower units. They correspond to *disengaging* in [1] although beginning of the course does not refer to time but to the sequence of the units that are displayed in the LMS. K-means algorithm finds an optimal set of 5 clusters; as before, the *completing* cluster is split into 3 clusters.

At first, it may be surprising that the three characterizations: *Clicks only*, *Elapsed time* and *Elapsed time and assessment scores* give exactly the same clusters: *completing* and *auditing*. With some consideration, this result is understandable: what distinguishes the most two learners is when one has accessed an element and the other not. A *completing* student has accessed

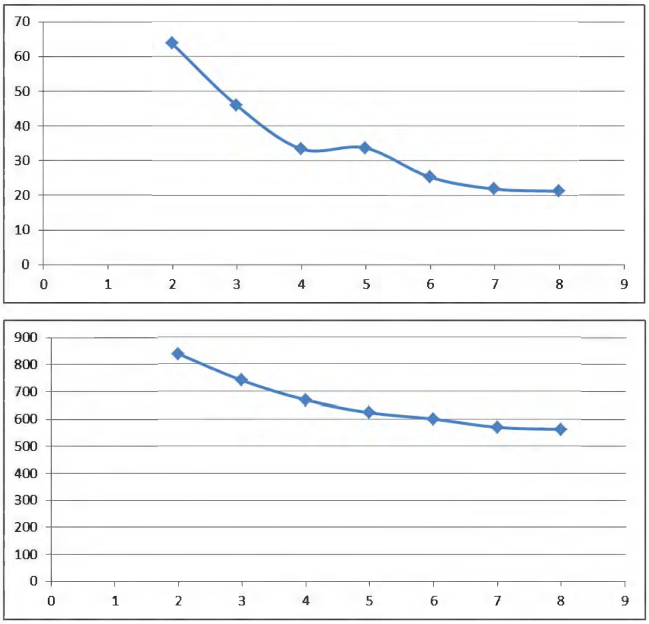


Figure 2: Plot of K against average within distance scenario clicks only for JavaFX (above) and AWT (below).

[2] Rapid Miner. See: https://rapidminer.com/

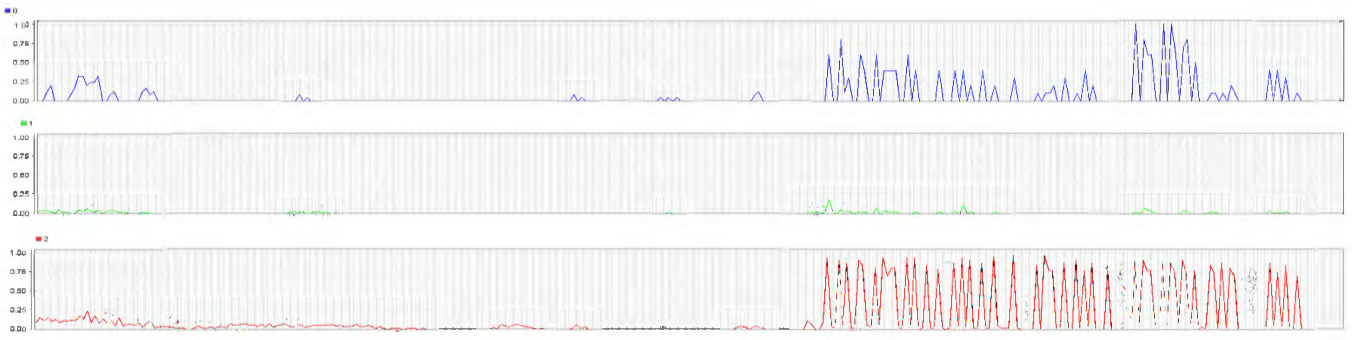


Figure 3: Plot of the centroids of the 3 clusters returned by X-means in the AWT course. The x-axis lists all the assessments of the course (self-assessments left part, exercises and feedback on time right side); the y-axis gives the scaled score of the center per element.

much more elements of the course than an *auditing* student; these two behaviors are discovered by X-means. The characterization *assessment scores* reduces the number of features used to perform clustering (interactions with LOs such as text or videos are omitted) and allows for distinguishing a sub-category in the *auditing* group: *disengaging*; those learners are completing activities primarily in the first unit of the course and then stop. They have hardly engaged with the course in the following units, what makes them similar to *auditing* students in the three other ways: they have engaged infrequently with exercises and have looked at few learning objects.

4.2 AWT

The first three approaches (*Clicks only*, *Elapsed time* and *Elapsed time and assessment scores*) lead to no meaningful results for the AWT course. On the one hand, K-means does not show a natural clustering of the data for any of these ways: plotting K against the *average within distance* does not show any drop, as the curve for the AWT course in Figure 2 bottom shows. On the other hand, these three ways are not really adequate to describe the engagement of an individual student due to the digital content of this course: at the end of each unit, there is a .pdf file containing all the slides of this unit. A student might download only the .pdf file of each unit and look at it as much as s/he wants, another student might access all the slides online multiple times. From the interactions that are stored and evaluated, these two students look very different, yet their learning behaviors are similar. At the beginning of the course, 66 Students have requested PDFs, and this number of students decreased to the end of the course to 19. One third of all students have requested all PDFs.

In contrast, for the *Assessment scores* approach, X-means generated three definite clusters. Figure 3 shows a visualization of these three clusters; it lists all assessments of the course on the X-axis and shows the corresponding score of the clusters' centers on each element. Two parts are clearly distinguishable: a rather flat left part and a right part where the blue (top) and the red (bottom) lines show spikes. The rather flat left part corresponds to the self-assessments; generally, not many students rated themselves. The right part corresponds to the exercises and student feedback Cluster 1 contains 9 students inclusive the one who did not pass the final exam (the upper diagram with the blue line of Figure 3). Students in this cluster provided self-assessments in the first three units, and worked out exercises but did not achieve good scores. They remind of *Strong Starters* and *Returners* proposed in [6] when this vocabulary is adapted to the sequential order of the units instead of the first weeks of the course. To some extent, they exhibit also some kind of *completing* pattern in terms of exercises, because they completed almost half of them: on average 22 from a total of 48. Their average mark in the final exam is 2.03 which is slightly worse than the general average of 1.90. The biggest cluster contains 64 students (cluster 2, the diagram in the middle with the green line of Figure 3) and is similar to the pattern *auditing* because they did exercises infrequently if at all: on average 1 out of 48. However, they did access .pdf files. All learners who did not participate in the final exam fall into this cluster. The average mark of the students in this cluster who participated in the final exam is 2.23 (no-shows are neglected), which is below the general average. The last cluster contains 26 students and shows a *completing* pattern (cluster 3, the bottom chart with the red line of Figure 3). If one sorts the students according to the number of distinct exercises they have solved in the course, 25 of these 26 students are the top 25. They have worked on nearly all the exercises, on average 42 out of 49, and completed almost all of them correctly. The final exam mark in this completing cluster reaches 1.50 on average, a better mark than the overall average of 1.90. The last two clusters are interesting: a *completing* student does well in the final exam, while an *auditing* student does worse in the final exam or even does not attend it. Although, as opposed to [1], these patterns do not tell anything on when students accessed the assessments in the time schedule.

K-means algorithm finds an optimal set of 4 clusters. It finds exactly the same big cluster of 64 students and finds almost the same first cluster as X-means does. However, it splits the last cluster to isolate three students. Students in both groups still solved in average 42 exercises but they differ in how they engaged with self-assessments. The small group of 3 students rated 74 self-assessments in average and the other students only rated 3 self-assessments in average in the first units of the course.

5. DISCUSSIONS AND FUTURE WORK

Considering the particularities of our courses, we have defined four meaningful ways of characterizing an individual learning behavior. We have used X-means clustering to extract typical learning behaviors from two distinct university courses, an optional online JavaFX course and a compulsory face-to-face course about Advanced Web Technologies. Because of the small data sets, particularly for the JavaFX course, clustering is challenging. We found that students do not act at random. In the JavaFX course, we could derive evidence behaviors that remind of patterns found in [1]: *completing*, *auditing*, and *disengaging*. Only the *Assessment scores* way delivers reliable clusters for AWT. From the three clusters uncovered by X-means, two are particularly interesting. All students that were ultimately not

participating in the final exam were located in the *auditing* cluster. Other students in that cluster, who participate in the final exam, tend to do less well than average. Students of the *completing* cluster tend to pass the exam with very good marks. Note that *completing*, *auditing*, and *disengaging* in this paper are similar to [1] in terms of which kind of learning material has been accessed frequently or not; as opposed to [1], our approach does not provide information on when in the time schedule the material has been accessed.

The present results suggest that typical behaviors found in MOOCs can be transferred to other courses - with care. This situation bears similarities with predicting students at risk of deserting a course. Numerous articles show that models with good predictive power can be built to predict drop-off and also the performance of students in a course. These articles show also that there is no set of features and no classifier that works best in all contexts: no one-size fits all. On the contrary, the set of features and classifiers needs to be adjusted to the data and setting at hand to achieve a good predicting power. The work of [2] also supports this view for MOOCs. Our results suggest that the situation is the same for typical behaviors. We adjusted methods of others to our context and were able to extract interesting and interpretable typical behaviors from relatively small data sets. This work considers rather simple features like clicks and elapsed time. Future work should focus on a more sophisticated feature extraction.

In our setting, there is a time schedule, even if it is indicative only. It could make sense to devise ways of characterizing an individual behavior taking this time schedule into account. The method of [1] needs careful adaptation because a learner might be *on track* or *behind* and might also be *early*. Works on these lines have already begun. Preliminary work shows that four of the five students of the *completing* cluster of the JavaFX course began only after three weeks to engage with the course, while the majority of the *completing* cluster of the AWT course engaged with the course regularly each week. Another future work is to reflect on implications for the recommendation engine and the learning analytics module. Should the recommendation engine be adjusted to different typical behaviors for example? We plan to integrate these findings in the overall behavioral feedback shown to students.

6. ACKNOWLEDGMENTS

The authors would like to thank the whole Smart Learning team for their great work and many constructive ideas. We would also like to thank the instructors of the AWT course: Stephan Steglich, Louay Bassbouss, Stefan Pham, and André Paul. Special thanks go to Christian Fuhrhop, who proofread this paper and made a lot of practical suggestions.

The Smart Learning project is sponsored by the German Federal Ministry of Education and Research (Bundesministerium fuer Bildung und Forschung – BMBF) under the project funding number 01PD14002A.

REFERENCES

[1] Kizilcec, R.F., Piech, C., and Schneider, E. 2013. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In Third international conference on learning analytics and knowledge LAK'13, April 8-12, Leuven, Belgium. ACM, p. 170-179. doi>10.1145/2460296.2460330.

[2] Kidzínski, L., Sharma, K., Boroujeni, M. S., Dillenbourg, P. 2016. On generalizability of MOOC models. In Proceedings of the 9th International Conference on Educational Data Mining, EDM'2016, Raleigh, USA, June 29-July 2, p. 406-411.

[3] Krauss, C.; Merceron, A.; An, T.-S.; Zwicklbauer, M.; Arbanowski, S.. The Smart Learning Approach - A mobile Learning Companion Application. In: Proceedings of The Eighth International Conference on Mobile, Hybrid, and On-line Learning (eLmL 2016). IARIA, April 24 - 28, 2016 - Venice, Italy.

[4] Krauss, C. 2016. Smart Learning: Time-Dependent Context-Aware Learning Object Recommendations. Proceedings of the 29th International Florida AI Research Society Conference (FLAIRS-29), AAAI, Key Largo, p. 501-504.

[5] Kevan, J. M. and Ryan P. R., "Experience api: Flexible, decentralized and activity-centric data collection," Technology, Knowledge and Learning, vol. 21, no. 1, 2016, pp. 143–149.

[6] Ferguson, R. and Clow, D. 2015. Consistent commitment: Patterns of engagement across time in Massive Open Online Courses (MOOCs). Journal of Learning Analytics, 2(3), 55–80. http://dx.doi.org/10.18608/jla.2015.23.5.

[7] Gelman, B., Revelle, M., Domeniconi, C., Johri, A., and Veeramachaneni, K. 2016. Acting the Same Differently: A Cross-Course Comparison of User Behavior in MOOCs. In Proceedings of the 9th International Conference on Educational Data Mining, EDM'2016, Raleigh, USA, June 29-July 2, p. 376-381.

[8] An, T.-S., Dubois, F., Manthey, E., Merceron, A. 2016. Digitale Infrastruktur und Learning Analytics in Co-Design. In Proceedings of the Workshop Learning Analytics, co-located with the 13th e-Learning Conference of the German Society for Computer Science, Potsdam, Germany, September 11, 2016. http://ceur-ws.org/Vol-1669/.

[9] Daniel, J. 2012. Making sense of MOOCs: Musing in a maze of myth, paradox and possibility. Journal of Interactive Media in Education, 2012(3). http://dx.doi.org/10.5334/2012-18.

[10] Hill, P.. Emerging student patterns in MOOCs: A (revised) graphical view, 2013. http://mfeldstein.com/emerging-student-patterns-in-moocs-a-revised-graphical-view/ accessed March 3, 2017.

[11] Pelleg, D., and Moore, A. 2000. X-Means: Extending K-means with Efficient Estimation of the number of Clusters. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29, p. 727-734.

Gaze-based Detection of Mind Wandering during Lecture Viewing

Stephen Hutt[1], Jessica Hardey[1], Robert Bixler[1], Angela Stewart[1], Evan Risko[2] and Sidney K. D'Mello[1]
[1]University of Notre Dame, [2]University of Waterloo
118 Haggar Hall, Notre Dame, IN, 46556, USA
{shutt, sdmello}@nd.edu

ABSTRACT
We investigate the use of consumer-grade eye tracking to automatically detect Mind Wandering (MW) during learning from a recorded lecture, a key component of many Massive Open Online Courses (MOOCs). We considered two feature sets: stimulus-independent global gaze features (e.g., number of fixations, fixation duration), and stimulus-dependent local features. We trained Bayesian networks using the aforementioned features and students' self-reports of MW and validated them in a manner that generalized to new students. Our results indicated that models built with global features (F_1 MW = 0.47) outperformed those using local features (F_1 MW = 0.34) and a chance-level model (F_1 MW = 0.30). We discuss our results in the context of MOOC development as well as integrating MW detection into attention-aware MOOCs.

Keywords
eye-gaze, Massive Open Online Courses, lecture viewing, intelligent tutoring systems, mind wandering, attention-aware learning

1. INTRODUCTION
Imagine you are giving a lecture on population diversity, most of your audience is engaged; however, one or more of your students are displaying signs of inattentiveness (e.g., dozing off, staring blankly). You may call on such a student in the hope of bringing their attention back to the lecture. You may even suggest a short break if too many students appear to be inattentive. This adaptation to your lecture was only possible because you had the ability to continually monitor your students' levels of attentional focus and to alter your instruction in real-time.

Now imagine you are teaching a Massive Open Online Course (MOOC). Your students are no longer in the same room as you and in many cases are not viewing the lecture at the same time you are delivering it. You no longer have the ability to monitor students' attentional focus and adapt to signs of inattentiveness.

Despite the challenges for educators, MOOCs are an increasingly popular method amongst students for e-learning and distance learning [16]. They have also been popular in traditional learning environments as alternate ways for delivering material [27]. MOOCs are often distributed world-wide to a variety of students across platforms with no limitations on individual participation. While there are some advantages to MOOCs with respect to promoting access, little is known with regard to how they address individual learners' needs. MOOCs have long had issues with extremely high dropout rates [1, 37], far greater than those in 'traditional' classroom environments. Though there has been work tying students' experiences with MOOCs to the dropout rate [37], there has been little exploration as to individual user experiences and trends that lead to retention problems [1, 17].

As a step towards better understanding student engagement within MOOCS, we focus on one form of disengagement called mind wandering (MW). MW is defined as an attentional shift from task-related processing towards internal task-unrelated thoughts [31]. In the context of learning, both lab and field studies have consistently reported MW rates in the 20%-50% range [21, 26, 34]; work looking at specifically recorded lectures showed the MW rates to be 20-45% [26, 34]. Additionally, a recent meta-analysis revealed a negative correlation between MW and performance across a variety of tasks [23]. MW negatively impacts a learner's ability to attend to external events [30], to encode information into memory [29], and to comprehend learning materials [28, 30]. As a result, MW is generally found to have a negative impact on learning outcomes.

Attempts to assuage the cost of MW rely on knowing if MW has occurred. However, detecting MW is no easy task. Although MW is related to other forms of disengagement, such as boredom, behavioral disengagement, and off-task behaviors [2, 3, 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 appear similar to being on-task [31]. Second, the onset and duration of MW cannot be precisely measured because MW can occur outside of conscious awareness [32].

Despite these challenges, there has been some progress toward automatic detection of mind wandering (discussed as related works in Section 1.1). However, almost all of the current MW detectors focus on reading. In contrast, we consider MW detection while students view MOOC-like lectures, building and validating the first gaze-based MW detector during video lecture viewing. We focus on video lectures because they are a core component of many courses and are vital to MOOCs. As MOOCs and lecture capture systems become more popular, we envision a variety of challenges with respect to keeping students engaged when content delivery occurs outside of the classroom with the instructor not even present. In this work, we harness the use of a computer in content delivery to take a step towards an attention-aware MOOCs.

1.1 Related Work

In an early study attempting to detect MW in the context of learning [10], students were asked to read aloud a paragraph about biology, followed by either self-explaining or paraphrasing. Students self-reported how frequently they zoned out on a scale from 1 (all the time) to 7 (not at all). Reports were then grouped as either low (1-3 on the scale) or high (5-7 on the scale). Supervised machine learning methods were trained using acoustic-prosodic features to classify these instances, achieving 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.

Researchers have also 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 textual features (e.g., 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 [11] 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, 8, 11]. 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) [8] or eye-gaze [5], 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 an 11% improvement in detection accuracy above the best individual modality [4].

Beyond reading, Pham et al. [22] provide initial proof that MW detection is possible during lecture viewing. Students watched video lectures on a smart phone using 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 of MW detection in a MOOC-like learning environment.

Hutt et al. [15] focused on detecting MW during learning with an intelligent tutoring system (ITS). Students' eye gaze was tracked with a consumer grade eye tracker as they completed a 30-40 minute learning session with the ITS. Students reported MW by responding to pseudo-random thought probes throughout the session. A variety of supervised classification models were trained to detect MW from eye movements and basic contextual information (e.g., time within session), achieving student-independent MW detection that was 37% greater than chance.

Finally, Mills et al. [18] studied MW detection in the context of viewing a narrative film. This study used a research grade eye tracker to monitor eye movements from which content-free global gaze features (e.g., fixation duration) as well as content specific features were computed. The content specific features were generated from two areas of interest (AOIs): one from the saliency map of the image [14], and one specific to the film being watched. These AOIs were then used in conjunction with eye gaze to generate content specific (local) features (e.g., average distance of fixations from an AOI or intersections with the AOI). The key finding was that, unlike in reading tasks, models built using local features were more successful than those built from global gaze features, achieving a student-independent score of 29% above chance.

1.2 Current Study and Novelty

The novelty of this paper is two-fold. First, we build the first gaze-based detector of MW during video lecture viewing. We focus on eye tracking due to well-known relationships between visual attention and eye-movements. For example, MW has been associated with longer fixation durations [25] and more blinking in reading [33]. We use low-cost consumer-grade eye trackers to collect gaze data from participants as they view a recorded lecture (see Figure 1). Since research grade eye trackers can cost upwards of $40,000, the selection of affordable equipment (less than $150) increases the applicability of this work, enabling its eventual deployment in real world learning environments such as classrooms or students' homes.

Second, we compare MW detection with the more generalizable, global eye gaze features to AOI based local features. Global eye gaze features have previously been successful for detecting MW in learning contexts such as reading [7] and interacting with an ITS [15]; however, recent work involving narrative film comprehension found that AOI based features were more effective in that context [18]. We explore if the differences in visual style and production techniques between a recorded lecture (Figure 1) and a narrative film (Figure 2) influence the effectiveness of local features for detecting MW. This is a critical comparison because the global features are much more generalizable.

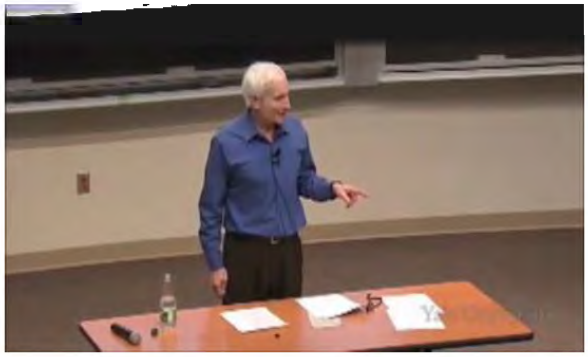


Figure 1. Example frame from recorded lecture

Figure 2. Example frame from narrative film

2. MW DETECTION

2.1 Procedure

Participants (or students) were 32 undergraduate students from a Canadian University, and they were compensated with course credit for their participation in the study. Participants watched a 24 minute lecture on population growth and were informed that there would be a test over what they had learned after watching the video. MW was defined as "Any thoughts that are not related to the material being presented", with examples such as "Concerns about an upcoming exam" and "Thoughts about dinner". Students also had the opportunity to ask questions regarding the instructions before the video began, but throughout the process, students had no control over the video.

Eye movements were monitored using a COTS eye-tracker called the EyeTribe that retails for $99. The eye tracker was placed just below the monitor on the desk.

2.2 Thought Probes

Mind wandering was measured during the recorded lecture using auditory thought probes, which is a standard approach in the literature [30]. Each student received 12 probes throughout the course of the recorded lecture that appeared at pre-determined times in the video. For each probe, the video paused and text was displayed on the screen asking, "In the moments prior to the probe were you MW?" Participants could then respond "1" for yes or "0" for no. Overall 31% of the probes were MW.

It is important to emphasize a few points about the method used to track MW. First, this method relies on self-reports because MW is an inherently internal phenomenon which requires self-awareness for reporting [32]. Second, self-reports of MW have been objectively linked to patterns in pupillometry [12], eye-gaze [25], and task performance [23], providing validity for this approach. However, at this time, there are no reliable neurophysiological or behavioral markers that can accurately substitute for the self-report methodology [32]. Indeed, this is the very reason we set out to build gaze-based MW detectors. The limits of thought probes are considered further in the Discussion section. For now, we note that our use of thought-probes to measure MW is consistent with the state of the art in the psychological and neuroscience literatures [32].

2.3 Feature Engineering

We calculated features from 30-second windows (window size was based on previous work [6, 15]) preceding each thought probe. We investigated two types of features: global gaze (from previous work [15]) as well as local features (based on [18]). Global gaze features focus on general gaze patterns and are independent of the content on the screen; whereas, local features encode where gaze is fixated on the screen.

2.3.1 Global Features

Eye movements were measured by fixations (i.e., points in which 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]. We considered six general measures across the 30-second window (bolded in Table 1) from which we computed the number, mean, median, minimum, maximum, standard deviation, range, kurtosis, and skew of the distributions, yielding 54 features. We also included three other features (see Table 1), yielding a total of 57 global gaze features.

Table 1. Eye-gaze features. Bolded cell indicates that nine descriptives (e.g., mean) were used as features (see Text)

Feature	Description
Fixation Duration	Elapsed time in ms of fixation
Saccade Duration	Elapsed time in ms of saccade
Saccade Length	Distance of saccade in pixels
Saccade Angle Absolute	Angle in degrees between the x-axis and the saccade
Saccade Angle Relative	Angle of the saccade relative to previous gaze point.
Saccade Velocity	Saccade Length / Saccade Duration
Fixation Dispersion	Root mean square of the distances of each fixation to the average fixation position
Horizontal Saccade Proportion	Proportion of saccades with relative angles <= 30 degrees above or below the horizontal axis
Fixation Saccade Ratio	ratio of fixation duration to saccade duration

2.3.2 Local Features

Local features were computed based on the relationship between eye movements and an area of interest (AOI). Two AOIs were defined for each frame of the lecture video that fell within the window: the most visually salient region of the frame, and the face of the lecturer. Visual saliency was determined using a MATLAB implementation of the Graph-Based Visual Saliency Algorithm [14] which produced a saliency map of pixel intensity from 0 to 1 for each frame that considered color, intensity, orientation, contrast, and movement. Determining the most visually salient region consisted of removing pixels with an intensity below a certain threshold (starting at 60% of the most intense pixel in the frame), leaving one or more regions of pixels as seen in Figure 4.

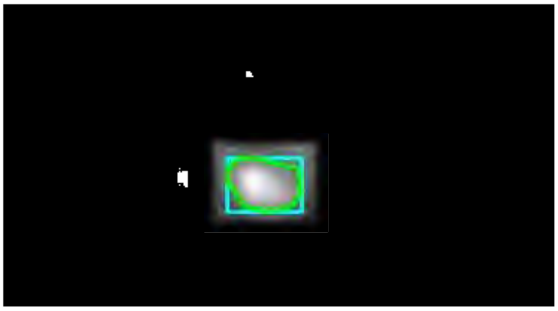


Figure 3. Example most salient region, lighter areas indicate higher saliency.

If the largest region had an area less than 2000 pixels (about 2% of the total area and a similar size to the face AOI), it was selected as the most visually salient region; otherwise, the process was repeated with a lower threshold. Figure 3 shows an example selection; in this case, the lecturer is gesturing, and the hand area was chosen as the most salient region. The face AOI was computed by detecting the facial location in the video using the commercially available software, Emotient [38]. The software provided the height and width of the face as well as the location

which was converted into a bounding box after adding a small buffer of 20 pixels to account for any tracker inaccuracies.

There were 17 features calculated from each AOI for a total of 34 features. The features can be divided into three types: (1) AOI distance, (2) AOI intersection, and (3) saccade landing. AOI distance features consisted of descriptive statistics (minimum, maximum, mean, median, standard deviation, skew, kurtosis, and range) of the distance between the center of the AOI and the fixation position for each frame where the AOI was present, for a total of eight AOI distance features per AOI. AOI intersection features captured the proportion of time that gaze was within the bounding box or within one or two degrees of visual angle from the bounding box, resulting in a total of three AOI intersection features per AOI. Saccade landing features consisted of counting the number of times saccades landed on an AOI, left an AOI, or occurred within an AOI. To account for tracking noise, an additional set of saccade landing features were computed that counted the same events if they occurred within one degree of visual angle from the AOI, for a total of six saccade landing features per AOI.

2.4 Model Building

We focused on Bayesian Networks as they yielded the best performance compared to several other standard classifiers on this task in our previous work [15]. We used the default implementation from the Weka data mining package [13]. We validated the models with a leave-one-participant-out cross-validation scheme. For each fold, probe responses of one participant are held out for testing, and the model is trained on the remaining probes. This process ensures that no instances of any individual participant could appear in both the training and testing sets within a fold. This process is then repeated for the number of participants.

In total, there were 384 probes during the lecture. Of those, 12 were discarded due to insufficient eye gaze data (< 1 fixation) in the respective window to compute all the global features. The remaining 372 instances were used across all feature sets to ensure a fair comparison. Students reported MW in 31% of the 372 instances, thereby leading to data skew. This imbalance between labels poses a challenge as supervised learning methods tend to bias predications towards the majority class label. To compensate for this concern, we use the SMOTE algorithm [9] to create synthetic instances of the minority class by interpolating feature values between an instance and its randomly chosen nearest neighbors until the classes were equated. SMOTE was *only done on the training sets;* testing sets were unaltered in order to ensure validity of the results.

2.5 Results

The classification results are shown in Table 2. Because our intention is to detect instances of MW, we focus on the precision, recall, and F_1 score of the MW class as our key metric. For comparison, a chance-level baseline was created by *randomly* assigning the MW label to 31% (i.e., the MW baserate) of the instances over 1,000 iterations and averaging the result.

The results indicated that, while all models outperform the chance baseline: (1) global features outperformed local features and (2) adding local features to the global features increased precision but decreased recall, leading to no improvement in F_1 MW over global features alone. The fact that the best results were obtained from global features is significant because these features are more likely to generalize across interaction contexts.

Table 2. MW detection results for the recorded lecture

Feature Set	F_1 MW	Precision MW	Recall MW
Global	0.47	0.39	0.62
Local	0.36	0.40	0.34
Global + Local	0.42	0.45	0.39
Chance	*0.30*	*0.30*	*0.30*

3. GENERAL DISCUSSION

MOOCs present an exciting new era for education, providing more resources for traditional and non-traditional students alike. However, little is known about user experience and student engagement [17] with MOOCs, and it is widely known that they are plagued with poor retention rates [37]. Attention is critical to learning, [23] and monitoring attentional states of students is a step towards better understanding the learning process. MW is one key attentional state that is negatively correlated with learning [21]. MW is a covert, internal state with no obvious behavioral markers, making it difficult to detect. Although strides have been made to detect MW using eye gaze in the context of self-paced reading, gaze-based MW detection has not yet been attempted in the context of recorded lectures, a key component of many MOOCs. This is a challenge we address in the current paper. In the remainder of this section, we discuss our main findings, potential applications, and discuss limitations and future work.

3.1 Main Findings

MW detection during reading is supported by decades of research on attention and eye movements [24]. Recent work has branched away from reading into more complex environments [15, 18] that are not afforded with predictable patterns of eye moments. We have shown that MW detection is possible in the context of viewing a recorded lecture. We were able to accurately classify MW with an F_1 of 0.47 which is a 56% improvement over chance. Although this result is modest, it is an important first step in detecting MW in this domain, especially using consumer-grade eye tracking equipment.

Since MW detection in the context of online learning is still in its infancy, it is important that we explore techniques that are both successful and generalizable. We considered two feature sets in this work: global eye gaze features, which have previously performed well at detecting MW during reading and while interacting with an ITS, and local features, based on AOIs, that have previously been shown to be successful predicting MW during narrative film viewing. In the context of lecture viewing, we have shown that global eye movements outperform local AOI-based features, contrasting previous work during narrative film viewing [18] that found the opposite pattern.

It is interesting to consider why AOIs were less successful in this context as opposed to narrative film viewing. One suggestion lies in the different styles of the two media. Commercial, narrative films are directed with the viewer in mind, directing the audience's attention to whatever is pertinent. In many cases, films are produced by professionals with years of experience and numerous qualifications in their art form. In contrast, a recorded lecture involves far more basic film production techniques, and in many cases the film audience is the secondary audience; the lecture itself is designed for the audience in the room. Our methods rely on automated AOI detection. It may be that these style differences affect that detection, having a downstream effect on the features generated from those AOIs. Further research would be required to confirm this hypothesis.

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 hope is that these models can be deployed at scale and can be used to improve engagement and learning from MOOCs. 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. The combination of student-independent models and consumer grade eye tracking increases our confidence that the models will generalize more broadly to applications outside of the laboratory, though this claim requires further empirical validation.

3.2 Applications

Lecture videos play a major role in online learning with MOOCs, so our MW detectors can be quite beneficial in that context. Our detectors could be implemented to provide real time updates to the MOOC software regarding the students' attention. Should a student be MW, the MOOC software could then adopt a variety of potential intervention strategies to refocus attention to the learning task. This could include simply pausing the video, asking a content-specific question, or asking the student to self-explain content that has recently been covered. Both interleaved questions [34] and self-explanations [20] have been shown to be effective in focusing attention. Students who answer incorrectly could then be encouraged to further review material and try again or could be redirected to an earlier point in the video. These approaches would give them multiple opportunities to correct the learning deficits attributed to MW.

It is important to consider that such interventions rely on MW detection which is inherently imperfect. The detector may issue a false alarm, suggesting that a student is MW when (s)he is not, or it could miss that a student is MW. In our view, MW detection does not need to be perfect as long as there is a modicum of accuracy. Imperfect detection can be addressed with a probabilistic approach, where the detector outputs a MW likelihood that is then used to determine whether an intervention is triggered (i.e., if the likelihood of MW is 70%, then there is a 70% chance of an intervention). The interventions should also be designed to "fail-soft" in that there are no harmful effects to learning if delivered incorrectly.

A further application is to inform the development of future MOOCs. Data from students' attention patterns whilst interacting with a MOOC video can be used to improve course structure (e.g. number of lectures and lecture length as well as course content such as individual explanations).

3.3 Limitations

We designed our approach to include a low-cost eye tracker, however, consumer models have a lower sampling-rate, limiting the accuracy of eye-gaze data compared to research-grade eye trackers. Furthermore, a key limitation was that we considered one lecture, so generalizability to other lectures is unknown. In addition, data was collected in a quiet lab environment; for better ecological validity we would need to explore more authentic learning environments (e.g. homes or libraries).

A further limitation relates to the use of thought probes which require users to be mindful of their MW and respond honestly. Although this methodology has been previously validated [12, 23, 25] there is no clear alternative to track a highly internal state like MW outside of measuring brain activity in an fMRI scanner. One futuristic possibility is to combine self-reports and wearable electroencephalography (EEG) as a means of collecting more accurate MW responses, but it is unclear if this can be done in more realistic contexts.

3.4 Future Work

The results discussed here invite several possibilities for improvement that we will address as future work. First, we will explore eye movements in different lectures. Having shown that global gaze models are applicable in this context, we will explore if we can train a model on one recorded lecture and use that model on other lectures and other topics. We will also explore cross training to other educational environments, to gain a better understanding of the differences and similarities in eye movements and attention across learning situations.

Another potential avenue is to integrate the detector into a MOOC to detect MW in real time. Here, the MW probes will be based upon the detectors real time assessment of students' attention instead of pre-prescribed or pseudo random probing. We can then better evaluate our detectors by comparing the probabilistic assessment of MW to students' responses to probes. Providing this refinement is successful, we could then use the detector to create a MOOC environment that intervenes in real time.

4. CONCLUSION

The popularity of MOOCs has ushered in an exciting time for students everywhere while also bringing challenges for educators. Advances in consumer grade eye tracking allow us to take a step towards a better understanding of how students engage with MOOCs on a larger scale. We have shown that we can detect MW in recorded lectures at above chance level. While much MW research has focused on the context of reading, our findings suggest that it might be possible to apply research on eye gaze, attention, and learning to this new context, thereby affording new discoveries about how students learn and interact with MOOCs while designing interfaces to sustain attention during learning.

5. 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.

6. REFERENCES

[1] Adamopoulos, P. 2013. What Makes a Great MOOC? An Interdisciplinary Analysis of Student Retention in Online Courses. *International Conference on Information Systems* (2013), 21.

[2] Arroyo, I. et al. 2007. Repairing disengagement with non-invasive interventions. *Artificial Intelligence in Education* (Amsterdam, The Netherlands, 2007), 195–202.

[3] Baker, R.S.J. d. 2007. Modeling and understanding students' off-task behavior in intelligent tutoring systems. *SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2007), 1059–1068.

[4] Bixler, R. et al. 2015. Automatic detection of mind wandering during reading using gaze and physiology. *International Conference on Multimodal Interaction* (2015), 299–306.

[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. 2016. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction*. 26, 1 (2016), 33–68.

[7] Bixler, R. and D'Mello, S.K. 2014. Toward fully automated person-independent detection of mind wandering. *User Modeling, Adaptation, and Personalization* (Aalborg, Denmark, 2014), 37–48.

[8] Blanchard, N. et al. 2014. Automated physiological-based detection of mind wandering during learning. *Intelligent Tutoring Systems* (Switzerland, 2014), 55–60.

[9] Chawla, N.V. et al. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*. 16, 1 (Jun. 2002), 321–357.

[10] Drummond, J. and Litman, D. 2010. In the zone: Towards detecting student zoning out using supervised machine learning. *Intelligent Tutoring Systems* (Pittsburgh, PA, USA, 2010), 306–308.

[11] Franklin, M.S. et al. 2011. Catching the mind in flight: using behavioral indices to detect mindless reading in real time. *Psychonomic Bulletin & Review*. 18, 5 (Oct. 2011), 992–997.

[12] Franklin, M.S. et al. 2013. Window to the wandering mind: pupillometry of spontaneous thought while reading. *The Quarterly Journal of Experimental Psychology*. 66, 12 (2013), 2289–2294.

[13] Hall, M. et al. 2009. The WEKA data mining software: An update. *SIGKDD Explorations*. 11, 1 (Nov. 2009), 10–18.

[14] Harel, J. et al. 2006. Graph-based visual saliency. *NIPS* (2006), 5.

[15] Hutt, S. et al. 2016. The eyes have it: gaze-based detection of mind wandering during learning with an intelligent tutoring system. *The 9th International Conference on Educational Data Mining* (Raleigh, NC, USA, 2016), 86–93.

[16] Liyanagunawardena, T. et al. 2013. MOOCs: A systematic study of the published literature 2008-2012. *The International Review of Research in Open and Distributed Learning*. 14, 3 (2013), 202–227.

[17] Milligan, C. et al. 2013. Patterns of engagement in massive open online courses. *Journal of Online Learning with Technology*. 9, 2 (2013), 149–159.

[18] Mills, C. et al. 2016. Automatic gaze-based detection of mind wandering during film viewing. *The 9th International Conference on Educational Data Mining*. (Raleigh, NC, USA, 2016).

[19] Mills, C. et al. 2015. Toward a real-time (day) dreamcatcher: sensor-free detection of mind wandering during online reading. *The 8th International Conference of Educational Data Mining* (Madrid, Spain, 2015), 786–789.

[20] Moss, J. et al. 2013. The nature of mind wandering during reading varies with the cognitive control demands of the reading strategy. *Brain Research*. 1539, (2013), 48–60.

[21] Olney, A.M. et al. 2015. 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] Pham, P. and Wang, J. 2015. AttentiveLearner: improving mobile MOOC learning via implicit heart rate tracking. *Artificial Intelligence in Education* (Madrid, Spain, 2015), 367–376.

[23] 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.

[24] Rayner, K. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*. 124, 3 (Nov. 1998), 372–422.

[25] Reichle, E.D. et al. 2010. Eye movements during mindless reading. *Psychol Sci*. 21, 9 (Sep. 2010), 1300–1310.

[26] Risko, E.F. et al. 2013. Everyday attention: Mind wandering and computer use during lectures. *Computers & Education*. 68, (2013), 275–283.

[27] Sandeen, C. 2013. Integrating MOOCS into traditional higher education: The emerging "MOOC 3.0" era. *Change: The magazine of higher learning*. 45, 6 (Nov. 2013), 34–39.

[28] Schooler, J.W. et al. 2004. Zoning out while reading: Evidence for dissociations between experience and metaconsciousness. *Thinking and seeing: Visual metacognition in adults and children*. MIT Press. 203–226.

[29] Seibert, P.S. and Ellis, H.C. 1991. Irrelevant thoughts, emotional mood states, and cognitive task performance. *Memory & Cognition*. 19, 5 (Sep. 1991), 507–513.

[30] Smallwood, J. et al. 2008. When attention matters: the curious incident of the wandering mind. *Memory & Cognition*. 36, 6 (Sep. 2008), 1144–1150.

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

[32] Smallwood, J. and Schooler, J.W. 2015. The science of mind wandering: Empirically navigating the stream of consciousness. *Annual Review of Psychology*. 66, (2015), 487–518.

[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] Szpunar, K.K. et al. 2013. Mind wandering and education: from the classroom to online learning. *Frontiers in Psychology*. 4, (2013), 495.

[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. *Behavior Research 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.

[37] Zheng, S. et al. 2015. Understanding student motivation, behaviors and perceptions in MOOCs. *Computer Supported Cooperative Work; Social Computing* (New York, NY, USA, 2015), 1882–1895.

[38] 2016. *Emotient module: Facial expression emotion analysis*.

Sequence Modelling For Analysing Student Interaction with Educational Systems

Christian Hansen, Casper Hansen, Niklas Hjuler, Stephen Alstrup, Christina Lioma
Department of Computer Science
University of Copenhagen, Denmark
{chrh,bnq,hjuler,s.alstrup,c.lioma}@di.ku.dk

ABSTRACT

The analysis of log data generated by online educational systems is an important task for improving the systems, and furthering our knowledge of how students learn. This paper uses previously unseen log data from Edulab, the largest provider of digital learning for mathematics in Denmark, to analyse the sessions of its users, where 1.08 million student sessions are extracted from a subset of their data. We propose to model students as a distribution of different underlying student behaviours, where the sequence of actions from each session belongs to an underlying student behaviour. We model student behaviour as Markov chains, such that a student is modelled as a distribution of Markov chains, which are estimated using a modified k-means clustering algorithm. The resulting Markov chains are readily interpretable, and in a qualitative analysis around 125,000 student sessions are identified as exhibiting unproductive student behaviour. Based on our results this student representation is promising, especially for educational systems offering many different learning usages, and offers an alternative to common approaches like modelling student behaviour as a single Markov chain often done in the literature.

Keywords

Markov Chains, Sequence Modelling, Clustering

1. INTRODUCTION AND RELATED WORK

How students interact with educational systems is today an important topic. Knowledge of how students interact with a given system can give insight in how students learn, and directions for the further development of the system based on actual use. The interaction can be studied both by explicit studies [7] directly observing student interaction *in situ*, or by the use of log data collected automatically by the use of the system as done in this paper.

Analysis of log data is often viewed as an unsupervised clustering problem at the student level [4, 8]. Our work takes another direction and focuses on the action sequence level. For clustering sequences, Markov models are popular as they provide a convenient way of modelling the transitions and dependencies of the sequences [9]. For action sequence mining, both hidden and explicit models have been used depending on the tested hypothesis, and on whether the states are explicit or implicit. Beal et al. use hidden Markov models for student prediction, assuming underlying hidden states of engagement, which can be clustered [2]. Köck and Paramythis use explicit states for analysing problem solving activity sequences, as the states in this case are explicit and therefore appear directly in the log [9].

The choice of clustering of the Markov models depends on the application area. Klingler et al. did student modelling by the use of explicit Markov chains, and the clustering was done by different similarity measures defined on the Markov chains themselves [8], e.g. euclidean distance between transitional probabilities, or Jensen-Shannon Divergence between the stationary probabilities of the chains. When individual sequences are clustered, an underlying assumption of the data coming from a mixture of Markov chains has been used [10], where the individual chains represent the cluster centres, and the task is finding both the chains and the mixing coefficients.

The work presented in this paper is using discrete Markov chain models for action sequence analysis, on log data[1] acquired from the company Edulab. Edulab is the largest provider of digital learning for mathematics in Denmark, having 75% of all schools as customers, and receiving more than 1 million student answers a day. Using a mixture of Markov chains, we assume that each chain will represent a prototype student behaviour. So the underlying assumption in this work is that each student can be modelled as behaving according to some underlying behaviour during each session, and a student can then be seen as a distribution over different behaviours. Edulab's product offers many different ways of learning mathematics, ranging from question-heavy workloads to video and text lessons, and other activities depending on whether the student is in class or at home. This allows to model a student as "distributed" over different behaviours, in contrast to a single student behaviour model of how the student usually interacts with the system.

We reason that mixture of Markov chains will allow for a qualitative study of what type of behaviour each chain rep-

[1]The data is proprietary and not publicly available

resents, and thus ultimately it can be used to show how a student uses the educational system.

Mixtures of Markov models can be solved by the EM algorithm, which however is notoriously slow to run for large amounts of data, and only local optimal solutions are found [6]. In this paper we need fast processing in order to analyse the large amounts of data produced by Edulab, so we simplify the assumptions on the underlying Markov chains, which allows for a modified version of k-means clustering.

Initial cluster centres, representing underlying student behaviour, can be chosen by domain experts and then refined through the clustering. However, since the true number of underlying clusters is unknown, it is difficult for an expert to predefine sensible cluster centres for a range of different numbers of clusters. In this work we first perform simulations to consider the effect of starting at the correct locations versus adding noise to the correct location until the starting points are completely random. Based on these results clustering is done on the Edulab dataset, and a qualitative analysis is performed on the resulting Markov chains. This shows how students are distributed among the Markov chains, and how unproductive system usage can be detected using the Markov chains.

In summary the primary research questions this paper addresses are: 1) to what extent can students be modelled as a distribution over underlying usage behaviours which is changing across sessions, and 2) how this modelling leads to insight in future improvements of the system for the producers of educational systems.

2. DATA

The data used in this work is produced by matematikfessor.dk, a Danish mathematics portal made by Edulab that spans the curriculum for students aged 6 to 16. The website offers both video and text lessons in combination with exercises covering the whole curriculum, such that it can be used as a primary tool for learning, and not only supplementary. Log data generated by the grade levels corresponding to students of age 12 to 14 for the 2016 school year is used (from August 2016 to February 2017). An action in this system can either be watching a lesson, which contains either a video or text description, or answering a question. Lessons and questions both have a topic id, specifying the general topic of the question or lesson. The data statistics are summarized in Table 1. The lessons and questions can be assigned as homework or done freely by the students (this study does not differentiate between whether it is homework or not). It should be noted that a lesson takes significantly longer time doing than answering a question hence the lower ratio of lessons, compared to other actions, in Table 1.

The logs do not contain information about when a session is started or finished, so we define a session as a sequence of actions, where the time between two actions is less than 15 minutes. A student has on average 12.5 sessions (standard deviation of 13.3), and the histogram of the number of actions in each action sequence can be seen in Figure 1, where sequence lengths larger than 200 have been removed from the plot for the purpose of visualization. When a student interacts with the system his actions are stored and seen as

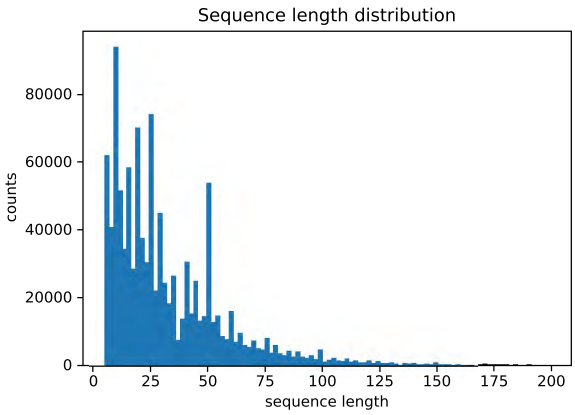


Figure 1: The distribution of action sequence lengths with lengths larger than 200 removed.

Number of sequences	1.08M
Number of actions	37.5M
Number of lessons	1.35M
Number of correctly answered questions	27.44M
Number of wrongly answered questions	8.71M

Table 1: Data statistics. The number of lessons and question answers sum to the number of actions.

an action sequence, an example of one is:

$$Qr_1^{t_1}, Qw_2^{t_2}, L_3^{t_1}, Qw_4^{t_3}, Qr_5^{t_1}, Qr_6^{t_1}, Qr_7^{t_1} \quad (1)$$

Qr is a correctly answered question, Qw is an incorrectly answered question, and L is a lesson. The subscript denotes the action number in a temporal ordering, and the superscript denotes the topic id, which is associated with each lesson and question.

3. METHOD

Our method for action sequence clustering will be explained in this section, and is based on modelling interactions with the system as Markov chains. Our Markov chain model with its transitions is shown in Figure 2. Our model consists of 8 states as will now be explained with their abbreviations in parentheses. These abbreviations are used for visualizing the resulting Markov chains from the clustering. The first two are start (S) and end (E). The rest consists of three general states: Doing a lesson (L), answering a question right (Qr), or answering a question wrong (Qw). Each lesson and question have an associated topic id, which might change from action to action creating the last three states: doing a lesson in another topic than the previous action (L_c), answering a question right in another topic (Qr_c), and answering a question wrong in another topic (Qw_c). If we consider the sequence described in Equation 1, then that

would correspond to visiting the following states

$$S \to Qr \to Qw\_c \to L\_c \to$$
$$Qw\_c \to Qr\_c \to Qr \to Qr \to E \quad (2)$$

The pipeline for clustering has the following procedure.

1. For every session we extract a sequence of actions $A_1, ..., A_n$, and each action sequence corresponds to a path in the used Markov chain model.

2. Since the Markov chains are unknown, priors $P_1, ..., P_k$ (which themselves are Markov chains) are generated at random such that each edge shown in Figure 2 has a transition probability taken uniformly at random from 0 and 1. Each random chain is normalized such that each state's outgoing transitional probabilities sum to one. These priors function is the pendant to the usual initial cluster centers, which most often are random data points. Generating a Markov chain from a randomly chosen point would however not work in our case, since many zero valued transition probabilities would occur.

3. Each action sequence is assigned to the prior which was most likely to generate it, i.e.

$$\arg\max_{1 \le j \le k} \left(\prod_{i=1}^{m} p^j_{b_{i-1}, b_i} \right) \quad (3)$$

   where $p^j_{b_{i-1}, b_i}$ is the transition probability from state b_{i-1} to b_i in prior P_j, m is the number of transitions between states, and k is the number of priors.

4. After each action sequence has been associated with a prior, then each prior is updated by generating the Markov chain most probable given its associated action sequences. This is done by counting the state transitions in each sequence in a new Markov chain model, and normalizing afterwards.

5. Points 3 and 4 are ideally reiterated until convergence, i.e. no action sequence changes its associated prior. However for computational reasons we stop iterating after less than 5% of the sequences have changed their assigned prior.

The clustering technique is very similar to ordinary k-means clustering, with the major difference that the clustering is not dependent on a similarity measure directly on the sequence, but dependent on the Markov chains generated by the clustering. Comparing to ordinary k-means clustering, the produced chains in each iteration are analogous to the ordinary cluster center found by some mean. The mixture model could also be estimated by the EM algorithm [1], which has the benefit that sequences that do not belong to a single clear cluster, i.e. that have multiple highly probable chains, will weight in on all of them. This has the downside that clusters take longer to be separated, and the convergence is therefore slower. Under the assumption of the chains being distinct, each sequence will mostly weight on a single chain, and here the k-means clustering method and EM algorithm will perform very similarly. For the data from Edulab we assume most of the chains to be distinct,

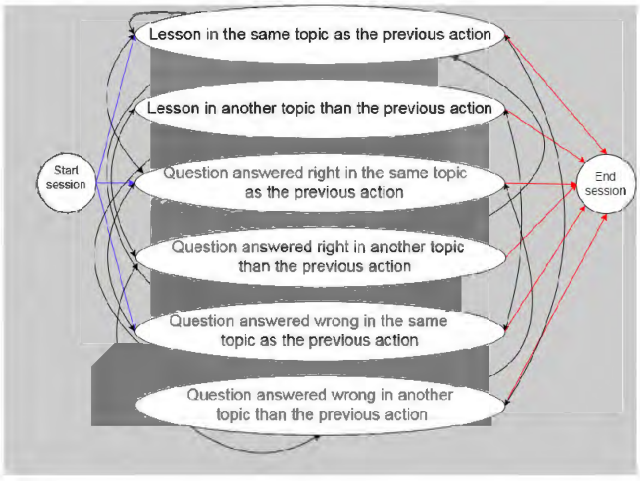


Figure 2: Markov chain representing the possible states and transitions. Note the transitions each way do not have to be equal.

but not necessarily all. In addition a very large number of sequences will have to be clustered in the future when the full dataset is used, and not restricted as done for this paper. We are therefore mostly interested in how well the k-means clustering approach performs as it is more computationally feasible when the data size is increased.

The above procedure leaves two challenges: 1) How do we know the resulting Markov chains are close to the real ones? and 2) How to estimate the number of priors? We address these points next.

The first point is dealt with using synthetic data, where k random Markov chains are made, and each action sequence is generated from one of those chosen uniformly at random. In order to ensure a suitable length of the generated action sequences, the ingoing probabilities to the end state are fixed to allow for an average sequence length of 20. After generating the synthetic data, the most probable Markov chain for each sequence is assigned as its label, and the goal in the clustering is to be able to capture these clusters. Note, that since each sequence is randomly generated using the chosen Markov chain, then its most probable Markov chain might not be the one generating it. To determine the ability to capture the original clusters we consider the average purity of the resulting clusters:

$$Average_{purity} = \frac{1}{n} \sum_{i=1}^{n} \frac{\max_{1 \le j \le k}(|C_j \cap S_i|)}{|S_i|} \quad (4)$$

Where S_i is an estimated cluster, C_j is the true cluster, n is the number of clusters, and k is the number of true clusters. An average purity of 1 represents that the method fully captures the original clusters. The underlying Markov chains are unknown on real data, so increasingly noisy versions of the underlying Markov chains are experimented with as priors, to show how the method is expected to perform under real circumstances.

In the case of real data, the true underlying Markov chains are unknown, so in this case the sum of the log likelihoods is calculated for the sequences to their most probable prior:

$$\text{sum of log likelihood} = \sum_{i=1}^{n} \log\left(\mathcal{L}(s_i|P_i^*)\right) \quad (5)$$

where s_i is an action sequence, P_i^* is the prior most likely to generate action sequence s_i, and $\mathcal{L}(s_i|P_i^*)$ is the likelihood that P_i^* generates s_i.

The second point mentioned earlier, about estimating the number of priors, can be solved using either the average purity in the synthetic case, or from the sum of log likelihoods in the real case. The sum of log likelihoods as a function of k will be monotonically increasing, but the slope will decrease as k exceeds its true underlying value. Since the method starts with randomly chosen priors, it is repeated a number of times, and the solution with the largest log likelihood is chosen for each value of k.

4. SIMULATED EXPERIMENT WITH NOISY PRIORS

There are two approaches for estimating the Markov chains for the Edulab data set. 1) The prior Markov chains can be chosen by domain experts - by specifying common sequences we would expect to find in the data, and then refine them during the clustering. 2) The second approach is as described in the method section, starting with random chains, and running k-means multiple times, and taking the clustering which gives the highest sum of log likelihoods. To measure how the method behaves as the initial priors are increasingly noisy versions of the underlying Markov chains, k-means is run with the priors chosen as:

$$P_i = (1-\alpha)P_i^* + \alpha P_{rand} \quad (6)$$

Where all Ps are Markov chains represented by matrices of transitional probabilities, and α is the noise parameter. P_i is the i^{th} prior, P_i^* is the i^{th} underlying Markov chain used when generating the synthetic data, and P_{rand} is a random Markov chain. The higher α, the more noisy the initial prior is.

In Figure 3, we see how the average purity behaves as a function of noise parameter α. The experiment is run for $k = 6$, and 6 random chains are generated. The transition probabilities to the end state are fixed at 0.05 for all states for all chains to allow for sequences of average length 20. 50000 sequences are sampled uniformly from the 6 chains. The modified k-means is then run with the priors varying depending on α, and the experiments are run 10 times and purity is the average over the 10 runs. First we note that even with using the modified k-means algorithm and not the EM algorithm the resulting average purities are quite high. It is seen that even with $\alpha = 1$ representing completely random priors, the reduction in purity is not too large compared to starting with the same priors as the data is generated from. Even starting with the same priors which generated the data does not guarantee perfect purity, which is expected as there are some sequences that are almost as likely under multiple chains, so small differences in the data determined Markov chains will move them from one chain to another. Based on the above result we will not define the priors by an expert, and instead let them be random. This has the benefit of being more manageable than handcrafting specific priors for each choice of k, which would be very difficult to do in a meaningful way when k is large.

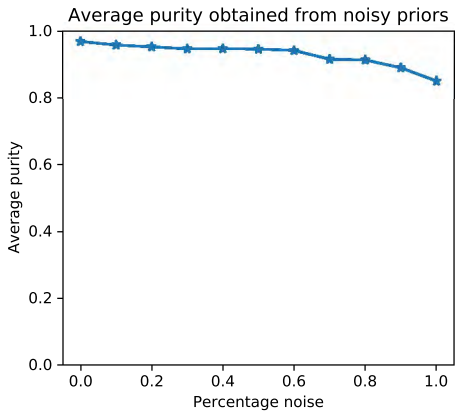


Figure 3: Average purity as a function of increasingly more noisy priors. A completely random prior (1.0 on the x axis) is able to perform well.

5. REAL DATA EXPERIMENT

5.1 Choosing the number of clusters

The problem of determining the number of clusters is common for all unsupervised learning tasks. In this paper we consider the sum of the log likelihoods for the action sequences. A common approach is the use of the "elbow" heuristic, where the choice of k is chosen based on the slope of the sum of log likelihoods function over k.

In order to argue that there is structure in the data, and that the method is able to capture this structure, a randomized experiment is made. The randomized experiment consists of randomly permuting each sequence (but keeping the start and end states), and seeing how the sum of log likelihoods is affected by it. If there is no structure originally in the sequences, then one can not expect it to perform better than the permuted data.

In Figure 4 we see that the sums of log likelihoods are considerably lower in the permuted data set, with only slightly higher sum of log likelihoods when $k = 20$ compared to $k = 2$ for the real data set. The action sequences therefore have structure which the Markov chain captures, and it is therefore not just random chains that the k-means clustering produces. Since the chains capture some inherent structure in the data, it is meaningful to analyse the individual chains with regards to what user behaviour they capture.

There is not an obvious breaking point in the sum of log likelihoods, but the increase before $k = 6$ is large, while the increase for $k > 10$ is notably smaller, so a value of k between 6-10 is sensible. We will in the qualitative assessment of the chains use $k = 6$.

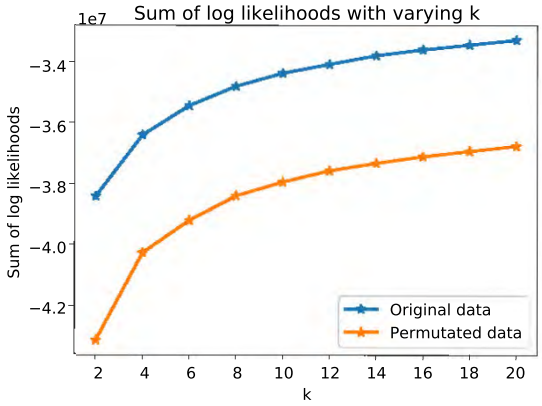


Figure 4: Sum of likelihoods for the best performing clusters for each k. Each experiment is run 5 times for each k. The permutations of each sequence is done for each value of k in each of the 5 times.

5.2 Qualitative assessment of Markov chains

This section will make qualitative assessments of what the different resulting Markov chains represent with regards to what type of user behaviour they capture. Even with six chains there is some similarity between some chains, so in this section we will focus on the three most distinct chains shown in Figure 5. The thickness of the arrows is proportional to the transitional probability for each state, except the ending state. The transitional probabilities are sorted and only drawn until 70% of the probability mass is covered. For the ending state, 70% of the incoming transitional probabilities are drawn.

In general not all chains can be described as either being a positive or negative usage of the system. Chain 2 captures usage where most of the questions being answered are either right or wrong, and there is very little mixing between taking lessons and answering a question. Usage like this could indicate an unproductive session for students, since they are mostly getting all questions right or all questions wrong, and research shows that students feel more intrinsic pleasure when the difficulty level is slightly challenging [5] leading to more engaged sessions [3]. Similarly, watching lessons without engaging with the material via questions leads to students not training the learned material, which is important for the learning process.

Chain 6 can be described as a positive usage of the system, as the most probable transitions lead to a question being correctly answered, except for the two transitions in the lessons. Generally students are focused on one topic at a time.

Chain 4 has high transitional probability when switching between topics, so this could indicate a session with a primary focus on repetition as the topic is varying, and students most often answer questions from another topic than the watched lessons.

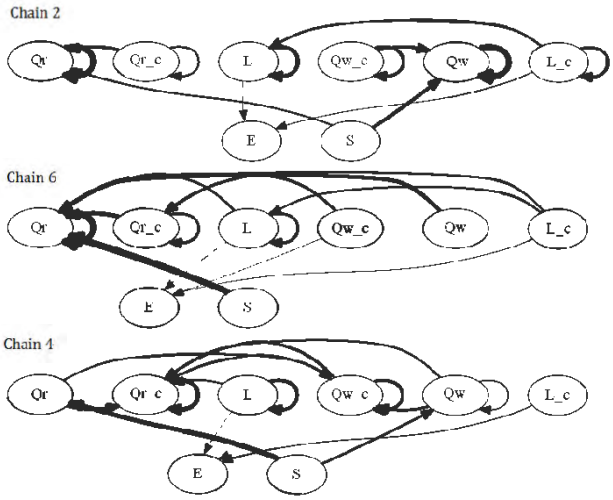


Figure 5: Chains 2, 6, and 4 of the six chains. The thickness of the arrows is proportional to the transitional probability for each state, except the ending state. The transitional probabilities are sorted and only drawn until 70% of the probability mass is covered. For the ending state 70% of the incoming transitional probabilities are drawn. State abbreviations are explained in section 3.

	Num. sequences	Avg. sequence length
chain 1	295,792	34.81
chain 2	126,683	36.88
chain 3	198,736	26.79
chain 4	131,460	28.79
chain 5	194,174	36.12
chain 6	144,121	44.85

Table 2: The number of sequences and average length of sequences for each Markov chain

The distribution of the sessions over the chains can be seen in Table 2.

The length of the sequences is varying, but no single chain in general captures either the very short or very long sequences. Instead a combination of shorter and longer sequences is captured by each chain. The most common chain can be seen in Fig 6. This chain is similar to chain 4 (Fig 5), but with more topic changes and more wrongly answered questions when changing topics, which can be seen in the self loop for Qw_c. Chain 4 is also shorter on average. As seen in Table 2, generally all six chains contain a large amount of sequences on average. This indicates that the system usage does indeed vary, and is not limited to all sequences of the same length defining the same use of the system. If one considers each user's distribution of Markov chains, then on average each user has 3.5 different types of sessions out of 6 with a standard deviation of 1.5. This supports the assumption that a single Markov chain is not optimal for user profiling for educational systems similar to the one generating our data, where there is a lot of user freedom in what

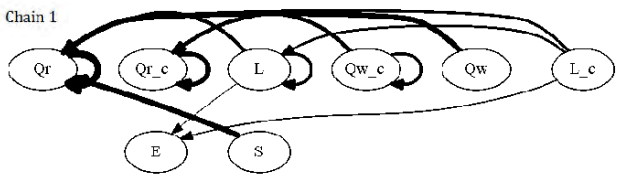


Figure 6: Chain 1, the most common chain. State abbreviations are explained in section 3.

activities they engage in.

6. DISCUSSION AND CONCLUSION

In this work first order Markov chains have been used, but it is generally known that the action sequences do not fulfil the Markov property of transition to a state only being dependent on the previous state. No order of Markov chain will completely capture the underlying transition between states, as the usage is dependent on many external factors which are unknown, but higher order chains would be able to capture more complex dynamics in the usage. Even though the Markov property is violated, Markov chains are still very widely used in educational data mining [4, 8], and provide a good tool for comparisons of action sequences across different lengths, focusing on the flow of actions taken. In future work an interesting extension would be considering time dependent Markov models, such that the transitional probabilities are dependent on how long the states have been unchanging. This would allow for more interpretative models, e.g. we could see when the probability of a session ending gets high.

When inspecting the Markov chains produced by the clustering, chain number 2 indicated suboptimal or unproductive usage of the system, where the students either experience questions that are too easy or too hard, or never train what they learn in the lessons. The chain has 126,683 sessions in its cluster, and it is therefore a significant amount of sessions where the learning outcome most likely could be improved. Based on this it could be recommended to have a few obligatory questions after a lesson to strongly encourage the student to use what they have just learned, and detect negative spirals where the students are always wrong by recommending lessons to help the student move forward.

Modelling the student as a distribution over Markov chains, which can be considered usage patterns, results in a vector representation of the individual students. This representation allows to apply standard techniques directly on the student model, compared to working on more complex student models. An example is the issue of drift in student behaviour over time, corresponding to some learning, or wider cognitive development of the student. This problem has also been considered in a similar context in [8], where distances between single Markov chains on a student level were estimated. However, in our setting standard methods could readily be used to detect this type of drift and potentially alert the teacher.

The work presented shows a qualitative study of the proposed student representation, and experiments using synthetic data show that our methodology is able to capture the underlying generative Markov chains very well, when the number of chains has been estimated. A source for future work will be using the student vectors in a predictive task, such that quantitative measures can be acquired. An interesting path would be using knowledge tracing methods over the different session types, to see if there are any unexpected differences between the knowledge acquired by the student depending on the type of session - i.e. the kind of Markov chain the session originates from.

7. ACKNOWLEDGMENTS

The work is supported by the Innovation Fund Denmark through the DABAI project.

8. REFERENCES

[1] D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, New York, NY, USA, 2012.

[2] C. Beal, S. Mitra, and P. R. Cohen. Modeling learning patterns of students with a tutoring system using hidden markov models. In *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, pages 238–245, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press.

[3] M. Csikszentmihalyi and I. Csikszentmihalyi. *Optimal Experience: Psychological Studies of Flow in Consciousness*. Cambridge University Press, 1992.

[4] L. Faucon, L. Kidzinski, and P. Dillenbourg. Semi-markov model for simulating mooc students. In T. Barnes, M. Chi, and M. Feng, editors, *EDM*, pages 358–363. International Educational Data Mining Society (IEDMS), 2016.

[5] J. Gottlieb, P.-Y. Oudeyer, M. Lopes, and A. Baranes. Information-seeking, curiosity, and attention: computational and neural mechanisms. *Trends in Cognitive Sciences*, 17(11):585–93, Nov. 2013.

[6] R. Gupta, R. Kumar, and S. Vassilvitskii. On mixtures of markov chains. In *Advances in Neural Information Processing Systems*, pages 3441–3449, 2016.

[7] S. Hutt, C. Mills, S. White, P. J. Donnelly, and S. K. D'Mello. The eyes have it: Gaze-based detection of mind wandering during learning with an intelligent tutoring system. In *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016, Raleigh, North Carolina, USA, June 29 - July 2, 2016*, pages 86–93, 2016.

[8] S. Klingler, T. Käser, B. Solenthaler, and M. Gross. Temporally Coherent Clustering of Student Data. In *Proceedings of EDM*, pages 102–109, 2016.

[9] M. Köck and A. Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *User Modeling and User-Adapted Interaction*, 21(1):51–97, 2011.

[10] Y. Yang, Q. Yang, W. Lu, S. J. Pan, R. Pan, C. Lu, L. Li, and Z. Qin. Preprocessing time series data for classification with application to crm. In S. Zhang and R. Jarvis, editors, *Australian Conference on Artificial Intelligence*, volume 3809 of *Lecture Notes in Computer Science*, pages 133–142. Springer, 2005.

Predicting Prospective Peer Helpers to Provide Just-In-Time Help to Users in Question and Answer Forums

Oluwabukola Mayowa Ishola, Gordon McCalla
ARIES Laboratory, Department of Computer Science
University of Saskatchewan, Saskatoon Canada
bukola.ishola@usask.ca , mccalla@cs.usask.ca

ABSTRACT

Question and answer forums are becoming more popular as increasing numbers of lifelong learners rely on such forums to receive help about their learning needs. Stack Overflow (SO) is an example of such a forum used by millions of programmers. The ability of users to receive timely answers to questions is crucial to the sustainability of such forums and for successful lifelong learning. In SO we have observed that the number of questions answered within 15 minutes have diminished with more questions taking a longer time to get answered or remaining unanswered in some cases. This suggests the need for an effective approach in predicting prospective helpers who can provide timely answers to the questions. In this paper, we seek to explore strategies to match helpers and help seekers. In particular we wish to use these strategies to predict which SO users will provide timely answers to questions asked in SO, and then compare these predictions to the users who actually answered the questions. In making these predictions we looked at 3 time frames of user data: 1 month, 3 months and 6 months. We used 5 basic strategies: frequency, knowledgeability, eagerness, willingness, recency; and we compared the success rates of each strategy in making predictions on 3 different success criteria: predicting the first answerer, predicting the answerer most liked by the asker of the question, and predicting the answerer rated most highly by other SO users. We then incorporated a timeliness measure, which takes into consideration how quickly the user provides answers to questions in the past, which helped us to achieve a higher success rate. The results of our study are an improvement over a similar previous study of SO and we hope will form the basis of methods for recommending peers in online forums who can provide just-in-time help to lifelong learners as their knowledge needs evolve and change.

Keywords

peer help, lifelong learning, peer matching

1. INTRODUCTION

Professional lifelong learners depend on online learning forums to help to meet their learning needs [2]. Our research is focused on supporting lifelong learners as they interact in such open-ended learning environments. Stack Overflow (SO) is an example of an online question and answer (Q&A) forum which supports millions of programmers. Over time, the answer response times to questions have increased and the number of unanswered questions has also increased. According to Asaduzzaman et. al. [1], failure of the questions asked to attract expert users is the top reason for unanswered questions, accounting for about 21.75% of unanswered questions. Receiving prompt answers to questions is important to the sustainability of a Q&A forum [2] and for successful lifelong learning.

While research efforts have been employed in the past in predicting potential peer helpers within a classroom-learning environment which encompasses just hundreds of students [4, 8,10], a new challenge arises in an online learning environment that is open ended with thousands or millions of potential helpers with varied expertise and learning interests. The need for an appropriate recommendation technique that scales up to millions of available users[1], and also aligns with the knowledge, interests and competency of the helper could be necessary. Greer et al. [4] in their study (similar to other studies [3,8,10]) employed the availability, helpfulness, technical ability and social ability of the helper as strategies considered in selecting the appropriate peer helper from the available users.

In a previous study using SO users as surrogates for lifelong learners, we employed a tag-based Naïve Bayes model to predict the answer performance of users using their previous activity in the forum [6]. The possibility of this model to predict poor answers even before they are provided could be used to help to reduce the frequency of poor answers within SO. In this new study, our goal is to predict helpers who are likely to provide answers to users' questions quickly ("just-in-time"). We also aim to determine how much information about the user is sufficient to predict the helper (to deal with issues such as those raised by Kay and Kummerfeld [7] about how much information must be usefully retained about the user in lifelong learning contexts). Finally, we compare the results from this study with the topic modelling approach used by Tian et al. [9]. We hope this study will augment such studies as [3, 4, 8, 10] in providing peer helper seeking strategies that scale to very large numbers of users.

2. RELATED WORK

In supporting learners in computerized learning environments human helpers and intelligent agents have been employed. Merrill et. al. [8] compared the help provided by peer helpers with that provided by intelligent agents and conclusions from this study show that human helpers provide more flexible and subtle help. Similarly, Greer et al. [4], building on earlier work in finding peer helpers in workplace environments [3], built the iHelp system to help computer science students find potential peer helpers among their classmates who are ready, willing and able to help in overcoming impasses. In addition, Vassileva et al. [10] in their study with iHelp incorporated the social characteristics of the helper into determining an appropriate helper, gleaned from the

[1] We will use the term "user" in this paper rather than "learner" when specifically discussing SO users since they are likely not explicitly learners in their own minds. However, in the future most professionals will be using such forums to meet their lifelong learning goals. The term "learner" then will be highly appropriate. Since our research is aimed at helping develop tools for such professional lifelong learners, especially tools that support personalization to each such learner, it is, we believe, deeply and broadly relevant to advanced learning technology.

online activities of the helper such as votes received by the helper, questions asked, answers provided, and the marks received on assignments.

While these studies [3,4,10] have all successfully recommended just-in-time helpers for a relatively small number of students within classroom and workplace settings, in a typical question and answer forum, the number of users ranges from thousands to millions of users with more varied knowledge interests [5]. The sustainability of such a large-scale question and answer forum is dependent on providing quick responses to questions [2]. A study by Bhat et al. [2] reveals that in Stack Overflow, although most of the questions get answered in less than 1 hour, about 30% of the questions have a response time of 1 day with about 344,000 questions having a response time greater than 1 day. In addressing the increasing number of unanswered questions, Bhat et al. [2] revealed the importance of assigning appropriate tags to questions; Asaduzzaman et al. [1] predicted how long a question will remain unanswered; and Tian et al. [9] predicted the best answerers to questions using a topic modelling approach. Yang and Manandhar [11] identified the topic modelling approach as a less effective approach that is too general while the use of question tags was proposed as a more informative approach. The study by Tian et al. [9] in predicting best answerers achieved a success rate of 21.5% while recommending 100 users who could answer the question. This reveals the need to explore other methodologies in predicting best answerers to questions.

3. ANALYSIS OF QUESTION RESPONSE TIME AND UNANSWERED QUESTIONS IN STACK OVERFLOW

SO is a question and answer forum that provides a platform to support millions of programmers by providing opportunities for them to ask questions and obtain answers from peers [5]. In cases where users do not receive answers form their peers, the user could provide answers to their own questions or sometimes, the questions remain unanswered. Key to the success of such a forum is the ability of users to receive prompt answers to their questions [2]. We studied the answer response time of questions in SO from January 2009 to December 2015, the distribution of questions answered by question askers themselves, and the proportion of unanswered questions. We defined the answer response time as the time difference between the times when a question is asked to when it receives the first answer. Figure 1 shows the answer response time of questions for each of 6 defined time intervals (within 15 minutes, within 1 hour, within 1 day, within 1 week, within 1 month and over a month) for each year under consideration.

Figure 1 shows that the majority of questions in SO get answered within 15 minutes, although we also observe a continuous decrease over time in the percentage of questions answered within 15 minutes. In fact, in 2015 just 36% of the questions were answered within 15 minutes compared to 2009 when about 57% of the questions were answered within 15 minutes. Also, questions with response times above 15 minutes have continually increased. In fact, some of the questions which received late answers were actually answered by the question askers themselves. Specifically, the total number of questions in this category has increased from 1,946 in 2009 to 18,479 in 2015 as shown in Table 1. In fact, some of these questions never get answered. Figure 2 shows a rapid growth in the number of unanswered questions.

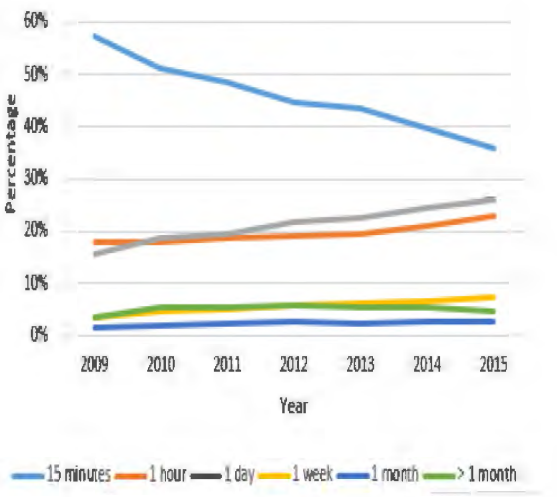


Figure 1: Response Time between Question Creation Date and First Answer Creation Date

Table 1: Questions Answered by the Question Asker

Year	Frequency
2009	1,946
2010	3,091
2011	6,701
2012	11,877
2013	16,936
2014	17,405
2015	18,479

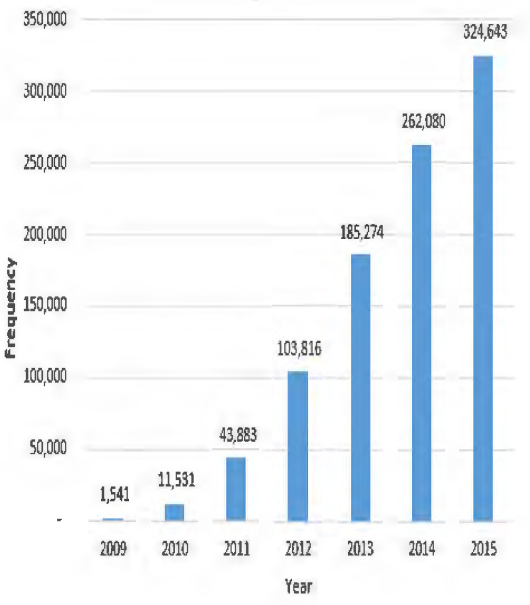


Figure 2. Number of Unanswered Questions

While this growth is partly a result of an increase in the number of questions asked in SO, we believe a growth from 1,541 in 2009 to 324,643 in 2015 is worth addressing. Moreso, Asaduzzaman et. al. [1] identified that the inability of questions to attract expert users is one of the main reasons they remain unanswered. Of course, not

receiving answers to questions or having to answer your own question yourself could deter the user from subsequently using the forum. The goal of our research is to support users who depend on online forums to receive answers to their questions. We believe the ability to predict prospective answerers for questions is the first step at supporting users to achieve this goal.

4. RANKING STRATEGIES

Results in section 3 suggest the need to support users in question and answer forums with the aim of decreasing the answer response time to questions. Our study seeks to predict such potential just-in-time peer helpers using 5 strategies for choosing such a helper. Each of these strategies considers the relevance of the question to online activities and the demonstrated knowledge in answers of the potential helpers (other users) in the past (we defined this by the co-occurrence of tags contained in the question with tags contained in the answers provided by the potential helper in the past). For each proposed strategy, personalized scores are assigned to each prospective helper based on their suitability to answer a question, as described below.

4.1 Frequency

The frequency strategy measures how frequently the prospective helper has answered questions relevant to a particular question under consideration in the past. The higher the frequency of interaction with relevant questions in the past, the more likely the user would be to answer the question. The frequency score was computed by counting the number of answer posts A relevant to the question $tag(i)$ for user u as shown in equation 1 below:

$$Score_{ui}^{freq} = \sum A(i)_u \quad (1)$$

The prospective helpers with higher scores are ranked as better helpers based on this strategy.

4.2 Knowledgeability

Knowledgeability shows how much a prospective helper knows about the question based on the number of up votes the user has earned in answering past questions with the same tag (in SO questions and answers are voted upon to show how useful and appropriate they are). This is computed as shown in equation 2 below:

$$Score_{ui}^{know} = \sum Upvotes\ (A(i)_u) \quad (2)$$

that is the sum of all upvotes to answer posts A relevant to question $tag(i)$ for user u. Prospective helpers with a higher number of up votes would be ranked as better based on this strategy.

4.3 Eagerness

Eagerness is based on monitoring the online activity of a prospective helper as depicted by the proportion of answers they have provided in the past relevant to the question compared to the total number of answers provided by the user to all questions, as shown in equation 3 below. The eagerness measure depicts the probability that a user will answer a question related to $tag(i)$:

$$Score_{ui}^{eag} = \frac{Score_{ui}^{freq}}{N_u^a} \quad (3)$$

N_u^a represents the total number of answers provided by the user to all questions. This strategy seeks to measure the interest of the user in answering questions related to $tag(i)$ by considering the proportion of relevant questions answered. We assume that users will provide more answers to questions they are more interested in; therefore the higher the proportion of relevant questions answered, the higher the likelihood the helper would be interested in answering the particular question under consideration. Prospective helpers with higher scores are ranked higher.

4.4 Willingness

This measure is a combination of how active and eager the user has been in answering questions related to the question tag in the past. That is, a user who is eager to answer questions like the question under consideration and has answered such questions a lot should be more willing to answer the question under consideration. The Bayes theorem is applied in computing this peer matching measure as shown in equation (4) below:

$$P\left(U_u^a \middle| tag(i)\right) = \frac{P(tag(i)|U_u^a) * P(U_u^a)}{P\left(tag(i)\right)} \quad (4)$$

where $P(tag(i)|U_u^a)$ is the likelihood of an answer to a question related to $tag(i)$ will be given by a user u, which is computed as shown in equation (4a) below:

$$P(tag(i)|U_u^a) = \frac{Score_{ui}^{freq}}{N(i)_a} \quad (4a)$$

$N(i)_a$ represents the total number of answers provided to $tag(i)$ by all users. $P(U_u^a)$ is the prior probability of a user u answering a question related to $tag(i)$ which is equivalent to the eagerness of the user as computed in equation (3) above. $P\left(tag(i)\right)$ is the probability that a question related to $tag(i)$ will be asked (this is the same for all prospective helpers). To maximize the posterior probability as shown in equation (4), the numerator is maximized since the denominator is common to all the prospective helpers. The willingness score is therefore computed as shown in equation (4b) below (we substituted values from equation (4a) and (3) into equation (4)):

$$Score_{ui}^{will} = \frac{Score_{ui}^{freq}}{N(i)_a} * Score_{ui}^{eag} \quad (4b)$$

Prospective helpers with higher willingness score are ranked higher.

4.5 Recency

The recency strategy corresponds to how actively and recently the prospective helper has provided answers to relevant questions. The recency score is computed for each prospective helper based on the timestamp of the latest answer A provided relevant to the question $tag(i)$ as shown in equation 5 below:

$$Score_{ui}^{rec} = latest(Time\ A(i))_u \quad (5)$$

This simply means that the recency score for a user u who has provided answers A to questions with $tag(i)$ will be the timestamp of their latest answer (the maximum time). Under this measure prospective helpers who have answered related questions more recently would be ranked higher than those who answered such questions earlier. As the interests of potential helpers could evolve [5], providing answers to relevant questions in recent times could imply the prospective helper is still interested in answering questions related to the question tags. Although Greer et al. [4] argued that helpers who have recently provided help should be exempt, to avoid overworking a peer helper in SO, this might not be as true, as users might still be willing to provide help with the goal of earning some incentive from the forum (this could be the earning of a reputation score or of various badges).

5. EXPERIMENTAL EVALUATION AND RESULTS

The goal of our study is to explore the effectiveness of different peer-helper matching strategies in terms of their ability to predict a relevant peer-helper who will provide quick answers. For each of the strategies described in section 4, we evaluated their effectiveness using the historical SO data of each prospective helper going back 1 month, 3 months and 6 months from the time a question was asked. For this study we only focused on *java*[2] questions (53,731 of them) that received at least one answer within the first hour of creation with 254,766 prospective helpers to choose from. These represent questions that were answered fairly much in time which we feel would provide a good rationale in evaluating the effectiveness of the various strategies in predicting the just-in-time answerers. Likewise, we regarded only users who were available online within the first hour the question was created to be users who would be prospective helpers, as in a real life situation; they are the set of users who are more likely to view the questions earlier and provide quicker response. Also, we employed the one hour time frame in defining the online users as it aligns with the time frame of the questions considered in this study.

We also need a success measure for our predictions. Similar to the study by Tian et al. [9], we deem it a success if a user in the top N ranked users computed by a strategy is also a user who actually answered the question under consideration in SO. The success rate S@N for each strategy can then be computed by dividing the total number of successes by the total number of questions as shown in equation 6 below.

$$S@N = \frac{Total\ Number\ of\ Successes}{Total\ Number\ of\ Questions} * 100\% \quad (6)$$

We can use different values of N to get a glimpse into how our prediction would perform as the number of prospective helpers predicted increases. In our study we used N = 1, 5, 10, and 20. Finally, we wanted to compare the effectiveness of our strategies in three different prediction criteria: predicting the answerer who responded first in SO, predicting the answerer who gave the best answer according to the user who asked the question, and predicting the answerer whose answer other SO users ranked as having the best score.

***Predicting the first answerer*:** This criterion evaluates the ranked list of prospective helpers predicted for each of the strategies with the aim to know their effectiveness at predicting the user who will first provide an answer to the question. The results in table 2 show that considering the willingness of a prospective helper has the highest success rate of 55.86% with S@20 using a time frame of 6 months.

***Predicting the best answerer*:** In SO, from the numerous answers provided to a question, the question asker can mark only one of the answers as accepted which indicates the best answer according to the asker [9]. The goal of this evaluation criteria is to determine the success of the measures at identifying the best answerer from the ranked list of prospective helpers suggested. The results are shown in table 3 below. As in predicting the first answerer, we observed that the willingness peer matching strategy has the highest success rate of 54.62% with S@20 using the 6 months defined time line.

***Predicting the answerer with the highest score*:** Other community (SO) members also have the privilege to vote on the answers provided if they wish. In some cases the answer voted as best by the question asker might not necessarily be the answer with the highest score according to the community. With this evaluation criterion we want to examine the effectiveness of the peer matching strategies at predicting the user with the highest score. Results from this evaluation are shown in table 4 below. Amongst the 7 strategies considered, again we observed willingness of the prospective users has the highest success rate at predicting the user who obtained the highest success with a success rate of 56% with S@20 using the 6 months defined time line.

Overall, with the 3 evaluation criteria we achieved the highest success rate with the willingness measure and the least success with the recency strategy. Also, we observed that as the number of months increases from 1 to 6 months, we did not see any tremendous difference in the success rate for all the strategies. Tables 2 - 4 show (unsurprisingly) that as N increases, the success rate of the prediction also increases. Comparing all 3 evaluation criteria, we achieved the highest success while predicting the user with the highest score, although the success rate obtained with the other criteria (i.e. predicting the first answerer and best answerer) did not differ significantly using S@20. In the next section, we show how we attempted to improve the performance of these strategies by including an additional measure called *timeliness*.

6. PREDICTION OF JUST-IN-TIME HELPERS

The main goal of this study is to predict helpers just-in-time, i.e. helpers who would provide answers as quickly as possible. Therefore we included a *timeliness* criterion that takes into consideration how quickly a prospective helper would provide an answer to a question. We used the 15 minutes time frame as it represents the average time in which most questions are answered (although, the percentage of questions answered within this time frame has decreased as shown in section 3). For each prospective helper, we computed the timeliness measure as shown in equation (7):

$$Score_u^{Tim} = \frac{N_u^{t \le 15}}{N_u^a} \quad (7)$$

$N_u^{t \le 15}$represents the number of questions the user answered within 15 minutes in the past while N_u^a represents the total number of answers provided by user u. To see how well our various strategies work in predicting such just-in-time helpers, we multiplied the timeliness score $Score_u^{Tim}$ obtained by each user by their respective score on each of the other strategies except for the recency strategy. We excluded the recency strategy in this prediction as it is the weakest measure as shown in tables 2-4. Moreover, the recency score computed as shown in equation 7 is a timestamp value which cannot be multiplied by the timeliness score as can the numeric values obtained with other strategies. Finally, since we did not observe any major differences when we used the 1 month history data of the prospective helper as compared to the 6 month history, in predicting the just-in-time helpers we only employed the history data of the prospective answerers over the 1 month time frame. This also saved a lot of computational time. The results obtained are shown in tables 5-7 for each of the evaluation criteria.

[2] We focused on questions containing *java* tags as this is the most used programming related tag in SO.

Table 2: Success Rate at Predicting the First Answerer

First Answerer	1 Month				3 Months				6 Months			
	S@1	S@5	S@10	S@20	S@1	S@5	S@10	S@20	S@1	S@5	S@10	S@20
frequency	5.40%	18.87%	31.65%	49.13%	5.27%	18.93%	31.37%	48.23%	5.81%	20.00%	33.13%	50.81%
recency	2.39%	11.31%	20.30%	33.60%	2.61%	11.67%	20.67%	33.96%	2.80%	12.66%	21.81%	35.59%
eagerness	1.81%	9.89%	21.29%	43.57%	1.88%	10.09%	21.53%	43.82%	2.01%	10.32%	23.15%	47.00%
knowledgeability	5.59%	17.97%	28.10%	39.52%	5.50%	17.85%	28.05%	39.32%	5.97%	19.03%	29.78%	41.94%
willingness	5.70%	21.06%	35.89%	**54.20%**	5.58%	21.11%	35.35%	**52.90%**	6.06%	22.43%	37.44%	**55.86%**

Table 3: Success Rate at Predicting the Best Answerer

Best Answerer	1 Month				3 Months				6 Months			
	S@1	S@5	S@10	S@20	S@1	S@5	S@10	S@20	S@1	S@5	S@10	S@20
frequency	5.27%	19.60%	31.84%	48.25%	5.27%	19.58%	31.72%	47.26%	5.77%	20.78%	33.34%	50.26%
recency	2.91%	12.20%	21.19%	33.91%	3.17%	12.55%	21.48%	34.35%	3.53%	13.70%	22.84%	36.12%
eagerness	1.75%	9.36%	19.98%	41.03%	1.89%	9.76%	20.69%	41.43%	1.97%	9.90%	22.06%	44.61%
knowledgeability	5.58%	19.18%	29.24%	40.66%	5.58%	18.99%	29.27%	40.66%	5.97%	20.33%	31.22%	43.54%
willingness	5.58%	21.40%	35.40%	**52.47%**	5.57%	21.30%	35.08%	**51.52%**	6.00%	22.80%	37.29%	**54.62%**

Table 4: Success Rate at Predicting the Answerer with the Highest Score

Highest Score	1 Month				3 Months				6 Months			
	S@1	S@5	S@10	S@20	S@1	S@5	S@10	S@20	S@1	S@5	S@10	S@20
frequency	5.43%	19.96%	32.48%	49.38%	5.47%	20.28%	32.46%	48.43%	5.90%	21.49%	34.34%	51.37%
recency	2.88%	12.26%	21.62%	34.90%	3.20%	12.92%	22.11%	35.33%	3.66%	14.11%	23.40%	37.13%
eagerness	1.82%	9.30%	20.09%	42.12%	1.94%	9.96%	21.18%	42.88%	2.02%	10.19%	22.61%	46.05%
knowledgeability	5.79%	19.99%	30.29%	41.73%	5.76%	19.92%	30.36%	41.80%	6.12%	21.16%	32.32%	44.63%
willingness	5.66%	21.71%	36.23%	**53.63%**	5.67%	21.89%	36.09%	**52.78%**	6.10%	23.45%	38.52%	**56.00%**

Table 5. Timeliness Success at Predicting the First Answerer

First Answerer Timeliness	1 Month			
	S@1	S@5	S@10	S@20
frequency	6.54%	21.86%	36.16%	55.01%
eagerness	5.46%	26.71%	43.31%	**63.15%**
knowledgeability	6.06%	20.10%	30.45%	41.54%
willingness	6.91%	24.89%	40.55%	60.34%

Table 6. Timeliness Success at Predicting the Best Answerer

Best Answerer Timeliness	1 Month			
	S@1	S@5	S@10	S@20
frequency	6.10%	20.95%	34.09%	50.84%
eagerness	3.80%	20.95%	35.27%	53.76%
knowledgeability	5.85%	20.19%	30.38%	41.45%
willingness	6.45%	23.64%	37.91%	**55.34%**

Table 7. Timeliness Success at Predicting the Answerer with the Highest Score

Highest Score Timeliness	1 Month			
	S@1	S@5	S@10	S@20
frequency	6.21%	21.46%	34.98%	51.94%
eagerness	3.90%	21.47%	36.47%	55.34%
knowledgeability	6.02%	21.06%	31.38%	42.54%
willingness	6.48%	24.30%	38.93%	**56.65%**

7. DISCUSSION

The aim of our research is to support lifelong learners as they interact with peers in open ended learning environments like SO. As lifelong learners are responsible for their own learning [7], millions of them depend on such learning forums to meet their learning needs on a daily basis. Obtaining timely answers to questions is important [2] in supporting lifelong learners and in enhancing the sustainability of such an online learning community. However, we observed (as shown in section 2) that the answer response times to questions have increased and in some cases the question askers have to answer their own questions themselves, which can deter the lifelong learner. In this study, we address this problem by predicting prospective users who are likely to provide the most timely answers to their question.

Previous studies by Greer et al. [3, 4] and Vassileva et al. [10] have identified the various strategies that could be used in predicting the prospective helpers within the classroom and workplace learning environments. In this study we explored the effectiveness of the various strategies at predicting prospective helpers in SO, an environment with vastly more learners seeking answers to their questions than in academic classes. We achieved the highest success rate S@20 of 54.20% using the 1 month time line with the willingness strategy. Also, with the recency measure, performing the poorest amongst all the measures defined, our study affirms the claim by Greer et al. [2] that helpers who have recently provided help would be less likely to provide answers and they should be exempted to avoid overworking a peer helper.

We improved upon the results obtained from each of the strategies described in section 4, by including an additional criterion called *timeliness*. This criterion takes into consideration the probability

that a user would answer a question quickly. We achieved a maximum success rate S@20 of 63.15% (eagerness), 55.34% (willingness) and 56.65% (willingness) in predicting, respectively, the first answerer, the best answerer, and the answerer who will provide the highest score. These values represent an improvement in the success rate from 43.57% to 63.15% (eagerness), 52.47% to 55.34% (willingness), 53.63% to 56.65% (willingness) in predicting the first answerer, best answerer and the answerer who will provide the highest score respectively using the 1 month time frame (comparing our results from tables 2-4 with results obtained in tables 5-7). While these results likely require improvement, these values are an improvement over the previous work by Tian et al. [9] whom obtained a success rate S@20 of 12.57% and S@100 of 23.06% while predicting the best answerer using the topic modelling approach. We believe the results obtained in this study for all the strategies defined outperforms this previous work. The variation in our results from those of Tian et al. is presumably because our study was restricted to questions that were answered fairly much on time (i.e. questions with at least one answerer within the first hour the question was created). We focused on these sets of questions because the goal of our study is to predict the just-in-time helpers who will provide quick answers to the questions in which case, questions answered late would not suffice. Although Yang and Manandhar [11] argued for the use of the topic modelling approach in predicting the best answerer, our results suggest that this is a less informative approach.

For each of the peer matching strategies, we also studied their performance in predicting the relevant peer helpers using the history data for prospective peer helpers for the periods of 1 month, 3 months and 6 months. Our aim is to understand the tradeoff of using older data about the user vs newer data. As Kay and Kummerfield [7] already identified, there is a trade-off between the usefulness of retaining older information about the lifelong learner and preserving only the recent data. Our results show that employing older information (6 months) about the learner was at best only marginally better when compared to the results achieved with the newer information (1 month). This confirms an earlier study [5] we did in predicting (again in SO) what the user would want to learn in the future, where we showed that employing shorter term information about the user's past behavior proved more effective in predicting what the user would be learning in future

While we feel that we have achieved good prediction accuracy with our strategies (especially as compared to other studies), we would still like to enhance the accuracy to ensure the usefulness of our strategies in a real learning environment. So, in our next experiment, we aim to further improve on our results, pushing them well above our current success rates if we can. Our aim will be to develop new strategies that can identify users who would have been likely to help answer the question quickly. Overall, we feel this research is a promising first step for being able to show how we can find good peer helpers to help professional lifelong learners who are keeping themselves up-to-date through interactions with their peers in online forums.

8. ACKNOWLEDGMENTS

We would like to thank David Edgar Kiprop Lelei for his help with presenting the paper. Aso, we like to thank the Natural Sciences and Engineering Research Council of Canada and the University of Saskatchewan for providing funding to support this research.

9. REFERENCES

[1] Asaduzzaman, M., Mashiyat, A. S., Roy, C. K., & Schneider, K. A. (2013, May). Answering questions about unanswered questions of stack overflow. 10th IEEE Working Conference on Mining Software Repositories (MSR 2013) (pp. 97-100).

[2] Bhat, V., Gokhale, A., Jadhav, R., Pudipeddi, J., & Akoglu, L. (2014, August). Min (e) d your tags: Analysis of question response time in stackoverflow. IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014) (pp. 328-335).

[3] Greer, J. E., McCalla, G., Collins, J. A., Kumar, V. S., Meagher, P., & Vassileva, J. (1998). Supporting peer help and collaboration in distributed workplace environments. International Journal of Artificial Intelligence in Education (IJAIED), 9, pp. 159-177.

[4] Greer, J., McCalla, G., Cooke, J., Collins, J., Kumar, V., Bishop, A., & Vassileva, J. (1998, August). The intelligent helpdesk: Supporting peer-help in a university course. International Conference on Intelligent Tutoring Systems (ITS 1998) (pp. 494-503).

[5] Ishola, O. M., & McCalla, G. (2016, September). Detecting and supporting the evolving knowledge interests of lifelong professionals. European Conference on Technology Enhanced Learning (EC-TEL 2016) (pp. 595-599).

[6] Ishola, O. M., & McCalla, G. (2017, June) Personalized tag-based knowledge diagnosis to predict the quality of answers in a community of learners. International Conference on Artificial Intelligence in Education (AIED 2017), (to appear).

[7] Kay, J., & Kummerfield, B. (2009, June). Lifelong user modelling goals, issues and challenges. In Lifelong User Modelling Workshop at the International Conference on User Modeling, Adaptation and Personalization (UMAP 2009) (pp. 27-34).

[8] Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. The Journal of the Learning Sciences, 2(3), pp. 277-305.

[9] Tian, Y., Kochhar, P. S., Lim, E. P., Zhu, F., & Lo, D. (2013, November). Predicting best answerers for new questions: An approach leveraging topic modeling and collaborative voting. Workshops at the International Conference on Social Informatics (SocInfo 2013) (pp. 55-68).

[10] Vassileva, J., Greer, J., McCalla, G., Deters, R., Zapata, D., Mudgal, C., & Grant, S. (1999, July). A multi-agent approach to the design of peer-help environments. International Conference on Artificial Intelligence in Education (AIED 1999) (pp. 38-45).

[11] Yang, B., & Manandhar, S. (2014, August). Tag-based expert recommendation in community question answering. IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014) (pp. 960-963).

Combining Machine Learning and Natural Language Processing to Assess Literary Text Comprehension

Renu Balyan
Arizona State University
Tempe, AZ, USA
renu.balyan@asu.edu

Kathryn S. McCarthy
Arizona State University
Tempe, AZ, USA
ksmccar1@asu.edu

Danielle S. McNamara
Arizona State University
Tempe, AZ, USA
dsmcnamara@asu.edu

ABSTRACT
This study examined how machine learning and natural language processing (NLP) techniques can be leveraged to assess the interpretive behavior that is required for successful literary text comprehension. We compared the accuracy of seven different machine learning classification algorithms in predicting human ratings of student essays about literary works. Three types of NLP feature sets: unigrams (single content words), elaborative (new) n-grams, and linguistic features were used to classify idea units (paraphrase, text-based inference, interpretive inference). The most accurate classifications emerged using all three NLP features sets in combination, with accuracy ranging from 0.61 to 0.94 (F=0.18 to 0.81). Random Forests, which employs multiple decision trees and a bagging approach, was the most accurate classifier for these data. In contrast, the single classifier, Trees, which tends to "overfit" the data during training, was the least accurate. Ensemble classifiers were generally more accurate than single classifiers. However, Support Vector Machines accuracy was comparable to that of the ensemble classifiers. This is likely due to Support Vector Machines' unique ability to support high dimension feature spaces. The findings suggest that combining the power of NLP and machine learning is an effective means of automating literary text comprehension assessment.

Keywords
Natural language processing; supervised machine learning; classification; interpretation

1. INTRODUCTION
Text comprehension researchers employ a variety of methods to assess how people process and understand the things that they read. The majority of this work has focused on how readers comprehend expository or informational texts (e.g., science textbooks or historical accounts) and simple narratives (e.g., brief plot-based texts). Much less work has been done to investigate the kinds of processes that occur when readers read literary texts, such as the poems, short stories, and novels assigned in English-Language Arts classrooms [1]. More so than in other text domains, literary text comprehension requires the construction of interpretations that go beyond the literal story to speak to a deeper meaning about the world at large [2].

In order to measure interpretation and assess literary comprehension, researchers have relied on collecting students' essays about the text. The essay can then be scored in a variety of ways to address different questions about the comprehension process [3]. Unfortunately, reliably evaluating essays is both time and resource intensive. In other text domains, researchers have begun to develop natural language processing (NLP) tools to automate this scoring [4,5]. With this in mind, our goal was to develop a means of automatically assessing students' essays about literary texts, with particular attention readers' interpretation of a text's potential deeper meaning.

Our purpose was to investigate if NLP and machine learning could be combined and leveraged to accurately predict human ratings of students' essays. We drew upon existing text comprehension research to identify and extract three NLP feature sets that were relevant to literary text comprehension. These feature sets were used to compare seven machine learning classification algorithms in their ability to classify idea units in student essays as literal (paraphrase or text-based inferences) or interpretive.

1.1 Text Comprehension
The field of text comprehension investigates the complex activities involved in how people read, process, and understand text. As people read, they generate a mental representation, or mental model. The quality, structure, and durability of this representation reflect the reader's comprehension of the text [6,7]. A critical aspect of this mental representation is the inclusion of inferences. Inferences connect different parts of the text or connect information from the text to information from prior knowledge. Those who generate more inferences have a more elaborated mental representation [6,7]. Importantly, different types of texts and tasks afford different amounts and types of inferences [8]. For example, readers studying for an upcoming test generate explanatory and predictive inferences, whereas readers reading for fun generate personal association inferences. These different types of inferences suggest readers are engaging in different processes and are constructing different mental representations of the text [9]. Given the importance of inferences in successful text comprehension, a majority of text research is aimed at understanding when and how inferences are constructed [10].

1.2 Literary Comprehension
In the study of literary text comprehension, researchers are interested in interpretive inferences. Interpretive inferences reflect a representation of the author's message or deeper meaning [11]. Take for example, the story of the Tortoise and the Hare. A reader may make text-based inferences to maintain a coherent representation of the events of the text. A reader might generate the inference *The tortoise was able to pass the hare because the hare was sleeping* to explain why the slow tortoise was able to beat the speedy hare. In contrast, a reader might generate an interpretive inference that goes beyond the story world to address the moral or message of the story, such as *It is better for someone to be perseverant than talented.* Research indicates that expert literary readers (e.g., English Department faculty or graduate students) allocate more effort to generating interpretive inferences, whereas

novices, who tend to have less domain-specific reading goals and strategies, tend to merely paraphrase, or restate the plot.

Notably, there is no one "right" interpretation, but rather a multitude of possibilities that may be more or less supportable by the evidence in the text [11,12]. Indeed, some might argue that the moral of the Tortoise and the Hare is not about the tortoise's achievement, but instead reflects a cautionary message about the hare's behavior, such as *People should not be over-confident.* As such, assessing interpretation is more difficult than evaluation of performance in well-defined domains that have a single correct answer. To capture and assess interpretations, researchers have relied on open-ended measures, such as think-aloud protocols, in which readers talk aloud about their processing as they read through the text [13,14,15] and through post-reading essays in which students construct responses to various writing prompts [16]. The transcribed think-aloud data and essays are then parsed into sentences or idea units and scored for the kinds of paraphrases and inferences present. In order to reliably categorize the idea units and essay quality, experts develop and refine a codebook that is then used to train raters. These raters work both independently and collaboratively to reach a satisfactory metric of reliability, such as percent agreement or intra-class correlation.

1.3 Natural Language Processing

More recently, a push has been made to incorporate NLP in text comprehension research [17]. Linguistic features from existing texts are extracted using NLP tools [18]. These tools draw upon corpora of large sets of texts and human ratings to measure aspects of language, such as word overlap, semantic similarity, and cohesion. NLP tools can be used to identify and measure linguistic features that reliably predict human essay ratings [4].

2. DATA & METHODS

2.1 Corpus

The corpus included 346 essays written by college students from two experiments investigating literary interpretation [16,19]. The essays were written about two different short stories from different literary genres (science-fiction, surrealist). In the behavioral experiments, participants received differing reading instructions and writing prompts that biased readers towards paraphrasing or interpretation.

2.2 Human Ratings

Four expert raters scored the set of essays using a previously developed codebook [16]. Essays were parsed into idea units (n = 4,111) and each idea unit was labeled as verbatim, paraphrase, text-based inference, or interpretive inference (Table 1). Given the low amount of verbatim units, verbatim and paraphrase were collapsed into a single paraphrase type.

2.3 Classification Algorithms

Machine learning investigates how machines can automatically learn to make accurate predictions based on past observations. Classification is a form of machine learning that uses a supervised approach. In supervised machine learning, the model learns from a set of data with the class labels already assigned. The model uses this existing classification to make classifications on new data.

Data classification consists of two steps; a learning step (or training phase), and a classification step. In the learning step, a classification algorithm builds a model by "learning from" a training set composed of database tuples, and their associated class labels. A training set may be represented as (X, Y), where X_i is an n-dimensional attribute vector, $X_i=(x_1, x_2,\ldots x_n)$ depicting n measurements made on the tuple from n database attributes, respectively A_1, A_2,..A_n. Each attribute represents a 'feature' of X. Each X_i belongs to a pre-defined class label, represented as Y_i [20]. In the classification step, the trained model is used to predict class labels for a test set of new data set that has not been used during model training. This test data is used to determine the accuracy of a classification algorithm, or *classifier*.

Some of the most commonly used classification algorithms are Naïve Bayes [21], Decision Trees [22], Maximum Entropy [23,24], Neural Networks [25], and Support Vector Machines [26,27]. In addition, researchers also employ ensemble techniques that use more than one of the classifying algorithms. These ensemble algorithms include Bagging [28], Boosting [29], Stacking [30], and Random Forests [31].

2.3.1 Naïve Bayesian

Naïve Bayesian algorithm is based on the Bayes' theorem of posterior probability. It is a probabilistic learning method. It assumes that the effect of an attribute value on a given class is independent of other attributes values [21].

Table 1. Idea unit identification: Definitions and examples
(From McCarthy & Goldman, 2015)

Type	Description	Example from *Harrison Bergeron*	Example from *The Elephant*
Verbatim	Copied directly from the text	*The Handicapper General, came into the studio with a double-barreled ten-gauge shotgun. She fired twice, and the Emperor and the Empress were dead before they hit the floor.*	*The schoolchildren who had witnessed the scene in the zoo soon started neglecting their studies and turned into hooligans. It is reported they drink liquor and break windows. And they no longer believe in elephants.*
Paraphrase	Rewording of the sentences from the text; Summary or combining of multiple sentences from the text	*Then [Harrison] and the ballerina were killed by Diana Moon Glampers, the Handicapper General.*	*After seeing this the students gave up on education became drunks and stopped believing in elephants.*
Text-Based Inference	Reasoning-based on information presented in the story, with some use of prior knowledge; connecting information from two parts of the text	*Diana Moon Glampers killed them because they tried to show their true selves.*	*After being deceived by the fake elephant, the children became poor students, and grew up behaving badly because they were lied to*
Interpretive Inference	Inferences that reflect nonliteral, interpretive interpretations of the text	*It shows what kind of a place the world can turn out to be if we let [the government] get out of control.*	*The theme is that being lied to ends the innocence of the young boys and girls.*

2.3.2 Decision Trees

The Decision Trees learning method approximates discrete-valued target functions. The learned function is represented as a decision tree, which is further represented as a set of if-then rules. Each node in the tree specifies a test of some attribute, and one of the possible values of the attribute represents a branch in the tree. The attribute considered for a node is based on the statistical property, information gain [22].

2.3.3 Maximum Entropy (MaxEnt)

MaxEnt models work on a simple principle, and choose a model that is consistent with all of the given facts. The models are based on what is known, and do not make any assumptions about the unknowns [23,24].

2.3.4 Neural Networks

Neural Networks is a computational approach based on a collection of neural units. It is an attempt to model the information processing capabilities of the human nervous system. These models are self-learning, and use a back-propagation algorithm for updating the weights based on feedback [25,32].

2.3.5 Support Vector Machine (SVM)

SVM constructs a hyperplane that separates the data into classes. SVMs are efficient for high-dimensional feature spaces and are among the best supervised learning algorithms [26,27].

2.3.6 Bagging

Bagging (or Bootstrap Aggregation), is a meta-algorithm that considers multiple classifiers. It creates bootstrap samples of a training set using sampling with replacement. Bagging trains each model in the ensemble using each bootstrap sample, and performs classification based on majority voting from trained classifiers [28].

2.3.7 Boosting

Boosting, a meta-algorithm that incrementally builds an ensemble by iteratively training weak learners or classifiers. While training new models, it emphasizes instances that are misclassified by the previous models. Thus, each model is trained on weighted data from the previous model performance. The final result is the weighted sum of the results of all of the classifiers [29].

2.3.8 Stacking

Stacking (or stacked generalization), combines multiple classifiers generated by different learning algorithms on a single data set. This algorithm works by first generating a set of base-classifiers, and then trains a meta-level classifier to combine the outputs of the base-classifiers [30].

2.3.9 Random Forests

Random Forests (or random decision forest) is designed to overcome the "overfitting" problem of decision trees. Random Forests constructs a multitude of decision trees in the training phase, and uses majority voting for classification [31,33,34].

2.4 Feature Sets

Three NLP feature sets were identified as theoretically relevant to the objective: unigrams, linguistic characteristic scores, and "elaborative" (new) unigrams.

2.4.1 Unigrams

Unigrams are the individual content words present in the idea units. The value of a unigram feature was the frequency of that unigram in the corpus. Some of the most common words appearing in the idea units are *elephant* (>1000), *story* (575), *zoo* (429), *handicap* (361), *government* (323), *believe* (158), and *think* (147).

2.4.2 Linguistic Characteristics

The second set of features considered were the linguistic characteristic scores. Ideas that reflect events from the text are likely to be more concrete, whereas those that are interpretive reflect themes (e.g., freedom, loss of innocence) are more abstract [35]. Thus, both *concreteness* and *imagability* were included as indices. Related to the greater sophistication in interpretive language, we also included *word familiarity* and *age of acquisition*. These linguistics characteristics were derived from merging norms of human ratings from three sources [36,37,38]. Details of merging are provided in appendix 2 of the MRC Psycholinguistic Database User Manual [39]. The characteristics, as defined by McNamara and colleagues [40], appear in Table 2.

Table 2. Descriptions of relevant linguistic characteristics
(From McNamara, Graesser, McCarthy, and Cai, 2014)

Linguistic Characteristic	Description
Concreteness	The degree to which a word is non-abstract
Imagability	How easy it is to construct image of a word in one's mind
Familiarity	How familiar a word is to an adult
Age of Acquisition	The age at which a word first appears in a child's vocabulary

2.4.3 Elaborative n-grams

The third feature set was the frequency of "elaborative" n-grams. These were words (unigrams), two consecutive words (bigrams) or three consecutive words (trigrams) that were new in the sense that they appeared in the idea units, but not in the original story. In addition, frequency of occurrence of a set of cue words or phrases that indicate an interpretive idea unit was included in this feature set.

We used a set of 'R' packages for implementing classification algorithms, and extracting the feature sets. The 'R' packages used for classification include 'RTextTools', 'e1071', 'randomForest', 'nnet', 'MASS', and 'caret'. The packages used for text mining, and extracting n-grams from the idea units and essays were 'tm', 'tau', 'openNLP', 'qdap', and 'quanteda'.

3. EXPERIMENTS & RESULTS

3.1 Feature Selection

The three NLP feature categories (frequency of unigrams, linguistic features of words, and number of "elaborative" n-grams and cue words) were tested in seven experiments.

The total number of unigrams extracted from the idea units was 4,406, resulting in a frequency matrix of 4,111 X 4,406 dimensions. This was more than the number of idea units in the corpus. As a means of reducing the dimensions in the data set, highly correlated unigrams (Pearson $r > .65$) were removed. However, this exercise did not significantly reduce the dimensions. It was noted that many of the unigrams did not appear frequently. Several frequency thresholds were tested to determine a frequency that would reduce dimensions, but not overly affect the accuracy of the model. It was determined that a frequency threshold of 10 was sufficient. Including only those unigrams that appeared in the corpus at least 10 times reduced the feature dimensions from 4,406 to 609.

For the second set of features we considered an initial set of 56 linguistic characteristics. The linguistic features included *concreteness, familiarity, imagability* and *age of acquisition* scores

for all the words, content words, function words, and all words with or without keywords. These features were extracted using two NLP tools: the Tool for the Automatic Analysis of Lexical Sophistication [41] and the Tool for Automatic Analysis of Text Cohesion [42]. Highly correlated (Pearson r >.85) features were removed, yielding 18 linguistic features for the classification tests.

For the "elaborative" n-grams feature set (unigrams, bigrams, and trigrams present in the idea units, but not the original story and cue words), the bigrams and trigrams were found to be highly correlated (Pearson r > 0.85). Consequently, only trigrams were included. In total, three features were used in the elaborative n-gram feature set for classification.

This final feature set was used to classify each idea unit as paraphrase, text-based inference, or interpretive inference using ML classification algorithms. Similar approaches have been used to classify other kinds of texts [43].

3.2 Idea Unit Classification

After experimenting with a large number of classification algorithms, we selected four machine learning classification algorithms (Trees, Support Vector Machine [SVM], Neural Networks, Maximum Entropy [MaxEnt]), as well as three ensemble approaches (Bagging, Boosting, Random Forests) to classify the idea units. Multiclass classification algorithms and 10-fold cross-validation were used in seven experiments to test the feature sets (609 unigrams, 18 linguistic features, and 3 elaborative n-grams) individually and in combination. Summary of classification accuracy for all the algorithms is presented in Table 3.

The bold entries in Table 3 indicate the maximum accuracy for each of the features. Random Forests achieved the highest accuracy for all experiments except when using elaborative n-grams as features. The Boosting algorithm classifier achieved the maximum accuracy in this case.

The italicized entries in Table 3 indicate the maximum accuracy achieved by a classification algorithm. Generally, the classification algorithms achieved high accuracy when a combination of all features was used. The accuracy for the algorithms varied between 0.77 and 0.94 when considering a combination of all the features, except for the Trees algorithm where the accuracy was quite low, 0.61. In fact, the accuracy for the Trees algorithm was low in all cases irrespective of the features considered.

F-scores for the three types of idea units produced by participants (interpretive, paraphrase, text-based) are summarized in Tables 4 and 5 for single classifiers and ensemble of classifiers, respectively. The bold numbers indicate the highest F-score for each type of idea unit. For the single classifiers, SVM achieved the highest F-score for paraphrases (F = 0.81) and for interpretive inferences (F = 0.73). MaxEnt obtained the highest F-score for single classifiers for text-based inferences (F = 0.42). For ensemble classifiers, Random Forests again performed the best, with the highest F-scores for paraphrases (F = 0.80) and interpretive inferences (F = 0.70). The Bagging algorithm achieved the highest F-score (0.30) for text-based inferences in ensemble category. The F-scores for identifying text-based inferences were relatively low, suggesting a machine learning approach may be better suited for identifying paraphrases and interpretations. The NAs in Table 4 indicate that the algorithm did not classify any idea unit as text-based.

Table 3. Accuracy for different classification algorithms with different feature combinations

[1]Unigrams (n=609); [2]Linguistic Features (n=18); [3]Elaborative n-grams (n=3; unigrams, trigrams, cue words)

	Classification Algorithm						
Feature	**SVM**	**Trees**	**MaxEnt**	**NeuralNets**	**Boosting**	**Bagging**	**Random Forests**
UNI[1]	0.75	0.58	0.81	*0.77*	0.73	0.75	**0.86**
LIN[2]	0.80	0.56	0.55	0.58	0.77	0.92	**0.94**
ENC[3]	0.64	0.60	0.58	0.62	***0.79***	0.63	0.61
UNI + LIN	0.77	0.58	*0.83*	0.76	0.74	0.92	***0.95***
UNI + ENC	0.78	*0.61*	0.80	*0.77*	0.77	0.82	**0.88**
LIN + ENC	*0.92*	0.59	0.62	0.63	*0.79*	*0.93*	**0.94**
UNI + LIN+ ENC	0.81	*0.61*	0.82	*0.77*	*0.79*	*0.93*	**0.94**

Table 4. F-Scores for Single classifiers

[1]Unigrams (n=609); [2]Linguistic Features (n=18); [3]Elaborative n-grams (n=3; unigrams, trigrams, cue words); [4]Interpretive; [5]Paraphrase; [6]Text-based Inference

	SVM			Trees			MaxEnt			NeuralNets		
Feature	**Inter[4]**	**Para[5]**	**TB[6]**	**Inter**	**Para**	**TB**	**Inter**	**Para**	**TB**	**Inter**	**Para**	**TB**
UNI[1]	0.71	0.80	0.28	0.44	0.71	NA	0.65	0.76	0.36	0.63	0.76	0.13
LIN[2]	0.45	0.73	0.13	0.27	0.70	NA	0.52	0.66	0.30	0.46	0.73	NA
ENC[3]	0.46	0.73	0.03	0.52	0.73	NA	0.50	0.72	NA	0.57	0.74	NA
UNI + LIN	0.70	**0.81**	0.35	0.49	0.72	NA	0.66	0.77	0.41	0.64	0.79	0.08
UNI + ENC	**0.73**	**0.81**	0.34	0.55	0.74	NA	0.69	0.78	0.38	0.62	0.73	0.18
LIN + ENC	0.48	0.73	0.11	0.50	0.73	NA	0.58	0.74	0.25	0.61	0.77	NA
UNI+LIN+ENC	0.72	**0.81**	0.36	0.55	0.74	0.30	0.70	0.79	**0.42**	0.63	0.79	0.06

Table 5. F-Scores for Ensemble classifiers

[1]Unigrams (n=609); [2]Linguistic Features (n=18); [3]Elaborative n-grams (n=3; unigrams, trigrams, cue words); [4]Interpretive; [5]Paraphrase; [6]Text-based Inference

	Boosting			Bagging			Random Forests		
Feature	**Inter[4]**	**Para[5]**	**TB[6]**	**Inter**	**Para**	**TB**	**Inter**	**Para**	**TB**
UNI[1]	0.65	0.77	0.06	0.65	0.76	0.17	0.68	0.79	0.27
LIN[2]	0.49	0.70	0.09	0.51	0.72	0.26	0.51	0.74	0.21
ENC[3]	0.52	0.73	0.06	0.51	0.73	0.18	0.53	0.74	0.02
UNI + LIN	0.57	0.73	0.12	0.61	0.76	0.27	0.67	0.78	0.23
UNI + ENC	0.62	0.76	0.07	0.66	0.77	0.27	**0.70**	**0.80**	0.28
LIN + ENC	0.55	0.73	0.23	0.57	0.75	0.25	0.58	0.77	0.21
UNI +LIN + ENC	0.61	0.76	0.18	0.63	0.79	**0.30**	0.67	0.79	0.23

4. CONCLUSIONS

This study demonstrates that a classification approach using unigrams, linguistic features, and "elaborative" n-grams can be used to accurately predict human ratings of idea unit classification for essays about literary works.

This study indicated that ensemble classification algorithms were, generally, more accurate than single classifiers. Random Forests, which is an ensemble of decision trees and uses a bagging approach, was the most accurate classifier and had the highest F-scores for most types of idea units. In contrast, the single classifier Trees showed relatively low accuracy. This finding is consistent with previous work that suggests Trees "overfits" to training data and, as a result, performs poorly on test data [44].

Interestingly, performance from the single classifier SVM was comparable to the ensemble classifiers. This classifier may have been highly accurate due to the fact that our data had a large amount of features under consideration. SVM is designed to support high-dimension spaces and data that may not be linearly separable.

This study provides a model for how machine learning and NLP can be used to assess literary text comprehension. In addition to being economical for researchers recruiting large samples and collecting large amounts of essay data, the approach can also be implemented in other automated writing evaluators (AWEs) to provide domain-specific assessment and feedback.

The presence of interpretive inferences suggests that a reader has successfully moved beyond the literal to engage in domain-appropriate interpretations. However, interpretive inferences are not necessarily indicative of higher quality literary text comprehension. Literary comprehension requires not only generating interpretations, but also justifying those interpretations with evidence from the text as well as appeals to cultural and literary norms [1,45]. Hence, good essays are likely to have a relatively even distribution of the various types of ideas (e.g., both inferences and interpretations). Our future plans include assessing the essays holistically and develop algorithms to predict those scores. Our ultimate objective is to better understand the relations between idea unit types and essay quality as well as to further the development of automated assessment of literary comprehension.

5. ACKNOWLEDGMENTS

This research was supported in part by IES Grants R305A150176, R305A130124, and R305A120707, as well as ONR Grants N00014-14-1-0343 and N00014-17-1-2300. Opinions, conclusions, or recommendations do not necessarily reflect the views of the IES or ONR.

6. REFERENCES

[1] McCarthy, K. S. 2015. Reading beyond the lines: A critical review of cognitive approaches to literary interpretation and comprehension. *Scientific Study of Literature* 5, 1(Jan. 2015), 99-128.

[2] Goldman, S. R., McCarthy, K. S., and Burkett, C. 2015. Interpretive inferences in literature. In, *Inferences during reading*, E. O'Brien, A. Cook, and R. Lorch, Eds. Cambridge University Press, New York, NY. 386-415.

[3] McCarthy, K. S., Kopp, K. J., Allen, L. K., and McNamara, D. S. under review. Methods of studying text: Memory, comprehension, and learning. In *Handbook of Research Methods in Human Memory*, H. Otani and B. Schwartz, Eds. Routledge.

[4] Crossley, S., Kyle, K., Davenport, J., and McNamara, D. S. 2016. Automatic assessment of constructed response data in a chemistry tutor. In *Proceedings of the 9th International Conference on Educational Data Mining*, T. Barnes, M. Chi, & M. Feng, Eds. (Raleigh, NC, June 29 – July 2, 2016). EDM'16, International Educational Data Mining Society, 336-340.

[5] Wiley, J., Hastings, P., Blaum, D., Jaeger, A. J., Hughes, S., Wallace, P., Griffin, T. D., and Britt, M. A. 2017. Different approaches to assessing the quality of explanations following a multiple-document inquiry activity in science. *International Journal of Artificial Intelligence in Education* (2017), 1-33.

[6] Kintsch, W. 1988. The role of knowledge in discourse comprehension: A construction-integration model. *Psychol. Rev.* 95 (Apr. 1998), 163-182.

[7] Kintsch, W. 1998. *Comprehension: A paradigm for cognition*. Cambridge University Press, Cambridge, England.

[8] Van den Broek, P., Young, M., Tzeng, T., and Linderholm, T. 1999. The Landscape Model of reading: Inferences and the online construction of memory representation. In *The construction of mental representations during reading*, H. van Oostendorp and S. R. Goldman, Eds. Psychology Press, 1999. 71–98.

[9] Van den Broek P., Lorch, R.F., Linderholm, T., and Gustafson, M. 2001. The effects of readers' goals on inference generation and memory for texts. *Memory & Cognition* 29, 8 (Dec. 2001), 1081-1087.

[10] McNamara, D. S. and Magliano, J. P. 2009. Towards a comprehensive model of comprehension. In B. Ross (Ed.), *Psychol Learn. Motiv*. 51 (Dec. 2009), Elsevier Science. New York, NY, 297-384.

[11] Langer, J. A. 2010. *Envisioning Literature: Literary understanding and literature instruction, 2nd edition*. Teachers College Press, New York, NY.

[12] Levine, S. and Horton, W. S. 2013. Using affective appraisal to help readers construct literary interpretations. *Scientific Study of Literature* 3, 1 (Jan. 2013), 105-136.

[13] Burkett, C. and Goldman, S. R. 2016. "Getting the Point" of Literature: Relations Between Processing and Interpretation. *Discourse Processes* 53, 5-6 (Jul. 2016), 457-487.

[14] Graves, B. and Frederiksen, C. H. 1991. Literary expertise in the description of fictional narrative. *Poetics* 20, 1(Feb. 1991), 1-26.

[15] Zeitz, C. M. 1994. Expert-novice differences in memory, abstraction, and reasoning in the domain of literature. *Cognition and Instruction* 12, 4(Dec. 1994), 277-312.

[16] McCarthy, K. S. and Goldman, S. R. 2015. Comprehension of short stories: Effects of task instructions on literary interpretation. *Discourse Processes* 52, 7 (Oct. 2015), 585-608.

[17] Crossley, S. A., Allen, L. K., and McNamara, D. S. 2014. Analyzing discourse processing using a simple natural language processing tool (SiNLP). *Discourse Processes* 51, 5-6 (Jul. 2014), 511-534.

[18] Jurafsky, D. and Martin, J. H. 2009. *Speech and Language Processing, 2nd edition*. Prentice-Hall, NJ.

[19] McCarthy, K.S. and Goldman, S. R. in prep. Effects of Genre Familiarity on Interpretive Behavior.

[20] Han, J., Kamber, M., and Pei, J. 2012. *Data Mining Concepts and Techniques, 3rd edition.* Elsevier.

[21] McCallum, A. and Nigam, K. 1998. *A comparison of event models for naive Bayes text classification*. In AAAI-98 Workshop on Learning for Text Categorization, Tech. rep. WS-98-05, AAAI Press.

[22] Mitchell, T. M. 1997. *Machine Learning*. McGraw-Hill, New York.

[23] Rosenfeld, R. 1994. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. Doctoral thesis, Carnegie Mellon University.

[24] Ratnaparkhi, A. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. Doctoral thesis, University of Pennsylvania.

[25] Zhang, G. P. 2000. Neural Networks for Classification: A Survey. *Trans. Sys. Man Cyber* Part C 30, 4 (November 2000), 451-462.

[26] Joachims, T. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. In *Proceedings of 10th European Conference on Machine Learning* (April 21-23). ECML'98. Springer-Verlag London, UK, 137-142.

[27] Dumais, S. T., Platt, J., Heckerman, D., and Sahami, M. 1998. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management* (Bethesda, Maryland, USA, November 02 - 07, 1998). CIKM'98. ACM, New York, NY, 148-155.

[28] Breiman, L. 1996. Bagging predictors. *Machine Learning* 24, 2 (Aug. 1996), 123-140.

[29] Krogh, A. and Vedelsby, J. 1994. Neural network ensembles, cross validation, and active learning. In *Proceedings of 7th International Conference on Neural Information Processing Systems* (Denver, Colorado). NIPS'94. MIT Press Cambridge, MA, USA, 231-238.

[30] Wolpert, D. 1992. Stacked generalization. *Neural Networks* 5, 2, 241–260.

[31] Schapire, R. E. and Singer, Y. 1999. BoosTexter: A boosting-based system for text categorization. *Machine Learning* 39, 2-3, 135-168.

[32] Rojas, R. 1996. *Neural Networks - A Systematic Introduction*. Springer-Verlag, Berlin.

[33] Schölkopf, B. and Smola, A. J. 2002. *Learning with Kernels*. MIT Press, Cambridge, MA.

[34] Ho, T. K. 1995. Random Decision Forests. In *Proceedings of the 3rd International Conference on Document Analysis and Recognition* (Montreal, QC, August 14-15, 1995). ICDAR'95, IEEE Computer Society Washington, DC, USA, 278–282.

[35] Rabinowitz, P. 1987. *Before reading: Narrative conventions and the politics of interpretation*. Ohio State University Press, Columbus, Ohio.

[36] Paivio, A., Yuille, J. C., and Madigan, S. A. 1968. Concreteness, imagery, and meaningfulness values for 925 nouns. *J. Exp. Psycho*. 76, 1p2 (Jan. 1968), 1-25.

[37] Toglia, M. P. and Battig, W. F. 1978. *Handbook of semantic word norms*. Lawrence Erlbaum.

[38] Gilhooly, K. J. and Logie, R. H. 1980. Age-of-acquisition, imagery, concreteness, familiarity, and ambiguity measures for 1,944 words. *Behav. Res. Methods & Instrum*. 12, 4, 395-427.

[39] Coltheart, M. 1981. The MRC Psycholinguistic Database. Q. *J. Exp. Psychol*. A 33, 4, 497–505.

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

[41] Kyle, K. and Crossley, S. A. 2015. Automatically assessing lexical sophistication: Indices, tools, findings, and application. *TESOL Quarterly* 49, 4 (Dec. 2015), 757-786.

[42] Crossley, S. A., Kyle, K., and McNamara, D. S. 2016. The tool for the automatic analysis of text cohesion (TAACO): Automatic assessment of local, global, and text cohesion. *Behav. Res. Methods*. 48, 4 (Dec. 2016), 1227-1237.

[43] Jarvis, S. and Crossley, S. (Eds.). 2012. *Approaching language transfer through text classification: Explorations in the detection-based approach*. Bristol, UK: Multilingual Matters.

[44] Chen, C., Liaw, A., and Breiman, L. 2004. *Using random forest to learn imbalanced data*. University of California, Berkeley, 110.

[45] Sosa, T., Hall, A. H., Goldman, S. R., and Lee, C. D. 2016. Developing symbolic interpretation through literary argumentation. *J. Learn. Sci.* 25, 1 (Dec. 2015), 93-132.

Predicting Student Retention from Behavior in an Online Orientation Course

Shimin Kai[1], Juan Miguel L. Andres[2], Luc Paquette[3], Ryan S. Baker[2], Kati Molnar[4], Harriet Watkins[4], Michael Moore[4]

[1] Teachers College, Columbia University, 525 W125th Street, New York, NY 10027

[2] Graduate School of Education, University of Pennsylvania, 3700 Walnut St., Philadelphia, PA 19104

[3] College of Education, University of Illinois Urbana-Champaign, 1310 S. 6th St. Champaign, IL 61820

[4] University of Arkansas System *e*Versity, 2402 N. University Ave., Little Rock, AR 72207

smk2184@tc.columbia.edu, {andresju, rybaker}@gse.upenn.edu, lpaq@illinois.edu, {kmolnar, hwatkins}@eversity.uasys.edu, mmoore@uasys.edu

ABSTRACT

As higher education institutions develop fully online course programs to provide better access for the non-traditional learner, there is increasing interest in identifying students who may be at risk of attrition and poor performance in these online course programs. In our study, we investigate the effectiveness of an online orientation course in improving student retention in an online college program. Using student activity data from the orientation course, Engage, we make use of machine learning methods to develop prediction models of whether students will be retained and continue to register for program-specific courses in the *e*Versity program. We then discuss the implications of our findings on improvements that may be made to the existing orientation course to improve student retention in the program.

Keywords

Prediction modeling, online orientation course, student retention

1. INTRODUCTION

With the widespread development of online learning programs in institutes of higher learning, access to a college education has improved by a considerable amount. Despite increased enrollment rates within these online degree programs, however, student attrition or dropout rates also tend to be correspondingly higher than in traditional face-to-face degree programs [4, 21]. Dropout can occur early for many students in online programs; some students drop out even before they register for their first course [24]. As such, it has become increasingly important for facilitators and administrators to identify factors that may influence attrition and retention in these online course offerings, and implement targeted interventions to increase retention.

Some of these targeted interventions involve the use of machine learning to provide timely information on student progress within a course to teachers and facilitators [1, 12, 17]. These interventions allow them to identify at-risk students earlier on within an online course, and take steps to encourage student retention. Another type of intervention involves the development of online orientation courses taken before the beginning of the program. These courses aim to provide students with the support and resources they may need during their progression through the program [3, 8]. A combination of the above interventions may also be implemented where machine learning models are developed to identify patterns in student behavior within online orientation courses themselves, which could help inform teachers and facilitators of students at risk of dropout even earlier on within an online program.

In this study, we use machine learning to investigate student behavior within a required online orientation course, Engage, for students registered in an online university, *e*Versity. *e*Versity is a completely online course program established and developed by the University of Arkansas System (UAS). Using student data in this online orientation course, we developed a model that allows us to predict the likelihood of their continued participation in the online college program, through their registration in future program-specific courses.

2. LITERATURE REVIEW

There has been extensive research in recent years to identify factors that lead to low student retention rates, particularly within the context of online learning programs [9, 16, 25]. Attrition and retention can be defined in several ways. Since this paper is focused on an online course program that emphasizes learning at students' own pace and preferred time(s), we make use of the definition proposed by Pascarella and Terenzini [22] (p.374), where retention is defined as progressive re-enrollment, whether continuous from one term to the next, or temporarily interrupted and then resumed, until completion with a degree.

Several researchers have found that student dropout rates in online courses are due to a variety of circumstances, including personal, job, or technology-related reasons [25], and are typically independent of demographic factors such as gender and race [2, 11, 25]. Park et al. [20] also found that organizational support and course relevance are better predictors than demographic variables, and significantly predict student persistence as well as student dropout in online course programs. Both O'Brien & Renner [18] and Jung et al. [14] replicated these findings and found that online courses that increase opportunities for student interaction, such as group

work, tend to improve student engagement, thereby reducing student dropout.

A popular intervention that has been implemented to improve student retention, based on these findings, is the development of orientation courses that seek to provide new students with organizational support, guidance, and resources that they may need to support their online learning. Studies have found that such online orientation courses can be effective at improving retention and the overall student learning experience [5, 8, 13].

Other interventions have focused on providing information to instructors, academic advisors, and facilitators on which students are at risk, so that the student can be contacted and better supported [1, 12, 17]. Increasingly, these types of interventions have been driven using automated models that can identify students who are at risk of dropping out or performing poorly, so that instructors and facilitators can focus intervention efforts on the students who are most likely to be benefit from an intervention. The use of data mining techniques has enabled course facilitators to identify at-risk students early on within a course. For instance, Dekker and colleagues [7] made use of data mining techniques to identify students at risk of dropping out from an electrical engineering program, after the first semester of their studies, or even before they enter the program. In another study, Lauria et al. [15] developed models to predict student performance based on course management system data as well as student academic records.

Such models have then been used by higher education institutions to provide support through early interventions to at-risk students. This type of intervention has been developed and implemented by various universities and companies, including Purdue University, Marist College, Civitas Learning, and ZogoTech [1, 10, 12, 17]. Arnold & Pistilli's work [1], for example, examines the development and implementation of Course Signals at Purdue University. Course Signals makes use of learning analytics to help course faculty provide accurate real-time feedback to their students about whether they are on track to succeed in their current course. Analyses of student performance showed that students who participated in at least one Course Signals course achieved better grades and experience higher retention rates than their peers who did not participate in any Course Signals courses. Similarly, Fritz [10] makes use of learning analytics to develop an intervention called "Check My Activities", where students are given the opportunity to compare their online course activity against an anonymous summary of their peers in the course, thus providing early system feedback directly to the students so that they are more aware of their own levels of engagement within a course.

3. EVERSITY – ONLINE LEARNING

The *e*Versity is a fully online institution for the University of Arkansas System, which is comprised of institutions of higher education across the state. The mission of *e*Versity is to provide online education specifically for adult learners; in particular, at-risk learners who may have previously dropped out of college and may require additional support to be successful academically. Currently the *e*Versity student population is 65% female, 69% white, 27% black or African American, and the average age is 36. Each academic term runs for a short 6 weeks to allow enrolled students maximum flexibility in fitting the online courses within their schedules.

To better serve students, *e*Versity offers a free credit-barring orientation course, Engage. This course fulfills two functions, both related to the goal of improving student retention: to introduce students to the tools and information they need to be successful in an online learning environment, and for the institution to get to know its students. Engage also aims to provide resources and guidance to new students as they continue on to register in program-specific online courses within *e*Versity. Upon enrollment in the *e*Versity program during any of the seven terms throughout the year, students are automatically registered in Engage. Within Engage, information is organized into 6 Steps: Welcome, Getting to Know You, Funding My Future, Supporting My Academic Success, Developing My Learning Plan, and My Financial Plan. Students are free to explore the six course sections at their own pace within the six-week academic term.

To ensure student participation within each section of the course, students are required to complete knowledge checks and assessments at the end of each Step before they can access the next Step. These assessments and checkpoints help students to process the information provided within each Step, and provide students with practice opportunities to complete work in online formats that will be commonly used within later program-specific courses, such as uploading assignments and journal entries, and taking online quizzes. Completion of the Engage course is required for students who wish to continue on to register for program-specific courses on *e*Versity.

4. METHODS

4.1 Orientation Course Data

The dataset used for analysis was obtained from the Blackboard online learning system, and included student data from the first rollouts of the Engage course in the October 2015 and January 2016 terms. As discussed above, each term spans approximately six weeks. The data set provided resource access information per student, including date accessed and page accessed, as well as actions performed while on these pages. Resources accessed and respective actions include:

1. Journals: add journal entry, view draft, edit journal entry
2. Assessments: launch assessment, review attempt, save attempt, submit assessment
3. Assignments: upload assignment
4. Discussion Boards: discussion entry, discussion reply
5. Messages: view messages, email instructor, email select students
6. Gradebook: check grade

We also obtained demographic data consisting of each student's age, gender, race, whether or not their parents attended college, and whether or not they registered for a class in any of the three academic terms immediately following the completion of the Engage orientation course. Of the cohort, a total of 151 students registered for courses after completing the Engage orientation course.

We then built a prediction model to identify which student features are more strongly associated with future registration in for-credit courses on *e*Versity.

4.2 Data Cleaning and Feature Generation

The data set obtained from *e*Versity included resource access data, and demographic and enrollment data. It represented 97,298 page accesses and actions across 325 students.

During their use of Engage, these students interacted with course content (i.e., video lectures), journals, assessments in the form of online quizzes, assignments, discussion boards, messages, and the gradebook. Each transaction within the access log contained a user ID, date stamp (with no time data available), page accessed, and, where relevant, the action performed.

The features investigated in this study included:

1. Total counts – total number of times student accessed each resource regardless of what action they performed (e.g., total count for journal access is the sum of the total count of journal access to write a new post and the total count of journal access to edit an existing post)
2. Days till first access – number of days since start of interaction until a student accessed any of the resources and performed each of their specific actions
3. Days between – average number of days between specific resources accesses and actions performed (e.g., average number of days between two journal views, average number of days between creation of a journal post and editing or submitting the same journal post)
4. Inactivity – average number of days inactive (i.e., number of days between any two transactions)
5. Descriptive statistics – average, standard deviation, minimum, and maximum values per resource access across days the student interacted with Engage

In calculating these features, we excluded behaviors that were required to complete the Engage course. Completing the Engage course was required in order for a student to continue on to register for a program-specific course, so any feature required to complete Engage would be tautologically connected to registering for a program-specific course. Specifically, we excluded student activity around completing assessments, uploading assignments, and adding journal entries. We thus removed these features in order to identify other student actions that may be related to future student registration in an *e*Versity course, but are not explicitly required for the student to register in an *e*Versity course.

4.3 Prediction Modeling

Prediction models of student activity were created using RapidMiner 5.3 in order to determine which combination best predicts whether a student will register in a program-specific course after completing Engage. We attempted to predict this variable using J-Rip classification and J-48 decision trees, with 10-fold student-level cross-validation. Cross-validation splits the data points into *N* equal-size groups. In the case of the current study, data points were split into 10 groups. It then trains on all groups but one, and tests on the last group, and does so for each possible combination.

J-48 decision trees, the RapidMiner Weka Expansion Pack implementation of the C4.5 algorithm, can handle both numerical and categorical predictor variables. The algorithm repeatedly looks for the feature which best splits the data in terms of predictive power for each variable. It later prunes out branches that turn out to have low predictive power. Different branches can have different sets of features. In cases where numerical predictors are used, the algorithm tries to find the optimal split. J-Rip is the RapidMiner Weka Expansion Pack implementation of the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [6], a propositional rule learner. J-Rip produces a set of rules, through stages of growing and pruning, that account for all classes and minimizes error.

Model variable selection was conducted using forward selection, where the feature that most increases fit is added to the current model, until no additional features improve the model. The resultant models' performance was assessed using Cohen's Kappa and AUC ROC. Kappa indicates the degree to which the detector is better than chance at identifying a modeled construct. 0 means that the model is no better than chance, and 1 means perfect performance. AUC ROC is the area under the ROC curve, and is also the probability that given 1 instance of 'registered' and 1 instance of 'not registered', the model is able to tell which instance is which. It is computed using the A' implementation to control for artificially high AUC ROC estimates due to having multiple data points with the same confidence. An AUC ROC value of 0.5 indicates chance level of performance, while a value of 1 means perfect accuracy.

4.4 Demographic Cross-Validation

Some prior research has shown that prediction models may have different levels of accuracy for different subgroups within the data set [19]. To determine whether this was a concern, we evaluated the performance of the models across different demographic groups in our data set. After the models had been developed and cross-validated, we took the model's prediction on the test sets and evaluated their performance on sub sets of the data based on the different demographic groups in our sample. In particular, we compared the performance of the model by gender (male versus female), race (white versus African-American) and parents' college education (parents attended college versus parents did not attend college). In addition to the majority of white and African-American students analyzed, 7 students were Native American. This number of students was insufficient to allow for a valid calculation. We then calculated performance metrics for each of these demographic groups.

5. RESULTS

5.1 Model and Performance

Prediction models created using the W-J48 and W-JRip classification algorithms resulted in high kappa and AUC values. Both algorithms used resulted in comparably high performance. As such, we will discuss both of these models below. The full set of models run and their respective performance values can be found in Table 1.

Table 1. Cross-validated performance of models of student enrollment with different classification algorithms

Classifier	Kappa	AUC
J-48	0.806	0.925
J-Rip	0.825	0.913

5.1.1 J-48 Model

With the J-48 model, a total of four features were selected in some folds of the cross-validation, but not all of them were selected in the final model fit on all data:

- number of days before grades were first checked by the student,
- minimum number of times grades were checked by the student,
- total number of views of online messages within the course platform, and
- total number of views of the Discussion Board Reply page.

The four features initially selected in some of the cross-validation folds indicate that students who checked their course grades earlier and more frequently, responded more to discussion board posts, and viewed in-course messages more frequently were more likely to register in a program-specific *e*Versity course after completing Engage.

The final decision tree generated using this algorithm contained 3 leaf nodes and 2 decision nodes. The decision tree generated by the prediction model is shown in Figure 1.

As can be seen in the figure, only 2 of the selected features had strong enough associations with future course registration to be included in the pruned decision tree built on all data: Number of views of the Discussion Board Reply page, and the number of days till the first time the student checks their course grades.

The decision tree generated with the J-48 model, shown in Figure 1, provides an indication of how each student's future course registration is predicted, and the confidence level assessed for each student's prediction.

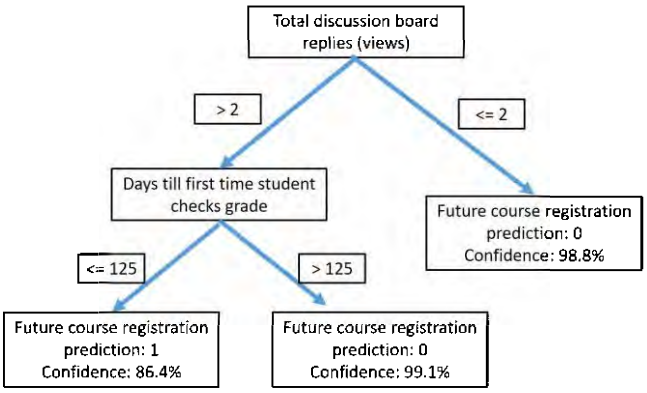


Figure 1: Visual representation of the decision tree generated by the J-48 algorithm

The decision tree in Figure 1 shows that a student who has made fewer attempts to respond in the discussion board is less likely to register in a program-specific course in the future, with a confidence of 98.8%. Similarly, we can see that students who checked their course grades earlier on during the term were more likely to register for a program-specific course afterwards, with a confidence of 86.4%. In contrast, students who only viewed their course grades much later after the start of the orientation course or not at all had a 99.1% confidence of not registering for another *e*Versity course in the future.

5.1.2 J-Rip Model

In the J-Rip model, on the other hand, only one feature was selected: the total number of views of the Discussion Board Reply page. Based on the J-Rip model classification rules, students who viewed the Discussion Board Reply page more often (>= 3 times) within the duration of the orientation course had a higher probability of registering in an *e*Versity course afterwards, with a confidence of 82.4%. In contrast, students who viewed the Discussion Board Reply page 3 times or fewer during the course had a lower likelihood of registering in another course later on, with a confidence of 98.8%.

The J-48 and J-Rip models obtained comparable performance metrics, with the J-48 model having a marginally higher AUC value than the J-Rip model, and the J-Rip model having a slightly higher Kappa value than its J-48 counterpart. This implies that the J-Rip model had a higher proportion of correct predictions when thresholded, but because only one classification rule was selected, there were only 2 confidence values that were associated with these predictions, hence resulting in a lower AUC value. In contrast, more features were selected in the J-48 model (and more differentiations were made), which could explain the slightly higher AUC value for that model than the J-Rip model.

5.2 Performance for Demographic Groups

We then tested both the cross-validated predictions models by three sets of demographic comparisons: gender (male .vs. female), race (white .vs. African-American) and whether the student's parents attended college or not. For the J-48 model, we found that it performed relatively well across all the demographic groups tested, and close to the performance values obtained in the overall model. The model performances of the various demographic groups are listed in Table 2 below. Our J-48 model performed at similar levels for most of the demographic groups that were tested. However, it performed marginally worse for African-American students (Kappa = 0.728, AUC = 0.905). When compared to the model's performance on the full data set (Kappa = 0.806, AUC = 0.925), its performance was still quite good in absolute terms even for this group.

Table 2. Performance of J-48 models of student enrollment for different demographic groups

Group	Kappa	AUC
Female	0.833	0.894
Male	0.753	0.946
African-American	0.728	0.905
White	0.826	0.932
Parents attended college	0.763	0.908
Parents did not attend college	0.829	0.933

Similarly, we found that our J-Rip model performed at comparable levels of performance across different demographics when compared to performance on the full data set. As with the J-48 model, the J-Rip model was least accurate for African-

American students, but still obtained good predictions, with Kappa = 0.748, AUC = 0.907.

Table 3. Performance of J-Rip models of student enrollment for different demographic groups

Group	Kappa	AUC
Female	0.833	0.937
Male	0.811	0.875
African-American	0.748	0.907
White	0.751	0.906
Parents attended college	0.774	0.896
Parents did not attend college	0.854	0.921

These findings suggest that the models obtained here are reliable across demographic groups, indicating that they can be used without concern regarding equity in their predictions.

6. DISCUSSION

To increase access to higher education for non-traditional students, institutions of higher learning have increasingly embraced online learning platforms to provide greater flexibility for working adults looking to return to school. Despite easier access, student retention and attrition has remained an important issue that online orientation courses like Engage aim to address.

In our study on students taking the orientation course Engage, we generated a total of 139 features based on student actions within the Blackboard course platform and developed models to predict future student registration in a program-specific for-credit course within the state of Arkansas's online *e*Versity. The features selected by our model were able to predict with high confidence levels the likelihood that students would register in a program-specific course after the orientation course. It is also notable that both the J-48 and J-Rip models selected the same feature (total number of views of Discussion Board Reply page) to be positively associated with future course registration. This finding echoes and provides support for earlier research suggesting that student participation in discussion boards is associated with better retention and achievement [18, 23].

The features selected in both our models, while not surprising, provide important implications that help guide administrators and facilitators to design interventions that can better identify at-risk students who may not continue on after the orientation course. For instance, the feature of discussion board reply views appeared to have a very strong association with future registration in an *e*Versity course. According to previous research, students' interactions within a course help improve student retention rates [14, 23]. Students who accessed the Discussion Board Reply page more often are more likely to be interacting with other students and course facilitators. In this manner, these students may experience greater engagement in the course and the *e*Versity program, which in turn could explain the association between the students' usage of the discussion board and future course registration within *e*Versity.

Within the J-48 model, three other features were selected in addition to discussion board reply views. The total number of views of the Messages page was also included in some models during cross-validation, even though it was not included in the final decision tree built on the entire data set. Like the Discussion Board Reply page views feature, this feature suggests that students who have more interactions with other students and course facilitators are more likely to register in another *e*Versity course afterwards.

Features on the number of days and frequency of the student checking of course grades appear to have positive associations with future course registration as well. From the decision tree generated with the J-48 algorithm, students that only view their course grades after a long period of time have a high likelihood of not registering for another *e*Versity course in the future. This can be another useful indicator of students who may not be as engaged in the *e*Versity program and their achievement in the orientation course, and who have a lower likelihood of registering for another *e*Versity course.

After developing our models, we tested their reliability across different demographic groups. We found that the models performed equally well across students of different race and gender, as well as between groups of students with parents who attended or did not attend college. These findings suggest that our model is not overtly biased towards or against a specific demographic group.

Based on our models' performance and the features selected, course administrators and facilitators could make further improvements to Engage to increase student retention in the online *e*Versity program. Since some of the selected features involve student interactions, course facilitators could try to embed more interactive activities within Engage to encourage students to reach out to their peers as well as to the program facilitators, and participate more actively in *e*Versity's social community. Given that discussion board views had high predictive power for future course registration within *e*Versity, Engage course facilitators could encourage student participation in discussion boards early on in the course, and maintain a stronger presence within discussion boards to provide a more robust and consistent form of support for students embarking on the *e*Versity program. Nevertheless, it is worth noting that student participation in discussion boards may also be a proxy for student interest in the course content or their overall goal of studying within *e*Versity. Actions taken by course facilitators to encourage student participation in discussion boards may not be as helpful in increasing student engagement or interest in the course content. Alternatively, it may be more effective for course facilitators to tweak the discussion board activities to ensure that they are optimally interesting and relevant to the learners participating in the orientation course.

7. CONCLUSION

In this study, we made use of student interaction data from a credit-baring online orientation course, Engage, in a completely online university, to build a prediction model of student registration in future program-specific courses. The prediction models were developed using machine learning algorithms and tested across different demographic groups. Two algorithms were tested; the performance of both models was high, and the models provide indicators that predict future student registration in program-specific courses within the online *e*Versity program. These prediction models thus provide *e*Versity administrators and course facilitators with fine-grained information on student behavior within the orientation course that could improve student retention on *e*Versity. As such, further improvements

could be made to the orientation course Engage to accurately target students at risk of dropping out of the online *e*Versity program, and provide further support to these students at an earlier stage in their higher education journey.

8. ACKNOWLEDGEMENTS

We would like to thank the Bill and Melinda Gates Foundation and the DLRN network for their support for our work. In addition, we would like to thank George Siemens, Candace Thille, Carolyn Rose, Carol Lashman, and Anita Crawley, for their helpful suggestions.

9. REFERENCES

[1] Arnold, K.E. et al. 2012. Course signals at Purdue: Using learning analytics to increase student success. *2nd International Conference on Learning Analytics and Knowledge*. May (2012), 2–5.

[2] Boston, W.E. et al. 2011. Comprehensive Assessment of Student Retention in Online Learning Environments. *School of Arts and Humanities, APUS*. Paper 1 (2011).

[3] Brewer, S. a. and Yucedag-Ozcan, A. 2012. Educational persistence: Self-efficacy and topics in a college orientation course. *Journal of College Student Retention: Research, Theory and Practice*. 14, 4 (2012), 451–465.

[4] Carr, S. 2000. As distance education comes of age, the challenge is keeping the students. *Chronicle of Higher Education*. 46, 23 (2000).

[5] Carruth, A. K.; Broussard, P. C.; Waldmeier, V. P.; Gauthier, D. M.; Mixon, G. 2014. Graduate Nursing Online Orientation Course: Transitioning for Success. *Journal of Nursing Education*. 49, March (2014), 14–17.

[6] Cohen, W.W. 1995. Fast effective rule induction. *Twelfth International Conference on Machine Learning*. (1995), 115–123.

[7] Dekker, G.W.. et al. 2009. Predicting students drop out: A case study. *EDM'09 - Educational Data Mining 2009: 2nd International Conference on Educational Data Mining*. (2009), 41–50.

[8] Derby, D.C. and Smith, T. 2004. An orientation course and community college retention. *Community College Journal of Research and Practice*. 28, 9 (2004), 763–773.

[9] Fike, D.S. and Fike, R. 2008. Predictors of First-Year Student Retention in the Community College. *Community College Review*. 36, 2 (2008), 68–88.

[10] Fritz, J. 2011. Classroom walls that talk: Using online course activity data of successful students to raise self-awareness of underperforming peers. *Internet and Higher Education*. 14, 2 (2011), 89–97.

[11] Hoskins, S.L. and Hooff, J.C. Van 2005. Motivation and ability: Which students use online learning and what influence does it have on their achievement? *Communications*. 36, 2 (2005).

[12] Jayaprakash, S.M. et al. 2014. Early alert of academically at-risk students: An open source analytics initiative. *Journal of Learning Analytics*. 1, 1 (2014), 6–47.

[13] Jones, K.R. 2013. Developing and implementing a mandatory online student orientation. *Journal of Asynchronous Learning Networks*. 17, 1 (2013), 43–45.

[14] Jung, I. et al. 2010. Effects of different types of interaction on learning achievement, satisfaction and participation in web-based instruction. *Innovations in Education and Teaching International*. 39, 2 (2010), 153–162.

[15] Lauría, E.J.M. et al. 2012. Mining academic data to improve college student retention : An open source perspective. *Proceedings of the Second International Conference on Learning Analytics And Knowledge - LAK '12*. May (2012), 139–142.

[16] Lee, Y. and Choi, J. 2011. A review of online course dropout research: Implications for practice and future research. *Educational Technology Research and Development*. 59, 5 (2011), 593–618.

[17] Milliron, M.D. et al. 2014. Insight and action analytics: Three case studies to consider. *Research and Practice in Assessment*. 9, (2014), 70–89.

[18] O'Brien, B. and Renner, A.L. 2002. Online student retention: Can it be done? *World Conference on Educational Multimedia, Hypermedia and Telecommunications* (2002).

[19] Ocumpaugh, J. et al. 2014. Population validity for Educational Data Mining models: A case study in affect detection. *British Journal of Educational Technology*. 45, 3 (2014), 487–501.

[20] Park, J.-H. and Choi, H.J. 2009. Factors Influencing Adult Learners´Decision to Drop Out or Persist in Online Learning. *Educational Technology & Society*. 12, 4 (2009), 207–217.

[21] Parker, A. 1999. A study of variables that predict dropout from distance education. *International Journal of Educational Technology*. 1, 2 (1999), 1–10.

[22] Pascarella, E.T. and Terenzini, P.T. 2005. How college affects students: A third decade of research. *How College Affects Students: A Third Decade of Research*.

[23] Roberts, J. and Styron, R. 2010. Student satisfaction and persistence: factors vital to student retention. *Research in Higher Education Journal*. 6, 3 (2010), 1–18.

[24] Tyler-Smith, K. 2006. Early Attrition among first time eLearners: A review of factors that contribute to drop-out, withdrawal and non-completion rates of adult learners undertaking eLearning programmes. *Journal of Online Learning and Teaching*. 2, 2 (2006), 73–85.

[25] Willging, P.A. and Johnson, S.D. 2009. Factors that influence students' decision to dropout of online courses. *Journal of Asynchronous Learning Network*. 13, 3 (2009), 115–127.

Inferring Frequently Asked Questions from Student Question Answering Forums

Renuka Sindhgatta
IBM Research - India
Bangalore, India
renuka.sr@in.ibm.com

Smit Marvaniya
IBM Research - India
Bangalore, India
smarvani@in.ibm.com

Tejas I Dhamecha
IBM Research - India
Bangalore, India
tidhamecha@in.ibm.com

Bikram Sengupta
IBM Research - India
Bangalore, India
bsengupt@in.ibm.com

ABSTRACT

Question answering forums in online learning environments provide a valuable opportunity to gain insights as to what students are asking. Understanding frequently asked questions and topics on which questions are asked can help instructors in focusing on specific areas in the course content and correct students' confusions or misconceptions. An underlying task in inferring frequently asked questions is to identify similar questions based on their content. In this work, we use hierarchical agglomerative clustering that exploits similarities between words and their distributed representations, reflecting both lexical and semantic similarity of questions. We empirically evaluate our results on real world labeled dataset to demonstrate the effectiveness of the method. In addition, we report the results of inferring frequently asked questions from discussion forums of online learning environment providing lectures to middle school and high school students.

Keywords

frequently asked questions, agglomerative clustering, question similarity, community question answering.

1. INTRODUCTION

Self-paced online learning environments provide valuable learning resources to a large number of students. A primary mechanism of interactions between the students are the discussion forums. These forums enable students to ask questions, answer questions and collaboratively learn. *Question answering forums*, are discussions forums where every thread is a question posted by a student - much like the community question answering (CQA) platforms such as StackOverflow[1], Quora[2]. Over time, a large number of students may post similar questions that could indicate topics susceptible to confusions, misconceptions or course content requiring further explanations. Most question answering forums allow a student or user to search similar questions present in the archives, using information retrieval technique. While searching similar questions is useful for a student, it provides limited view to an instructor on frequently asked questions. A potential way to aid manual identification of common or frequently asked questions, in such forums is to employ clustering, so that semantically related questions are grouped together.

Motivating Example: Table 1 lists examples of sample groups of similar questions posed by middle and high school students on Khan Academy[3]. These groupings or question clusters can help an instructor identify key concerns or confusions among students. The instructor could address confusions by providing additional content on the specific topic. For example, many students are asking questions on the slope of vertical or horizontal line. Having a view of question clusters, can be valuable to the instructor and help in refining course content.

Partition-based clustering methods such as k-means, k-mediods, k-means++ [9] need prior information about the number of clusters required. Providing number of clusters as input can be very hard for the instructors. Hence, in this work we use hierarchical clustering [9] that does not have an input requirement. Dendrograms (a tree of clusters), that capture results of hierarchical clustering, can allow instructors to extract clusters of different granularities without having to re-run the clustering algorithm. Further, most algorithms of hierarchical clustering, provide the flexibility to choose a distance metric that we utilize in this work.

Existing work on processing CQA archives, identify or rank similar questions given a new question [12]. While the problem of estimating relevance of questions to address a new question is a related to estimating similarities between questions to identify clusters, much of the work done to address

[1]www.stackoverflow.com
[2]www.quora.com
[3]www.khanacademy.org

Table 1: Examples of frequently asked questions.

C#	Video Lecture	Student Questions
C1	Graphing a line in slope intercept form	What would the line look like if the slope was a zero?
		What is the slope of a horizontal line?
		what about vertical lines? do they have slope?
		Would a vertical line imply an undefined slope, and would a horizontal line imply a zero slope?
C2	Proof of Limit $sin(x)/x$	Why not use L'hopital's rule?
		can you use l'hopital 's rule to prove this limit ?
		you can also use l'hopital 's rule to turn $sinx/x$ turn into into $cosx/1$
		Can you also prove this limit using L'Hopital's rule?
		Just use l'hopital's rule for that ...$sin(x)/x == cos(x)/1$ and $cos(x)$ for $x- > 0 = 1$

the former problem, uses supervised learning approaches that require labeled datasets for training and building models.

Our Contributions: We address the problem of inferring frequently asked questions (FAQ) by harnessing a distance metric that that uses the similarity of the words in the question using a lexical database (such as WordNet[4]) and the word embedding space representation that depicts contextual similarity of words. We further provide a flexible way of cutting the output of the clustering algorithm, *dendrogram*, allowing the end user to identify clusters of questions. A range, specifying the number of points needed to define a cluster is taken as input. The generated clusters are sorted by the distance metric, thus enabling instructors to filter and identify relevant question clusters.

2. RELATED WORK

In this section we position our work in the context of existing literature along two directions: (1) Analyzing textual content available in student discussion forums, (2) Processing questions in community-based question answering (CQA) systems.

2.1 Student Discussion Forums

There has been a growing body of research on analyzing the textual discussion forum data in Massively Open Online Courses (MOOCs).

A precursor to analyzing questions is determining the utterance of students or classifying the dialog act of the students (such as asking questions, giving feedback or agreeing and disagreeing). Ezen-can et al. [4], apply k-medoids clustering algorithm and qualitatively evaluate the clusters to group dialog acts and topics. In our work, we analyze posts that are categorized as questions. Topic analysis of MOOC discussion content using Structural Topic Model (STM) has been explored by Reich et al. [15]. While topic labels are useful in providing a broad overview of the themes that are attracting student discussions, they do not help the instructor in analyzing finer details of what students are asking or answering. In one of the recent work Thushari et al. [2], present a 'topic-wise organization' of discussion posts by using Latent Dirichlet allocation (LDA) on the discussion data. The authors present a topic visualization dashboard that would assist MOOCs staff in understanding emergent discussion themes or identifying popular topics [1]. Our work uses questions in the student question answering forums and evaluates the semantic similarity between pairs of questions to identify similar question clusters. The work presented here can be used on the subset of discussion posts that have been tagged or organized into a topic.

In addition, discussion forum data has been utilized for a wide variety of purposes, recent among these is the analysis of information seeking behavior of students (that includes querying, refining the query, reading and browsing), while they learn programming [8]. Sentiment analysis in discussion forums [18], examining relationship between students' discussion behaviors and their learning [17] [6], explore various possibilities of using the forum as a rich source of data.

2.2 Community Question Answering (CQA)

The popularity of CQA indicates that users find them useful in finding answers to their questions. However, there are several issues related to CQA that has led to a large body of research: 1) Identifying good and relevant answers to questions can help users filter noise in the responses. 2) Identifying questions that may be repeated or closely related to previously asked questions can help eliminate redundancy. The latter issue, relates very closely to the problem we address in our work.

One of the recent tasks in SemEval 2016 [12] dealt with identifying and ranking a set of 10 related questions given a new question. The participating teams in the task, built supervised machine learning models that used distributed representation of words, knowledge graphs to define lexical and semantic features [5], neural network approaches including convolution neural nets (CNN) or Long short term memory (LSTM) networks [11], [16], [13]. The focus of their work is to rank the questions in a relevant manner considering semantic similarity. A prerequisite to using these approaches in practice, is the need of a labeled dataset. In our work, we use an unsupervised method that circumvents the need for labeled data.

Clustering questions answers (QA) from the CQA systems to ease tasks such as tagging has been less explored. In one of the recent works [14], the authors identify clusters of related QA. The approach is based on classical k-means clustering algorithm, but mixes the similarities of the questions and answers to define an objective function that is optimized over

[4]https://wordnet.princeton.edu/

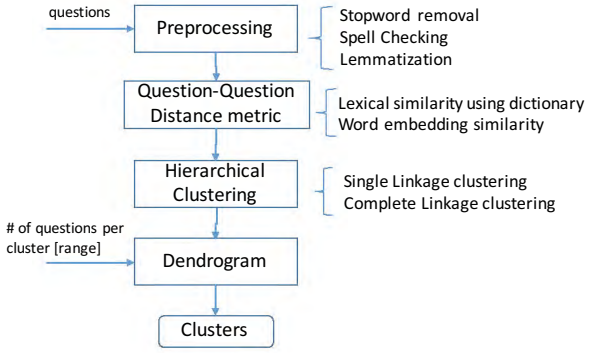


Figure 1: Identifying commonly asked questions.

multiple iterations. While our goal is to cluster questions and use an unsupervised model, we do not rely on the answer information, primarily because the answers given by peers students may contain irrelevant information, especially with students from middle school.

3. IDENTIFYING COMMON QUESTIONS

Our method to infer or identify commonly asked questions is organized into multiple steps, as shown in Figure 1. The first step deals with preprocessing the question to remove any noise. Next, we focus on the key aspect of any clustering algorithm; the choice of (dis)similarity function or distance metric between a question pair. The hierarchical clustering algorithm uses the distance metric to derive the output as a dendrogram. Finally, the dendrogram is partitioned and the clusters are identified.

3.1 Preprocessing

In the preprocessing phase, for each question we filter all URL, email addresses or other similar such patterns which may be irrelevant in the context of the data being analyzed. The misspellings are corrected using the WordNet database. Stopwords are removed and the remaining words in each question are lemmatized to their base forms using the lemmatizer provided by Stanford Core NLP parser[5]

3.2 Question-Question Distance Metric

The distance function uses the combination of both the lexical and word embedding similarity. We define the distance metric between question pairs q_i, q_j as follows:

$$dist(q_i, q_j) = ((\Omega \cdot D_{bow}(q_i, q_j))^x + ((1-\Omega) \cdot D_{vec}(q_i, q_j)^x)^{1/x} \quad (1)$$

where, $D_{bow}(q_i, q_j)$ is the distance computed based on the lexical similarity and D_{vec} is the distance computed based on word embeddings for question pair (q_i, q_j). The following section describes the distance metrics in detail. The distance function Ω is the weight associated with lexical or word embedding based distance. As stated by the authors in [14], the metric represented as $(a^x + b^x)^{1/x}$ approximates to $max\{a, b\}$ for high positive values of x and to $min\{a, b\}$ for high negative values of x.

[5]http://stanfordnlp.github.io/CoreNLP/

3.2.1 Lexical Similarity

Each question is represented as a bag of words vector. The dimension of the vector being the vocabulary size of the question corpus W. Each word w_i in the question and its associated synonyms are identified from the WordNet lexical database. The words are weighted by their *idf* measure. The *idf* measure is given by

$$idf(w_i) = log\left(\frac{|D|}{df(w_i)}\right) \quad (2)$$

where, D is the corpus size and $df(w_i)$ is the number of documents containing w_i. Similarity between two question $Sim_{bow}(q_i, q_j)$ is computed using the cosine similarity of the question vectors. The distance is defined as:

$$D_{bow}(q_i, q_j) \equiv 1 - Sim_{bow}(q_i, q_j) \quad (3)$$

3.2.2 Word Embedding Similarity

Each question is represented as a weighted combination of embeddings of words in the question. The word vector v_w for each word w in the question is identified using the distributed representation of words generated by the word2vec tool [10]. Each question q is represented as:

$$V_q = \frac{1}{|q|} \sum_{w \in q} log(\frac{|D|}{df(w)}) \cdot v_w \quad (4)$$

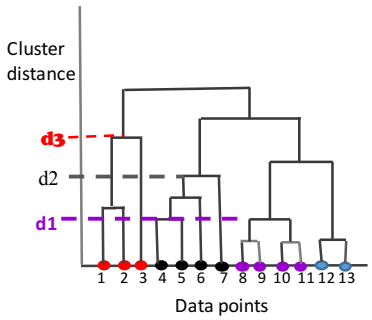
Similarity between two question $Sim_{vec}(q_i, q_j)$ is computed using the cosine similarity of the question vectors. The distance between question pairs q_i, q_j is defined as:

$$D_{vec}(q_i, q_j) \equiv 1 - Sim_{vec}(q_i, q_j) \quad (5)$$

3.3 Hierarchical Clustering

We use agglomerative hierarchical clustering. Initially, each question is in its own cluster. The nearest clusters are merged until there is only one cluster left. The end result is a cluster tree or dendrogram. The tree can be cut at any level to produce different clusters. There are two types of clustering methods. The *Single Linkage* approach, merges two clusters by considering the minimum distance between the points in clusters to be merged. In *Complete Linkage* approach, two clusters are merged by considering the maximum distance between the points in the clusters. Complete linkage clustering results in more compact clusters as the merge criterion considers all points in the cluster. We use complete linkage clustering. The worst case run time complexity of agglomerative clustering is $\mathcal{O}(n^2 \log n)$ which makes it too slow for large datasets. The primary advantage of the clustering approach is that it does not require any prior input to generate the cluster tree.

We evaluated another clustering algorithm Density-based spatial clustering of applications with noise (DBSCAN) [3], which has a worst case run time complexity of $\mathcal{O}(n^2)$. The inputs to the DBSCAN, are the minimum number of points to form a cluster and the distance threshold *eps* such that, for every point in the cluster, there exists another point in the same cluster whose distance is less than the *eps*. Selecting distance threshold as an input can be a challenge. The resulting clusters can vary significantly with *eps*.



(a) Dendrogram

Input cluster size range = [3,4]

Cluster	Rank	Points
C1	d1	{8,9,10,11}
C2	d2	{4,5,6,7}
C3	d3	{1,2,3}

(b) Resulting clusters

Figure 2: (a) Dendrogram (b) Clusters identified for input range of number of points.

3.4 Dendrogram

The output of the hierarchical clustering is a dendrogram as shown in Figure 2(a). A typical approach is to cut the dendrogram at a specific distance and identify the resultant clusters. However, a dendrogram can be cut at different distances based on the domain or application specific information. In our scenario, an important input from the instructor, is the minimum number of points or questions in cluster, for it to be considered as a FAQ. An instructor may decide, that she would like to address groups of at least 4 similar questions, or provide a range of question sizes as input. Figure 2(b) depicts such a scenario of wanting a range of [3, 4] questions in each cluster. We use number of questions as the input and provide a list of question clusters sorted by the cluster distance. Hence, clusters that are linked with lower distance values form good quality clusters. As the distance function increases, the quality of the resulting cluster would be poor.

4. EXPERIMENTAL EVALUATION

In this section, we evaluate our method for identifying FAQ. We use a labeled data set from a CQA archive and create reference clusters.

4.1 Data

To evaluate the suitability of our approach, we use SemEval 2016 Task 3 dataset that contains questions and answers from Qatar Living forum [12]. The data relevant for our evaluation contain questions categorized as *Original question*. For each original question, a set of 10 related questions are annotated as *PerfectMatch*, *Relevant* and *Irrelevant*. Using the labeled information, we build a set of reference clusters or ground truth, which contain the original question and the related questions that are either *PerfectMatch* or *Relevant*. Table 2 contains the details of the data set. The test dataset contained of 770 questions.

Table 2: SemEval 2016 Task3 dataset used.

Questions		Training	Test
Original Questions		200	70
Related Questions	*Total*	1,999	700
	Relevant	606	152
	PerfectMatch	181	81
	Irrelevant	1,212	467
Total		2,199	770

4.2 Evaluation Metrics

The quality of clustering is measured using F-Measure, combining the precision and recall scores used in information retrieval [7]. Each generated cluster C_{gen} is treated as a result of the query and each reference cluster C_{ref} is considered as the desired set of documents or points:

$$precision(C_{gen}, C_{ref}) = \frac{C_{gen} \cap C_{ref}}{C_{gen}} \quad (6)$$

$$recall(C_{gen}, C_{ref}) = \frac{C_{gen} \cap C_{ref}}{C_{ref}} \quad (7)$$

$$F - Measure(C_{gen}, C_{ref}) = \frac{2 \cdot precision \cdot recall}{precision + recall} \quad (8)$$

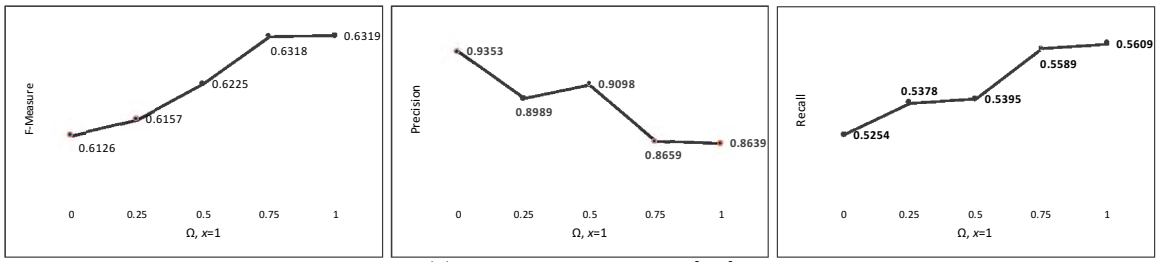
The average precision, recall and F-Measure values are computed for each cluster containing the "original question". For the purpose of evaluation, we use the test data set and identify the partition or the distance threshold at which the maximum average F-Measure is obtained.

4.3 Results

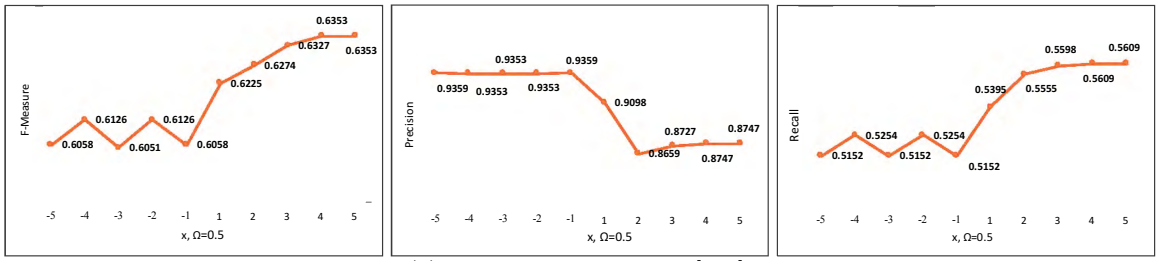
The results of our approach are presented in Figure 3. We evaluate the cluster measures by considering the question-question distance metric using various values of Ω and x. High F-Measure and recall is achieved when we use lexical similarity as the primary distance metric. Using word embedding as a primary similarity metric results in higher precision, which could be suitable in scenarios where the data is noisy or contains large number of irrelevant questions. Figure 3(a) has varying weights associated to lexical and word embedding based similarity. When $x = 0.5$, a balance between high precision and high recall is achieved. Further, Figure 3(b), shows the metrics achieved by varying Ω. Here, the best results are achieved with $\Omega = 4$, with an F-measure of 0.653, a precision of 0.874 and recall of 0.5609. The SemEval 2016 Task 3 participants reported unofficial precision, recall and F-Measure values. Here, for each original question, *Relevant'* and *PerfectMatch* questions are categorized as true pairs and *Irrelevant* questions are categorized as false pairs. The precision values reported by the top 4 participants ranged from 0.636 to 0.763. The recall values were higher and ranged from 0.553 to 0.759. The F-Measure was between 0.64 and 0.71. The results of our method are comparable and encouraging as we have used an unsupervised model.

5. INFERRING FAQ FROM STUDENT QA SYSTEM

In order to verify the relevance of the approach, we ran the clustering tool on a student question answering platform. The dataset for the analysis, was extracted from the Khan Academy, by permission, using screen scapping pro-



(a) $x = 1$, varying Ω from [0,1]



(b) $\Omega = 0.5$, varying x from [-5,5]

Figure 3: F-Measure, Precision and Recall values by varying Ω and x.

Table 3: Sample FAQ inferred using proposed method from Khan Academy question answering forum.

C#	Video Lecture	Student Questions
C1	Graphing a line in slope intercept form	what do the b stand for in the equation y = mx + b ?
		what do the m stand of in y = mx + b ?
		why do we use m and b in the equation y = mx + b ?
		what if the m and the b be zero in the equation y = mx + b ?
C2	Introduction to limits	isn't 0/0 indeterminate not undefined
		Sal said that 0/0 is undefined. Shouldn't it be not a number?
		At 1:18, why is 0 divided by 0 undefined? My teacher taught us it's 0...
		is 0/0 undefined, or one? and Why?
		I thought that 0/0 is called a indeterminant not undefined. Correct my logic please
		WHY is anything divided by 0 considered as undefined??
C3	Definition of function	I'm trying to understand but, I see what he is doing but what ever he is saying is in slow motion so I don't understand. And what is a piecewise function
		Do you have a video where they give you a graph of a piecewise function, but need to find the rule?
		How to find inequalities for piecewise functions?
		How do you graph piecewise functions?
		what is a piecewise function?
C4	Proof of sin x by x	i m a class 9 student and dont have 100% knowledge on trigonometry (just went through his videos once) so i dnt get what i am missing here: should he prove that for 3rd and 2nd quadrant as well?!
		Is this statement is not applicable to 2nd &3rd quadrants ? Why?
		exactly why does this only apply to 1st and 4th quadrant why not, 2nd and 3rd?
		what about the 2nd and 3rd quadrants?
		X would not be negative in the 4th quadrant.,x is only negative in 2nd and 3rd quadrant.
		why is he working in the first and fourth quadrants only? because the absolute value remains the same in all quadrants
		@14:22 Khan says that cos(x) is always the x value in the first and fourth quadrants. Doesn't he mean that cos(x) and x have the same sign in the first and fourth quadrants?
		Why do we consider x only in the first and the fourth quadrant? Does it change the result if we need to consider all the quadrants?
		I feel like I understand everything except going into the fourth quadrant. From 8:32 to the end of the video, he is discussing the fourth quadrant.
		Why go into the fourth quadrant, and why does he stay away from the second and third quadrant?
		why is he working in the first and fourth quadrants only? because the absolute value remains the same in all quadrants

tocol. We considered micro lectures of 8^{th} grade mathematics and micro lectures covering differential calculus. On the learning platform, each micro lecture video has easy access to the page where questions for that lecture, can be asked or viewed. Asking questions is voluntary. Each learner can view questions that have been previously asked by their peers. Once a question is asked, a discussion thread is initiated with peer students providing answers. The data set contains about 22000 questions from 300 video lectures. As questions are asked in the context of a given micro lecture,

we infer the FAQs for each lecture. This helps us reduce the running time of our clustering algorithm.

5.1 Discussion

Table 3 presents a subset of the clusters or FAQs extracted. Four example clusters or FAQ are presented. We were able to extract 4 to 10 questions in each of the sample clusters. We observed several clusters with irrelevant questions, that resulted from poor semantic match when the question content contained numerous mathematical expressions, symbols and less text. Our results can improve with domain specific preprocessing. The current preprocessing step does not parse or process mathematical expressions. Identifying expressions and tagging them as a special tokens for computing question-question distance could provide better results. We noticed several abbreviations in the questions, that were not handled by our preprocessing step. In addition, many students had questions related to content presented at specific time periods in the video lectures. Annotating terms representing video lecture time period, as a part of preprocessing could help ascertain intervals of time within the lectures, where students are seeking more information. Such domain specific processing of content in questions could help improve the question-question distance metric and reduce noise in the generated clusters.

6. CONCLUSION

Our goal in this work was to identify FAQ from the question answering systems of online learning environments. We used agglomerative clustering, an unsupervised learning approach, to identify the FAQ as it did not require any prior inputs to identify groups of questions. A distance metric was defined to harnesses similarity based on bag of words and word embeddings. Our empirical evaluation on labeled dataset shows the effectiveness of our approach, with the precision and F-Measure values comparable to the existing methods that use supervised models. We extracted questions asked by students from Khan Academy and FAQ was extracted for each topic. In future, we would include the answers provided by students in identifying similar questions. The answers can be filtered based on the votes received, student popularity and other related answers in the posts. This would result in improving the quality of extracted FAQ.

7. REFERENCES

[1] T. Atapattu, K. Falkner, and H. Tarmazdi. Topic-wise classification of MOOC discussions: A visual analytics approach. In *International Conference on Educational Data Mining*, 2016.

[2] T. Atapattu and K. E. Falkner. A framework for topic generation and labeling from MOOC discussions. In *ACM Conference on Learning @ Scale*, 2016.

[3] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *International Conference on Knowledge Discovery and Data Mining*, 1996.

[4] A. Ezen-Can, K. E. Boyer, S. Kellogg, and S. Booth. Unsupervised modeling for understanding MOOC discussion forums: a learning analytics approach. In *International Conference on Learning Analytics And Knowledge*, 2015.

[5] M. Franco-Salvador, S. Kar, T. Solorio, and P. Rosso. UH-PRHLT at semeval-2016 task 3: Combining lexical and semantic-based features for community question answering. In *International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, 2016.

[6] J. Huang, A. Dasgupta, A. Ghosh, J. Manning, and M. Sanders. Superposter behavior in mooc forums. In *ACM Conference on Learning @ Scale*, 2014.

[7] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *International Conference on Knowledge Discovery and Data Mining*, 1999.

[8] Y. Lu and S. I. Hsiao. Seeking programming-related information from large scaled discussion forums, help or harm? In *International Conference on Educational Data*, 2016.

[9] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[10] T. Mikolov, W. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013.

[11] M. Mohtarami, Y. Belinkov, W. Hsu, Y. Zhang, T. Lei, K. Bar, S. Cyphers, and J. Glass. SLS at semeval-2016 task 3: Neural-based approaches for ranking in community question answering. In *International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, 2016.

[12] P. Nakov, L. Màrquez, A. Moschitti, W. Magdy, H. Mubarak, A. A. Freihat, J. Glass, and B. Randeree. Semeval-2016 task 3: Community question answering. In *International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, 2016.

[13] H. Nassif, M. Mohtarami, and J. Glass. Learning semantic relatedness in community question answering using neural models. *Association for Computational Linguistics*, page 137, 2016.

[14] D. P. Mixkmeans: Clustering question-answer archives. In *Conference on Empirical Methods in Natural Language Processing*, 2016.

[15] J. Reich, D. Tingley, J. Leder-Luis, M. E. Roberts, and B. M. Stewart. Computer assisted reading and discovery for student generated text in massive open online courses. *Journal of Learning Analytics*, 2015.

[16] S. Romeo, G. D. S. Martino, A. Barrón-Cedeño, A. Moschitti, Y. Belinkov, W. Hsu, Y. Zhang, M. Mohtarami, and J. R. Glass. Neural attention for learning to rank questions in community question answering. In *International Conference on Computational Linguistics*, 2016.

[17] X. Wang, D. Yang, M. Wen, K. R. Koedinger, and C. P. Rosé. Investigating how student's cognitive behavior in MOOC discussion forum affect learning gains. In *International Conference on Educational Data Mining*, 2015.

[18] M. Wen, D. Yang, and C. P. Rosé. Sentiment analysis in MOOC discussion forums: What does it tell us? In *International Conference on Educational Data Mining*, 2014.

On the Prevalence of Multiple-Account Cheating in Massive Open Online Learning

A replication study

Yingying Bao, Guanliang Chen* and Claudia Hauff
Web Information Systems
Delft University of Technology
Delft, the Netherlands
Y.Bao-1@student.tudelft.nl
{guanliang.chen, c.hauff}@tudelft.nl

ABSTRACT

Massive Open Online Courses (MOOCs) are a promising form of online education. However, the occurrence of academic dishonesty has been threatening MOOC certificates' effectiveness as a serious tool for recruiters and employers. Recently, a large-scale study on the log traces from more than one hundred MOOCs created by Harvard and MIT has identified a specific cheating strategy viable in MOOCs: *Copying Answers using Multiple Existences Online* (CAMEO). In essence, learners create several accounts on a MOOC platform, request assessment solutions via some of the accounts, and then submit these "harvested" solutions in their main account to receive credit. In our work, we *replicate* the CAMEO implementation and apply it to ten edX MOOCs created by the Delft University of Technology. Our results show that in those MOOCs, 1.9% of certificates were likely earned through CAMEO cheating, a number comparable to the fraction of cheating observed in Harvard and MIT MOOCs.

Keywords

MOOCs, Academic Dishonesty, Multiple-Account Cheating, Educational Data Mining

1. INTRODUCTION

Cheating is generally defined as using dishonest means to gain an undeserved reward of ability or to get rid of an embarrassing situation [3]. Academic dishonesty is a type of cheating that occurs in relation to an academic exercise. It is a widespread occurrence across different levels and forms of education [4]. There are diverse cheating strategies adopted by students to implement academic dishonesty such as impersonation, bringing notes into the exam hall, using an unauthorized digital device, and so on.

MOOCs, which are courses designed with open access for a large number of online participants, have become a vital part of scalable and large-scale education. However, the effectiveness of MOOCs has been threatened by academic dishonesty. For instance, as early as 2012, some instructors have voiced concerns about various forms of cheating in their MOOCs [7].

One of the main issues in exploring the issue of cheating in MOOCs is the general lack of ground truth data — MOOC providers may be reluctant to confront learners (as a definite proof of cheating is difficult to come by and a time-consuming endeavour) and MOOC learners are reluctant to admit their misbehaviour. Recently, Northcutt et al. [5] proposed a first approach to automatically detect a particular kind of cheating purely based on the log data that is collected in major MOOC platforms; they termed this method CAMEO or *Copying Answers using Multiple Existence Online*. In brief, this method is able to detect learners that cheat in the following way: (1) A learner registers multiple accounts on a MOOC platform and enrolls in a MOOC of interest with all these accounts; one of those registered accounts is the learner's *main* account. (2) The learner uses some of the registered accounts to randomly submit answers to assessment questions (which in MOOCs are often multiple-choice or fill-in-the-blank questions to enable automatic grading) as a way to *harvest* the correct solutions. This is made possible by a design decision of major MOOC platforms which allows learners to check their submitted solutions immediately after submission. (3) The learner then submits the harvested solutions through the main account, allowing the learner to successfully complete the course and earn a certificate. Commonly, achieving 60% (or a similar percentage) of all possible points is sufficient to receive a MOOC certificate.

Among the many potential ways of cheating in MOOCs, CAMEO is of particular concern for a number of reasons: (1) the CAMEO cheating strategy can be performed by every learner individually, it does not require learners to collaborate with others; (2) CAMEO cheating is efficient and easy to execute as it directly utilizes the solutions provided in a MOOC; and (3) CAMEO cheating can be applied across many different MOOCs, largely independent of the subject

*The author's research is supported by the *Extension School* of the Delft University of Technology.

or course level.

Northcutt et al. [5] observed CAMEO cheating in 69 Coursera MOOCs (out of 115 investigated) provided by MIT and Harvard University; among those 69, approximately 1.3% of the certificates were issued to learners identified as CAMEO users. Given that MOOCs provided by different universities usually attract varying sets of learners, in this work, we investigate the following two **R**esearch **Q**uestions:

RQ1 What is the *prevalence* of CAMEO cheating in the MOOCs provided by TU Delft?

RQ2 What are *characteristics* of learners identified to have employed the CAMEO strategy?

To answer these questions, we implement the detection approach as described in [5] and apply it on the log traces of 10 edX MOOCs. We find that 1.9% of the certificates are earned by CAMEO learners (our answer to **RQ1**), with some types of MOOCs more prone to cheating than others. While we did not observe any CAMEO behaviour in a MOOC on political debates, we found more than 6% of certificates to be CAMEO certificates in a business and technical course respectively. With respect to **RQ2**, we observe cheating to be most prevalent mid-course and to be more prevalent in some user demographics than others.

2. RELATED WORK

There are a few works proposed to investigate the prevalence of cheating in MOOCs. Two of the earliest works were proposed by [5] and [6]. Both of these two works focused on the detection of CAMEO cheating based on learnersâĂŹ traces in MOOCs provided by MIT on edX.

In [5], 1.3% of the certificates among 69 MOOCs covering different subjects were earned by learners who adopted CAMEO cheating strategies. Learners who applied CAMEO are more likely to be young, male and international than the other certified learners. In [6], the number is 10.3% of the certificates in an introductory physics MOOC.

In both of these works, researchers set patterns of CAMEO and select learners whose behaviors satisfy the patterns. There are overlaps between the criteria adopted by the two works. Ruiperez-Valiente et al. [6] has relatively more detailed assumptions to CAMEO in different modes. Northcutt et al. [5] was conducted in more than 100 MOOCs, which helps to avoid the accidental bias in the prevalence of CAMEO caused by courses.

Compared to these works, our goal is to investigate the prevalence of this cheating behavior in the MOOCs provided by TU Delft and what the common characteristics are among the detected cheaters.

3. DETECTION METHOD

In this section we recap the main assumptions that underpin Northcutt et al. [5]'s approach. Note that these assumptions are derived from intuitions about MOOC learners' (or more generally online users') behaviours on the learning platform. Our implementation of the approach matches the original paper's algorithmic formulation as closely as possible.

- **CAMEO users hold at least two accounts.** Each CAMEO user (i.e. a learner who cheats to gain an advantage in a MOOC) should use one or more accounts to harvest solutions (so-called Harvest Account(s)) and one main account to submit the correct solutions (i.e., the Master Account) so as to earn the certificate. Initially, every possible pair of user accounts having enrolled in a particular MOOC is a candidate Master/Harvester pair.

- **CAMEO users harvest solutions before entering them into their Master Account.** In other words, for questions that learners cheat on, the candidate Harvester Account should precede the candidate Master Account in time for the gathering of solutions.

- **CAMEO users quickly pass collected solutions from Harvester Accounts to Master Account.** It is reasonable to assume that a cheater may simultaneously log in both the Harvest Account and the Master Account, and once the learner collects the correct solutions, he may immediately submit the correct solutions through the Master Account. This assumption requires the time difference between the correct submission from the candidate Master Account and the request to solutions from the candidate Harvester Account to be small.

- **Master Accounts are certified, the Harvester Accounts are not.** Given that Harvester Accounts are mainly used to gather correct solutions via randomly submitting answers, more often than not, the Harvester Accounts do not reach the passing threshold of a MOOC. At the same time, the Master Accounts should perform well in that respect and earn a certificate.

- **Master Account and Harvester Account are connected via IP addresses.** As noted before, a CAMEO user may simultaneously log into multiple accounts on one and the same or different devices in the same location; thus, it is likely that Master and Harvester account share a common logged IP address during the MOOC.

In the CAMEO approach, these intuitions are transformed into filtering rules (that filter the initially created account pairs) and only candidate Master/Harvester pairs that meet all of these criteria are considered to be CAMEO users, that is, learners who cheat through multiple account usage in a MOOC. Most of these rules contain ad-hoc parameters (e.g. the time limit between a Harvester and Master account submission); we have followed the parameter settings described in [5] in our implementation.

4. EXPERIMENT

4.1 Dataset

Our study is based on the log data generated during 10 edX MOOCs (eight different MOOCs of which two ran twice) which were provided by TU Delft between 2014 and 2016. The MOOCs cover various scientific areas including data science, programming paradigms, biotechnology, business and political science. An overview of the MOOCs, including the number of enrolled learners and the number of certificates earned is shown in Table 1.

Table 1: Overview of the ten MOOCs included in this study. **#Enrollments** shows the number of user accounts that registered for each MOOC and **#Certificates** lists the number of registered participants that achieved a certificate (the passing threshold is 50% for Frame101x and 60% for all other MOOCs). Note that FP101x and EX101x are listed twice, as they both ran in two different time periods.

Course Code	Course Title	Session	#Enrollments	#Certificates
FP101x	Functional Programming	2014 Fall	37,940	1,356
CTB3365DWx	Drinking Water Treatment	2014 Fall	10,458	246
EX101x	Data Analysis	2015 Spring	33,515	2,190
Frame101x	Framing: How Politicians Debate	2015 Spring	34,017	919
Calc001x	Pre-university Calculus	2015 Summer	27,857	358
EX101x	Data Analysis	2015 Fall	21,041	1,156
IB01x	Industrial Biotechnology	2015 Fall	8,143	329
FP101x	Functional Programming	2015 Fall	20,936	1,143
RI101x	Responsible Innovation	2016 Spring	2,741	113
CTB3365sTx	Urban Sewage Treatment	2016 Spring	9,566	361

Table 2: Overview of the detected CAMEO users and the percentage of certificates gained by CAMEO users. The last row shows the numbers across all ten MOOCs.

Course Code	#CAMEO Users	% CAMEO Certificates
FP101x (2014)	13	0.96%
CTB3365DWx	4	1.63%
EX101x (2015S)	27	1.23%
Frame101x	0	0
Calc001x	13	3.63%
EX101x (2015F)	20	1.73%
IB01x	12	3.65%
FP101x (2015)	16	1.40%
RI101x	7	6.19%
CTB3365sTx	25	6.93%
Total	137	1.89%

4.2 CAMEO Detection Results

For each of the MOOCs, we present the number of detected CAMEO users (and subsequently the percentage of certificates gained through CAMEO) in Table 2. CAMEO users are detected in 9 out of the 10 MOOCs and overall account for 137 (or 1.89%) of all certificates. This percentage is slightly higher than Northcutt et al. [5]'s (1.3%). The percentages vary across courses, with *Urban Sewage Treatment* being the MOOC with the largest percentage of CAMEO learners, nearly 7%. On the other hand, our only MOOC without CAMEO cheating detected is *Framing: How Politicians Debate*. In future work we will investigate this variance in CAMEO between courses; we hypothesize that for participants in Frame101x a certificate has less intrinsic value (the self-development aspect is more important) and thus cheating is less likely to occur.

4.3 Verification of CAMEO Users

To explore how plausible the detection results are — i.e., are the detected account pairs actually belonging to the same learner and did the learner indeed cheat — we manually verified key account characteristics. It is sensible for instance to assume that at least some CAMEO users register with the same/similar name across the Harvester and Master Account. Indeed, among our 137 detected CAMEO users, 20% have similar or even same registered full names attached to their Harvester and Master Accounts[1]. To provide the reader with some intuition on the similarities, we now describe for a randomly picked CAMEO user in our dataset the similarities between the detected Master and Harvester Account:

- The Harvester & Master Account have the same registered full name.
- The registered email addresses of the Harvester & Master Account contain a common long character sequence (eight characters).
- The Harvester & Master Account utilize the same IP address to answer every question.
- The Harvester & Master Account submit answers within 60 seconds for every harvested question and the Harvester Account always submits before the Master Account.
- The Harvester Account submits answers for all questions in the course, but the correctness is only 11.5%.

Based on these observations, we are highly confident that the learner is indeed a CAMEO user.

4.4 Characteristics of CAMEO Users

To gain a better understanding of the detected CAMEO users, we analyze their characteristics and patterns. With respect to the nationality of the certified learners, we find them to come mainly from the US, the Netherlands and the UK. However, the detected CAMEO users are mainly from India (27), the US (12) and Germany (7).

We are also interested in the motivation of CAMEO cheaters, i.e., what drives them to cheat in MOOCs. Intuitively, we believe that most CAMEO users to be strongly goal-oriented with the goal being the certificate (instead of the goal being related to knowledge gains). To verify this intuition, we compute how many detected CAMEO users would be

[1]We compute the similarity between two account names according to the Ratcliff/Obershelp sequence match method [1].

Table 3: Overview of the identified CAMEO learners and their certificate status (pass or fail) if the assessments points they gained through CAMEO were removed.

Course Code	Pass w/o CAMEO	Fail w/o CAMEO
FP101x (2014)	2	11
CTB3365DWx	0	4
EX101x (2015S)	3	24
Frame101x	0	0
Calc001x	0	13
EX101x (2015F)	4	16
IB01x	0	12
FP101x (2015)	2	14
RI101x	1	6
CTB3365sTx	0	25
Total	12	125

able to earn a certificate without CAMEO cheating. Specifically, we calculate the grades of CAMEO users on the condition that they only receive credits for questions they did not cheat on and evaluate whether the scores are sufficient to pass the course. As shown in Table 3, nearly 90% of the CAMEO users cannot pass the MOOCs without cheating, which implies that most of the CAMEO users are purely certificate-driven.

We also investigate *when* CAMEO users are most likely to cheat during the course of a MOOC. To this end, we select FP101x (2014 and 2015) and EX101x (2015 Spring and 2015 Fall) for analysis as the grading strategies adopted across the four MOOCs are very similar: almost all questions (more than 100 per course) are worth a single point and the final grade is simply based on the fraction of questions the learner answered correctly (with 60% of correct answers being the passing threshold). Figures 1 (FP101x) and 2 (EX101x) show the number of identified CAMEO users that resort to the CAMEO strategy across the different course weeks.

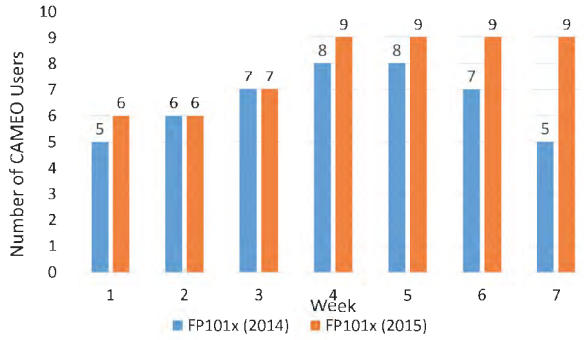


Figure 1: Average Number of CAMEO Cheater Cheating on per Question in Different Weeks in FP101x.

Few learners resort to CAMEO in the first two weeks of the course, while course weeks 3, 4, 5 and 6 attract the most cheating. This is not overly surprising considering the fact that the questions in later weeks are usually more difficult than those in early weeks. The trend of decreased CAMEO in the final week(s) can be explained by the fact that the

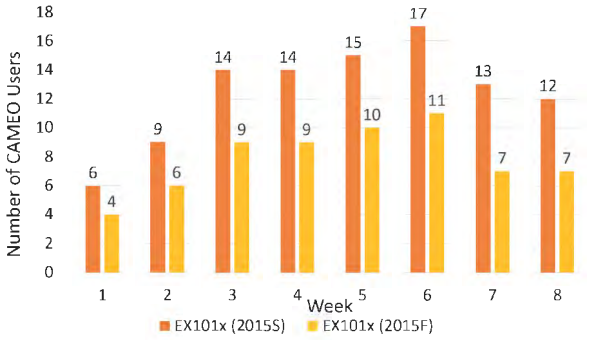


Figure 2: Average Number of CAMEO Cheater Cheating on per Question in Different Weeks in EX101x.

edX platform provides a *Progress* page where each learner can check his progress towards the passing threshold. For a learner whose main goal is the certificate, the realization of that goal (which can occur already as early as week 5 as the passing threshold is 60%) is likely to reduce or stop his CAMEO behaviour.

5. CONCLUSION

We successfully replicated the CAMEO strategy formalized in [5] and applied it to a novel set of MOOCs. Overall, we found similar percentages of CAMEO cheating in TU Delft MOOCs (1.9% vs. 1.3%), albeit with the limitation that we only explored 10 MOOCs (vs. 115 by MIT/Harvard). We are currently enlarging the study to include all 50 MOOCs that are provided by TU Delft. Our future work will place a greater emphasis on the demographic analysis of CAMEO users and on ways to reduce and prevent such cheating — either through technological means or ethical appeals and moral reminders [2].

References

[1] Paul E Black. Ratcliff/obershelp pattern recognition. *Dictionary of Algorithms and Data Structures*, 17, 2004.

[2] Henry Corrigan-Gibbs, Nakull Gupta, Curtis Northcutt, Edward Cutrell, and William Thies. Measuring and maximizing the effectiveness of honor codes in online courses. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 223–228. ACM, 2015.

[3] Melanie Ghoul, Ashleigh S Griffin, and Stuart A West. Toward an evolutionary definition of cheating. *Evolution*, 68(2):318–331, 2014.

[4] Donald L McCabe, Kenneth D Butterfield, and Linda K Trevino. *Cheating in college: Why students do it and what educators can do about it.* JHU Press, 2012.

[5] Curtis G Northcutt, Andrew D Ho, and Isaac L Chuang. Detecting and preventing "multiple-account" cheating in massive open online courses. *Computers & Education*, 100:71–80, 2016.

[6] Jose A Ruiperez-Valiente, Giora Alexandron, Zhongzhou Chen, and David E Pritchard. Using multiple accounts for harvesting solutions in moocs. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 63–70. ACM, 2016.

[7] K Webley. Mooc brigade: Can online courses keep students from cheating? *Time*, 2012. Retrieved May 2017, from nation.time.com/2012/11/19/mooc-brigade-can-online-courses-keep-students-from-cheating/.

Clustering Student Sequential Trajectories Using Dynamic Time Warping

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

One of the most challenging tasks in the field of Educational Data Mining (EDM) is to cluster students directly based on system-student sequential moment-to-moment interactive trajectories. The objective of this study is to build a general temporal clustering framework that captures the distinct characteristics of students' sequential behaviors patterns, that tracks whether a student's learning experience is *unprofitable*, and can identify such an individual as early as possible so personalized learning can be offered. The central idea of our framework is based on Dynamic Time Warping (DTW), which calculates distance between any two temporal sequences even with different lengths. In this paper, we explore both the original DTW and our proposed normalized DTW to generate distance matrix and apply Hierarchical Clustering to the resulted distance matrix. To fully evaluate the power of our temporal sequential clustering framework, we calculate distance matrix at three types of granularity in the increasing order of: problem, level, and session across three training datasets. As expected, results show that clustering moment-to-moment temporal sequences at problem granularity is more effective than level and session granularity. In addition, our proposed normalized DTW is more effective than both original DTW and the baseline Euclidean distance.

Keywords

Clustering, distance matrix, dynamic time warping

1. INTRODUCTION

The impetus for the development of many Intelligent Tutoring Systems (ITSs) was the desire to capture the effective learning experience provided by human one-on-one instruction. ITSs have shown positive impact on learning but the degree of their effectiveness often depends on individual student's motivation, incoming competence, etc. In ITSs, the system-student interactions can be viewed as a sequential action-response process. Each of these interactions will affect the system-student's subsequent interactions. As one of the great promises of ITS is to support personalized learning [15], the system-student moment-to-moment interactive trajectories often have vastly different lengths while most existing clustering approaches including K-means and Hierarchical Clustering are not designed to directly handle such temporal sequential datasets. Therefore, the main objective of this research is to build and evaluate a general clustering framework that captures the distinct characteristics of system-students' sequential interactive behavioral patterns, that tracks whether a student's learning experience is *unprofitable*, and can identify such an individual as early as possible so personalized learning can be offered.

Previously, various clustering methods have been widely applied for different Educational Data Mining (EDM) applications such as temporally coherent clustering [7], collaborative learning [9], reading comprehension [13], handwritten coursework [4], and personalized e-learning [8]. However, as far as we know, most of the prior research has used datasets that consist of per-student feature vectors that *summarize* a student's entire interaction trajectory but do not consider the sequential nature of the interactions; or sequential data where the student's behavior is extracted as a sequence of feature vectors but the length of the sequence is *fixed*. Neither approach directly handles the moment-to-moment temporal dependency and different length of interactive trajectories. Therefore, we implement Dynamic Time Warping (DTW) [11] which calculates the distance between any two sequences of different lengths and also considers moment-to-moment dependencies.

We proposed a general temporal clustering framework that would firstly construct a specified distance matrix on the sequential dataset and then apply clustering approach on the resulted distance matrix. We tested our framework across three datasets collected in Fall 2015, Spring 2016 and Fall 2016 semesters. All participants were trained on a logic tutor named Deep Thought (DT) and they were assigned to different conditions based on how the tutor decided whether to assign a *Problem Solving* or a *Worked Example* on next problem. Two-three weeks after the training, all participants took a in-class midterm as the PostTest. Much to our surprise, empirical results showed no significant difference among different conditions on PostTest scores across all three semesters. So we explored whether our proposed

general temporal clustering framework would generate effective clusters to predict student PostTest scores. To do so, we explored three types of granularity in increasing order of problem, level and session. More specifically, a session contained a student's entire training session on the tutor which involved six levels and each level contained multiple training problems. For three types of granularity: problem granularity recorded students' problem-by-problem behaviors and thus had different lengths for different students since the number of problems that students solved on DT varied greatly: from 19 to 65; level granularity contained the sequential data with a fixed length of six, one per level, for each student; and session granularity had one single summarized feature vector for each student. In our case, we treated session granularity as the baseline for early detection and investigated the impact of different types of granularity on clustering results.

In this work, we applied three distance functions including DTW, normalized DTW and Euclidean distance, and implemented Hierarchical Clustering with four different linkage functions. Finally, we evaluated the goodness of clusters on PostTests. Our results showed that significant difference was consistently found among the discovered clusters when clustering student trajectories at problem granularity rather than level and session granularity, and the best result is found when using the first four out of six levels of trajectories rather than using entire trajectories. Therefore it suggested that using fine-grained problem granularity was more suitable for clustering student interactive trajectories than coarse-grained level and session granularity.

2. RELATED WORK

2.1 Previous Research on Clustering

Previous research has showed the value of clustering for various applications in EDM. For example, clustering has been widely used in student modeling. Yue Gong et al [3] implemented k-means on to identify clusters with distinct students' skill and then applied knowledge tracing model to model students from each cluster separately in order to detect students' knowledge level. They found that clustering had positive impact on student modeling, providing a good representation of student knowledge. Furthermore, Terry Peckham and Gord McCalla [13] utilized k-means in reading comprehension tasks and determined four different clusters based upon cognition skills including positive or negative reading, scanning or scrolling behaviors.

Relatively little research has done to directly cluster student trajectories. Generally speaking, most of the prior research used either per-student feature vectors or the sequential data with fixed length on such task. For the former case, Ke Niu et al [12] extracted the feature vector per learner through analyzing his/her behavior and then applied spectral clustering algorithm to classify students' performance in order to provide benefit for personalized services. They categorized students' performance into nine classes and evaluated clustering results based on accuracy. Similarly, Gholam Montazer [10] proposed hybrid clustering method to group learners in E-learning systems and evaluated clustering results by comparing clustering labels with the ground truth labels.

For using sequential data but with fixed length, Severin Klingler et al [7] designed a pipeline for evolutionary clustering on student behavior sequential data with fixed length in order to group students at any time point and to identify the change of clusters over time. Particularly, Markov Chain model is applied to transfer the original behavior data as well as to capture the moment-to-moment temporal dependency. The optimal number of clusters is selected based upon the best model, evaluated by Akaike information criterion (AIC). Different from this work, we try to clustering the sequences with different lengths.

2.2 Application of DTW

DTW has been successfully applied to a variety of applications related to time series data, such as time series indexing [6], classification [14] and clustering in domains of astronomy, speech physiology, and medicine [1]. More specifically, Hesam Izakian et al [5] applied fuzzy clustering with DTW distance approach on UCR time series data sets and evaluated the performance of clustering methods based on precision value. In addition, Gançarski, Pierre et al [2] utilized DTW to capture the semantic proximity between urban blocks on spatial temporal topographic databases and implemented ascendant Hierarchical Clustering to detect the distinctive evolutions of urban blocks. Furthermore, Nurjahan Begum et al [1] explored DTW by adding pruning strategies and did the multidimensional time series clustering on different types of data sets in astronomy, speech physiology, medicine, entomology and astronomy domains. They evaluated performance of clustering approaches in term of accuracy.

As far as we know, this is the first study of applying DTW to the field of EDM by directly clustering student-system interactive sequential trajectories. Given the special nature of EDM, we further propose normalized DTW and find that normalized DTW is more effective to our task than original DTW.

3. METHODOLOGY

In this section, we first introduce the original and the proposed normalized DTW for calculating the distance matrix between any pair of student interactive trajectories, and then describe how we apply Hierarchical Clustering to identify clusters with distinctive behavior pattern and performance.

3.1 Distance Function

3.1.1 Dynamic Time Warping (DTW)

Given sequences $X = \{x_1, x_2, ..., x_N\}$ and $Y = \{y_1, y_2, ..., y_M\}$ with different lengths ($N \neq M$), a warping path W is an alignment between X and Y, involving *one-to-many* mapping for each pair of elements. The cost of a warping path is calculated by the sum of cost of each mapping pair. Furthermore, warping path contains three constraints: 1) *Endpoint constraint:* The alignment starts at pair $(1, 1)$ and ends at pair (N, M); 2) *Monotonicity constraint:* The order of elements in the path for both X and Y should be preserved same as the original order in X and Y respectively; 3) *Step size constraint:* the difference of index for both X and Y between two adjacent pairs in the path need to be no more than 1 step. In other words, pair (x_i, y_j) can be followed

by three possible pairs including (x_{i+1}, y_j), (x_i, y_{j+1}) and (x_{i+1}, y_{j+1}).

Dynamic Time warping (DTW) is a distance measure that searches the optimal warping path between two series. Particularly, we firstly construct a cost matrix C where each element $C(i,j)$ is a cost of the pair (x_i, y_j), specified by using Euclidean, Manhattan or other distance function. DTW is calculated based on dynamic programming. Initial step of DTW algorithm is defined as

$$DTW(i,j) = \begin{cases} \infty & \text{if } (i=0 \text{ or } j=0) \text{ and } i \neq j \\ 0 & \text{if } i=j=0 \end{cases}$$

The recursive function of DTW is defined as

$$DTW(i,j) = \min \begin{cases} DTW(i-1,j) + w_h \cdot C(i,j) \\ DTW(i,j-1) + w_v \cdot C(i,j) \\ DTW(i-1,j-1) + w_d \cdot C(i,j) \end{cases}$$

Where w_h, w_v, w_d are weight for horizontal, vertical and diagonal direction respectively. $DTW(i,j)$ denotes distance or cost between two sub sequences $\{x_1, ..., x_i\}$ and $\{y_1, ..., y_j\}$, and $DTW(N,M)$ indicates total cost of the optimal warping path.

In equally weighted case $(w_h, w_v, w_d) = (1,1,1)$, the recursive function has the preference on diagonal alignment direction because the diagonal alignment takes one-step cost while the combination of a vertical and a horizontal alignment takes two-steps cost. In order to counterbalance this preference, we can set $(w_h, w_v, w_d) = (1,1,2)$.

3.1.2 Normalized DTW

One potential issue of using the original DTW definition is that the longer the two sequences are, the larger their DTW value will be. Thus, its absolute value may not truly reflect the difference of the two sequences. Thus, we propose the normalized DTW, defined as dividing original DTW by the sum of lengths of two sequences as shown below:

$$DTW_{norm}(N,M) = \frac{DTW(N,M)}{N+M}$$

Each alignment in the warping path has a corresponding weight, selected from (w_h, w_v, w_d) and the sum of weights for all alignments equals to the sum of lengths of two sequences $(N+M)$. Therefore, the normalized DTW evaluates the average distance of alignments in the warping path for two sequences. We will empirically compare the effectiveness of the original DTW and our proposed normalized DTW.

3.2 Hierarchical Clustering

Our proposed framework uses Hierarchical Clustering because K-means cannot directly applied here. K-means needs to calculate the centroid of each cluster while we only have the DTW-based distance for each pair of trajectories.

To apply Hierarchical Clustering, we explore four linkage functions: average, median, complete and ward, which determine how to merge clusters based on the distance between the clusters. Our results show that the first three linkage methods generate extremely unbalanced clusters while the ward linkage discovers relatively balanced ones. Therefore, in the following, we will report our results using ward linkage only.

The optimal number of cluster is selected based upon the measurement called WCSS (*within cluster sum of squares*) [16] defined as

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} d(x_i, x_{i'})$$

Our results show that the optimal number of clusters is 4.

4. EXPERIMENT

4.1 Training Datasets

Our datasets were collected by training students on a logic DT tutor across three semesters: 2015 Fall, 2016 Spring and 2016 Fall referred as DT15F, DT16S and DT16F respectively. For each semester, students were randomly assigned into different conditions based on the pedagogical strategies employed by the tutor. *Pedagogical strategies* were policies used to decide whether give Problem Solving (PS) or Worked Examples (WE) as the next problem. In WE, students were given a detailed example showing the expert solution for the problem. In PS, by contrast, students were tasked with solving a particular problem. For different versions of DTs, we applied different types of data-driven approaches to induce pedagogical strategies [15]. There were a total of four, six and five conditions for DT15F, DT16S and DT16F respectively. One-way ANOVA results showed that there was no significant difference on PostTest scores among conditions across all three semesters: $F(158,1) = 0.728, p = 0.537$ for DT15F, $F(196,1) = 0.644, p = 0.667$ for DT16S and $F(188,1) = 0.445, p = 0.776$ for DT16F. More details were eliminated due to the limitation of space. While no significant was found among different conditions, different pedagogical policies resulted in quite different student-system interactive trajectories and our goal was to investigate whether the proposed temporal clustering framework would be more effective to predict PostTest scores and to discover the true temporal patterns during student training than the condition.

To best describe student learning trajectory, we considered the following 36 continuous features which could be grouped into three categories:

1 **Autonomy (AM):** the amount of work done by the student: such as the number of problems solved so far (*PSCount*) or the number of hints requested (*hintCount*).
2 **Temporal Situation (TS):** the time related information about the work process: such as the average time taken per problem (*avgTime*), or the total time for solving a problem (*TotalPSTime*).
3 **Student Action (SA):** the statistical measurement of student's behavior: such as the number of non-empty-click actions that students take (*actionCount*), or the number of clicks of applying rules for logic proof (*AppCount*).

To fully evaluate our proposed framework, we explored three types of granularity: 1) **Problem granularity** considered students' behaviors problem by problem. When training on DT, the number of problems that each student solved differed greatly and as a result, the length of student interactive sequences varied. For example, about 8%, 4% and 1% of students had more than 40 problems in their interac-

DT	Level	Hierarchical Clustering with Ward Linkage					
		Problem		Level			Session
		Normalized DTW	DTW	Normalized DTW	DTW	Euclidean	Euclidean
DT15F	3	**5.05(.027)***	**6.48(.012)***	3.84(.051).	1.40(.238)	1.30(.256)	2.26(.135)
	4	**10.3(.001)****	1.18(.279)	**5.86(.017)***	**7.67(.006)****	0.75(.388)	2.96(.087).
	5	**6.06(.015)***	1.71(.193)	**4.03(.046)***	2.55(.112)	1.93(.166)	1.81(.181)
	6	3.19(.076).	1.37(.244)	3.79(.053).	0.50(.480)	1.21(.272)	0.76(.385)
DT16S	3	**12.4(.000)*****	0.63(.427)	**8.94(.003)****	1.89(.171)	0.41(.521)	1.00(.318)
	4	**13.7(.000)*****	0(.995)	**10.0(.002)****	2.49(.117)	0.99(.319)	0.38(.536)
	5	**7.1(.008)****	0.84(.359)	**6.36(.013)***	3.75(.054).	0.39(.532)	**15.8(.000)*****
	6	3.11(.079).	0.05(.821)	0.53(.466)	0.67(.412)	0.06(.806)	**8.33(.004)****
DT16F	3	0.28(.594)	0.96(.328)	0.94(.333)	2.38(.124)	0.89(.344)	2.90(.090).
	4	**3.93(.049)***	1.97(.163)	2.61(.108)	3.32(.070).	0.52(.471)	1.14(.288)
	5	**4.76(.030)***	3.64(.058).	2.64(.058).	1.74(.189)	1.65(.201)	0.06(.798)
	6	**3.95(.048)***	**9.67(.002)****	2.27(.134)	1.92(.168)	1.27(.261)	0.0(.997)

Note: significant codes: 0.000 :'****'; 0.001: '***'; 0.01: '**'; 0.05: '*'; 0.1: '.'

Table 1: One way ANOVA using PostTest score as dependent measure and cluster as a factor

tive sequences in DT15F, DT16S and DT16F respectively. 2) **Level granularity** summarized students' behaviors for each level as a singe feature vector; since DT has six levels, the length of level interactive sequence is six for each student. 3) **Session granularity** summarized the students' entire training behaviors by a single feature vector.

Furthermore, there were 158, 196 and 188 students that participated in DT15F, DT16S and DT16F respectively. Combining semesters with three types of granularity, we had a total of 9 data sets.

4.2 Data Preprocessing

Our data-preprocessing involved two steps: 1) *Standardization.* To ensure that our state features measured at different scales would contribute equally to the distance functions, we standardized all features by subtracting mean and dividing standard deviation; 2) *Principle Component Analysis* (PCA), which is widely used for dimensionality reduction. PCA is able to generate mutually independent principle components (PCs) which cover the majority of variance information. We selected PCs with the corresponding variance larger than 1, thus 6-8 PCs were chosen for different training data sets.

4.3 Clustering Process

While most of previous clustering research on sequential trajectory used the entire trajectory, we investigated whether it was more effective to only use sub-sequential trajectories rather than the entire trajectories. This was especially important because we wanted to identify students with different learning patterns, especially the students with *unprofitable* learning as early as possible so personalized learning could be offered.

To do so, we recursively generated our nine training datasets, three types of granularity across three semesters, using subsequential trajectories from the beginning of the training up to each of the six levels separately. For example, 'Level4-Problem-DT16S' training dataset was generated by using problem-by-problem trajectories from the beginning of training process up to level 4 using DT16S. Then we followed the following three steps:

Distance matrix. We explored three types of distance matrices: DTW, normalized DTW and Euclidean distance. Euclidean distance was used as the baseline here.

Outlier Detection. Given that many clustering methods are often sensitive to outliers, we applied filtering approach to remove them from our training data. More specifically, for each type of distance matrix, we calculated the average distance for each student to all others and then obtained the mean μ and standard deviation σ for all students' average distances. We filtered out students whose average distances were larger than: $\mu + 2 * \sigma$.

Cluster Evaluation. We applied Hierarchical Clustering on distance matrices calculated above, and used PostTest scores to evaluate the effectiveness of the resulted clusters.

5. RESULT

As mentioned above, while the assigned condition did not seem to be a crucial factor to predict student PostTest scores, we explored whether our proposed temporal clustering framework could do better.

5.1 Cluster Evaluation

Table 1 summarized clustering results. In Table1, each row denoted clustering results of using student interactive subsequential trajectories, varying from using the first three levels up to the entire six levels. For instance, 'Level 4' used sequential data or summarized data points from the beginning of training process up to level 4. Note that we did not get good clustering results when using only the first two levels so their results were eliminated from the table. This was probably because there were a lot of noises in the first two levels as some students were still getting used to the

DT	#Student	Dependent Measures: F-ratio(p-value)						
		PostTest	Interaction	WrongApp	hintCount	avgstepTime	avgTime	TotalTime
DT15F	155	10.31(**.001**)	40.54(**.000**)	21.55(**.000**)	6.79(**.010**)	0.01(.919)	17.15(**.000**)	20(**.000**)
DT16S	190	13.69(**.000**)	47.59(**.000**)	67.47(**.000**)	99.73(**.000**)	2.77(.097)	28.76(**.001**)	36.21(**.000**)
DT16F	178	3.93(**.048**)	2.28(.133)	0.16(.691)	5.99(**.015**)	13.45 (**.000**)	0.31(.58)	0.20(.655)

DT	Cluster	Size	Dependent Measures: Mean(Standard Deviation)						
			PostTest (score)	Interaction (count)	WrongApp (count)	hintCount (count)	avgstepTime (sec)	avgTime (min)	TotalTime (hour)
DT15F	C1	47	**84.84**(21.64)	**1052**(432)	**80**(47)	**21**(34)	6.01(1.86)	**5.45**(2.31)	**1.75**(0.73)
	C2	26	76.92(26.02)	1259(662)	110(79)	44(45)	**10.88**(3.21)	11.47(5.52)	3.76(1.97)
	C3	55	72.35(28.77)	**2021**(752)	**214**(155)	**76**(61)	8.31(3.00)	**12.64**(5.14)	**4.49**(1.76)
	C4	27	**66.58**(24.49)	1706(600)	154(101)	26(30)	**5.48**(1.73)	7.88(3.28)	2.60(1.04)
DT16S	C1	112	**91.04**(16.53)	**1242**(519)	**104**(64)	**13**(12)	5.89(2.32)	**5.98**(3.60)	**2.06**(1.22)
	C2	41	83.99(23.83)	1483(660)	140(91)	22(16)	**9.37**(3.73)	10.72(4.33)	3.66(1.42)
	C3	14	**70.98**(27.14)	**2186**(551)	275(170)	39(28)	**5.04**(1.84)	8.84(4.50)	3.16(1.73)
	C4	23	78.66(26.08)	2058(994)	**278**(205)	**65**(52)	6.81(2.08)	**10.91**(8.07)	**4.05**(2.98)
DT16F	C1	40	79.61(20.67)	**1216**(500)	122(92)	17(21)	**8.76**(2.53)	9.11(5.43)	3.15(1.94)
	C2	44	88.21(16.35)	1713(867)	147(98)	16(15)	**4.19**(0.94)	**5.71**(2.93)	2.03(1.09)
	C3	35	**78.57**(25.87)	**2335**(887)	**276**(182)	**43**(34)	6.26(1.74)	**11.25**(4.62)	4.09(1.84)
	C4	59	**90.09**(13.95)	1440(528)	**116**(66)	25(28)	5.99(1.48)	7.12(3.30)	2.47(1.19)

Table 2: result of one way anova on dependent measurements for best clustering assignment

tutor. Each cell in Table 1 denoted one-way ANOVA results using PostTest score as the dependent measure and clusters as the factor in the format of *F-ratio*(*p-value*). The bold numbers showed that significant differences were found among clusters on PostTest scores. Each column represented different types of granularity using different distance functions: DTW, normalized DTW and Euclidean. For problem granularity, we only applied DTW and normalized DTW approaches because Euclidean distance could not be applied on sequential trajectories with different lengths. For level granularity, we utilized all three distance functions. Note that when calculating Euclidean distance, we first calculated distance for each level separately and then summed them up. For session granularity, all three distance functions were equivalent in that all became Euclidean distance.

Granularity Comparison. Table 1 showed that among three types of granularity, problem granularity was most suitable for clustering because significant differences were found across all three datasets and across all levels of subsequences on PostTest scores when using problem granularity. This finding was consistent with our hypothesis that directly clustering student moment-to-moment fine grained trajectories indeed provide benefit to discover the underline characteristics of student learning processes.

Distance Function Comparison. To compare the three distance functions, we only focused on the level granularity since it was the only one that involved all three distance matrices. Table 1 showed that both original and normalized DTW outperformed Euclidean distance because no significant differences were found among the clusters using Euclidean distance. To compare the two types of DTW, we focused on both problem and level granularity. Table 1 showed normalized DTW could induce more robust and consistent results than DTW. In short, among the three distance functions, our proposed normalized DTW was the best.

Sub-sequences Comparison. Table 1 showed that consistently significant results were found for all problem granularity data sets using normalized DTW and sub-sequential trajectories up to first four or five levels. Interestingly, using the entire sequential data may be not as effective as using sub-sequences in that for DT15F and DT16S datasets, no significant difference was found when using problem granularity on the entire trajectories.

Variable	Definition
PostTest	the score of student's post test
Interaction	number of student's actions
WrongApp	number of wrong application of rules
hintCount	number of hints that students take
avgstepTime	average time per step
avgTime	average time per problem
TotalTime	time of completing the training process

Table 3: Variables and Definitions

5.2 Clusters Analysis

Table 1 showed that the consistent significant results was found when we clustered on problem granularity using normalized DTW on sub-sequences from beginning of training process to the level 4 across the three semesters' datasets. Therefore, in the following, we will shed some lights on characteristics of the discovered clusters.

Table 2 showed one-way ANOVA results on seven dependent measures using clusters as the factor. Particularly, we bolded p values which were less than 0.05. We found

that there was significant difference on all variables except *avgstepTime* for DT15F and DT16S. Additionally, significant difference existed on three variables including *PostTest*, *hintCount* and *avgstepTime* for DT16F. In order to investigate how much difference existed among clusters based on selected variables, we presented the mean and standard deviation for each pair of cluster and variable in Table 2. We highlighted the mean of variables that were significantly different from others, either extremely large or small. We analyzed the difference among clusters for three semesters separately shown as follows.

1. DT15F. C1 had the highest *PostTest* while C4 had the lowest one among four clusters. C1 had the lowest *Interaction*, *WrongApp*, *hintCount* and *TotalTime* among four clusters. Although C2 and C3 had similar *PostTest*, C2 contained dramatically larger *Interaction*, *WrongApp* and *hintCount* than C3. Furthermore, C3 had the largest value of *Interaction avgTime* and *TotalTime*.

2. DT16S. C1 had the highest *PostTest* and the lowest *Interaction*, on the contrary, C3 had the lowest *PostTest* and the highest *Interaction* among four clusters. Although C2 and C4 had the closed *PostTest*, C4 contained higher *WrongApp* and *hintCount* than C2.

3. DT16F. C4 had the highest *PostTest*, while C3 had the lowest one. Although C2 performed closed to C4, C2 had higher *WrongApp* than C4. Furthermore, C1 had the lowest *Interaction* and the highest *avgstepTime* while C3 contained the highest *Interaction* and *WrongApp*.

In short, our results showed that our discovered clusters indeed had the distinctive interactive patterns and could predict students PostTest better than their assigned conditions.

6. CONCLUSIONS & FUTURE WORK

In this paper, we proposed the temporal clustering framework to directly cluster student interactive trajectories. Particularly, we explored three different distance functions and three types of granularity. Results showed that normalized DTW is the most effective function for generating distance matrix; problem granularity is more effective than level and session granularity. More importantly, through clustering statistical analysis, we were able to identify distinctive patterns among clusters during the learning process, which could provide benefit to the personalized learning. For the future work, we will modify distance matrix by combining kernel function with DTW approach given sequential data containing both continuous and discrete features in order to generate effective distance matrix.

7. ACKNOWLEDGEMENTS

This research was supported by the NSF Grant 1432156 "Educational Data Mining for Individualized Instruction in STEM Learning Environments".

8. REFERENCES

[1] N. Begum, L. Ulanova, J. Wang, and E. Keogh. Accelerating dynamic time warping clustering with a novel admissible pruning strategy. In *Process of ACM SIGKDD*, 2015.

[2] P. Gançarski, A. Puissant, and F. Petitjean. Use of symbolic dynamic time warping in hierarchical clustering of urban fabric evolutions extracted from spatiotemporal topographic databases. *AI Communications*, 2016.

[3] Y. Gong, J. E. Beck, and N. T. Heffernan. Using multiple dirichlet distributions to improve parameter plausibility. In *Educational Data Mining*, 2010.

[4] J. Herold, A. Zundel, and T. F. Stahovich. Mining meaningful patterns from students' handwritten coursework. *Proceedings of EDM*, 2013.

[5] H. Izakian, W. Pedrycz, and I. Jamal. Fuzzy clustering of time series data using dynamic time warping distance. *Engineering Applications of Artificial Intelligence*, 2015.

[6] E. Keogh. Exact indexing of dynamic time warping. In *Proceedings of VLDB*, 2002.

[7] S. Klingler, T. Käser, B. Solenthaler, and M. Gross. Temporally coherent clustering of student data. In *Proceedings of EDM*, 2016.

[8] M. Köck and A. Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *UMUAI*, 2011.

[9] R. M. Maldonado, K. Yacef, J. Kay, A. Kharrufa, and A. Al-Qaraghuli. Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop. In *EDM*, 2010.

[10] G. A. Montazer and M. S. Rezaei. A new approach in e-learners grouping using hybrid clustering method. In *ICEELI*, 2012.

[11] M. Müller. Dynamic time warping. *Information retrieval for music and motion*, 2007.

[12] K. Niu and Z. Niu. A coupled user clustering algorithm for web-based learning systems. In *EDM*, 2016.

[13] T. Peckham and G. McCalla. Patterns in reading comprehension tasks. *EDM*, 2012.

[14] F. Petitjean, G. Forestier, G. I. Webb, A. E. Nicholson, Y. Chen, and E. Keogh. Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm. *KAIS*, 2016.

[15] S. Shen and M. Chi. Reinforcement learning: the sooner the better, or the later the better? In *Proceedings of UMAP*, 2016.

[16] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society*, 2001.

Learner Affect Through the Looking Glass: Characterization and Detection of Confusion in Online Courses

Ziheng Zeng, Snigdha Chaturvedi, Suma Bhat
University of Illinois
Urbana-Champaign, USA
{zzeng13, snigdha, spbhat2}@illinois.edu

ABSTRACT

Characterizing the nature of students' affective and emotional states and detecting them is of fundamental importance in online course platforms. In this paper, we study this problem by using discussion forum posts derived from large open online courses. We find that posts identified as encoding confusion are actually manifestations of different learner affects pertaining to their informational needs–primarily seeking factual answers. We quantitatively demonstrate that the use of content-related linguistic features and community-related features derived from a post serve as reliable detectors of confusion while widely *outperforming* currently available algorithms of confusion detection. We also point out that several prediction tasks in this domain (e.g., confusion and urgency detection) can be correlated, and that a model trained for one task can effectively be used for making predictions on the other task without requiring labeled examples. Finally, we highlight a very significant problem of adapting the classifier to unseen courses.

Keywords

Confusion characterization, discussion forum analysis

1. INTRODUCTION

Discussion fora constitute a central feature of learner interaction in online course platforms, where learners post questions, opinions, and concerns, which are viewed, rated and answered by fellow-learners and/or teaching staff. In the particular instance of courses affording only virtual interactions, such as at-scale learning environments, forum posts constitute rich repositories of students' affective and emotional states captured in real time. The focus of this study is on *characterizing* the nature of students' affective and emotional states, manually identified as confusion in forum posts and developing automatic methods to *detect* them. Here, as in [25] and [2], we operationalize the definition of confusion as a state in which a student hits an impasse and is uncertain of how to move forward. As such, the reasons for confusion could be attributed to lack of clarity on the topic discussed or technical shortcomings of the learning interface, among others. Examples of such posts are shown in Table 1.

Table 1: Posts representing confusion and its absence.

I have also problems with the section "Pre course Survey" I have completed this section several times about 10, I have the final message "Thanks" but at each new connection appears in my courseware "pre course Survey (please complete)" Please help me, what I have to do ? (**Confusion**)
Interesting! How often we say those things to others without really understanding what we are saying. That must have been a powerful experience! Excellent! (**No confusion**)

The strong connection between learner affect, engagement, and learning outcomes has long been understood but studies on their effect on continued participation in internet-based learning environments such as MOOCs is only emerging (e.g.,[25, 2]). In addition to constituting supporting evidence to understand this association, mechanisms to automatically detect learner affect encoded via confusion in discussion fora serve the following ends. Firstly, they inform us about the aspects of a course that are frustrating for learners and hence need improvement [24, 21, 11]. Second, they can aid a timely and accurate intervention to struggling learners by providing critical insights into their emotional states[25], eventually leading to success of this critical course component.

For instance, when a student expresses confusion or misunderstanding about a concept, the immediacy with which the confusion is addressed impacts student satisfaction and course progress. Because of this, and the demands of an at-scale learning environment, efficient and automatic detection of confusion has become more important than ever before. With a steady increase in the number of courses on online course catalogs, and with limited means to control the instructor-to-student ratio in online platforms, the problem of detecting confusion as expressed in online fora is timely. Despite the critical need, relatively few studies analyze confusion in course discussion forum posts [25, 2].

While the explicit purpose of discussion fora is to engage the users in a way that develops a sense of community and communication within large-scale online courses, the posts themselves serve as proxy for learner affect and emotions expressed in various forms. Detecting this encoded affect from posts is an important challenge for natural language processing algorithms. This is because, at the outset, a post indicating confusion could be construed to be a question. Since question posts and confusion posts–forms of information seeking behavior–are remarkably similar, one would expect that approaches to detect questions (e.g., [7]) ought to be directly applicable. However, this is not always the case. Many times confusion posts do not have an explicit question making the two prob-

lems of question detection and confusion detection closely related but not the same. This makes the detection of confusion in a post a non-trivial problem partly because, for posts containing a question, the questions tend to occur with other declarative sentences. A second difficulty is the use of different question styles (informal, where standard features such as the question mark are likely to be absent or where the question is worded without a question mark). Hence, simple heuristics of using question mark or 5W1H words (who, what, which, where, why, how) are rendered inadequate.

Additionally, as observed in [18], finding patterns to identify non-questions is more challenging than finding patterns in questions (since they usually do not share common lexical and/or syntactic patterns). This is directly applicable to confusion posts where posts not indicative of confusion have diverse intent.

Prior studies in this direction (e.g., [6, 2, 25]) have led to the use of linguistic and structural features available from the discussion forum. While similar in spirit to these prior studies, this study sets itself apart from them in many ways. Firstly, we identify that confusion detection is different from simple/complex question detection. In order to solve this problem more effectively, we point out that the community needs a characterization of confusion instead of treating it as yet-another text-classification task. We present an in-depth analysis of types of 'confused posts' using high-quality and reliable manual annotations (Section 4). Motivated by this analysis, we then design features to detect confusion automatically in a supervised framework. We also point out that several prediction tasks in this domain (such as confusion and urgency detection) are correlated, and demonstrate that a model trained for one task can effectively be utilized for making predictions on the other task without requiring labeled examples. Finally, we highlight a very significant problem concerning the applicability of such classifiers to unseen courses. We summarize our contributions below:

Characterizing affective states and informational needs: We observe that nearly half of the posts encoding confusion and considered urgent pertain to users seeking answers to factual questions. Aside from indicating an information need, these posts are also used to report course-specific issues such as concerns with assignments or quizzes as well as to report course-related technical issues (e.g., unavailability of a lecture video or a peer-assessment grade).

Efficient confusion detection: We quantitatively demonstrate that our use of content-related linguistic features of a post and a set of community-related features associated with it serve as reliable detectors of confusion while widely *outperforming* currently available algorithms of confusion detection.

Combined confusion and urgency detection: We show that the trained confusion classifier also functions as an efficient urgency detector when tested on confusion posts also labeled as 'Urgent'.

Scaling the effort to other courses and domains: Based on the dataset, we make concrete suggestions to explore domain adaptation towards building course-generic classifiers. Rather than aiming for course-independent classifiers, our proposal is to harness the utility of available course-specific classifiers for an unseen course, based on suitably defined cross-domain similarities.

By means of a thorough quantitative evaluation of our proposed features in a supervised machine learning model, we demonstrate its effectiveness as a *scalable* and *efficient* model for automatic detection of confusion that generalizes well to unseen courses.

2. RELATED PRIOR WORK

Confusion and its impact on learners: Studies modeling confusion and exploring its relation to learner affect have found that even though students seem to struggle when confused, the situation leads them to attempt to resolve barriers to their understanding of complex concepts [16, 10, 8]. However, it has also been pointed out that remaining confused has a negative effect that leads to student disengagement and eventual dropout, thus making it imperative that confusion be resolved immediately [15, 25]. This necessity is more immediate in the context of learning at scale given the impersonal and the distant nature of the learning process[14, 19]. Thus, detecting learner affect, particularly with respect to understanding the material has the potential to contribute to the design of interventions as shown in prior studies (e.g.,[9, 22]) can lead to increased learning effectiveness in computer-based learning environments such as online courses.

Detecting confusion: Focusing on MOOCs, where the only venue for learner-instructor interaction is the discussion forum, studies are now beginning to explore automated mechanisms to provide timely learner support by analyzing forum content. These include, predicting when instructor intervention is needed [5, 6], monitoring student's opinion towards the course [20], recommending questions to users for assisting students seeking answers [23], identifying acceptable answers [13], organizing the forum content into aspects or topics along with their sentiments to help instructors in promptly addressing common issues [17], identifying posts that express confusion to predict points of eventual student dropout [25], and detecting posts that express confusion to then map confused posts to course video clips as a way to automate interventions [2]. A common feature of these approaches to detect confusion is their reliance on textual and structural features of the discussion forums to design effective algorithms.

While [25] uses a set of linguistic features to detect confusion, it disregards the structural features (e.g. the number of times a post has been read or the number of up-votes) that are found to be useful in detecting the informational need or urgency [6], [2] uses a set of structural features in combination with a linguistic feature in addition to also relying on the other dimensions of a post, such as expression of a sentiment and the sense of urgency. This latter reliance on the other dimensions is not realistic given the manual effort of assigning the labels for sentiment and urgency (needed to design corresponding classifiers). Our study shares similarities with these prior studies in that we rely on the discussion forum information, but differs from them by the use of a novel set of features that encode content-related aspects of forum posts to account for and structural aspects of the forum posts.

We compare the performance of our detection approach to that in [2] and show that our approach *outperforms* current state-of-the-art by a wide margin both in-domain and across course domains. In addition, differing from prior work, we show that our confusion classifier can simultaneously detect urgency, thereby addressing the need for immediacy for learning effectiveness.

3. DATA DESCRIPTION

The forum posts analyzed in this study are from the Stanford MOOC Posts dataset, a corpus composed of 29,604 anonymized learner forum posts from eleven Stanford University public online classes [1]. The posts are taken from three course domains: Humani-

Table 2: Summary of posts from the three discussion forums

Category	No. of Posts	Not Confused	Confused	Confused & Urgent (%)	No. of sentences per post (mean, sd)
Education	9878	6714	640	67.5	(3.6, 2.8)
Humanities	9723	1358	2257	86.4	(4.5, 4.7)
Medicine	10001	1581	1598	38.9	(4.3, 3.7)

ties/Sciences, Medicine, and Education, with about 10,000 posts in each set.

A salient feature of the dataset is that each post is available with manually assigned labels for six dimensions indicating *confusion*, *urgency*, *question*, *opinion*, *answer*, and *sentiment*. We encourage the readers to refer to [1] for more details. In our study, we only consider the dimensions of *Confusion* and *Urgency*:

Confusion - encodes the extent to which the post expresses confusion, on a scale of 1 (expert knowledge) to 7 (extreme confusion);

Urgency - denoting the extent to which the post is interpreted to be urgent and requires that an instructor respond to the post with 1 denoting 'not urgent at all' and 7 denoting 'extremely urgent';

We divide the posts into two groups–"confused" and "not confused" based on their gold *Confusion* scores. A score above 4 is considered a *Confused* post, whereas a score below 4 is regarded as a *Not confused* one (we disregard posts with score $= 4$ from the analyses). Likewise, an Urgency score above 4 is regarded as an Urgent post, whereas a score of 4 and below is regarded as a non-Urgent post. A summary of the data set is provided in Table 2.

4. CHARACTERIZING CONFUSION

To understand how confusion is expressed in forum posts, two of the authors independently coded a random sample of 200 posts from the entire data set for the following 6 types:

1. **Factual**, if the post seeks clarification of a factual aspect of the course material, as in the post, "Does this mean logistic regression always gives adjusted ratios and the manually computed ratios are unadjusted?"
2. **Course-specific**, if the user seeks a course-specific clarification, such as "Dear Staff, Can you give atleast 2 attempts for each quiz. Giving only one attempt is making us loose interest in the course. Kindly consider."
3. **Course-technical**, if the user seeks clarification on technical aspects of the course. For example, "I am trying to download 5.R.RData, but I cannot open it, can please let me know how I can open this file. With kind regards,"
4. **Recommendation**, if the user is seeking a recommendation. For instance, consider the following post. "another question would you use this form throughout the whole essay? or would you shorten it after using the full phrase?"
5. **Frustration**, where the user expresses frustration, as in, "I had the same issue. Am I bad at finding the check button and bad at math???"
6. **Other**, for posts that belong to none of the above 5 types.

The inter-rater reliability, κ, was 0.81. Based on the instances where both coders agreed, we characterize the type of posts. True to the fact that the discussion forum is an avenue for learners to seek learning support from fellow learners, the most popular post type is *Factual* (54% of the annotated posts), where learners seek to clarify their misunderstandings of concepts presented in the course. This post type is then followed by *Course specific* (27%) and *Course technical* (12%). The remaining posts were categorized as *Recommendation* (3%), *Frustration* (2%) and *Other* (2%).

Overall, these observations confirm that posts indicative of confusion need to be addressed in a timely manner; even though some of them may not be explicit questions, they echo the information seeking nature and the uncertainty encoded in posts that are explicit questions. Additionally, we hypothesize that the inherent difference in the nature of affective states encoded as confusion could be responsible for the inconclusive nature of the effect of confusion on learning outcomes (e.g., confusion positively impacting learning in [10] and negatively impacting outcomes in [25].

5. DETECTING CONFUSION

Our next focus is on building a confusion detector that will allow for automatic identification of confusing posts to facilitate immediate response thereby enhancing the learning experience and reducing learner frustration. Towards this end, the confusion-detection features can be grouped into two categories: content-related and community-related features.

Content-related features: These features analyze the textual content of the post:

1. Automated readability index (ARI): Readability indices are designed to measure how understandable a piece of text is. We hypothesize that the posts encoding confusion, owing to their information seeking nature as well as owing to the tendency of learners to post verbatim course content, have higher readability indices (i.e., are more difficult to read) than those posts that do not encode confusion.
2. Post length in words;
3. Unigrams: These binary features encode whether a word occurred in the post or not.
4. Topicality (LDA): These features use supervised Latent Dirichlet Allocation (LDA) [4] to generate the LDA labels as features. Towards this, we first perform a preprocessing step involving stop-word removal (including numbers and punctuation); stemming; and removing high-frequency (top 1%) and low-frequency words (occurring fewer than 5 times). Then a supervised LDA (sLDA) model is obtained with the confusion labels. Here we use the confusion labels for each post to obtain two sets of LDA words (associated with presence/absence of confusion). This model predicts a label (confusion or not) based on the words in the post that occur in the respective LDA set.
5. Question mark: Since confusion is often expressed via questions, this feature checks for presence of a question mark.

Community-related features: A second set of predictors of whether a post encodes confusion or not is obtained by observing how the community of learners reacts to a post. In particular, a post that is of general interest to learners (such as one that is seeking a factual clarification, or that seeks resolution for a course-related technical problem) would be read by several viewers, thus leading to a rela-

Table 3: Performance of our approach and the two baselines. 'NR' stands for results that were not reported in the respective paper.

Course	Model	Accuracy	Precision	Recall	F-measure	Cohen's Kappa
Humanities	Our Model	84.38	90.38	77.16	**83.14**	0.69
	Unigrams Model[3]	71.99	71.00	82.21	75.28	0.44
	YouEDU[2]	NR	77.80	64.20	70.00	0.62
Education	Our Model	80.04	79.44	81.02	80.00	0.60
	Unigrams Model[3]	82.03	78.76	87.81	**82.96**	0.64
	YouEDU[2]	NR	NR	NR	38.30	0.36
Medicine	Our Model	83.75	86.67	80.14	**83.16**	0.67
	Unigrams Model[3]	70.39	72.82	65.33	68.69	0.41
	YouEDU[2]	NR	69.90	58.90	62.70	0.56

tively higher number of reads. Likewise, posts encoding confusion are considered important resulting in higher up-votes. Accordingly, our set of features includes the number of (i) reads and (ii) up-votes of the post.

We cast the task of confusion detection as one of binary classification, where posts expressing confusion constitute the positive class. For the purpose of this study we do not use the confusion-types identified in the characterization. We trained an Elastic-net model, which is a regularization approach that uses a mixture of L_1 and L_2 penalties to perform variable selection [26].

6. EXPERIMENTS

Datasets: From Table 2 we can see that for majority of the courses, the data is biased towards the negative (not-confusion) class. This makes learning difficult, especially for the positive (confusion) class. In order to alleviate this problem, for each course, we down-sample the negative class (randomly) such that the two classes are balanced. Additionally, forum posts from 'Education', contains very few (640) confusion posts. This resulted in a very small resampled dataset for this course (compared to the posts in Humanities and Medicine) after down-sampling the negative class. Noting that this dataset was prone to over-fitting due to very few posts as compared to the number of features, we up-sampled the positive class to twice its original size before down-sampling the negative class as before.

We also tokenized the content of the posts; removed stopwords (175 unique words); stemmed [12]; and removed infrequent words (with count less than 5). The final vocabulary lists for these courses contained about 2400, 1400, and 1750 words respectively.

Evaluation Measure: From the perspective of helping students, the positive (confusion) class, indicative of learner affect, is more important than the negative class. An ideal classifier would, therefore, identify all confusion posts bringing them to the instructor's attention (high recall for the positive class). Additionally, a high precision for the positive class is also important so that the instructor's efforts are not wasted in analyzing false-positives. Therefore, it seems natural to evaluate models using the F-measure of the positive class (in-line with related prior work). For the sake of completeness, we also report accuracy and Cohen's Kappa.

6.1 Confusion Detection

Table 3 compares 10-fold CV results of our model with two prominent baselines: (i) Unlike our model, our first baseline [2] uses manual annotation for dimensions such as Opinion and Question (apart from ground truth confusion labels for training). We include their performance as reported in their paper. (ii) The second baseline [3] uses only Unigram features. We replicated this baseline in our experiments. Also, a random baseline would get a score of 50%. However, we do not include this result in the tables for clarity.

We can see that, for Humanities and Medicine, our model performs significantly better than the baselines. For instance, for the Humanities course, our model achieves 10.4% and 18.8% relative improvements in F-measure over the two baselines. Similarly, on the Medicine course, our model achieves 21.1% and 32.3% relative improvements in F-measure. Our model's Cohen's Kappa (and accuracy when reported) are also better than the baselines. *This indicates the utility of our features in not only learning the positive class, but also performing well on the overall classification task.*

For the Education course, our model outperforms the YouEDU[2] model significantly. Our model achieves an F-measure of 80.0% as opposed to only 38.3% by the YouEDU model. We would like to remind the reader that the data for the Education course was particularly skewed towards the negative class (not-confusion) with only 6.5% of the posts belonging to the positive class (confusion). *This stark difference in performances of the two models, emphasizes the need for models that can pay particular emphasis on the minority class, which in this case is more significant than the majority class.*

Interestingly, for this course, the performance of our model is comparable to the unigrams model [3], with the latter performing slightly better. Both the models use the same dataset and so neither suffers from the rare-class problem. The seemingly disadvantageous nature of our features for this course is not consistent with the results obtained for the other two courses, and requires further investigation. However, in general, the features proposed in our approach provide a considerable boost in performance.

6.2 Effect of Degree of Confusion

As mentioned in the data description, the dimension of Confusion was annotated on a scale of 1-7 (denoting the degree of confusion), which could be potentially construed to correspond to a scale of affective states. While we had conflated all the positive confusion levels (rep. negative levels) for the purpose of detection, here we evaluated the performance of our detector on its ability to detect the degree of confusion. We examined the performance (here, accuracy) at every Confusion degree and report the results in Table 5. We observe that the accuracy monotonically increases with confusion level, suggesting the classifiers suitability for real applications (e.g., potentially informative to instructional designers).

6.3 Feature ablation analysis

Table 4 compares the predictive importance of our various features by removing them one at a time. For convenience, the first row for each course depicts the performance with the full feature set (same as Table 3). From the table, 'Unigram' and 'Question-mark' seem to be the most valuable. For instance, the model for Education re-

Table 4: Feature ablation. For each course, the top row corresponds to the complete feature set. The subsequent rows represent performance with one of the features removed. Removing any feature (except 'LDA') decreases performance, indicating its utility.

Course	Feature-class	Removed Feature	Accuracy	Precision	Recall	F-measure	Cohen's Kappa
Humanities	–	None	84.38	90.38	77.16	**83.14**	0.69
	Community-related	Number of Reads	84.16	89.86	77.24	82.96	0.54
		Score	84.24	89.86	77.40	83.06	0.55
	Content-related	ARI	84.12	89.66	77.40	82.95	0.55
		Post Length	83.76	89.40	76.88	82.53	0.54
		Unigrams	80.73	88.44	70.82	78.53	0.24
		LDA	84.52	90.01	78.05	83.43	0.70
		Question Mark	70.91	72.64	73.03	72.00	0.53
Education	–	None	80.04	79.44	81.02	**80.00**	0.60
	Community-related	Number of Reads	76.99	75.67	79.30	77.26	0.54
		Score	77.46	75.98	80.16	77.88	0.55
	Content-related	ARI	77.62	76.04	80.47	78.04	0.55
		Post Length	76.88	75.42	79.53	77.25	0.54
		Unigrams	62.15	60.67	69.77	64.70	0.24
		LDA	85.16	83.33	87.97	85.44	0.70
		Question Mark	76.64	73.86	82.58	77.88	0.53
Medicine	–	None	83.75	86.67	80.14	**83.16**	0.67
	Community-related	Number of Reads	83.65	86.41	80.20	83.10	0.67
		Score	83.72	86.49	80.26	83.17	0.67
	Content-related	ARI	83.78	86.51	80.39	83.25	0.68
		Post Length	83.72	86.59	80.14	83.13	0.67
		Unigrams	80.43	86.65	72.93	78.91	0.61
		LDA	83.62	86.06	80.64	83.14	0.67
		Question Mark	70.04	73.90	62.49	67.55	0.40

Table 5: Accuracy of the model in detecting Confusion at different levels. Numbers in () show number of instances. Performance improves with increasing scores. Confusion at levels higher than 5.5 did not have sufficient instances.

Course	4.5	5	5.5
Education	0.76 (521)	0.80 (93)	0.87 (24)
Humanities	0.69 (463)	0.79 (553)	0.79 (190)
Medicine	0.71 (641)	0.86 (762)	0.90(154)

lies heavily on the Unigram features (removing which decreases the F-measure from 80% to 64.7%). Removing any of the other features like 'Number of reads', 'Post Length' also hurt model performance, albeit to a lower degree. Experiments reveal that the inclusion of LDA as a feature hurts more than helping the model's performance. Overall, we can conclude that removing most of our features reduces the performance of the model to various degrees, indicating their utility.

6.4 Testing on Unseen Courses

Our supervised model requires having labeled training data. However, considering the short duration of most online courses, manually annotations for an ongoing course is not only expensive but also infeasible due to time and privacy constraints. Hence, domain-independence of such classifiers is extremely desirable. In our next experiment, we test a given model on an unseen course in order to estimate the domain-independence of existing methods. Table 6 shows the results of this experiment. The last column of the table shows the change in model's performance when tested on a course not seen during training. We can see that the model performance always decreases when it is tested on a new course. However, the decrease can be expected to depend on the difference in the class-conditional distributions of the train and the test sets. *From this perspective, one could argue that the post from Humanities and Medicine are more similar to each other than to the posts from Education, as far as this task is concerned.* From instance, when a model trained on data from Humanities is tested on data from Medicine, and vice-versa, the decrease in F-measure is only about of 4 points. On the other hand, the model suffers a much greater decrease in performance when it is trained on data from Medicine (or Humanities) and is tested on data from Education, and vice-versa.

This result indicates that domain-adaptation methods, that aim to build course-independent classifiers, should not blindly aim for classifiers that perform well on all courses. Instead, a more opportunistic alternative would be based on assessing the similarity between the data from the source (training) and the target (testing) courses.

6.5 Urgency Prediction

In Table 2 we can see that there is a high correlation between the 'Confused' and 'Urgent' labelings. For instance, 86.4% of the posts from Humanities labeled as 'Confused' are also labeled as 'Urgent'. Therefore, it would be of interest to investigate how well a model trained for detecting confusion would perform on the task of detecting urgency. Table 7 shows the results of this experiment. For this table we train our model using ground-truth Confusion labeling, and use the trained model to make predictions on the test instances. We then judge model's performance by comparing predicted positive/negative class with the ground truth Urgent/not-urgent class. Note that we use urgent/not-urgent labelings only during evaluation and not training. Like before, we are primarily interested in the F-measure of the positive (urgent) class. From the table we can see that we achieve a reasonably high F-measure especially for Humanities (75.78%) and Medicine (80.68%). This suggests that for the two related tasks, classifiers trained for one task could be used for the other task with little modifications.

7. FUTURE DIRECTIONS

We have presented detailed analysis of posts indicative of confusion from a collection of discussion forum posts from learners on online courses spanning 3 domains. Our detailed manual analysis of the types of confusion posts suggests that subsequent explo-

Table 6: Model performance decreases when tested on unseen courses. Performance drops indicate a need for more aggressive domain-adaptation efforts on *diverse* pairs (like Education-Humanities), as compared to *similar* ones (Humanities-Medicine).

train-Course	test-Course	Acc.	Precision	Recall	F-measure	Kappa	Change in F-measure
Humanities		84.38	90.38	77.16	83.14	0.69	–
Education	Humanities	70.25	67.86	76.95	72.12	0.40	-11.02
Medicine		79.16	78.95	79.53	79.24	0.58	-4.10
Education		80.04	79.44	81.02	80.00	0.60	–
Humanities	Education	71.88	81.60	56.48	66.76	0.44	-13.24
Medicine		70.82	77.17	59.14	66.96	0.42	-13.04
Medicine		83.75	86.67	80.14	83.16	0.67	–
Humanities	Medicine	81.06	87.03	73.00	79.40	0.62	-3.76
Education		65.15	61.40	81.59	70.07	0.30	-13.09

Table 7: Model trained for detecting confusion performs well on the Urgency prediction task without using urgency labels.

Course	Accuracy	Precision	Recall	F	Kappa
Humanities	80.50	72.07	80.59	75.78	0.60
Medicine	83.02	76.57	85.54	80.68	0.66
Education	61.95	30.13	88.15	44.10	0.26

rations could consider more specific models involving dedicated components for each of the confusion types.

Future work could also focus on supplementing our results with qualitative analyses, e.g. via interviews of learners, to explore specific findings in greater depth. Another related direction for future exploration is the inclusion of clickstream information in the analysis to afford a broader view of learner-content interactions in the presence of confusion.

8. ACKNOWLEDGMENTS

This work is supported in part by the IBM-ILLINOIS Center for Cognitive Computing Systems Research (C3SR) – a research collaboration as part of the IBM Cognitive Horizons Network.

9. REFERENCES

[1] A. Agrawal and A. Paepcke. The stanford mooc posts dataset, December 2014. Available from http://datastage.stanford.edu/StanfordMoocPosts/.

[2] A. Agrawal, J. Venkatraman, S. Leonard, and A. Paepcke. YouEDU: addressing confusion in mooc discussion forums by recommending instructional video clips. In *EDM 2015*, pages 297–304. ACM, 2015.

[3] A. Bakharia. Towards cross-domain mooc forum post classification. Learning @ Scale, pages 253–256. ACM, 2016.

[4] D. M. Blei and J. D. McAuliffe. Supervised topic models. In *NIPS*, pages 121–128, 2007.

[5] M. K. Chandrasekaran, M. Kan, B. C. Y. Tan, and K. Ragupathi. Learning instructor intervention from MOOC forums: Early results and issues. In *EDM*, pages 218–225, 2015.

[6] S. Chaturvedi, D. Goldwasser, and H. Daumé III. Predicting instructor's intervention in MOOC forums. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1501–1511, 2014.

[7] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun. Finding question-answer pairs from online forums. In *ACM SIGIR conference on Research and development in information retrieval*, pages 467–474. ACM, 2008.

[8] S. Craig, A. Graesser, J. Sullins, and B. Gholson. Affect and learning: an exploratory look into the role of affect in learning with autotutor. *Journal of educational media*, 29(3):241–250, 2004.

[9] R. S. J. de Baker, S. M. Gowda, M. Wixon, J. Kalka, A. Z. Wagner, A. Salvi, V. Aleven, G. Kusbit, J. Ocumpaugh, and L. M. Rossi. Sensor-free automated detection of affect in a cognitive tutor for algebra. In *EDM*, pages 126–133, 2012.

[10] S. D'Mello, B. Lehman, R. Pekrun, and A. Graesser. Confusion can be beneficial for learning. *Learning and Instruction*, 29:153–170, 2014.

[11] C. Geigle and C. Zhai. Scaling up online question answering via similar question retrieval. In *L@S*, pages 257–260, 2016.

[12] K. Hornik. Snowball: Snowball stemmers, 2007. R package version 0.0-1.

[13] M. Jenders, R. Krestel, and F. Naumann. Which answer is best?: Predicting accepted answers in MOOC forums. In *WWW*, pages 679–684. ACM, 2016.

[14] R. W. Larson and M. H. Richards. Boredom in the middle school years: Blaming schools versus blaming students. *American journal of education*, pages 418–443, 1991.

[15] D. M. C. Lee, M. M. T. Rodrigo, R. S. J. de Baker, J. O. Sugay, and A. Coronel. Exploring the relationship between novice programmer confusion and achievement. In *International Conference on Affective Computing and Intelligent Interaction*, pages 175–184. Springer, 2011.

[16] B. Lehman, S. D'Mello, and A. Graesser. Interventions to regulate confusion during learning. In *Conference on Intelligent Tutoring Systems*, pages 576–578. Springer, 2012.

[17] A. Ramesh, S. H. Kumar, J. R. Foulds, and L. Getoor. Weakly supervised models of aspect-sentiment for online course discussion forums. In *ACL*, pages 74–83, 2015.

[18] K. Wang and T. Chua. Exploiting salient patterns for question detection and question retrieval in community-based question answering. In *COLING*, pages 1155–1163, 2010.

[19] Y.-C. Wang, R. Kraut, and J. M. Levine. To stay or leave?: the relationship of emotional and informational support to commitment in online health support groups. In *ACM conference on Computer Supported Cooperative Work*, pages 833–842, 2012.

[20] M. Wen, D. Yang, and C. P. Rosé. Sentiment analysis in MOOC discussion forums: What does it tell us? In *EDM*, pages 130–137, 2014.

[21] A. F. Wise, Y. Cui, and J. Vytasek. Bringing order to chaos in MOOC discussion forums with content-related thread identification. In *Conference on Learning Analytics & Knowledge, LAK*, pages 188–197. ACM, 2016.

[22] B. Woolf, W. Burleson, I. Arroyo, T. Dragon, D. Cooper, and R. Picard. Affect-aware tutors: recognising and responding to student affect. *International Journal of Learning Technology*, 4(3-4):129–164, 2009.

[23] D. Yang, D. Adamson, and C. P. Rosé. Question recommendation with constraints for massive open online courses. In *Eighth ACM Conference on Recommender Systems, RecSys*, pages 49–56, 2014.

[24] D. Yang, M. Piergallini, I. K. Howley, and C. P. Rosé. Forum thread recommendation for massive open online courses. In *EDM*, pages 257–260, 2014.

[25] D. Yang, M. Wen, I. Howley, R. Kraut, and C. Rose. Exploring the effect of confusion in discussion forums of massive open online courses. In *Learning@ Scale*, pages 121–130. ACM, 2015.

[26] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Modeling Classifiers for Virtual Internships Without Participant Data

Dipesh Gautam
The University of Memphis
Memphis, TN 38152
dgautam@memphis.edu

Zachari Swiecki, David W. Shaffer
University of Wisconsin-Madison
Madison, WI 53706
{swiecki,dws}@wisc.edu

Arthur C. Graesser, Vasile Rus
The University of Memphis
Memphis, TN 38152
{graesser,vrus}@memphis.edu

ABSTRACT

Virtual internships are online simulations of professional practice where students play the role of interns at a fictional company. During virtual internships, participants complete activities and then submit write-ups in the form of short answers, digital notebook entries. Prior work used classifiers trained on participant data to automatically assess notebook entries from these learning environments. However, when teachers create new internships using available authoring tools, no such data exists. We evaluate a method for generating classifiers using specifications provided by teachers during their authoring process instead of participant data. Our models rely on Latent Semantic Analysis based and Neural Network based semantic similarity approaches in which notebook entries are compared to ideal, expert generated responses. We also investigated a Regular Expression based model. The experiments on the proposed models on unseen data showed high precision and recall values for some classifiers using a similarity based approach. Regular Expression based classifiers performed better where the other two approaches did not, suggesting that these approaches may complement one another in future work.

Keywords

Automated assessment, text classification, LSA, neural network, semantic similarity, regular expressions,

1. INTRODUCTION

Recently, authoring tools have been developed that let teachers customize and create new versions of digital learning environments such as intelligent tutoring systems and simulations [15]. However, if these environments use integrated automated systems, such as classifiers, customization can be problematic: a new environment invalidates previous automated systems and participant data does not yet exist to train new ones. Therefore, teachers who author these learning environments must implement them, at least initially, without a key component of the technology.

For example, virtual internships are online simulations of professional practice where participants play the role of interns at a fictional company [14]. During virtual internships, participants complete activities and submit work in the form of digital notebook entries. Typically, these are short answer responses ranging from a few sentences to a paragraph in length. Prior work has investigated automated assessment of notebook entries by training classifiers on participant data [10]. However, since the development of the Virtual Internship Authoring Tool [18], teachers can now customize activities and their notebook requirements. Thus, previously developed classifiers may no longer be valid and, initially, participant data is not available to use for model training.

In this paper, we present and test a method that addresses this issue by generating classifiers from specifications that teachers provide during the authoring process rather than waiting to generate them from participant data. Ultimately, these classifiers will be integrated into a fully automated assessment system that will score participant notebook entries. In this study, however, we only report on the development of classifiers for determining whether teacher defined requirements are present or absent in an entry, not classifiers that assign a final assessment.

2. BACKGROUND

Several automated essay scoring systems [3, 8, 16] have been developed to tackle the challenges of costs, reliability, generality and scalability while assessing open-ended essays. Previous researches on automated essay scoring focused on the argumentative power of an entire essay, while in our case, the student generated content is typically short text the length of a sentence or paragraph. Also, the focus of our assessment is to classify the content based on the presence or absence of semantic content defined by teachers during their authoring process. This means that style and higher-level constructs, such as rhetorical structure, are less important in our task compared to essay scoring and that factors that focus more on content measures are more important. Therefore, we limit our work to a semantic similarity approach and Regular Expression (RegEx) matching approach to identify the presence of targeted semantic content in participant generated text.

Various methods of text similarity measures have been used from the very early years of information retrieval. One of the simplest approach is to use the lexical overlap between the texts, however this approach does not consider the semantic relation between the words. Salton & Lesk [13] used is term frequency based vector model for documents similarity. Such model fails when two texts with same meaning have few overlapping words. Other approaches use knowledge base such as WordNet to find semantically similar words in two text [4, 9]. However these approaches face challenges of word sense disambiguation. Other approaches use LSA or LDA methods that rely on large corpus and do not face word sense disambiguation challenge [11].

Rus et al.[11] 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 [5]. They used the most significant indices to build a prediction model that can identify true and false paraphrases and also several categories of paraphrase types. Our work is significantly different than their work as our classifier

model does not rely on participant generated content (we develop classifiers from teachers specifications of content before any participant response is available), secondly our paraphrase detection model measures semantic relation between the text without depending on linguistic features such as content word counts.

Our LSA based similarity method relies on the combination of constituent words a phrase. Hence the similarity score will be more biased towards phrases having common words. While the Neural Network (NN) based semantic similarity method proposed by [7, 17], which we also explored, projects the phrase pairs into common low dimensional space hence the similarity score obtained will be more consistent irrespective of the presence of common words in the phrases.

Our work closely relies on previous works [2, 4, 9] where the authors proposed methods to measure the semantic similarity between texts. The authors in [2] and [4] used knowledge bases such as WordNet while the authors in [9] used word to word similarity and vectorial representation of words derived using Latent Semantic Analysis (LSA) to compute the semantic similarity of two given texts. In addition to these methods, we used in our work presented here phrase vectors generated using Neural Network based models [7, 17]

Our work is also partially related to the work by Cai et al.[1], which proposed methods to evaluate student answer in an intelligent tutoring system. They used LSA and RegEx to assess student answers. Their work showed that the carefully created RegEx had high correlation with human raters' scores. They also noted that the correlation increased when the expected answers created by experts were combined with the previous students' answers to assess new student answers.

3. METHODS

We developed three different types of classifier models and evaluated their performances separately.

To generate our classifiers, we worked with data from one teacher as she authored an activity in the virtual internship, *Land Science*. In *Land Science*, participants work to design a city zoning plan that balances the demands of stakeholders who advocate for indicators of community health. In the activity that this teacher customized, participants describe their proposed zoning changes in a notebook entry. In the first step of our method, the teacher defines assessment criteria for an entry in terms of core concepts, or the key semantic content they want to be present or absent in an entry. For this entry, the teacher defined five core concepts (see Table 1). Next, she constructed six example entries and identified the chunks of text in each example that expressed each concept. In addition, she provided lists of keywords for each core concept that she expected to be present in participant notebook entries.

Afterward, we developed various classifiers for each core concept based on the teacher provided items: sample responses, core concepts, and concept keywords. In this paper, we report three such classifier types; The LSA based semantic similarity threshold classifier, the NN based semantic similarity classifier, and the RegEx based classifier.

In both the LSA based and NN based classifiers, we use a sliding window to search for the most similar chunk in an intern's notebook entry. That is, for each teacher-defined chunk, we slide a window of equal size over the student entry. For each such participant-chunk identified by the sliding window over the student's notebook entry, we calculate the semantic similarity of the text within the window to the teacher-defined chunk. After the similarity of all windows to a teacher-chunk has been calculated, we assign the highest value as the similarity score for a given core concept. For LSA based classifiers, we calculated the similarity score using SEMILAR [12]. For the NN based classifier, we calculated similarity score using the Sent2Vec[1] tool. Since both the tools are capable of taking phrases or sentences as input, we give the chunks as input phrase, hence in the rest of the sections, we call these chunks as phrases.

If the highest similarity score is high enough, e.g. higher than a threshold, we decide the target core concept is present in the student response. Otherwise, we infer the student respond does not include the core concept. That is, we developed a semantic similarity based classifier for assessing students' responses.

In order to choose a threshold for the similarity based classifiers, we derived a threshold by calculating the similarity score between the chunks of each of the core concepts tagged by the teacher for both LSA based and NN based methods. See the experiment section for details.

To test the validity of our approach, we developed classifiers for each target concept and then tested them using 199 participant entries coded by humans for the presence or absence of each core concept.

Because our initial thresholds were created without the aid of participant data, we expected that better thresholds would exist. We therefore sought to compare the performance of our classifiers using two different thresholds, the *derived* thresholds above and *ideal* thresholds (described in more detail below). To calculate the ideal threshold for each classifier we varied the semantic similarity thresholds from zero to one and obtained precision and recall measures for each threshold using participant data.

For the RegEx based classifiers, we used the teacher provided keywords, which were generated without using participant data, to create regular expression lists for each core concept. We infer that the target core concept is present in a given entry as long as any of its associated keywords are present, as determined by regular expression matching. Therefore, in contrast to the LSA and NN models, a threshold is not required for the RegEx classifiers.

The semantic similarity approach minimizes the teachers' input which encouraged us to adopt it for assessing participant responses with respect to containing (or not) targeted, required concepts. This method is also relatively easy to automate, meaning that after the teacher has made a small set of specifications, classifiers can be developed without further human input. The RegEx approach is less flexible compared to the semantic similarity approach as novel expressions of a core concept, not encoded yet in the regular expressions, are less likely to be correctly identified. However, the RegEx is capable of identifying core concepts that are characterized by a closed set of keywords and semantic similarity may not be able to perform as needed.

[1]https://www.microsoft.com/en-us/download/details.aspx?id=52365

4. EXPERIMENTS AND RESULTS

First, we describe the data set we used in our experiments and then present the results obtained with our automatically generated classifiers. We also apply these classifiers to participant generated notebook entries to assess the performance of our models on unseen data.

4.1 Data Set

As we mentioned above, our classifiers were generated from specifications made by a teacher as she customized an activity in *Land Science*. To evaluate our method and test how our classifiers would perform on unseen data, we selected 199 participant entries from prior, uncustomized, implementations of *Land Science*. We took these entries from *uncustomized* versions of the activity the teacher in this study worked to customize. In this case, the customizations to this activity's notebook requirements and assessment criteria, as defined by the core concepts, were not drastically different from the requirements and criteria of the original activity. Thus, this situation provided a case where we could test our classifiers on data that was expected to contain some distribution of the core concepts. In general, however, our method for generating classifiers is meant to accommodate both small customizations, such as we have here, and more drastic ones, such as a case where a teacher creates an entirely new activity. Therefore, we cannot always expect to have such similar data for testing.

The 199 participant entries were manually coded for each core concept by two raters. Both raters had worked with the teacher in this study to define the core concepts and had extensive prior experience coding notebook entries from *Land Science*. Using the process of *social moderation* [6], the raters agreed on the presence or absence of each core concept for each of the 199 entries. From Table 1, we see that the distributions of some concepts are balanced (C2), while others are skewed (C5). However, because we built classifiers based on the textual features of teacher samples, skewness should have a small effect on the performance of the model.

Table 1. Distribution of concepts in data set

Concept	Notations	#Concepts	%Concepts
land use changes	C1	141	72.860
original land use configuration	C2	114	57.280
location of land use change	C3	79	39.690
indicator changes	C4	128	64.320
stakeholder demands	C5	46	23.110

4.2 Threshold Initialization Method

To derive a similarity score threshold, which is needed for the semantic similarity based classifiers, we calculated the similarity scores between the tagged chunks of text for each core concept in the teacher provided examples. Next, we calculated the average and standard deviation of these scores and set our threshold as the average similarity minus one standard deviation for each core concept. The values we obtained using this approach are reported in Table 2, where the last column is the derived threshold for each classifier. Table 2 shows thresholds for both LSA based similarity and the NN based model.

Phrase similarity based on LSA relies on the combination of constituent words a phrases. Hence the similarity score will be more biased towards phrases having common words. While the NN based semantic similarity method [7, 17] projects the phrase pairs into common low dimensional space hence the similarity score obtained will be more consistent irrespective of the presence of common words in the phrases.

Table 2. Derived threshold for LSA based and NN based similarity method

Classifier		Avg.	Std.	Avg. - Std.
C1	LSA	0.584516	0.228474	0.356042
	NN	0.437065	0.122893	0.314172
C2	LSA	0.239488	0.189726	0.049762
	NN	0.242053	0.168682	0.073372
C3	LSA	0.696795	0.103681	0.593114
	NN	0.523347	0.077424	0.445923
C4	LSA	0.278877	0.170271	0.108607
	NN	0.174579	0.124677	0.049902
C5	LSA	0.466482	0.196369	0.270113
	NN	0.149499	0.096005	0.053494

Note: **Avg.**=average similarity score, **Std**=standard deviation.

In Table 2 it is also observed that the standard deviations of similarity scores for NN based models are less than that of the LSA based semantic similarity model in all the five classifiers. This validates our previous understanding that LSA based similarity measures is more biased towards phrases with high degree of word overlap and gives lower score for the phrases with lower degree of or word overlap, resulting high variation in the score. On the other hand, NN based method does not suffer from such biasedness.

4.3 Results

We now present precision and recall results for LSA based and NN based models for the derived thresholds presented earlier and for ideal thresholds (described next). Afterward, we present results for the RegEx based classifiers.

As an alternative to deriving classifiers based on teacher-specified input, we wanted to see how well our methods performed when trained on actual, participant data. That is, when the threshold used in the classifiers to make the final decision was fit based on actual participant data. We call such participant data-trained threshold, the ideal threshold. This ideal threshold could only be computed when participant data is available, which is a major constraint when developing a new internship, as we pointed out earlier.

Figure 1 and 2 shows the precision and recall plot for increasing thresholds of LSA based and NN based similarity methods. These plots were obtained by comparing the model classifications to the manual classifications on the 199 participant entries. It is generally seen that whenever precision increases at a particular threshold, the recall decreases or vice versa. The point of intersection of the precision and recall for a particular classifier gives the ideal precision and recall—that is, the classifier has

balanced performance in terms of precision and recall. From the figure, it is clear that if we want fewer false negatives, for example, the value of the threshold should be increased. In such a case, the precision will be compromised. Therefore, the threshold should be chosen carefully not to compromise either precision or recall to an undesirable extent.

The results obtained with ideal and derived thresholds are summarized in Table 3. These data suggest that, for the ideal thresholds, the LSA based classifiers for core concepts C1 through C4 performed well with the lowest precision and recall value being 0.72. However, the NN based classifiers outperformed the LSA classifiers for all core concepts other than C2. LSA based models depend on the overlapping content words in phrases and the performance suffers in cases where the phrases contain out of vocabulary words. Out of vocabulary here means the LSA similarity relies on pre-built vocabulary from a large corpus that does not contain some of the words, such as proper nouns that are specific to Land Science. However, NN based similarity models rely on letter trigrams from a very large corpus, and every input phrase is converted to letter trigrams. Therefore, the NN based models are capable of capturing the semantics even when there are out of vocabulary words in the phrases or context of the phrases. Hence, the NN based classifiers are superior for these concepts. However, for C2, the NN based classifier lagged in performance by 2% in precision and recall compared to the LSA based classifier because the teacher samples used for C2 contained only short phrases with very few context words and some of the overlapping words in the phrases boosted LSA based classifiers. The classifier C5 performed poorly for both LSA and NN based classifiers.

Table 3. Precision and recall for ideal and derived thresholds for LSA based and NN based similarity method

Classifier		Threshold		Precision		Recall	
		I	**D**	**I**	**D**	**I**	**D**
C1	LSA	0.36	0.35	0.84	0.82	0.84	0.86
	NN	0.34	0.31	0.86	0.84	0.86	0.92
C2	LSA	0.80	0.05	0.80	0.57	0.80	1.00
	NN	0.52	0.07	0.78	0.57	0.78	1.00
C3	LSA	0.38	0.59	0.82	0.92	0.82	0.80
	NN	0.36	0.44	0.86	0.96	0.86	0.78
C4	LSA	0.56	0.11	0.72	0.64	0.72	1.00
	NN	0.46	0.05	0.74	0.64	0.74	1.00
C5	LSA	1.00	0.27	0	0.22	0	0.98
	NN	0.80	0.05	0	0.23	0	1.00

Note: **I**=ideal, **D**=derived.

For the LSA based classifiers, the highest precision using derived thresholds was 0.92 with recall of 0.80 for C3 and the lowest precision was 0.22 with recall of 0.98 for C5. As we saw with the derived thresholds, NN based classifiers generally outperformed their LSA based classifiers counterparts, with the exception of the recall for concept C3

The results in Table 3 suggest that a good threshold could be derived without participants' data. The high recall and precision using derived thresholds for concepts C1 and C3 suggest the possibility of assessing the core concepts in participant notebook entries with classifiers generated using only the teacher's sample responses. However, when compared to the results using the ideal thresholds, classifiers C2, C4 and C5 did not perform well; their derived thresholds differed largely from their ideal thresholds, and their precision and recall suffered. The relatively low derived threshold values for these concepts suggests that their associated examples, which were used to calculated the thresholds, were semantically dissimilar. Dissimilar examples for a given concept could imply an ill-defined concept and that the provided examples do not represent it well. Alternatively, dissimilar examples could imply a complex or varied concept that requires highly different examples to represent it fully. Because we cannot distinguish between these cases automatically, we plan in future work to set a best guess threshold of 0.5 in such cases.

Table 4. Performance of regular expression model

Concepts	Precision	Recall
C1	0.963	0.551
C2	0.640	1.000
C3	1.000	0.746
C4	0.791	0.890
C5	0.894	0.739

Table 4 shows the precision and recall of RegEx based classifiers. Here the performance for concepts C2, C4, and C5 is more interesting when we compare those values with the previously discussed result. For example, the precision and recall for C5 improved impressively with values 0.89 and 0.73 respectively, whereas in previous case those values were either undefined or 0 precision with recall 1. Furthermore, the precisions of C1 and C3 are high, however the recalls are relatively low. Qualitatively investigating these results suggested that participants entries expressed these concepts in a variety of ways that were not captured by the regular expression lists.

Given that we see improvements for some core concepts using the regular expression based approach, these results suggest that the teacher provided samples on which the similarity measures where based may not have included a variety of key terms that could indicate the presence or absence of these core concepts. Comparing the sample responses and the keywords provided revealed that the samples indeed did not contain many of the keywords in the list. In some cases, the keywords were synonyms or other instances of particular kinds of words provided in the sample responses. For example, in *Land Science*, there are sixteen stakeholders who give demands on zoning plans. The core concept C5, *stakeholder demands*, is meant to capture references to these 16 stakeholders in participant notebook entries. Examining the teacher provided samples, we found that only four stakeholders were covered, while the keyword list for the core concept mentioned all sixteen. We plan in future experiments to either ask teachers to provide enough samples to cover finite sets of semantic content such as this or to incorporate the provided keyword list into the semantic similarity methods as extra samples.

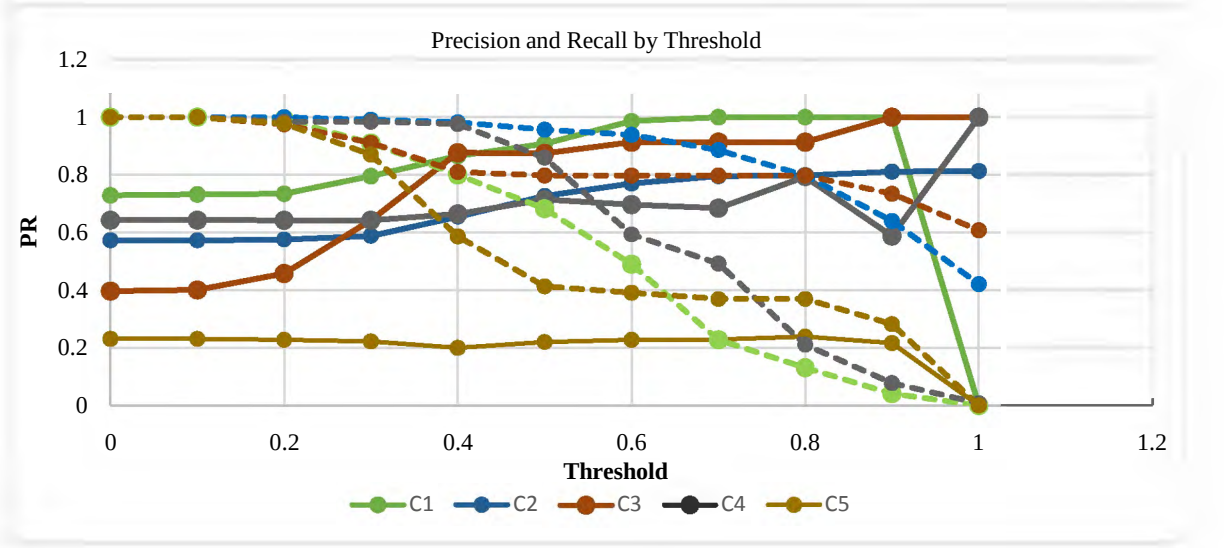


Figure 1. Precision and recall for LSA based similarity thresholds (solid lines are precision; dotted lines are recall)

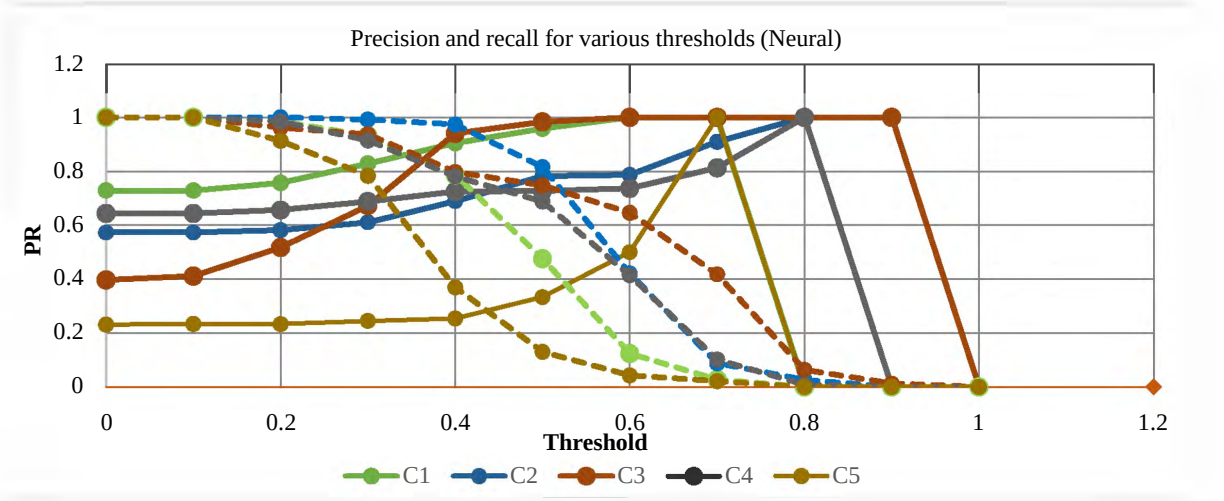


Figure 2. Precision and recall for neural network based similarity thresholds (solid lines are precision; dotted lines are recall)

5. CONCLUSIONS

In this paper, we investigated a method for creating classifiers for virtual internship notebook entries using teacher provided specifications without the use of participant data. Our classifiers used LSA based and NN based semantic similarity methods to capture the general semantic relationships among concepts. We also investigated regular expression based classifiers. The results are impressive in the sense that some classifiers, using both LSA and NN, gave high precision and recall values using thresholds derived without participant data, which suggests that our general method is plausible.

Furthermore, the superiority of the NN classifiers over the LSA classifiers suggests that NN methods are preferable when the participant responses vary widely in terms of style, content, and word overlaps with the teacher provided sample response.

The improved performance for some core concepts, such as C5, using regular expression based classifiers implies that such classifiers performed better for concepts whose sample responses did not contain a variety of keywords, despite the benefits we saw for NN models. These results suggest that, in some cases, teachers may need to provide more exhaustive samples, and that provided keywords and regular expression based classifiers may supplement a semantic similarity approach.

In future work, we will investigate a method to combine the classifiers in order to better understand how performance of one model is boosted by another in the scenario where participants responses vary widely compared to the sample responses. We will also see how the performance be affected by setting up the thresholds to 0.5 for concepts C2, C4 and C5.

Our work has several limitations; most obviously, we used participant data in to evaluate the performance of some of our classifiers. In the real use case of our method, we cannot expect to have such data available. We want to make clear, however, that our purpose in using participant data was not to train better classifiers, but to evaluate our method for generating them. Thus, our results suggest that this method can produce classifiers that would perform well on unseen data, but more refinements are needed.

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] Cai, Z., Graesser, A. C., Forsyth, C., Burkett, C., Millis, K., Wallace, P., ... & 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 systems* (pp. 429-433).

[2] Corley, C., & Mihalcea, R. (2005). Measuring the Semantic Similarity of Texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*. Ann Arbor, MI.

[3] Dikli, S. (2006). An Overview of Automated Scoring of Essays. *Journal of Technology, Learning, and Assessment, 5*(1).

[4] Fernando, S. & Stevenson, M. (2008). A semantic similarity approach to paraphrase detection, In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics* (pp. 45-52).

[5] 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

[6] Herrenkohl, L. R., & Cornelius, L. (2013). Investigating elementary students' scientific and historical argumentation. *Journal of the Learning Sciences*, *22*(3), 413–461.

[7] Huang, P. S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013, October). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (pp. 2333-2338). ACM.

[8] Leacock, C., and Chodorow, M. 2003. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities, 37*(4), 389-405.

[9] Lintean, M. C., & Rus, V. (2012, May). Measuring Semantic Similarity in Short Texts through Greedy Pairing and Word Semantics. *In FLAIRS Conference*.

[10] Rus, V., Gautam, D., Swieki, Z., & Shaffer, D. W. (2016, June). Assessing Student-Generated Design Justifications in Virtual Engineering Internships. In *Educational Data Mining* 2016.

[11] Rus, V., Lintean M., Graesser, A.C., & McNamara, D.S. (2009). Assessing Student Paraphrases Using Lexical Semantics and Word Weighting. *In Proceedings of the 14th International Conference on Artificial Intelligence in Education*, Brighton, UK.

[12] Rus, V., Lintean, M. C., Banjade, R., Niraula, N. B., & Stefanescu, D. (2013, August). SEMILAR: The Semantic Similarity Toolkit. In *ACL (Conference System Demonstrations)* (pp. 163-168).

[13] Salton, G., and Lesk, M. 1971. Computer evaluation of indexing and text processing. Prentice Hall, Ing. Englewood Cliffs, New Jersey. 143–180.

[14] Shaffer, D. W. (2006). *How Computer Games Help Children Learn*. Macmillan.

[15] Shaffer, D. W., Ruis, A. R., & Graesser, A. C. (2015). Authoring Networked Learner Models in ComplexDomains. In *Design recommendations for intelligent tutoring systems*, 179.

[16] Shermis, M.D. & Burstein, J. (2003). *Automated Essay Scoring: A Cross Disciplinary Perspective*. Lawrence Erlbaum Associates, Mahwah (2003).

[17] Shen, Y., He, X., Gao, J., Deng, L., & Mesnil, G. (2014, November). A latent semantic model with convolutional-pooling structure for information retrieval. *In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (pp. 101-110). ACM.

[18] Swiecki, Z., Midsfelt, M., Stoddard, J., Shaffer, D.W., (in press). Dependency-Centered Design as an Approach to Pedagogical Authoring. InBaek, Y. (Ed.) *Game-Based Learning: Theory Strategies and Performance Outcomes*. Hauppauge, NY: NOVA.

Convolutional Neural Network for Automatic Detection of Sociomoral Reasoning Level

Ange Tato
Université du Québec à Montréal

Roger Nkambou
Université du Québec à Montréal

Aude Dufresne
Université de Montréal

Miriam H. Beauchamp
Université de Montréal

ABSTRACT

We propose a model that employs convolutional neural networks (CNN) to evaluate sociomoral reasoning maturity, a key social ability, necessary for adaptive social functioning. Our model is used in a serious game to evaluate learners. It uses pre-annotated textual data (verbatims) and a coding scheme (SoMoral) applied by experts in psychology. State of the art text classification algorithms (Support Vector Machine, Naïve Bayes, etc.) achieved low results in our context in contrary to the CNN that achieved best results with little fine tuning on the input data representation. We use a simple but efficient input data vectors representation learnt directly from the dataset without loosing the sentences 'semantic'. We present a series of experiments with 5 baseline text classification algorithms and 4 baseline data representation. The results show that our model can predict the level of sociomoral reasoning with about 92% of accuracy. Our findings allow not only to advance the text-mining field but also the user modeling in highly social adaptive systems.

Keywords: Convolutional neural networks, data vectors representation, text classification, moral reasoning, social skills, serious game, learner model.

1. INTRODUCTION

Sociomoral reasoning (SMR) is a socio-cognitive construct essential for appropriate decision-making in social contexts, as well as for social adaptation. It is commonly defined as how individuals think about moral emotions and conventions that govern social interactions in their everyday lives [2]. The ability to predict and identify individual's sociomoral reasoning maturity level is a key step to quantifying peoples' social functioning and can be used to identify those at-risk for maladaptive social behaviour and orient them towards appropriate services. We propose a model and a simple input data representation for predicting the level of SMR maturity of an individual based on the justifications they provide when solving sociomoral dilemmas. A computerized test was designed, the Socio-Moral Reasoning Aptitude Level (So-Moral), in which children and adolescents are presented with visual social dilemmas from everyday life and asked to determine how they would react and provide a justification for their answer [21]. A serious game was designed based on the original tasks, and our model was designed to evaluate subjects using existing verbatims and scoring by experts that use the moral maturity coding scheme inspired by a cognitive-developmental approach [7]. The proposed model can be seen as a supervised text classification task.

Text classification is the task of automatically assigning classes to sentences or documents. There exist several supervised classification algorithms that have achieved good results in text classification tasks (Sentiment analysis [15, 19], topic mining [5], etc.) such as Support Vector Machine (SVM), Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA) or MLP (Multilayer perceptron) [1]. While their primary use has been in image classification and speech recognition, deep learning techniques (such as Convolutional Neural Networks) have recently been used for text classification and have achieved remarkable results [8, 11, 23]. A text document is characterized by the words it contains, and consequently the representation of textual data is only based on its words [10]. Thus, an important feature in text classification is the word vector representation of input data. Bag-of-words (BoW) vectors representation is the simplest and most widely used representation where vectors indicate which words appear in the documents without preserving word order. Vectors from BoW lack semantics and are usually huge and sparse. Alternative solutions have been proposed such as n-gram models [18] (bi-gram, tri-gram, etc.), word2vec or wordnet. However, to be effective, models that use n-gram or word2vec require a huge dataset and sentences or words that are frequently observed. Similarly, the use of wordnet is language dependant.

To benefit from word order and the annotated dataset, we built our classifier using CNN and a simple but effective data representation approach called *class-based representation (CBR)*. CNNs are neural networks with layers representing convolving filters applied to local features [12]. The application of CNN on text classification makes use of the 1D structure (word order) of text data so that each unit in the convolution layer responds to a small region of a document (a sequence or pattern of words) [8]. CNN can extract deep features from data which can improve discriminate classes.

1.1 The *Les Dilemmes* serious game

One of the objectives underlying the development of the proposed CNN is to implement the automated scoring mechanism in a serious video game called *Les Dilemmes*. It is a first-person serious game which aims to assess and train the social reasoning skills of the player. It is a virtual environment offering an interactive context which is emotionally, socially and cognitively rich. Players face different socio-moral dilemmas in a 3D environment in which they have to make decisions and are asked to provide oral justifications for the choices they make. They can also ask the opinions of virtual friends (non-player characters) in the game. Their answers are selected from previous recorded verbatims from the different moral maturity levels according to the coding scheme (SoMoral [2]). The learner (player) model implemented in the learning environment includes 3 keys dimensions: the affective state, the cognitive profile and, the sociomoral reasoning profile. Therefore, sociomoral reasoning skill is part of the player model

implemented in the game. As stated in [3], a learner model that can accurately represent the learner longitudinally in a game leads to efficient adaptation, which in turn helps increase player satisfaction and his motivation. To this end, it is important to ensure the effectiveness of the learner model before deploying the system for real uses.

Through this work, we aim to build an effective model of the sociomoral facet of the player. The level of sociomoral reasoning of an individual is determined from its verbal justifications provided when solving the dilemmas. This involves the implementation of a model for automatic measurement of this level during the game. We have a dataset of verbatims coming from the SoMoral experimentation already annotated by experts and a description (a paragraph with key concepts) associated with each different level (or class) of maturity. This paper aims to propose a machine learning model that can accurately assess the sociomoral reasoning skill level of a player based on his verbatim. In our knowledge, there is no research that deals with the automatic classification of sociomoral reasoning skills as part of learner-player social behaviour in serious games.

1.2 Sociomoral reasoning skill levels

The original So-Moral task includes five different levels of sociomoral reasoning [2]: (1) Authoritarian-based consequences, (2) Egocentric exchanges, (3) Interpersonal Focus, (4) Societal Regulation and (5) Societal Evaluation. Transition levels (i.e. 1.5, 2.5, 3.5, 4.5) are used to account for verbatims that provide elements of two reasoning stages and show a sequential progression from one stage to another. Occasionally, a verbatim is assigned to two different closed levels (1 being the maximum deviation) when two independent experts annotate the data for rater reliability purposes.

1.3 Dataset

The dataset consists of a benchmark of 691 verbatims (in French) manually coded by experts. Verbatims are short or long text fragments containing at least one sentence. They are not equally distributed between levels. Table 1 shows the repartition of data where for example levels 4.5 and 5 have a smaller number of verbatims than other levels. Level 5 constitutes the highest level of maturity and it is therefore more rarely attributed to children and adolescent's socio-moral justifications. This implies that certain levels have very few examples to learn from. Of the 691 verbatims, 53 were classified as 0, which means that the verbatim does not represent one of the sociomoral reasoning levels (e.g., the answer provided by the participant was tangential and did not contain a justification of their social response). We do not consider these cases in our study, which reduces our corpus to 638 verbatims.

Table 1. Distribution of verbatims between levels

Class	Freq.	%	Class	Freq.	%
1	232	36.36	**1.5**	11	1.8
2	76	11.92	**2.5**	29	4.6
3	207	32.44	**3.5**	31	4.86
4	40	6.3	**4.5**	3	0.48
5	9	1.5			

2. BASELINE METHODS FOR SENTENCE CLASSIFICATION

Since verbatims are annotated text data, we investigated the use of some existing sentence classification algorithms. In this section, we expose state of the art methods for text classification that have shown good results on similar problems.

2.1 Input representation

Here, we present some representation techniques that we experimented on for determining sociomoral reasoning level.

Bag-of-words (BoW): BoW is a binary word presence representation (indicating whether a word is present or not in a sentence). Each distinct word in the dataset corresponds to a feature in the representation. Each labeled verbatim in the dataset is transformed to a vector of N columns, where N (the vocabulary size) is the total number of distinct words in the entire corpus.

Matrix Tf-idf: Tf-idf (Term Frequency-Inverse Document Frequency) representation allows evaluation of the importance of a term contained in a document relative to a collection. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps adjust for the fact that some words appear more frequently in general.

Dictionary of synonyms: We developed a tool to compare our representation model with an approach similar to that of wordnet and to make use of the concept lists from each level provided by experts). The tool takes two words, and for each word, extracts a set of synonyms from a free access online French synonyms database and then computes the intersection of those sets to determine whether the two words are related or not. The So-Moral scoring manuel provides a description of what types of justifications should be included at each level and a list of concepts that describe each level. This information was used by extracting keywords (we removed stop words). After this process, we obtained a list of 53 words representing all the levels, which are used as a vocabulary set for the data. Each word is represented by a vector of size 1*53. For each word from a verbatim and for each word from the vocabulary list, if the intersection of vectors is not null, then it is given a code of 1, otherwise, it is coded 0.

Word2vec: It is common in sentence classification to use publicly available word2vec vectors that are trained on over 100 billion words from Google news [11]. This technique usually works with sentences in English. Instead of directly using those pretrained vector representations, others try to learn those vectors directly from their dataset. We also attempted to represent our data with word2vec vectors that were trained on our corpus.

2.2 Supervised classification algorithms

There exist several supervised classification algorithms. Among them, we selected ones that generally produce excellent results in text classification.

SVM (Support Vector Machine): The learning algorithm consists of finding a hyperplane, which separates the levels appropriately by limiting the error rate of classification in the new data. The aim is to maximize the distance of the vectors close to the hyperplane for each of the levels, which avoids overfitting. Although this algorithm is more suited to binary class problems, the aim was to explore its behavior on our

dataset since it generally provides good outcome on text classification [1, 6].

NB (Naïve bayes): NB is a probabilistic classifier based on the Bayes theorem with a naive assumption of attribute independence [17]. It is generally used in the detection of spam, sentiments analysis and in the medical field. The principle is to compute the posterior probability of the class for a given document, and the class with the highest posterior probability is then assigned to the document. We chose to experiment with NB because it is fast [16] and easy to implement especially in real-time applications.

LSA (Latent Semantic Analysis): LSA is an algorithm that has been developed specifically for mining textual data. This algorithm allows us to take into account semantics, which very few algorithms offer. It is an interesting technique because it does not consider any information related to language processing (meaning of words, dictionaries etc.). This makes it possible to establish relations between a set of documents and the terms it contains by constructing "concepts" related to documents and terms [13, 20].

LDA: LDA is a machine learning technique that has revolutionized the extraction of latent subjects in texts [4]. It tries to create topic clustering of documents that are similar to each other. Each document is represented as a mixture of topics. We trained the LDA model on our 638 verbatims by setting the number of topics to 5 or 9 (depending on the problem). The classification of a new verbatim was achieved by computing the cosine similarity between the verbatim and each of the topic probability distribution vectors over words.

MLP (Multilayer Perceptron): MLP is a feedforward artificial neural network model with one or more layers between hidden layers that maps sets of input data onto a set of appropriate outputs. MLPs are widely used for pattern classification, recognition, prediction and approximation. MLPs are able to learn non-linear models, but require tuning of a number of hyperparameters such as the number of hidden neurons, layers, and iterations.

3. THE PROPOSED MODEL

3.1 Class-based Representation (CBR)

The poor distribution of verbatims over levels (see Table 1) and the fact that some of the keywords that can aid in the discrimination of levels appear just once or twice in the entire corpus, makes the application of some data representation techniques inaccurate (e.g. BoW, tf-idf). Also, verbatims are sentences that are semantically very rich and of varying sizes; state-of-the-art techniques often fail to accurately classify this type of data (see Section 5 for details).

We propose a simple yet fast and efficient representation model for data that use only an annotated dataset. Using this technique, we gain over 10% accuracy compared to all other classification techniques previously presented, and over 30% accuracy compared to some state-of-the-art representation models. A further advantage of the proposed representation is that it is not language dependent. It does not consider any information related to language processing (meaning of words, dictionaries etc.), which can be time consuming.

We represented each verbatim as a feature vector, whose values (1 or 0) accounted for the presence of a word in a level. The idea of CBR is simple: if a word appears in the verbatim of one level, then it must be semantically correlated with that class. In turn, if a word appears in verbatims from different classes, then it must be semantically correlated with all the classes, but has less significance than a word that appears only in verbatims from one of those levels. For example: we have 4 levels, and we have 2 verbatims from level1 and level 2 (see Table 2a); Table 2b shows the representation of two words in this specific case (1 means semantically correlated and 0 means uncorrelated).

Table 2. a) Examples of two verbatims

Verbatim	Class
*Parce que c'est mal et elle n'apprendrait pas de ses erreurs (*Because it's wrong and she won't learn from her mistakes)	3
C'est tricher (It's cheating)	1

b) Examples of CBR on the 2 verbatims from a).

Words\Classes	1	2	3	4
est	1	0	1	0
erreurs	0	0	1	0

The input data for the CNN model is a matrix with 5 columns and 88 lines, which correspond to the length (number of words) of the longest sentence of the corpus after data pre-processing.

3.2 The CNN Model

According to LeCun and colleagues [14], deep learning allows computational models composed of multiple layers of processing to learn data representations at multiple levels of abstraction. Deep learning techniques such as CNN have been shown to be effective for Natural Language Processing (NLP) and have achieved excellent results on sentence classification [11, 24], sentence modeling, and semantic parsing [9]. They can explore small text regions to learn useful features for categorization [8]. The CNN we are proposing requires as input a vector representation (88*5 or 88*9) of verbatims that preserves the internal order of words, as in class-based representation.

Parameter selection: The hyper-parameters of our CNN, such as the size of filters and the number of layers, were chosen based on the results obtained empirically from several tests on our dataset. The structure of the CNN consists of two layers of *convolution*, two layers of *maxpooling* and one layer fully *connected* to the output. The fully connected layer of our model uses 40 rectified linear units. The structure also includes two Filter windows, one of size 1x5 for the 5-level classifier (1x9 for the 9-level classifier) and the other of 2x1 in size. The first filter window is used to implement the convolution on the input data. Using a 1-dimension window here allows exploration of the data one word at a time in order to derive specific features associated with each word (which contributes to determining the semantics of the word). Following this step and the maxpooling of its output, another filter is used for a second convolution. This second convolution aims at extracting features related to word order (or text regions). A filter vector of 2x1 (for exploring the text regions) is used for this purpose. There are 20 filters in each convolutional layer. The *batchsize* was set to 500 and the number of iterations to 250.

4. EXPERIMENTS

Our experiments involve the five classification algorithms, Naive Bayes (NB), LSA (Latent semantic Analysis), LDA

(Latent Dirichlet Allocation), MLP (Multi Layer Perceptron) and Support Vector Machine (SVM) that we presented earlier in this paper. The goal of using all these algorithms is to compare the models obtained from them with that obtained from the CNN-based model. We explored existing input representations of data and compared results with the CBR.

4.1 Data pre-processing

For consistency between different input representations and algorithms, we used the same pre-processing steps for the data.

Stop-word removal: Generally, the very first step to reduce the vector size of the data is to remove stop-words (connective words, such as "a", "in", "the" in English). Alone, they are considered lacking semantic to give information to the classifier [1]. Unfortunately, the typical list of stop words for the French available online gave poor results in our classification task. Instead, we excluded common words in the verbatims, which were not discriminatory for the different SMR levels.

Lemmatization: This is the process of mapping words onto their base form [10]. For example, the words "installed", "installs" and "installing" are mapped to "install". This mapping makes the binary presence of word representation approaches treat words of different forms as a single feature, hence reducing the total number of features. We used the Stanford NLP tools to apply a French lemmatization to the verbatims.

4.2 Results

Accuracy is typically used as the standard measure for classification performance. However, for datasets with an unbalanced distribution such as the one used here, this measure can be illusory and not very informative about the errors being committed by the classifier. Instead of relying solely on accuracy, we used the F1 score (or F-measure) which takes into consideration both precision and recall. To provide a point of reference for our CNN model results using our proposed input representations, we first report the performance achieved using baseline techniques for sentence classification. We report Accuracy and F1-score over all datamining techniques and datasets in Tables 3 and 4. First, we used only the BoW and tf-idf representations as input representation for the algorithms. SVM was run with the RBF (Radial Basis Function) as kernel function. LSA is an algorithm which initially works with tf-idf representation, that is why we have the n/a (not applicable) mention. Table 3 shows the results. For a second experiment, we used dictionary of synonyms and the *CBR* representation as input representation techniques for the MLP and the CNN. We have kept only those 2 algorithms for the next step because of their good results compared to others on step 1.

Table 4 shows results. **Erreur ! Source du renvoi introuvable.**Figure 1 graphically shows the performance of MLP and CNN on both the dictionaries synonyms and *classes-based* techniques and on the 2 types of problems previously mentioned in section 2 (5 and 9 classes).

For all the algorithms, we trained the models on 75% of data (which is about 500 verbatims) and we tested on the remaining verbatims (138 verbatims).

5. DISCUSSION

We begin our discussion by looking at the most basic representations, those involving the BoW and the tf-idf (Table 3). We note that none of the 5 baseline algorithms were able to classify at least the half of the data with the BoW representation technique. Only NB and MLP were able to classify more than 50% with tf-idf. However, the F1 score remains relatively low in general. Furthermore, for SVM, LSA and LDA, all the verbatims in the test data were classified as level 1. For NB, they were classified into levels 1 and 3. The reason for these misclassifications can be seen in Table 1, where levels 1 and 3 are the most represented in the dataset. This brings us to the conclusion that those 2 representations depend strongly on the distribution of the data into classes. Despite the time-consuming learning, CNN and MLP gave the best results.

Table 3. Accuracy and F1 scores of 6 baseline algorithms for sociomoral reasoning level classification. The input data are represented with BoW and Tf-idf techniques.

Input representation	Measure	SVM	NB	LSA	LDA	MLP	CNN
BoW	**Accuracy**	43.00	49.28	N/A	38.96	49.91	**49.98**
	F1-score	12.71	**30.55**	N/A	25.24	28.32	29.18
Tf-idf	**Accuracy**	43.27	59.9	31.05	48.36	60.25	**63.00**
	F1-score	18.54	**46.81**	15.4	44.00	45.79	37.68

Table 4. Accuracy and F1 score of the CNN model and MLP for sociomoral reasoning level classification. The input data are represented with class-based and dictionary of synonyms techniques.

Input representation	Measure	MLP (5 classes)	CNN (5 classes)	MLP (9 classes)	CNN (9 classes)
Dictionary of synonyms	**Accuracy**	66.33	**71.25**	44.00	52.00
	F1-score	60.4	54.34	36.21	**64.37**
CBR	**Accuracy**	75.00	**85.8**	56.60	82.60
	F1-score	67.76	**83.76**	37.09	74.8

Table 5. Results of the CNN model and MLP with errors margins (1 for the 5 classes and 0.5 or the 9 classes).

Input representation	MLP (5 levels)	CNN (5 levels)	MLP (9 levels)	CNN (9 levels)
CBR	84.28	**92.00**	63.52	**84.00**

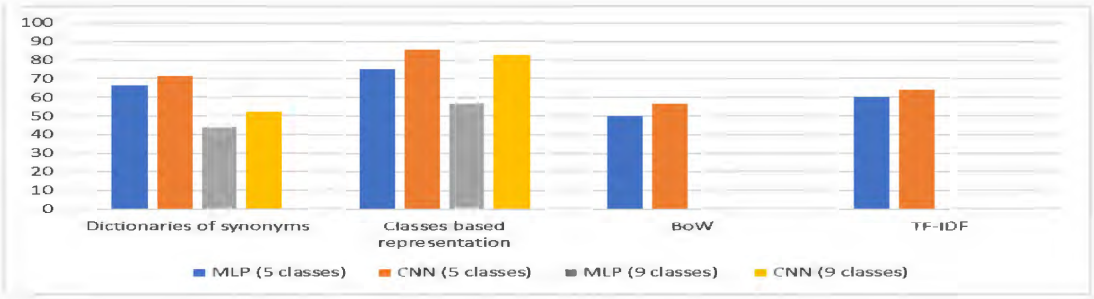


Figure 1. Variation of the accuracy of MLP and CNN, based on input data representation techniques.

In

Table 4, we ran our CNN model and MLP with the dictionary of synonyms and class-based techniques. We also considered the 5 and 9 levels problem. At first glance, we see that the CBR gives the best results compared to other representation techniques. The best result was obtained from our CNN model. The model provided 85% accuracy and 83% F1 score, which is an acceptable result for the problem. The training of the CNN took more time than other techniques because we needed to find the parameters that achieved the best results. We limited the number of iterations to 250 to avoid overfitting. Over the 250 iterations, we obtained poorer classification results on test data, but over 99% on training data. We also note that the results vary considerably based on the training set, suggesting that selection of the training set is an important part in the pre-processing of data for the CNN model.

Why does CNN give the best results?

The size of filters (number of lines) in the CNN can be compared to the idea behind N-Grams. The convolution is done on 2, 3 or 4 words at a time if the filters are respectively of size 2, 3 and 4. So, the CNN takes into account the order of the words in sentences. Another reason for the better results with the CNN compared to other techniques is that it can extract deep features (e.g., semantically grounded) using a series of convolutions, filters, feature maps and pooling on data, which help in the discrimination of data. The input data representation also contributes to this performance.

Real-life sociomoral reasoning classification

In manual scoring of socio-moral reasoning, different experts occasionally associate the same verbatim with different levels because of inherent variability between even expert raters. Taking into consideration that even experts can make errors, we retrained our model (on both 5-level and 9-level problems) by considering a margin error of 1 for the 5-level problem and of 0.5 for the 9-level problem. For example, if the model predicts that the level of verbatim v1 is 1 and that the real level is 1.5, then it is considered as a true classification.

Table 5 shows the results when error margins are considered. We can see that the CNN on the 5-level problem achieves exceptional results with an accuracy of 92%, which is the best so far.

6. CONCLUSION

We propose a model able to predict with over 90% accuracy the sociomoral reasoning skill level based on a textual verbatim. Specifically, we propose a simple but efficient input text data representation that can work with different classification algorithms. This work is a considerable contribution in sentence classification and in sociomoral reasoning maturity classification. Verbatims are typically manually annotated by experts. Our proposed model is intended to help them in this task and produces results that are comparable to the accuracy of independent raters, suggesting promising applications.

Contrary to state-of-the-art techniques in text classification, the CNN model we propose achieves the best results in our context. This is mainly due to its deep structure that can learn useful features from data. Despite the good results obtained by the CNN, parameters must be manually tuned and require many experiments to find the best results. MLP can be treated as a lexical mining technique on text, because all neurons on hidden layers receive information from all previous neurons (blind mining). The order or the meaning of words is not considered. On the other hand, CNN can capture deep features from data and thus the order (pattern or syntax mining) and the meaning (semantic mining) of words, if the representation is good enough. Since a sentence is fully defined by its syntax, lexis and semantics, a model considering those features will lead to better results in sentence classification and even NLP tasks. In our future works, we will develop a model based on a pooling of MLP and CNN techniques. We will also consider the use of the multiple channels features of CNN to combine different representation of sentences as reported by Kim [11] and Yin [22]. Similarly, while more complex data representations for text classification will undoubtedly continue to be developed, those deploying such technologies in real-life problems will likely be

attracted to simpler variants, which afford fast training and prediction times such as the CBR model that we propose. The only downside of our representation approach is that it requires a classified dataset. We will explore the combination of class-based approach and others interesting representation techniques that use RBM (Restricted Boltzmann Machine) or autoencoders in future work, in order to achieve 90% accuracy without adjustment for error margins. The proposed coding solution will be implemented in *the Les Dilemmes video game*. The next step will be the assessment of the efficiency of the sociomoral reasoning dimension as a learner model facet in a highly adaptive social serious video game.

7. ACKNOWLEDGMENTS

We would like to thank our colleagues from ABCs research lab who provided us with the So-Moral data.

8. REFERENCES

[1] Aggarwal, C.C. and C. Zhai, *A survey of text classification algorithms*, in *Mining text data*. 2012, Springer. p. 163-222.

[2] Beauchamp, M., J.J. Dooley, and V. Anderson, *A preliminary investigation of moral reasoning and empathy after traumatic brain injury in adolescents.* Brain injury, 2013. **27**(7-8): p. 896-902.

[3] Birk, M.V., et al. *Modeling Motivation in a Social Network Game using Player-Centric Traits and Personality Traits*. in *International Conference on User Modeling, Adaptation, and Personalization*. 2015. Springer. p. 18-30.

[4] Blei, D.M., A.Y. Ng, and M.I. Jordan, *Latent dirichlet allocation.* Journal of machine Learning research, 2003. **3**(Jan): p. 993-1022.

[5] Chen, Z. and B. Liu. *Mining topics in documents: standing on the shoulders of big data*. in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2014. ACM. p. 1116-1125.

[6] Gautam, G. and D. Yadav. *Sentiment analysis of twitter data using machine learning approaches and semantic analysis*. in *Contemporary computing (IC3), 2014 seventh international conference on*. 2014. IEEE. p. 437-442.

[7] Gibbs, J.C., *Moral development and reality: Beyond the theories of Kohlberg, Hoffman, and Haidt*. 2013: Oxford University Press.

[8] Johnson, R. and T. Zhang, *Effective use of word order for text categorization with convolutional neural networks.* arXiv preprint arXiv:1412.1058, 2014.

[9] Kalchbrenner, N., E. Grefenstette, and P. Blunsom, *A convolutional neural network for modelling sentences.* arXiv preprint arXiv:1404.2188, 2014.

[10] Khoo, A., Y. Marom, and D. Albrecht. *Experiments with sentence classification*. in *Proceedings of the 2006 Australasian language technology workshop*. 2006. p. 18-25.

[11] Kim, Y., *Convolutional neural networks for sentence classification.* arXiv preprint arXiv:1408.5882, 2014.

[12] Krizhevsky, A., I. Sutskever, and G.E. Hinton. *Imagenet classification with deep convolutional neural networks*. in *Advances in neural information processing systems*. 2012. p. 1097-1105.

[13] Landauer, T.K., *Latent semantic analysis*. 2006: Wiley Online Library.

[14] LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning*. Nature, 2015. **521**(7553): p. 436-444.

[15] Maas, A.L., et al. *Learning word vectors for sentiment analysis*. in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. 2011. Association for Computational Linguistics. p. 142-150.

[16] Narayanan, V., I. Arora, and A. Bhatia. *Fast and accurate sentiment classification using an enhanced Naive Bayes model*. in *International Conference on Intelligent Data Engineering and Automated Learning*. 2013. Springer. p. 194-201.

[17] Rish, I. *An empirical study of the naive Bayes classifier*. in *IJCAI 2001 workshop on empirical methods in artificial intelligence*. 2001. IBM New York. p. 41-46.

[18] Sidorov, G., et al., *Syntactic n-grams as machine learning features for natural language processing.* Expert Systems with Applications, 2014. **41**(3): p. 853-860.

[19] Tang, D., et al. *Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification*. in *ACL (1)*. 2014. p. 1555-1565.

[20] Tougas, J.E. and R.J. Spiteri, *Updating the partial singular value decomposition in latent semantic indexing.* Computational Statistics & Data Analysis, 2007. **52**(1): p. 174-183.

[21] Vera-Estay, E., et al., *All for One: Contributions of Age, Socioeconomic Factors, Executive Functioning, and Social Cognition to Moral Reasoning in Childhood.* Frontiers in psychology, 2016. **7**.

[22] Yin, W. and H. Schütze. *Multichannel variable-size convolution for sentence classification*. in *Proceedings of the Conference on Computational Natural Language Learning*. 2015. p. 204-214.

[23] Zhang, X. and Y. LeCun, *Text understanding from scratch.* arXiv preprint arXiv:1502.01710, 2015.

[24] Zhang, Y. and B. Wallace, *A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification.* arXiv preprint arXiv:1510.03820, 2015.

A Latent Factor Model For Instructor Content Preference Analysis

Jack Z. Wang
Rice University
jzwang@rice.edu

Andrew S. Lan
Princeton University
andrew.lan@princeton.edu

Phillip J. Grimaldi
Rice University
phillip.grimaldi@rice.edu

Richard G. Baraniuk
Rice University
richb@rice.edu

ABSTRACT

Existing personalized learning systems (PLSs) have primarily focused on providing learning analytics using data from learners. In this paper, we extend the capability of current PLSs by incorporating data from instructors. We propose a latent factor model that analyzes instructors' preferences in explicitly *excluding* particular questions from learners' assignments in a particular subject domain. We formulate the problem of predicting instructors' question exclusion preferences as a matrix factorization problem, and incorporate expert-labeled Bloom's Taxonomy tags on each question as a factor in our statistical model to improve model interpretability. Experimental results on a real-world educational dataset demonstrate that the proposed model achieves superior prediction performance compared to several other baseline methods commonly used in recommender systems. Additionally, by explicitly incorporating Bloom's Taxonomy, the model provides meaningful interpretations that help understand why instructors exclude certain questions. Since instructor preference data contains their insights after years of teaching experience, our proposed model has the potential to further improve the question recommendations that PLSs make for learners.

Keywords

personalized learning, educational data mining, latent factor model, Bloom's Taxonomy

1. INTRODUCTION

Today's education system has largely remained a "one-size-fits-all" learning experience in which the instructor selects a single learning action for all learners, ignoring their diverse backgrounds, interests, and goals. Modern machine learning (ML) techniques have led to a great acceleration in the development of personalized learning systems (PLSs) that have the potential to revolutionize education by delivering a high-quality and affordable personalized learning experience at large scale.

Current PLSs generally perform learning analytics using only learner data, overlooking data that instructors generate. However, when instructors are present in educational settings such as traditional classrooms, they generate important data that reveals how they prefer to interact with learning resources. Augmenting current learning analytics approaches by modeling instructors' preferences clearly provide advantages, since their preferences reflect years of teaching experience and thus provide valuable insights on how to utilize learning resources effectively. As a result, PLSs can refine their learning resource recommendations for learners using both learner data and these valuable insights. Additionally, analysis of instructor preferences for learning resources can serve as a starting point of recommending learning resource to learners when learner data is scarce such as at the beginning of a semester.

In this work, we focus on a specific instance of instructors' content[1] preferences. We collect instructors' preferences to exclude questions from being given to learners in their class via OpenStax Tutor[13], a personalized learning and teaching platform. OpenStax Tutor has a functionality to automatically *select* homework assignment questions for learners from a question corpus. At the same time, it allows instructors to *exclude* questions they do not want OpenStax Tutor to assign to learners in their classes from the corpus. While this exclusion option allows more flexibility for instructors to control homework assignment questions that learners receive, manually selecting questions to exclude from a (potentially huge) corpus is a labor-intensive process. As a result, analyzing instructors' question exclusion behavior has immediate utility in automating the question exclusion process.

1.1 Contributions

With the objective of analyzing instructors' preferences on assigning questions to learners on the OpenStax Tutor platform, we develop a novel latent factor model that predicts instructors' question preferences in a particular subject domain given previous records of whether instructors choose to *exclude* certain questions from homework assignments. The latent factor modeling approach is primarily inspired by SPARFA [10] which is a successful latent factor model for learner and content analysis. But more importantly, this approach allows flexible incorporation of prior knowledge in the form of meta-data into the model. Consequently, the model that we develop in this work can be easily extended to include additional information in the form of latent factors to explain instructors' question exclusion preferences,

[1]From now on, we will use the phrase "learning resources" and the word "content" interchangeably.

as well as be used in other educational data mining tasks where auxiliary information is available. Additionally, our proposed model incorporates expert-labeled Bloom's Taxonomy tags for each question to explain instructors' question exclusion preferences, based on the conjecture that instructors have varying inclinations towards different Bloom's Taxonomy tags[2].

Experimental results on a real-world educational dataset show that, compared to standard methods used in recommender systems, our model achieves higher overall accuracy in predicting instructors' question preferences. Additionally, we demonstrate that our model is highly interpretable in that the Bloom's Taxonomy explains question preferences of individual instructors, and reveals question preference patterns among instructors. Our analysis of the instructors' question exclusion preferences enables PLSs to incorporate instructors' insights on questions and potentially improve the quality of their personalized question recommendations.

We emphasize that our proposed model is not limited to analyzing instructors' question exclusion preferences; it can be easily modified to analyze instructors' preferences on a broader range of learning resources. Therefore, our work serves as an initial investigation into extending the capability of existing PLSs with the analysis of instructor learning resource interaction data.

1.2 Related Work

We formulate the problem of predicting instructors' question preferences as a matrix factorization problem underlying a recommender system. Recommender systems often rely on collaborative filtering (CF); the two most successful family of CF approaches to date are neighborhood-based methods and latent factor methods [4]. Neighborhood-based methods predict preferences based on neighbors chosen by some similarity measure. Latent factor methods, in particular, can be readily applied to education applications, resulting in tensor factorization for student modeling [15] and probabilistic models such as SPARFA [10], a primary source of inspiration for this work. However, these approaches, in their original form, do not have mechanisms to incorporate meta-data on learners and questions. Therefore, the explanatory power of these methods is usually limited. Our proposed model, on the other hand, extends the original latent factor model to explicitly include the Bloom's Taxonomy tag of each question as meta-data, providing additional interpretability and, at the same time, improves prediction accuracy.

Works including [6] and [12] incorporate external factors such as movie genres to improve users' movie rating prediction in the Netflix challenge [2], but their methods do not directly apply to education scenarios.

The work in [14] broadly describes a Bayesian approach to model instructors. While our work pursues a similar objective, we propose a concrete model with evaluations on a real-world dataset instead of a high-level overview. [11] uses the k-means clustering algorithm to recommend learning resources for instructors based on similar teaching styles among instructors. In addition to studying question type preferences, we approach the problem with a latent factor model instead of k-means clustering, yielding results that are more interpretable.

The work in [16] compares several models in predicting learners' next-term grades using various features including instructors' job title, rank, and tenure status. Our work, on the contrary, uses data that contains instructors' direct interaction with learning resources rather than simple demographic information.

[2]Bloom's Taxonomy hierarchically describes questions in terms of one of the six cognitive processes, including remembering, understanding, applying, analyzing, evaluating, and creating, in increasing cognitive complexity [9]. It describes the cognitive processes by which learners encounter and work with knowledge [1].

2. LATENT FACTOR MODEL

Let N, Q, K denote the total number of instructors, the total number of questions, and the total number of distinct Bloom's Taxonomy tags, respectively. Let $\mathbf{Y}$ be the binary-valued matrix of dimension N by Q that represents instructors' preference for a particular course, where $Y_{ij} = 1$ indicates instructor i explicitly denotes preference to exclude question j, and $Y_{ij} = 0$ indicates no preference. Also let $\mathbf{a}_j$ be a vector of dimension K that represents the question–Bloom's Taxonomy tag association for question j, where a_{jk} denotes the kth component of $\mathbf{a}_j$. $a_{jk} = 1$ indicates an association of question j with Bloom's Taxonomy tag k, and $a_{jk} = 0$ indicates no association.

With the above setup, we model $\mathbf{Y}$ as Bernoulli random variables:

$$Y_{ij} \sim \text{Ber}\big(\phi(\mathbf{p}_i^T \mathbf{a}_j + \mathbf{g}_i^T \mathbf{h}_j)\big), \tag{1}$$

Where the function $\phi(\cdot)$ is the sigmoid function:

$$\phi(x) = \frac{1}{1 + e^{-x}}.$$

In the model, $\mathbf{p}_i \in \mathbb{R}^K$, $\mathbf{g}_i \in \mathbb{R}^M$, $\mathbf{h}_j \in \mathbb{R}^M$ are model parameters to be estimated, where M is the dimension of $\mathbf{g}_i$ and $\mathbf{h}_j$ (we select the value of M via cross validation). Intuitively, the latent factor $\mathbf{p}_i$ represents the instructor Bloom's Taxonomy tag preference vector that reveals instructors' different preferences on each Bloom's Taxonomy tag. The latent factors $\mathbf{g}_i$ and $\mathbf{h}_j$ model additional factors that also contribute to explaining the observed data matrix $\mathbf{Y}$.

To compare the significance of the factor $\mathbf{p}_i$ against the factors $\mathbf{g}_i$ and $\mathbf{h}_j$, we use two simplified variants of the full model in Equation 1, namely P Model that involves only the factor $\mathbf{p}_i$, and GH Model that involves only factors $\mathbf{g}_i$ and $\mathbf{h}_j$:

$$\text{P Model:} \quad Y_{ij} \sim \text{Ber}\big(\phi(\mathbf{p}_i^T \mathbf{a}_j)\big) \tag{2}$$

$$\text{GH Model:} \quad Y_{ij} \sim \text{Ber}\big(\phi(\mathbf{g}_i^T \mathbf{h}_j)\big) \tag{3}$$

2.1 Optimization Algorithm

We formulate the maximum-likelihood parameter estimation problem for the proposed model as an optimization problem. The optimization objective is given by

$$\underset{\mathbf{P},\mathbf{G},\mathbf{H}}{\text{minimize}}\, f(\mathbf{P}, \mathbf{G}, \mathbf{H}),$$

where $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$ denotes the matrix of instructor Bloom's Taxonomy tag preference associations by stacking

the association vectors together. $\mathbf{G}$ and $\mathbf{H}$ are defined analogously. The cost function $f(\mathbf{P}, \mathbf{G}, \mathbf{H})$ is given by

$$f(\mathbf{P}, \mathbf{G}, \mathbf{H}) = \sum_{i=1}^{N}\sum_{j=1}^{Q} \log\Big(1 + \exp\big(-(\mathbf{p}_i^T \mathbf{a}_j + \mathbf{g}_i^T \mathbf{h}_j)\big)\Big) + \frac{\lambda}{2}\sum_{i=1}^{N} \|\mathbf{p}_i\|_2^2 + \frac{\gamma}{2}\sum_{i=1}^{N} \|\mathbf{g}_i\|_2^2 + \frac{\eta}{2}\sum_{j=1}^{Q} \|\mathbf{h}_j\|_2^2.$$

The last three terms in the cost function are regularization terms added to prevent overfitting. λ, γ, and η are regularization parameters for the factors $\mathbf{p}_i, \mathbf{g}_i, \mathbf{h}_j$, respectively.

The above optimization problem is non-convex, but the subproblems to optimize over each parameter while holding the others fixed are convex. We therefore employ block coordinate descent to efficiently find a local minima for the above optimization problem by iteratively updating each parameter in turn. The update equations for the parameters are given by

$$\mathbf{p}_i^{\text{new}} = \mathbf{p}_i^{\text{old}} - \delta \frac{\partial}{\partial \mathbf{p}_i} f(\mathbf{p}_i^{\text{old}}, \mathbf{g}_i^{\text{old}}, \mathbf{h}_j^{\text{old}})$$

$$\mathbf{g}_i^{\text{new}} = \mathbf{g}_i^{\text{old}} - \delta \frac{\partial}{\partial \mathbf{g}_i} f(\mathbf{p}_i^{\text{new}}, \mathbf{g}_i^{\text{old}}, \mathbf{h}_j^{\text{old}})$$

$$\mathbf{h}_j^{\text{new}} = \mathbf{h}_j^{\text{old}} - \delta \frac{\partial}{\partial \mathbf{h}_j} f(\mathbf{p}_i^{\text{new}}, \mathbf{g}_i^{\text{new}}, \mathbf{h}_j^{\text{old}}),$$

where δ is the step size. The gradients of the cost function with respect to each parameter are given by

$$\frac{\partial}{\partial \mathbf{p}_i} f(\mathbf{p}_i, \mathbf{g}_i, \mathbf{h}_j) = -\sum_{j=1}^{Q} \frac{\mathbf{a}_j}{1 + e^{-(\mathbf{p}_i^T \mathbf{a}_j + \mathbf{g}_i^T \mathbf{h}_j)}} + \lambda \mathbf{p}_i$$

$$\frac{\partial}{\partial \mathbf{g}_i} f(\mathbf{p}_i, \mathbf{g}_i, \mathbf{h}_j) = -\sum_{j=1}^{Q} \frac{\mathbf{h}_j}{1 + e^{-(\mathbf{p}_i^T \mathbf{a}_j + \mathbf{g}_i^T \mathbf{h}_j)}} + \gamma \mathbf{g}_i$$

$$\frac{\partial}{\partial \mathbf{h}_j} f(\mathbf{p}_i, \mathbf{g}_i, \mathbf{h}_j) = -\sum_{i=1}^{N} \frac{\mathbf{g}_i}{1 + e^{-(\mathbf{p}_i^T \mathbf{a}_j + \mathbf{g}_i^T \mathbf{h}_j)}} + \eta \mathbf{h}_j.$$

At the beginning of optimization, we randomly initialize the model parameters $\mathbf{p}_i$, $\mathbf{g}_i$, $\mathbf{h}_j$ for all i, j. In each optimization iteration, we first loop over all i's to update all $\mathbf{p}_i$ and $\mathbf{g}_i$ while holding all $\mathbf{h}_j$'s fixed, and then loop over all j's to update $\mathbf{h}_j$ using the newly calculated $\mathbf{p}_i$'s and $\mathbf{g}_i$'s. We repeat the above iterations until convergence, i.e., the difference of the cost function between two iterations falls below a predefined threshold.

Note that the inference problem for the P Model in Equation 2 is convex, and optimization is straightforward via gradient descent. Since the GH Model in Equation 3 involves two sets of parameters and has a non-convex inference problem, we employ the same block coordinate descent method as in the full model.

2.2 Model Extensions

We now enumerate possible extensions to the proposed model. First, we can incorporate additional prior information as latent factors in the model by simply including other modalities of meta-data as an additional inner product terms of two more latent factors inside the $\phi(\cdot)$ function. In this way, in each inner product term, one factor denotes the newly included meta-data modality, and the other characterizes the instructor's exclusion preference in terms of that specific modality of meta-data. Concretely, the extension of the model in Equation 1 has the following form:

$$Y_{ij} \sim \text{Ber}\left(\phi\left(\sum_{l=1}^{L} \mathbf{u}_i^{l\,T} \mathbf{v}_j^l + \mathbf{g}_i^T \mathbf{h}_j\right)\right), \tag{4}$$

where we have replaced the inner product term $\mathbf{p}_i^T \mathbf{a}_j$ in Equation 1 with a sum of L inner product terms. Each $\mathbf{u}_i^l$ and $\mathbf{v}_j^l$ model instructor and question association of a particular modality of meta-data. Additionally, the dimensions of $\mathbf{u}_i^l$ and $\mathbf{v}_j^l$ can vary for different l's depending on the mathematical representation of that meta-data modality.

Next, it is easy to see that the same approach can be applied to analyzing instructors' preferences on other learning resources. Although we specify in Equation 1 that Y_{ij} represents instructor i's preference for question j, Y_{ij} can naturally represent preferences to other contents types, by using j to index learning resources. Therefore, we can easily extend the proposed model in Equation 1 to analyze additional instructor preference data with a different preference data matrix $\mathbf{Y}$.

Table 1: Performance comparison between the proposed model and its variants, in terms of prediction accuracy (ACC) and area under operating characteristic curve (AUC). The proposed model achieves the best result among its two variants. The model involving the $\mathbf{g}_i$ and $\mathbf{h}_j$ factors achieves better performance than the model with the $\mathbf{p}_i$ factor alone.

	Metrics	
Models	ACC	AUC
Proposed Model	**0.9033±0.0045**	**0.9592±0.0061**
P Model	0.8880±0.0047	0.8908±0.0064
GH Model	0.9026±0.0048	0.9254±0.0058

3. EXPERIMENTS

We now evaluate the prediction performance of the proposed latent factor model using a real-world educational dataset. We further showcase the interpretability of the model by visualizing the instructor Bloom's Taxonomy tag preference vectors $\mathbf{p}_i$.

3.1 Dataset

We collect from OpenStax Tutor [13] 20 instructors' preferences on all 896 questions of the textbook "Concepts of Biology" that these instructors use in their classes, resulting in a fully observed data matrix $\mathbf{Y}$ of dimension 20 by 896. About 15% of all entries in $\mathbf{Y}$ have a value of 1, meaning that an instructor explicitly indicates to exclude a question, and the rest 0, meaning that there is no such indication. We remind the reader that excluding a question in OpenStax Tutor means that this question is excluded from the pool of questions that OpenStax Tutor selects from to assign to learners as personalized practice recommendations. We also collect the Bloom's Taxonomy tag for each question, labeled by domain experts, as meta-data on the questions. Since there are 6 distinct Bloom's Taxonomy tags

Table 2: Performance comparison between the proposed model and existing collaborative filtering methods in terms of the four metrics. The proposed model shows superior prediction performance compared to the other methods on all metrics.

	Models/Methods			
Metric	**Full Model**	UBCF	IBCF	FSVD
ACC	**0.9033±0.0045**	0.8961±0.0048	0.8895±0.0048	0.8896±0.0045
F-1	**0.6483±0.0128**	0.6007±0.0158	0.5696±0.0137	0.6185±0.0158
Precision	**0.7163±0.0222**	0.7070±0.0214	0.6928±0.0254	0.6964±0.0236
Recall	**0.6153±0.0227**	0.5226±0.0190	0.4954±0.0159	0.5661±0.0248

Table 3: Comparison between $\mathbf{p}_{ik}$ (second row for each instructor) and the percentage of questions they actually excluded under each Bloom's taxonomy tag k (first row for each instructor), for selected instructors. The values of $\mathbf{p}_i$ estimated by the proposed model closely resemble the actual number of questions each instructor excluded.

	Bloom's Taxonomy tag					
Instructor	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
$i=3$	0.9% 0.058	1.6% 0.083	0.5% 0.038	1.8% 0.216	0.0% 0.075	0.0% 0.084
$i=5$	16.9% 0.441	16.3% 0.448	19.0% 0.501	5.5% 0.360	21.1% 1.000	33.3% 0.858
$i=9$	63.1% 0.826	67.8% 1.000	72.4% 0.985	67.3% 0.924	42.1% 0.583	33.3% 0.215

in total, the dimension of the question–Bloom's Taxonomy tag association vector $\mathbf{a}_j$ is $K = 6$. The entries of $\mathbf{a}_j$ correspond to Bloom's Taxonomy tags in increasing levels of cognitive complexity, i.e., $k = 1$ represents "remembering", $k = 2$ represents "understanding", etc. Additionally, each question is only associated with one Bloom's Taxonomy in our dataset. Therefore, the values of $\mathbf{a}_j$ satisfy $a_{jk} \in \{0, 1\}$ and $\sum_k a_{jk} = 1$ for all j.

3.2 Experimental Setup

We compare our model and its variants against three methods frequently used in recommender systems: user-based collaborative filtering (UBCF), item-based collaborative filtering (IBCF), and funk singular value decomposition (FSVD). UBCF and IBCF use similarities among users (instructors) and items (questions), respectively, and predict a user's preference on an item based on the preferences of most similar users or items. FSVD makes the observation that the actual number of user and item types is much lower than the number of users and items, and therefore utilizes a low-rank model to model user–item interactions [4, 5]. [7] explain the detailed implementations and evaluation methods for UCBF, ICBF, and FSVD that we use in this paper.

We use a total of five metrics for model evaluation: (i) prediction accuracy (ACC), (ii) precision, (iii) recall, (iv) F-1 score, and (v) area under the receiver operating characteristic curve (AUC) of the resulting binary classifier [8]. Formulas for calculating metrics (i) through (iv) are shown below:

$$\begin{cases} \text{ACC} & = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}} \\ \text{precision} & = \frac{\text{TP}}{\text{TP+FP}} \\ \text{recall} & = \frac{\text{TP}}{\text{TP+FN}} \\ \text{F-1} & = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision+recall}}, \end{cases}$$

where TP denotes true positive, TN denotes true negative, FP denotes false positive, and FP denotes false negative. In the context of this paper, we treat preference for excluding a question, corresponding to $Y_{ij} = 1$, as the positive class. True positive means predicting the positive class when the ground truth is also positive. False positive means predicting the positive class when ground truth is negative, and the rest follows. All metrics take on values in $[0, 1]$, with larger values indicating better prediction performance. We perform two sets of comparisons, one between the full model and its two variants (the P and GH models) evaluated on the ACC and AUC metrics, and the other one between the full model and UCBF, IBCF, and FSVD using ACC, F-1, precision, and recall. Since the AUC metric is only appropriate for evaluating algorithms using probabilistic models, we do not evaluate the three CF methods that do not have an underlying probabilistic model.

We perform 5-fold cross validation for model selection, i.e. choosing the best set of parameters for each model, and model assessment, i.e. evaluating the best model on the test set, according to the train-validation-test split paradigm. First, we randomly select 20% of all observed data and set it aside as test set. We then randomly partition the remaining 80% of all data into four roughly equal-sized parts, fit the

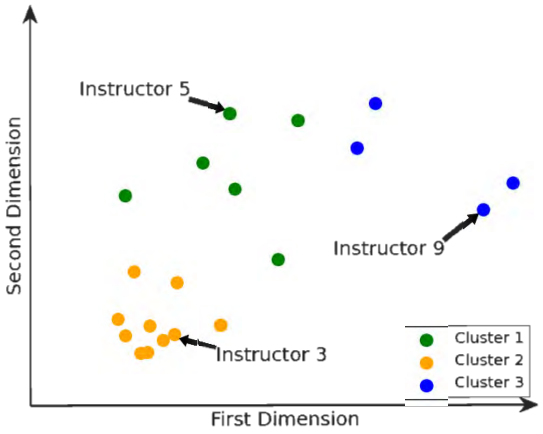


Figure 1: 2D projection of instructor Bloom's Taxonomy tag preference vectors using multidimensional scaling and clustering using k-means that shows instructors' diverse question exclusion preferences. Notice that instructors 3, 5, and 9 that we show to have very different question exclusion preferences also appear far apart in the plot.

model to first three of the four parts, and validate the fitted model using the fourth part of the data to select the values of the regularization parameters using grid-search. Finally, we select the best performing model, fit it on all data except for the test set, and evaluate its performance on the test set. We perform 20 random partitions of the data, average the evaluation results, and compare the best evaluation results of each method.

3.3 Results And Discussions

Table 3 shows results for the full model, UBCF, IBCF, and FSVD evaluated on the ACC, F-1, Precision, and Recall metrics. The relatively lower Recall scores of the full model compared to its ACC suggests that the proposed model still exhibits some albeit less tendency to avoid assigning an exclusion preference label than other methods. Nevertheless, comparing across columns, we see that the performance of the full model, regardless of the choice of metric, is significantly better than the rest of the models, showing promise for the proposed latent factor model in predicting instructors' question exclusion preferences.

Table 1 shows prediction performance results for the full model and its two variants evaluated on the ACC and AUC metrics. From the table, we observe that the full model achieves the best performance on both metrics. Further inspection of the results of the two variants reveals that the GH Model, which involves factors $\mathbf{g}_i$ and $\mathbf{h}_j$, achieves better results for both metrics than the P Model, which involves only factor $\mathbf{p}_i$. This implies that besides Bloom's Taxonomy, additional factors are needed in the latent factor model to better characterize instructors' question exclusion preferences. Even though Bloom's Taxonomy contribute only moderately to the prediction performance, the purpose of explicitly incorporating Bloom's Taxonomy, as stated earlier, is the power of interpretability it brings to the proposed model, which we demonstrate below.

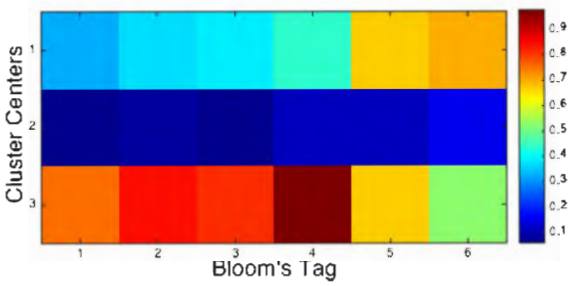


Figure 2: Heatmap visualization of the cluster centers that shows the radically different question exclusion preferences of each cluster of instructors.

First, we use the instructor Bloom's Taxonomy tag association vectors to interpret how instructors prefer to exclude certain questions in terms of Bloom's Taxonomy. Table 3 presents a comparison between the numerical values of entries in the instructor Bloom's Taxonomy tag preference vector $\mathbf{p}_i$ and the percentage of questions that the corresponding instructor excludes with each Bloom's Taxonomy tag, for a selected subset of instructors $i \in \{3, 5, 9\}$. Comparing the values in the two rows for each instructor i in the table, we observe that higher values of $\mathbf{p}_{ik}$ correspond to a higher percentage of the questions of Bloom's Taxonomy tag k that the instructor excludes. Therefore, $\mathbf{p}_{ik}$ reflects the degree to which instructor i prefers to exclude questions with Bloom's Taxonomy tag k. For example, we observe from the second row of instructor 5 that values of $\mathbf{p}_{ik}$ are high for $k = 5$ and $k = 6$, indicating that this instructor strongly prefers to exclude questions that involve more complex cognitive processes such as evaluating and creating. Second, the instructor Bloom's Taxonomy tag preference vectors uncover differences and patterns in instructors' Bloom's Taxonomy tag preferences. Comparing the second row of all instructors in Table 3, we see distinct preferences for different instructors. For example, values of $\mathbf{p}_{ik}$ for instructor 9 are high for $k = 1, 2, 3, 4$, indicating that this instructor strongly prefers to not assign questions that involve simpler cognitive processes such as remembering, understanding, applying and analyzing. Such preferences are opposite to those for instructor 5. Moreover, instructor 3 exhibits no obvious exclusion preference for any Bloom's Taxonomy tags by noting the small values of $\mathbf{p}_{ik}$ for $i = 3$, setting this instructor apart from both instructors 5 and 9.

We further visualize patterns in instructors' question preferences after projecting each $\mathbf{p}_i$ onto a 2-dimensional plane using multidimensional scaling [3]. We then run the K-means algorithm to group the instructors into 3 clusters. Figure 1 plots each $\mathbf{p}_i$ as a point in the 2-dimensional space, where the color of the point denotes the cluster that the point belongs to. The figure shows obvious clustering patterns, which means that instructors exhibit only a few patterns on their Bloom's Taxonomy tag preferences. Note that instructors 3, 5 and 9 are far apart in the figure and belong to different clusters. Figure 2 presents a heatmap visualization of the cluster centers that shows distinct Bloom's Taxonomy pref-

erences across the three instructor clusters. For example, the first and third clusters demonstrate almost entirely opposite Bloom's Taxonomy preferences, where the first cluster tends to exclude questions with more complex cognitive process, whereas the third cluster tends to exclude questions with simpler cognitive processes. On the other hand, the second cluster does not exhibit strong exclusion preferences for any particular Bloom's Taxonomy tag. Such clustering could help a PLS to recommend questions to an instructor that they might want to exclude, based on instructors that have demonstrated similar Bloom's Taxonomy preferences.

4. CONCLUSIONS AND FUTURE WORK

We have presented a latent factor model that predicts instructors' question preferences, and explicitly incorporates questions' Bloom's Taxonomy tags to improve model interpretability. Evaluated on a real-world educational dataset, our proposed model shows superior prediction performance over popular collaborative filtering methods frequently used in recommender systems. Additionally, we demonstrated model interpretability by showing that the Bloom's Taxonomy captures each instructor's question preferences reasonably well, and also visualized different Bloom's Taxonomy preference patterns across instructors. These encouraging results show the promise of using latent factor approach for instructors' content preferences modeling to 1) potentially automate the question exclusion process in OpenStax Tutor, and 2) more broadly, to improve various aspects of personalized learning systems such as intelligent content recommendation that takes into account of instructors' preferences.

To achieve these goals, the following avenues of future research seem appropriate. First, we used only one source of meta-data, i.e., Bloom's Taxonomy tags, in the proposed model. We have shown that the proposed model is easily extendable to accommodate additional meta-data; moreover, the performance comparison between the P Model and the GH Model shows the need to incorporate additional factors. Therefore, we plan to extend the proposed model to include other sources of meta-data, such as the textbook chapter or section that each question belongs to, to improve both prediction accuracy and model interpretability. Second, we focused on instructors' preferences in a very specific content, i.e., question exclusion. We are interested to see how well the proposed modeling approach can be adapted to analyze instructors' preference for other learning resources. Third, we also plan to expand our experiments from a single textbook to multiple textbooks and domains, in order to validate the proposed approach for analyzing instructor preferences on a wide range of contents and across different subject domains.

5. REFERENCES

[1] P. Armstrong. *Bloom's Taxonomy*, 2014. `https://cft.vanderbilt.edu/guides-sub-pages/blooms-taxonomy/`.

[2] R. M. Bell and Y. Koren. Lessons from the Netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, 9(2):75–79, Dec. 2007.

[3] I. Borg and P. J. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

[4] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human–Computer Interaction*, 4(2):81–173, Feb. 2011.

[5] S. Funk. *Netflix update: Try this at home*, 2006. `http://sifter.org/~simon/journal/20061211.html`.

[6] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *Proceedings of the 10th IEEE International Conference on Data Mining*, pages 176–185, Dec. 2010.

[7] M. Hahsler. *recommenderlab: Lab for Developing and Testing Recommender Algorithms*, 2016. R package version 0.2-1.

[8] J. Huang and C. X. Ling. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on knowledge and Data Engineering*, 17(3):299–310, Jan. 2005.

[9] D. R. Krathwohl. A revision of Bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218, Nov. 2002.

[10] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *Journal of Machine Learning Research*, 15(1):1959–2008, Jan. 2014.

[11] C. Limongelli, M. Lombardi, A. Marani, and F. Sciarrone. A teaching-style based social network for didactic building and sharing. In *Proceedings of the 16th International Conference on Artificial Intelligence in Education*, pages 774–777, Jul. 2013.

[12] M. G. Manzato. Discovering latent factors from movies genres for enhanced recommendation. In *Proceedings of the 6th ACM Conference on Recommender Systems*, pages 249–252, Sept. 2012.

[13] OpenStax Tutor. https://tutor.openstax.org/, 2016.

[14] Z. A. Pardos and N. T. Heffernan. Tutor modeling vs. student modeling. In *Proceedings of International Florida Artificial Intelligence Research Society Conference*, pages 420–425, May 2012.

[15] S. Sahebi, Y.-R. Lin, and P. Brusilovsky. Tensor factorization for student modeling and performance prediction in unstructured domain. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 502–506, June 2016.

[16] M. Sweeney, J. Lester, H. Rangwala, and A. Johri. Next-term student performance prediction: A recommender systems approach. *Journal of Educational Data Mining*, 26(1):33–68, Mar. 2016.

Mining Innovative Augmented Graph Grammars for Argument Diagrams through Novelty Selection

Linting Xue, Collin F. Lynch & Min Chi
North Carolina State University, Raleigh, North Carolina, USA
lxue3, cflynch, & mchi@ncsu.edu

ABSTRACT

Augmented Graph Grammars are a graph-based rule formalism that supports rich relational structures. They can be used to represent complex social networks, chemical structures, and student-produced argument diagrams for automated analysis or grading. In prior work we have shown that Evolutionary Computation (EC) can be applied to induce empirically-valid grammars for student-produced argument diagrams based upon fitness selection. However this research has shown that while the traditional EC algorithm does converge to an optimal fitness, premature convergence can lead to it getting stuck in local maxima, which may lead to undiscovered rules. In this work, we augmented the standard EC algorithm to induce more heterogeneous Augmented Graph Grammars by replacing the fitness selection with a novelty-based selection mechanism every ten generations. Our results show that this novelty selection increases the diversity of the population and produces better, and more heterogeneous, grammars.

Keywords

Heterogeneous Rules, Augmented Graph Grammars, Argument Diagrams, Evolutionary Computation, Novelty selection

1. INTRODUCTION

Intelligent tutoring systems, social-networking systems, and computer-supported collaborative platforms have grown increasingly prevalent in education (e.g. Pyrenees [15], LASAD [8], and CSCL [13]). Consequently, researchers have begun to collect large repositories of complex relational data representing student-produced conceptual or structural diagrams [8], structured user-system interaction logs [15], and personal relationships [13]. Researchers have generally analyzed this data via standard network analysis tools and gestalt relationships which allow us to assess general topological graph structures but which do not focus on individual graph features or graph rules (e.g. [15, 13]).

One of the primary goals of Graph-based Educational Data Mining is to automatically identify substructures that can reveal vital pedagogical information in graph data. These features include good sub-solutions and structural flaws in students' solutions, which can be used for automated guidance and grading [10]. Prior research has demonstrated that we can use hand-authored graph rules to evaluate student-produced argument diagrams [10]. But, hand-authored rules are expensive and time consuming to generate and do not always generalize well to novel contexts. Existing general purpose graph rule induction algorithms (e.g. [16, 2]) have limitations and are unsuited to the induction of generalized rules that use negation or other hierarchical elements [17].

Evolutionary Computation (EC), on the other hand, is both flexible and robust enough to induce complex graph structures and to deal with rich graph data. We have previously shown that EC can be used to automatically induce positive and negative graph rules for student-produced argument diagrams through fitness selection [17]. The induced rules can be used as features to provide hints for argument writing, and to detect structural flaws. Prior research also indicates that the induced graph rules from EC outperform all but one of the expert hand-authored rules and they outperform all of the rules induced by two general purpose graph grammar induction algorithms, Subdue [2] and gSpan [16]. However, prior research has shown that, while the traditional EC algorithm does converge to an optimal fitness, the premature convergence can lead to it getting stuck in local maxima, which may lead to undiscovered graph rules [6].

In this work, we augmented the standard EC algorithm to produce more heterogeneous Augmented Graph Grammars that can reflect innovative structures in student-produced argument diagrams. To that end, we incorporated a *novelty selection* mechanism into our EC system that was designed to enforce population diversity. The goal of this diversity was to explicitly retain novel *introns* and thus to reward the basic stepping stones of evolution both in the internal (*genospace*) and the external application space (*phenospace*), respectively. In this work, we experimented with two different novelty selection mechanisms: novel genotype selection and novel phenotype selection. Our research hypotheses is that *novelty selection will increase the diversity of the population and will produce better and more heterogeneous graph grammars when compared with pure fitness selection.*

2. BACKGROUND

2.1 Argument diagrams

Argument diagrams are graphical representations for real-world argumentation that reify the essential components of arguments such as *hypotheses* statements, *claims*, and *citations* as nodes and the *supporting*, *opposing*, and *clarification* relationships as arcs [11]. These complex elements can include text fields describing the node and arc types or free-text assertions, links to external resources and other data.

A sample student-produced diagram is shown in Figure 1. The diagram includes a *hypothesis* node at the bottom right, which contains two text fields, one for a conditional or *if* field, and the other for a consequent or *then* field. Two *citations* are connected to the hypothesis via *supporting* and *opposing* arcs colored green and red, respectively. They are also connected via a *comparison* arc. 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.

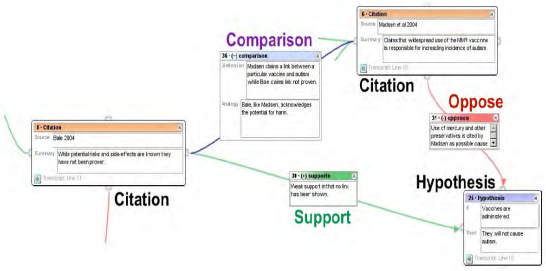
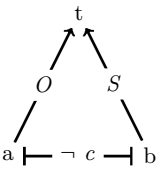


Figure 1: A student-produced Argument Diagram.

2.2 Augmented Graph Grammars

Augmented Graph Grammars (AGGs) are a graph-based rule formalism that supports rich relational structures [9]. AGGs are an extension of traditional graph grammars, which are composed of standard graph elements including *ground* nodes, *ground* arcs, and *variable* arcs which can match multiple items. In addition to these basic features, AGGs also support: complex node and arc types that contain sub-elements; negated elements which select for the nonexistence of subgraphs; generalized node and arc types 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 constraints. As such they are an ideal rule representation for the analysis of argument diagrams.

In prior work [10, 11], we collaborated with a group of domain experts to define a set of 77 a-priori argument rules encoded as grammars. These rules were designed to identify individual features of argument diagrams or sub-graphs that were consistent with high quality argumentation or which represented common structural flaws. We have shown that these hand-authored graph rules are correlated with the student-produced argument diagram grades and essay grades and they are empirically valid and can be used as



$$\begin{Bmatrix} t.Type = \text{“}claim\text{”} or \text{“}hypothesis\text{”} \\ a.Type = \text{“}citation\text{”} \\ b.Type = \text{“}citation\text{”} \\ c.Type = \text{“}comparison\text{”} \end{Bmatrix}$$

Figure 2: A hand-authored Augmented Graph Grammar.

the basis for predictive models of student grades. A sample hand-authored rule is shown in Figure 2. This rule is designed to identify cases where students use a citation a to oppose a claim or hypothesis node t via an opposing path O, and use the other citation b to support the node t via a supporting path S, however, the students do not include a comparison arc c between two citations a and b.

2.3 Evolutionary Computation

Evolutionary Computation (EC) is a general machine learning algorithm based upon Natural Selection. The algorithm starts with a population of candidate solutions, which may be generated at random or user-defined. The individual *solutions* are assessed by an objective measurement known as the *fitness function*. Subsequent generations are produced by a combination of *elitism* in which very fit individuals are cloned into the next generation, and fitness-proportional *reproduction* in which individuals are copied over with direct *mutations* or through *crossover* with other members in the population. The EC algorithm proceeds iteratively until a given fitness threshold is reached or until a fixed number of generations has passed. When compared with existing graph grammar induction algorithms, EC is much more flexible and robust. The behavior of the system is determined by the user-defined *solution representation*, *fitness function*, and the genetic operators including *mutation* and *crossover*.

In prior work, we applied EC to automatically induce a set of AGG rules on student-produced argument diagrams [17]. The induced rules support disjoint subgraphs, negation, and generalized elements. In that work, the solution representation was an individual graph rule. The fitness of each graph rule was accessed via Spearman's Rank Sum Correlation (ρ) [3] between the frequency with which a rule matches a diagram, and the argument grades. The mutation in the EC algorithm was basic point mutation that can add, delete, or modify existing nodes and arcs. Crossover was implemented using matrix crossover based upon the work of Stone, Pillmore, & Cyre [14].

2.4 Novelty Selection

Absolute fitness functions of the type that we used in our prior studies, are designed to reward *individual* progress toward an absolute objective in the search space without consideration for the population as a whole. Prior studies have shown that although the fitness function is driven to con-

verge to a fitness optimum, the *objective function* sometimes suffers from the pathology of local optima [6]. This is because the objective function only rewards improvements in performance with respect to the static objective, it does not necessarily reward diversity in the search space that can ultimately lead to other solutions. One approach that EC researchers have taken to address this problem is *Novelty Selection* that is, explicitly incorporating population diversity into the fitness metric or supporting diverse solutions irrespective of the fitness value [1, 5]. The goal in doing so is to encourage the development of good sub-solutions or *stepping stones* that can support novel solutions and avoid local optima.

Current novelty selection algorithms fall into one of two broad categories: novel *genotype* selection, or novel *phenotype* selection. In EC, the genotype of a solution is the basic solution structure or code that defines the solution, which corresponds to the set of genes in a real organism. The phenotype, by contrast, is the observed behavior of the solution when it is evaluated. In the context of our work, the genotype is the AGG structure while the phenotype is the way in which the rule maps to the graphs in our dataset. Thus the genotype is fixed while the phenotype is data-driven.

The novel genotype selection is focused on finding individuals that have a unique structure relative to the remainder of the population. Prior researchers have focused on applying user-defined metrics to calculate pairwise distances between members of the population [4, 1]. The metrics are necessarily representation specific. Maximally-unique individuals are then selected for reproduction or cloning in order to maintain genetic diversity. The primary shortcoming of this method is that computing pairwise distance can be computationally intractable (e.g. comparing neural networks which is NP-Hard) [5].

While novel genotype selection seeks individuals with unique genes, novel phenotype selection rewards individuals that *behave* differently according to some separate evaluating metric. This is usually based upon some user-defined distance function based upon prior knowledge of the domain. The goal of the metrics is to enforce coverage of the solution space and, as with the genotype selection, maximally unique individuals are selected for retention. The primary disadvantage of this approach is that given two individuals with comparable behavior but distinct genes we will discard one and will potentially lose good evolvable genes in the process [5].

3. METHODS

In order to compare the performance of novelty selection with traditional objective fitness selection, we implemented two novelty selection methods in EC with one rewarding novel rule structures (genotype) and the other rewarding rules that match a unique set of graphs in our dataset (phenotype). For the former metric, we select the novel rules according to the *diversity score*, which is calculated using a greedy graph-matching algorithm; for the latter one, the novel rules are rewarded based on the *behavior score* using the χ^2 test[3]. A large diversity or behavior score indicates that the specified rule is substantively different from the rest of the population.

3.1 Genotypic Distance - Diversity Score

We define the diversity score of an individual as its average genotypic distance from the remainder of the population. In order to compute this score, we developed a greedy graph matching algorithm that computes the distance based upon local-neighborhood similarity. The root intuition behind this algorithm is that if two graph grammars G_0 and G_1 are isomorphic then it should be possible to automatically align their local neighborhoods (individual nodes plus immediate neighbors). The algorithm returns a distance score between 0 and 1 inclusive. Here 0 means that the two grammars are completely isomorphic and 1 indicates they are wholly distinct from one another. The algorithm operates as follows:

First, we count the total number of nodes n in both grammars on a per-type basis. For example, Figure 3 shows two graph grammars G_0 and G_1. They have a total of 6 nodes of 5 types (A, B, C, D, E) and 4 arcs of 2 types $(1, 2)$. For category A, G_0 has one A node (A_0), while G_1 has two (A_0 & A_1), so $n_a = max(2, 1) = 2$. For the remaining types B, C, D and E, we have $n_b = n_c = n_d = n_e = 1$, and the total number of nodes n is 6.

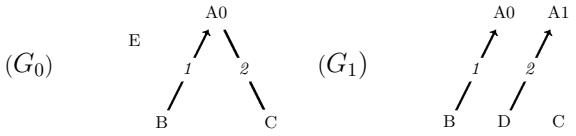


Figure 3: Example of two graph grammars with five categories of nodes $(A, B, C, D, E,)$ **and two categories of arcs** $(1, 2)$**.**

Second, we compute the individual similarity score $S = \{s_1, s_2, s_3, ..., s_i, ..., s_n\}$ for $i \in \{0, n\}$, where s_i indicates the similarity score for node N_i. For nodes of the same type, we use greedy search to find the best match for each node and then update the maximum similarity score of the whole grammar. The value of s_i is between -1 and 1, and is computed by the following formula:

$$s_i = \begin{cases} -1 & if\ N_i\ in\ G_0\ or\ G_1; \quad (1) \\ \dfrac{\#\ of\ shared\ neighbors}{total\ \#\ of\ neighbors\ in\ G_0\ and\ G1} & otherwise. \quad (2) \end{cases}$$

where $s_i = -1$ means that node N_i is in either G_0 or G_1 but not both; $s_i = 0$ indicates that node N_i is in both graphs, but they do not share any neighbour at all; $s_i = 1$ indicates that node N_i is in both graphs and they share the same neighbor(s) with the same arc(s). Note that if two nodes share a same neighbour but with different arcs, we do not count it as the same neighbour.

In the example shown in Figure 3, we have $S = \{s_a^1, s_a^2, s_b, s_c, s_d, s_e\}$. For A nodes, if we match $A_0 \in G_0$ with $A_0 \in G_1$, we have $s_a = \frac{1}{2}$; if we match $A_0 \in G_0$ with $A_1 \in G_1$, s_a is 0. Thus, the best match for $A_0 \in G_0$ is $A_0 \in G_1$ and update for $s_a^1 = \frac{1}{2}$. Now for $A_1 \in G_1$, we cannot find any node to match with, so $s_a^2 = -1$ using Equation (2). For the

B nodes, B is present in both graphs, they share the same neighbour (A) with the same arc type of (1), so $s_b = 1$. Similarly C nodes are present in both graphs, but they do not share any neighbours because $C \in G_1$ is isolated, so $s_c = \frac{0}{1} = 0$. For D and E, we have $s_d = s_e = -1$ because node D and E is just shown in one of the two graphs. Thus we have $S = \{\frac{1}{2}, -1, 1, 0, -1, -1\}$.

Finally, we use Euclidean distance to normalize the similarity scores to a distance score within a range of $[0, 1]$ by Equation (3). Then the diversity score for an individual is the average distance score to the remaining population.

$$D = \sqrt{\frac{\sum_{n=1}^{n}(1 - s_i)^2}{n * 2^2}} \tag{3}$$

3.2 Phenotypic Distance - Behavior Score

The behavior score of an individual is the average phenotypic distance between it and the remainder of the population. We use a data-driven definition of behavior. For each individual we define its *behavior signature* as a vector of positive integers representing the number of distinct subgraphs that it matches for each of the 104 graphs in our dataset. We then calculate the pairwise distance between individuals using the χ^2 test of independence [12]. χ^2 is a statistical test that measures divergence from the expected distribution assuming that one feature occurs independently of the others. It is often applied to evaluate the independence of two variables in mathematical statistics [7]. The null hypothesis of this test is that two variables are wholly independent. A p-value ≤ 0.05 of χ^2 test leads us to reject the null hypothesis and conclude that the variables are significantly correlated.

If two frequency sets are statistically independent from one another other according to the χ^2 test then we assign a phenotypic distance score as 1 indicating that the grammars are independent. If, however they are dependent then we assign a score of 0, meaning that the grammars are substantively similar given our dataset. We then calculate the average score for each individual to indicate its relative uniqueness within the population.

3.3 Dataset

For this study we used a dataset of 104 argument diagrams that was originally collected at the University of Pittsburgh in a course on Psychological Research Methods [10, 11]. The subgraph shown in Figure 1 was collected as part of this study. Students in the course were instructed to plan their written arguments graphically using LASAD, an online tool for argument diagramming and collaboration [8], and then to produce written essays. The diagramming ontology contained four types of nodes: *citation*, *claim*, *current study* and *hypothesis*; and four types of arcs: *supporting*, *opposing*, *comparison*, and *unspecified*. Current study nodes are used to represent factual information about the study such as the target population. Unspecified arcs represent cases where nodes provide clarification or concept definitions. At the end of the study, 104 paired diagrams and essays were collected. These diagrams and essays were graded by an experienced TA according to a parallel grading rubric.

4. EXPERIMENTS

In this work, we evaluated the impact of novelty selection on graph grammar induction by comparing the two types of novelty selection to a traditional objective-fitness approach. We ran three experiments to induce three sets of graph grammars using the different selection functions. The three experiments are Baseline, Geno, and Pheno respectively:

Baseline: we used traditional fitness function at each generation. The fitness function measures the correlation between the observed graph rule frequency and diagram grades.

Geno: we replaced the fitness function with novel genotype selection on every tenth generation. The novel genotype selection rewards grammars with novel structure for further evolution by cloning them to the next generation.

Pheno: we used the novel phenotype selection to reward graph grammars that have significantly different behaviours to the remaining population in every tenth generation.

For each experiment, we conducted a series of three evolutionary runs to explore the search space. In each run, we set a population size of 100 individuals and ran for 500 generations. The initial populations were composed of randomly generated grammars each of which contained between 3 and 10 elements. The nodes and arcs were all ground elements and were selected from a predefined ontology of basic types that matched the argument diagram ontology. The fitness function, crossover and mutation operators were the same as in our prior work discussed in section 2.3. On each evolutionary run, we harvested all graph grammars generated over the course of the run whose performance exceeded a threshold of ($\rho \geq 0.18$) and preserved them for later analysis. The threshold was chosen based upon a series of exploratory studies which showed that ρ values at or above this threshold were statistically significant.

5. RESULTS & ANALYSIS

After collecting the three sets of grammars, we applied the graph matching algorithm discussed in section 3.1 to identify the isomorphic rules, we then filtered the overlapping rules to obtain the unique rule sets. Table 1 shows the number of unique rules collected from each experiment along with the ρ values for the top three rules in each unique rule set. The top

Table 1: The number of unique rules above the threshold ($\rho \geq 0.18$) and the Spearman's Correlation value ρ for the top three best rules

Experiments	Unique rules	ρ value		
		1st	2nd	3rd
Baseline-Only	37	0.282	0.279	0.260
Geno-Only	**112**	**0.348**	**0.334**	**0.325**
Baseline ∩ Geno	146	0.371	0.369	0.362
Baseline-Only	26	0.282	0.260	0.254
Pheno-Only	**99**	**0.348**	**0.334**	**0.333**
Baseline ∩ Pheno	157	0.371	0.369	0.362

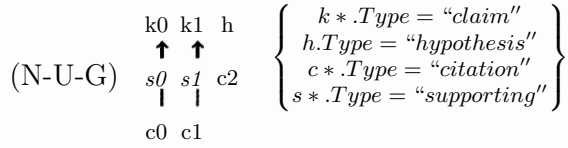


Figure 4: Best performing graph rule in Geno Only and Pheno Only with correlation ($\rho = 0.348$).

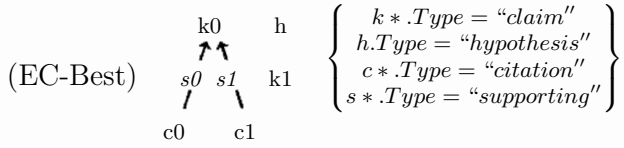


Figure 5: Best performing rule in EC experiment with the correlation ($\rho = 0.371$).

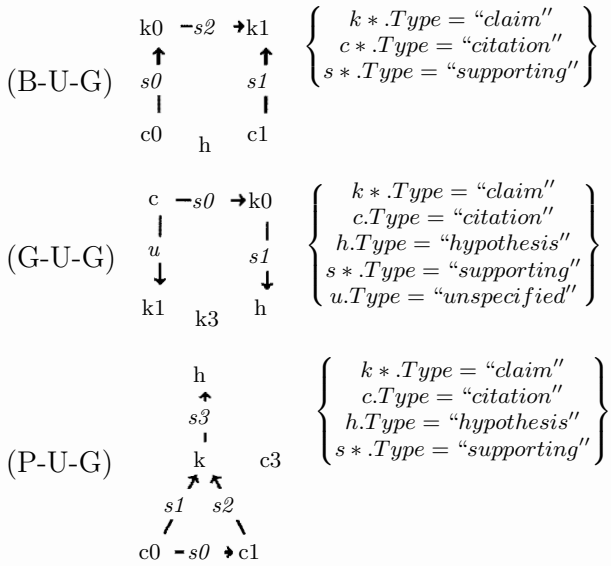


Figure 6: Example graph rules with unique structures. B-N-G: unique rule in Baseline with correlation ($\rho = 0.280$); G-N-G: unique rule in Geno experiment with correlation ($\rho = 0.197$); P-N-G: unique rule in Pheno experiment with correlation ($\rho = 0.182$).

three rows display the rules that are unique to the Baseline and Geno experiments along with the the overlapping rules shared between them (Baseline ∩ Geno). The bottom three rows show the rules that are unique to the Baseline and Pheno experiments, and the overlapping rules between them (Baseline ∩ Pheno).

As Table 1 indicates, after removing the isomorphic rules, the Geno and Pheno experiments still produced a large number of high-performing rules with Geno-Only having 112 unique rules and Pheno-Only having 99. The top three performing rules in Geno- and Pheno-Only outperform the rules in both the Baseline-Only. After examining these rules, we found that the top two rules in Geno- and Pheno-Only are isomorphic with the same performance and the best rule is shown in Figure 4. This rule contains 6 nodes with two citations ($c0$ & $c1$) supporting two claims ($k0$ & $k1$) and two isolated nodes, one hypothesis (h) and one citation ($c2$), which may or may not be connected to the remaining structure. This reflects an argument diagram where the students have two solid claims supported by different citations and where they include both a hypothesis and at least one other additional supporting citation. This rule captures another highly correlated feature in the student-produced argument diagrams that two claims are supported by two different citations.

The top three rules in Baseline ∩ Geno and Baseline ∩ Pheno outperform the rules in both Baseline-Only and the rules in Geno- and Pheno-Only. We also found that these three best rules are isomorphic with the same performance, meaning that all three fitness models are capable of identifying the best performing rules on our dataset. Figure 5 shows the best graph rule with the correlation ($\rho = 0.371$). It represents a rule with 5-nodes, two of which are citations ($c0$ & $c1$) that support a shared claim node ($k0$). The remaining nodes consist of a single claim ($k1$) and hypothesis (h) which may or may not be connected to the other elements. 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 that students have been encouraged to make in their arguments as it shows an ability to synthesize citated work to form a complex claim.

We also investigated the unique structures that were specific to each experiment. The structure refers to the sub-graph within a graph rule but without isolated node(s). When comparing the Baseline and Geno experiments, we found three unique structures that only show up in the Baseline experiment and six in Geno. When comparing the Baseline and Pheno experiments, we identified three unique structures in the Baseline experiment and four in the Pheno experiment respectively.

Figure 6 shows three example graph rules with unique structures in each experiment. B-U-G is a unique rule induced in the Baseline experiment, it matches cases where two citations ($c0$ & $c1$) support two claims ($k0$ & $k1$) and are connected via a supporting arc ($S2$) and where an isolated hypothesis (h) may or may not be connected to the remaining structure. This rule reflects a very interesting argument structure where the student used one citation to directly support a claim and the other citation to support this claim with another intermediate claim. G-U-G shows rule that was induced in the Geno experiment. It has one citation (c) that supports a claim ($k0$) which in turn supports a hypothesis (h). This citation is also connected to a claim ($k1$) with an unspecified arc (u). And it has an isolated claim ($k3$) which may or may not be connected to the remainder of the structure. This rule indicates another innovative use

of chaining support which students were encouraged to use and which is comparable to B-N-G.

P-U-G shows a graph rule from the Pheno experiment, it contains a connected structure with four arcs, and is the *most* complex rule above the threshold. This connected structure has two citations with one supporting another ($c0$ & $c1$) and then jointly supporting a shared claim (k) which in turn directly supports a hypothesis (h). The rule also contains an isolated citation ($c3$) which may or may not connect to the remaining structure. Conceptually this indicates a case where a grounded claim supports a research hypothesis. In the real word, it indicates that the author sought out closely-related sources of literature or noted important connections between them, then used this well-supported claim to support a research hypothesis, something which they had been encouraged to do in class.

6. CONCLUSION AND FUTURE WORK

In this work, we augmented the standard EC with two novelty section methods to induce Augmented Graph Grammars on student-produced argument diagrams by replacing the fitness function with a novelty selection function every ten generations. This novelty selection promotes diversity in the population by explicitly encouraging the production and maintenance of novel stepping stones or partial solutions in the genotypic and phenotypic spaces. Our experimental results indicate that, when compared to pure objective-fitness selection, the novelty-selection functions produced more heterogeneous and better-performing graph grammars. The unique rules that were induced by each experiment reflect some novel features in student-produced argument diagrams. The significance of this work is that the novelty selection can enhance EC to produce more empirically-valid rules that can be used for automatic grading.

In future work, we plan to work with domain experts to determine whether the rules are semantically valid, and whether or not they can serve as the basis for automatic hinting. We will also build an intelligent argument grading system to automatically grade and provide feedback on student-produced argument diagrams based on the induced graph grammars and other argument diagram features.

7. REFERENCES

[1] L. T. Bui, H. A. Abbass, and J. Branke. Multiobjective optimization for dynamic environments. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 3, pages 2349–2356. IEEE, 2005.

[2] 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.

[3] P. Dalgaard. *Introductory Statistics with R*. Springer Verlag New York Inc., 2002.

[4] E. D. De Jong, R. A. Watson, and J. B. Pollack. Reducing bloat and promoting diversity using multi-objective methods. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pages 11–18. Morgan Kaufmann Publishers Inc., 2001.

[5] S. Doncieux and J.-B. Mouret. Behavioral diversity measures for evolutionary robotics. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

[6] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial intelligence through simulated evolution*. John Wiley, 1966.

[7] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Tools with artificial intelligence, 1995. proceedings., seventh international conference on*, pages 388–391. IEEE, 1995.

[8] F. Loll and N. Pinkwart. Lasad: Flexible representations for computer-based collaborative argumentation. *International Journal of Human-Computer Studies*, 71:91–109, Januart 2013.

[9] C. F. Lynch. Agg: Augmented graph grammars for complex heterogeneous data. In 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.

[10] C. F. Lynch and K. D. Ashley. Empirically valid rules for ill-defined domains. In J. Stamper and Z. Pardos, editors, *Proceedings of The 7^{th} International Conference on EDM 2014*. IEDMS, 2014.

[11] 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.

[12] C. F. Lynch, L. Xue, and M. Chi. Evolving augmented graph grammars for argument analysis. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, pages 65–66. ACM, 2016.

[13] O. Noroozi, H. Biemans, M. C. Busstra, M. Mulder, V. Popov, and M. Chizari. Effects of the drewlite cscl platform on studentsâĂŹ learning outcomes. *Collaborative and distributed e-research: Innovations in technologies, strategies and applications*, pages 276–289, 2012.

[14] S. Stone, B. Pillmore, and W. Cyre. Crossover and mutation in genetic algorithms using graph-encoded chromosomes. *Unpublished*, March 2011.

[15] K. VanLehn, D. Bhembe, M. Chi, C. Lynch, K. Schulze, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, and M. Wintersgill. Implicit versus explicit learning of strategies in a non-procedural cognitive skill. In *International Conference on Intelligent Tutoring Systems*, pages 521–530. Springer, 2004.

[16] 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.

[17] L. Xue, C. Lynch, and M. Chi. Unnatural feature engineering: Evolving augmented graph grammars for argument diagrams. In T. Barnes, M. Chi, and M. Feng, editors, *EDM*, pages 255–262. IEDMS, 2016.

An Extended Learner Modeling Method to Assess Students' Learning Behaviors

Yi Dong
Institute for Software Integrated Systems
Vanderbilt University
Nashville, U.S.
yi.dong@vanderbilt.edu

Gautam Biswas
Institute for Software Integrated Systems
Vanderbilt University
Nashville, U.S.
gautam.biswas@vanderbilt.edu

ABSTRACT

This paper discusses a novel approach for developing more refined and accurate learner models from student data collected from Open Ended Learning Environments (OELEs). OELEs provide students choice in how they go about constructing solutions to problems, and students exhibit a variety of learning behaviors in such environments. Building accurate models from limited amount of student data is difficult; to address this we develop a methodology that uses Monte Carlo Tree Search methods to boost the initial set of student action sequences in such a way that we can learn more accurate models of students' learning behaviors. We use a HMM representation to model students' learning behaviors and demonstrate the effectiveness of our approach by running a case study on data collected from 98 students, who worked with the Betty's Brain system for four days. The results have interesting implications for learner modeling and its applications to adaptive scaffolding of students' learning behaviors and strategies as they learn from OELEs.

1. INTRODUCTION

In recent work on computer-based STEM learning environments, there has been a focus on developing OELEs, which provide students with a learning goal, usually in the form of a complex problem or a modeling task, and a set of tools that support the problem-solving/modeling task [1]. To succeed, these students need to make choices on how to structure the solution process, explore alternative solution paths, develop awareness of their own knowledge and problem-solving skills, and develop strategies that support more effective learning and problem solving [2].

Given the complexities students face in working with OELEs, it is imperative that effective scaffolding be provided to help them progress in their learning and problem solving tasks and achieve their learning goals. However, an important component of effective scaffolding is learner modeling that can accurately capture students' cognitive and metacognitive processes. In this work, we take on the challenge of using data-driven techniques to construct accurate models of learner behaviors and performance by analyzing the learners' activity data from OELEs.

Typically, data-driven methods require large volumes of rich data to support accurate and robust learner modeling. However, collecting such data from OELEs, especially in K-12 settings can be a difficult, time consuming process. To alleviate this problem, we propose a novel set of techniques that combine the use of Hidden Markov Modeling (HMM) [7], Monte Carlo Tree Search (MCTS) [3], and a reinforcement learning methodology [4] to generate artificial student activity data that simulates students behavior corresponding to learning activities captured in the log data. The original student data combined with the artificially generated data is then used to derive more accurate and complete models of students' behaviors and strategies used for learning.

In section 2, we briefly review the Betty's Brain OELE that we use for this work, and describe the overall learner modeling approach as well as the two more important techniques that we employ, i.e., HMMs and MCTS. Section 3 provides experimental results and evaluations of our learner modeling method by comparing analysis results of original data with data generated post-reinforcement learning. Section 4 presents the discussion and conclusions.

2. BACKGROUND

We implement the learner modeling methods starting from data collected from student work in the Betty's Brain OELE. Betty's Brain is a learning by teaching environment, where students utilize tools for *information acquisition*, *solution construction* and *solution assessment* to teach a virtual character named Betty by constructing a causal map [5]. The primary student actions in the Betty's Brain environment can be categorized as:

Information Acquisition (***IA***): It relates to actions, such as reading to learn new information (***read***) and searching for specific knowledge ***search***. Taking and viewing notes is also considered to be useful for information acquisition (***notes***).

Solution Construction (***SC***): In Betty's Brain, SC actions are causal map editing actions (***mapedit***), which include addition and deletion of concepts and adding,deleting or changing links in the causal map.

Solution Assessment (***SA***): It consists of asking Betty to take a quiz(***quiz***); answer questions (***query***); and to explain how she derived her answers using qualitative reasoning methods (***expl***). Besides, students can mark correctness of links that have been added to assist their solution assessment.

Students' performance is based on a map score that is computed by comparing their causal models with a pre-specified expert model. In our study, the expert model had 15 links, which implies that the students could achieve a max map score of 15. At any time, the students' map score is computed by number of correct links minus number of incorrect links in their constructed (partial) maps. Next, we describe the learner modeling approach applied to Betty's Brain.

2.1 General Approach

Figure 1 illustrates the general approach that we have developed for our learner modeling method. As a first step, we apply a HMM clustering method [6] that divides the student' behaviors into groups of similar behaviors. We then iteratively generate a more accurate HMM model for each group by running a MCTS algorithm that combined with a reinforcement learning approach to produces a number of additional student behavior sequences that provides more coverage of the students' learning behaviors. These additional sequences when combined with the original student data is used to learn a new HMM model that we believe is a more complete description of the students' learning behaviors.

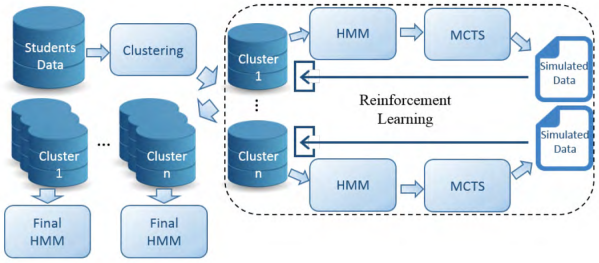


Figure 1: Architecture of the Overall Approach

2.2 HMM applied to Learner Modeling

A HMM is defined as a tuple, i.e., $\lambda = \{\mathbf{A}, \mathbf{B}, \pi\}$, where $\mathbf{A}$ and $\mathbf{B}$ represent state transition probability distribution and emission probability distribution matrices, respectively, while π is the initial state probability distribution [7]. Figure 2 presents the state diagram of a simple HMM example trained on two action sequences S_1 and S_2 with only 4 action types. Although not explicitly shown in the action sequences, the hidden states h_1 and h_2 can be interpreted as IA state (searching for and reading resources) and SC state (editing concept entities and causal links) respectively.

Based on the different probability distribution for each observation (action), the hidden states can be labeled by the primary actions associated with that state. The transitions between states capture changes in student activities over time, as also frequent patterns of activities, e.g., frequent occurrence of information acquisition followed by solution construction patterns.

2.3 Reinforcement Learning using MCTS

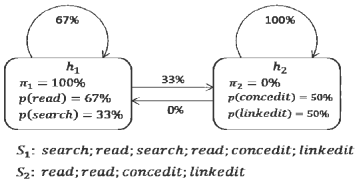


Figure 2: Simple HMM example.

To learn accurate and robust HMMs, it is important that the data set cover the range of behaviors a student exhibits in sufficiently large numbers.. However, given that we have limited student activity data on the system, we suffer from the data impoverishment problem. To address this problem, we propose a novel reinforcement learning method using Monte Carlo Tree Search (MCTS) and combine it with an initially derived HMM model to generate artificial data that matches students' learning behaviors. For generating action sequences that simulate actual students' behavior, we build the MCTS tree and traverse it to iteratively pick the next best node (with highest number of simulations) as the new action and add it to the tail of the sequence. In the reinforcement learning process as illustrated in Figure 1, we repeatedly generate simulated action sequences that maximize a specified reward function, and add them to the previously generated data. The reinforced data set is used to construct a refined version of the HMMs.

MCTS performs an iterative search with each iteration consists of 4 steps, i.e., ***Selection***, ***Expansion***, ***Simulation*** and ***Backpropagation*** [3]. In most MCTS implementations, the Upper Confidence bounds applied to Trees (UCT) algorithm is applied as the reward function for ***Selection***:

$$UCT = \frac{w_i}{n_i} + c\sqrt{\frac{\ln t}{n_i}} \quad (1)$$

where n_i is the number of simulations performed after adding the ith action; c is the exploration parameter with a typically chosen empirical value of $\sqrt{2}$; t is the total number of simulation runs for the parent node, which is equal to the sum of all the n_i; w_i is the sum of wins (1's) for all simulations after adding the ith action.

We adopt a similar reward function and compute the w_i value for generating action sequences that form a *Reinforced scaffolding model*. In this model, the normalized simulation results in the range of lowest-to-highest performance measure are summed up to compute w_i. For example, an action sequence has $w_i = 1$ when it achieves the max map score (i.e., 15) in Betty's Brain. This allows MCTS to better utilize coherence relations [8] to generate action sequences with more effective SC actions. The resulting HMM will favor the use of more coherent actions and be able to capture evolvement of learning behaviors/strategies that lead to better learning performance. Such behavioral and strategic evolvements can provide the basis for adaptive scaffolding.

We use the HMM to constrain the ***Expansion*** and ***Simu-***

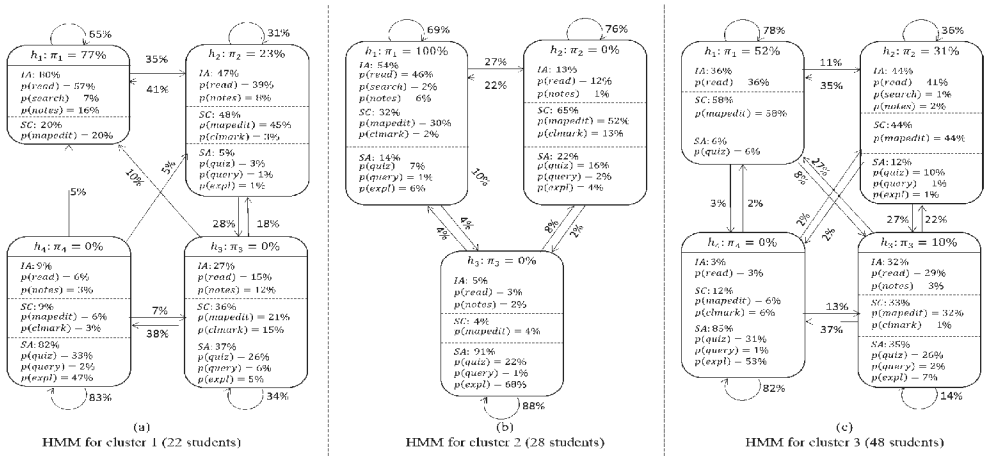


Figure 3: HMMs for the three clusters

Table 1: Comparison of the Three Clusters

	IA state	SA state	Balanced IA&SC state	Balanced SC&SA state	Search & Note Actions rate	Better strategic state transitions	$\overline{S_g}$	$\overline{S_m}$
Cluster 1	h_1	h_4	h_2	h_3	High	Yes	6.22	7.5
Cluster 2	h_1	h_3	-	-	Low	No	2.85	-2.25
Cluster 3	-	h_4	h_2	h_3	Low	Yes	5.61	3.79

lation steps to prevent expanding unvisited nodes and associated actions that are are not likely to occur in a given state. With these simulation and expansion policies, we can always generate action sequences that fit the HMM within a specified variance range. Figure 4 shows a simple example of generating artificial action sequence by applying MCTS.

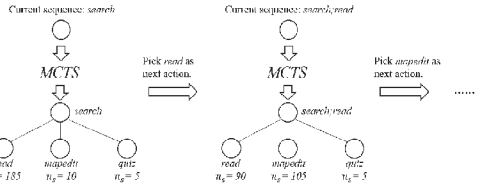


Figure 4: Simple example of applying MCTS for generating action sequence. n_s is the number of simulations performed during MCTS.

3. EXPERIMENTS AND ANALYSIS

We use data from a Betty' Brain study run with 98 6th grade middle school students in a science classroom for our experiments. A HMM clustering algorithm [6] is applied to discover groups of action sequences with high within-cluster homogeneities. This algorithm produced 3 clusters with the highest Partition Mutual Information value. HMMs for the three clusters are represented by the state diagrams shown in Figure 3, where h_i represents the ith hidden state with corresponding initial probability π_i. State transition probabilities are marked on the transition links while emission probability of an action a in a state diagram is given by $p(a)$. For measuring students performance in the different clusters, we denote the average pre- and post-test score gain as $\overline{S_g}$ and denote the average final causal map score of the group as $\overline{S_m}$. We combine this information to interpret and compare students' behaviors in the three different groups as shown in Table 1.

As we can see from Table 1, all three clusters have a SA state (primarily focusing on SA actions). However, Cluster 3 doesn't have an IA state, while Cluster 2 doesn't have states that balances efforts between IA & SC, and SC & SA. These balanced efforts are aimed to use acquired information or solution assessment results to support subsequent SC actions. Besides, only Cluster 1 maintains a good proportion of Search & Note actions which are considered to be more active as for acquiring information. Students in Cluster 1 and 3 did better in strategic state transitions, while for Cluster 2, self transitions dominated in all states. The performance measures of students in Cluster 1, i.e., $\overline{S_g}$ and $\overline{S_m}$, are the best among all three clusters.

3.1 Reinforced Scaffolding Model Analysis

The reinforced scaffolding model as described in section 2.3 is aimed to capture useful behavioral and strategic evolutions. To validate it, we analyze the generated reinforced HMMs along with artificial action sequences that equal the sample size of original data set. The reinforced HMMs are shown in Figure 5.

Compared to the original HMMs (Figure 3), the HMMs for the three clusters gradually converge to a isomorphic 3-state HMM structure. The differences between original and refined HMMs can be summarized as (1) the HMMs tend to redistribute the efforts made between IA & SC, as well as SC & SA, e.g., the proportion of IA in h_1 is decreased for

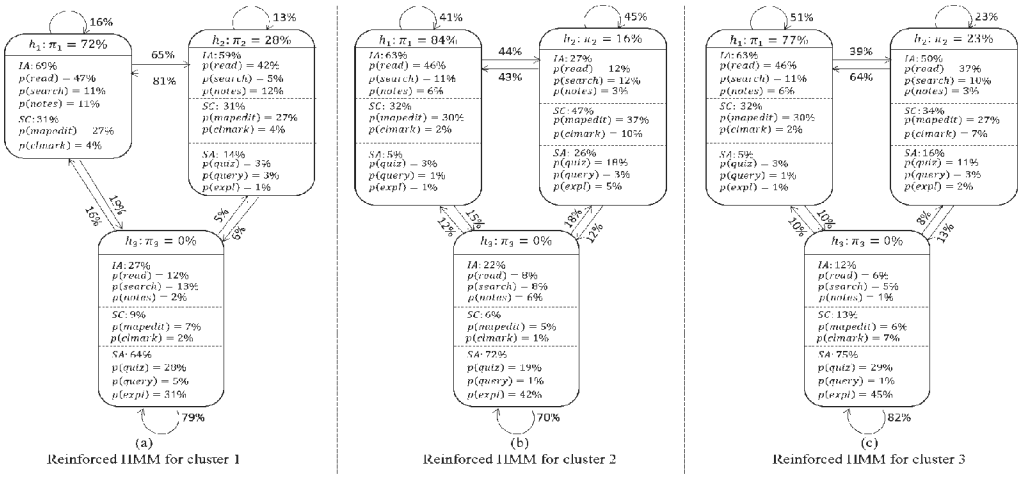


Figure 5: Reinforced Scaffolding HMMs for the three clusters

cluster 1 but it is increased for the other two clusters. Given the probability of IA supporting SC, $P_{ia-sc} = 0.43 \approx 3:7$ according to statistics, the reinforced HMMs tend to have all SC actions to be supported by at least one IA action by converging emission probability of IA and SC towards a ratio of 70% : 30%. This is because the SC actions being supported by IA actions have higher probability to be effective (the ratio for unsupported:supported *mapedit*s to be correct is 0.41 : 0.53); and (2) the usage frequency for actions, such as *search*, increase significantly, especially for clusters 2 and 3. An explanation for this phenomena is that in the few cases that *search* appeared in the original data set, it is very likely followed by a *read* that supports a subsequent *mapedit*. The original HMM captures this pattern by having a hidden state h_s with relatively high emission probability for *search*, *read* and *mapedit*. When it expands to a node with *search* action during MCTS, the posterior probability for the hidden state to remain in h_s is high and, therefore, further expansion can form this specific pattern and result in a higher chance of correct *mapedit*. Since the reward function is designed to optimize the causal map score, the reinforcement learning is likely to follow this pattern more frequently when generating artificial action sequences.

4. DISCUSSION AND CONCLUSIONS

In this paper, we proposed a novel reinforcement learning method for learner modeling, which integrated Hidden Markov Model and Monte Carlo Tree Search within a Reinforcement learning framework to generate more accurate learner models for groups of students. We applied the HMM clustering algorithm to divide students into groups based on their behaviors. Analysis and interpretation on these groups are presented to explain the clustering results.

We then used data of student activities collected from a study with the Betty's Brain OELE and generated reinforced data sets along with the *Reinforced scaffolding model*. The experiments showed promising results according to our interpretation, where we were able to generate and interpret reinforced HMMs by analyzing evolvements of learning behaviors that can lead to better performance in building causal maps.

In future work, we will develop scaffolding methods to support students' learning new, more productive behaviors and strategies as they work on the system. And it will be of interest to study how our reinforcement learning method works with longitudinal studies on students and collect data across longer periods of time to generate dynamic coherence models. Besides, we will collect data from other learning environments, or even data from other domains to see how well our modeling methods perform.

5. REFERENCES

[1] G. Biswas et al. From design to implementation to practice a learning by teaching system: Betty's Brain. *International Journal of Artificial Intelligence in Education*, 26(1):350–364, 2016.

[2] J. E. Brophy. *Motivating students to learn*. Routledge, 2013.

[3] C. B. Browne et al. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

[4] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[5] K. Leelawong and G. Biswas. Designing learning by teaching agents: The Betty's Brain system. *IJ Artificial Intelligence in Education*, 18(3):181–208, 2008.

[6] C. Li and G. Biswas. A Bayesian approach to temporal data clustering using Hidden Markov Models. In *ICML*, pages 543–550, 2000.

[7] L. R. Rabiner. Readings in speech recognition. chapter A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[8] J. R. Segedy, J. S. Kinnebrew, and G. Biswas. Using coherence analysis to characterize self-regulated learning behaviours in open-ended learning environments. *test*, 2(1):13–48, 2015.

Estimating Individual Treatment Effect from Educational Studies with Residual Counterfactual Networks

Siyuan Zhao
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609, USA
szhao@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609, USA
nth@wpi.edu

ABSTRACT

Personalized learning considers that the causal effects of a studied learning intervention may differ for the individual student. Making the inference about causal effects of studies interventions is a central problem. In this paper we propose the Residual Counterfactual Networks (RCN) for answering counterfactual inference questions, such as "Would this particular student benefit more from the video hint or the text hint when the student cannot solve a problem?". The model learns a balancing representation of students by minimizing the distance between the distributions of the control and the treated populations, and then uses a residual block to estimate the individual treatment effect based on the representation of the student. We run experiments on semi-simulated datasets and real-world educational online experiment datasets to evaluate the efficacy of our model. The results show that our model matches or outperforms the state-of-the-art.

Keywords

Counterfactual inference, deep residual learning, educational experiments, individual treatment effect

1. INTRODUCTION

The goal of personalized learning is to provide pedagogy, curriculum, and learning environments to meet the needs of individual students. For example, an Intelligent Tutor System (ITS) decides which hints would most benefit a specific student. If the ITS could infer what the student performance would be after receiving each hint, then it would simply choose the hint which leads to the best performance for the student. To make this possible, we might run an online educational experiment by randomly assigning students to one of the hints, and collect student performance. Then making predictions about causal effects of possible interventions (e.g. available hints) becomes a central problem in this case. In this paper we focus on the task of answering counterfactual questions [8] such as, "Would this particular student benefit more from the video hint or the text hint when the student cannot solve a problem?"

There are two ways of collecting data for counterfactual inference: randomized control trials (RCTs) and observational studies. In RCTs, participants (e.g. students) are randomly assigned to interventions (e.g. video hints or text hints), while participants in observational studies are not essentially randomly assigned to interventions. For example, consider the experiment of evaluating the efficacy of video hints and text hints for a certain problem. Under the design of RCT, students who need a hint would be randomly assigned to either the video hints or the text hints. In an observational study, students are assigned to one of the interventions based on their contextual information, such as knowledge level or personal preference.

[5] proposed Balancing Neural Networks (BNN) which can be applied to solve the counterfactual inference problem. They used a form of regularizer to enforce the similarity between the distributions of representations learned for populations with different interventions, for example, the representations for students who received text hints versus those who received video hints.This reduces the variance from fitting a model on one distribution and applying it to another. Because of random assignment to the interventions in RCTs, the distributions of the populations within different interventions are highly likely to be identical. However, in the observational study, we may end up with the situation where only male students receive video hints and female students receive text hints. Without enforcing the similarity between the distributions of representations for male and female students, it is not safe to make a prediction of the outcome if male students receive text hints. In machine learning, "domain adaptation" [7] refers to the dissimilarity of the distributions between the training data and the test data.

Recent work [6] has demonstrated that (deep) neural networks can be used with domain adaptation approaches to produce outstanding results on some domain adaptation benchmark datasets. Motivated by their work, we propose the Residual Counterfactual Networks (RCN) for the counterfactual inference to estimate the individual treatment effect and evaluate its efficacy in both a simulated dataset and a real-world dataset from an educational online experiment. The RCN extends the BNN by adding a residual block to estimate the individual treatment effect (ITE) based on the learned representation of participants. The idea of the resid-

ual block is originated from the state-of-the-art deep residual learning [2]. We enable the estimation of ITE by plugging several layers into neural networks to explicitly learn the residual function with reference to the learned representation.

The rest of the paper is organized as follows. Section 2 provides an overview of the problem setup of counterfactual inference for estimating the ITE. Section 3 details information of our model. Section 4 gives an overview of related work in this research area. Section 5 describes the datasets and evaluation metrics used to test our model. Section 6 presents the results of our model and compares them with other models. Finally, we discuss the results and conclude the paper.

2. PROBLEM SETUP

Let $\mathcal{T}$ be the set of proposed interventions we wish to consider, X the set of participants, and Y the set of possible outcomes. For each proposed intervention $t \in \mathcal{T}$, let $Y_t \in Y$ be the potential outcome for x when x is assigned to the intervention t. In randomized control trial (RCT) and observed study, only one outcome is observed for a given participant x; even if the participant is given an intervention and later the other, the participant is not in the same state. In machine learning, "bandit feedback" refers to this kind of partial feedback. The model described above is also known as the Rubin-Neyman causal model [11, 10].

We focus on a binary intervention set $\mathcal{T} = \{0, 1\}$, where intervention 1 is often referred as the "treated" and intervention 0 is the "control." In this scenario the ITE for a participant x is represented by the quantity of $Y_1(x) - Y_0(x)$. Knowing the quantity helps assign participant x to the best of the two interventions when making a decision is needed, for example, choosing the best intervention for a specific student when the student has a trouble solving a problem. However, we cannot directly calculate ITE due to the fact that we can only observe the outcome of one of the two interventions.

In this work we follow the common simplifying assumption of no-hidden confounding variables. This means that all the factors determining the outcome of each intervention are observed. This assumption can be formalized as the strong ignorability condition:

$$(Y_1, Y_0) \perp t|x, 0 < p(t = 1|x) < 1, \forall x.$$

Note that we cannot evaluate the validity of strong ignorability from data, and the validity must be determined by domain knowledge.

In the "treated" and the "control" setting, we refer to the observed and unobserved outcomes as the factual outcome $y^F(x)$, and the counterfactual outcome $y^{CF}(x)$ respectively. In other words, when the participant x is assigned to the "control" ($t = 0$), $y^F(x)$ is equal to $Y_1(x)$, and $y^{CF}(x)$ is equal to $Y_0(x)$. The other way around, $y^F(x)$ is equal to $Y_0(x)$, and $y^{CF}(x)$ is equal to $Y_1(x)$.

Given n samples $\left\{(x_i, t_i, y_i^F)\right\}_{i=1}^n$, where $y_i^F = t_i \cdot Y_1(x_i) + (1-t_i)Y_0(x_i)$, a common approach for estimating the ITE is to learn a function $f : X \times T \to Y$ such that $f(x_i, t_i) \approx y_i^F$. The estimated ITE is then:

$$\hat{ITE}(x_i) = \begin{cases} y_i^F - f(x_i, 1-t_i), & t_i = 1. \\ f(x_i, 1-t_i) - y_i^F, & t_i = 0. \end{cases}$$

We assume n samples $\left\{(x_i, t_i, y_i^F)\right\}_{i=1}^n$ form an empirical distribution $\hat{p}^F = \{(x_i, t_i)\}_{i=1}^n$. We call this empirical distribution $\hat{p}^F \sim p^F$ the empirical factual distribution. In order to calculate ITE, we need to infer the counterfactual outcome which is dependent on the empirical distribution $\hat{p}^{CF} = \{(x_i, 1-t_i)\}_{i=1}^n$. We call the empirical distribution $\hat{p}^{CF} \sim p^{CF}$. The p^F and p^{CF} may not be equal because the distributions of the control and the treated populations may be different. The inequality of two distributions may cause the counterfactual inference over a different distribution than the one observed from the experiment. In machine learning terms, this scenario is usually referred to as domain adaptation, where the distribution of features in test data are different than the distribution of features in training data.

3. MODEL

We proposed RCN to estimate individual treatment effect using counterfactual inference. The RCN first learns a balancing representation of deep features $\Phi : X \to R^d$, and then learns a residual mapping Δf on the representation to estimate the ITE. The structure of the RCN is shown in the left side of Figure 1.

To learn a representation of deep features Φ, the RCN uses fully connected layers with ReLu activation function, where $Relu(z) = max(0, z)$. We need to generalize from factual distribution to counterfactual distribution in the feature representation Φ to obtain accurate estimation of counterfactual outcome. The common successful approaches for domain adaptation encourage similarity between the latent feature representations w.r.t the different distributions. This similarity is often enforced by minimizing a certain distance between the domain-specific hidden features. The distance between two distributions is usually referred to as the discrepancy distance, introduced by [7], which is a hypothesis class dependent distance measure tailored for domain adaptation.

In this paper we use an Integral Probability Metric (IPM) measure of distance between two distributions $p_0 = p(x|t = 0)$, and $p_1 = p(x|t = 1)$, also known as the control and treated distributions. The IPM for p_0 and p_1 is defined as

$$\text{IPM}_{\mathcal{F}}(p_0, p_1) := \sup_{f \in \mathcal{F}} \left| \int_S f dp_0 - \int_S f dp_1 \right|,$$

where $\mathcal{F}$ is a class of real-valued bounded measurable functions on S.

The choice of functions is the crucial distinction between IPMs [15]. Two specific IPMs are used in our experiments: the Maximum Mean Discrepancy (MMD), and the Wasserstein distance. When $\mathcal{F} = \left\{f : \|f\|_{\mathcal{H}} \leqslant 1\right\}$, where $\mathcal{H}$ represents a reproducing kernel Hilbert space (RKHS) with k as its reproducing kernel, $\text{IPM}_{\mathcal{F}}$ is called MMD. In other words, the family of norm-1 reproducing kernel Hilbert space

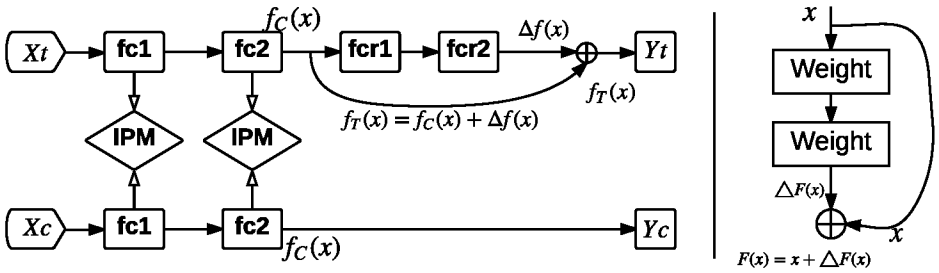


Figure 1: (left) Residual Counterfactual Networks for counterfactual inference. IPM is adopted on layers fc1 and fc2 to minimize the discrepancy distance of the deep features of the control and the treated populations. For the treated group, we add a residual block fcr1-fcr2 so that $f_T(x) = f_C(x) + \Delta f(x)$; (right) Residual block

(RKHS) functions lead to the MMD. The family of 1-Lipschitz functions $\mathcal{F} = \{f : \|f\|_L \leq 1\}$, where $\|f\|_L$ is the Lipschitz semi-norm of a bounded continuous real-valued function f, make IPM the Wasserstein distance. Both the Wasserstein and MMD metrics have consistent estimators which can be efficiently computed in the finite sample case [14]. The important property of IPM is that $p_0 = p_1$ iff $\text{IPM}_{\mathcal{F}}(p_0, p_1) = 0$.

The representation with reduction of the discrepancy between the control and the treated populations helps the model to focus on balancing features across two populations when inferring the counterfactual outcomes. For instance, if in an experiment, almost no male student ever received intervention A, inferring how male students would react to intervention A is highly prone to error and a more conservative use of the gender feature might be warranted.

After balancing the feature representations of the control and the treated populations, the next step is to infer the treatment effect for participant x. We adopt the residual block [2] to estimate the treatment effect.

As shown in the right side of Figure 1, $F(x)$ is the underlying desired function mapping. Instead of stacking a number of layers to fit the desired $F(x)$, we let stacked fully connected layers learn the residual mapping $\Delta f(x) = F(x) - x$. Then the origin mapping is converted into $\Delta f(x) + x$. The operation $\Delta f(x) + x$ is performed by a shortcut connection and an element-wise addition. Learning residual mapping is favored over fitting the desired mapping directly, because it is easier to find the residual with reference to an identity mapping than to learn the mapping as new.

The goal of the residual block is to approximate a residual function Δf such that $f_T(x) = f_C(x) + \Delta f(f_C(x))$, where f_C is the deep representation of participant x before being fed into the output layer, and f_T is the input to the output layer for the treated population. The output layer is a ridge linear regression to generate the final outcome. From the definition of the residual function Δf, we see that $\Delta f(x)$ is the estimated treatment effect for participant x, which is our interest in a control and treated experiment. With the residual block directly connected to fc2, the residual function $\Delta f(x)$ is dependent on the feature representation of participant x.

We plug in the residual block (shown in Figure 1) between fc2 layer and final output layer for the treated population in order to estimate the ITE. There is no residual block plugged in between fc2 layer and the final output layer for the control population. The final output layer $\varphi(\cdot)$ is a linear regression to calculate the predicted outcome, such that $Yc = \varphi(f_C(x))$, and $Yt = \varphi(f_T(x))$.

Recall the problem setup described above that there exist n samples $\{(x_i, t_i, y_i^F)\}_{i=1}^n$, where $y_i^F = t_i \cdot Y_1(x_i) + (1 - t_i)Y_0(x_i)$. In the control and the treated setting, we assume that $n_c (n_c > 0)$ samples $\left\{(x_i, 0, y_i^{(0)})\right\}_{i=1}^{n_c} \sim D_c$ are assigned to the control ($t = 0$), and $n_t (n_t > 0)$ samples $\left\{(x_i, 1, y_i^{(1)})\right\}_{i=1}^{n_t} \sim D_t$ are assigned to the treated ($t = 1$), such that $n = n_c + n_t$. As described above, RCN is an integration of deep feature learning, feature representation balancing, and treatment effect estimation in an end-to-end fashion with the loss function as such:

$$\min_{f_T = f_S + \Delta f(f_S)} \frac{1}{n_c} \sum_{i=1}^{n_c} L(f_c(\mathbf{x}_i), y_i^{(0)}) + \frac{1}{n_t} \sum_{i=1}^{n_t} L(f_t(\mathbf{x}_i), y_i^{(1)}) + \lambda \cdot \text{IPM}(D_c, D_t),$$

where λ is the tradeoff parameter for the IPM penalty, L is the loss function of the model. In the case of binary classification, L is the standard cross entropy. In the case of regression, L is root-mean-square error (RMSE). During the training, the model only has the access to the factual outcome.

4. RELATED WORK

From a conceptual point of view, our work is inspired by the work on domain adaptation and deep residual learning. [6] proposed the Residual Transfer Network that adopt MMD distance to learn transferable deep features from labeled data in the source domain and unlabeled data in the

target domain and adds a residual block to transfer the prediction classifier from the target domain to the source domain. The structure of our model is similar to that of their model. Deep residual learning is introduced by [2], the winner of the ImageNet ILSVRC 2015 challenge, to ease the training of deep networks. The residual block is designed to learn residual functions $\Delta F(\mathbf{x})$ with reference to the layer input $\mathbf{x}$. Reformulating layers to the residual block makes the training easier than directly learning the original functions $F(\mathbf{x}) = \Delta F(\mathbf{x}) + \mathbf{x}$.

Our model extends the work by [5, 13], where the authors build a connection between domain adaptation and counterfactual inference. They use IPMs, such as MMD and wasserstein distance, to learn a representation of the data which balances the control and treated distributions. The treatment assignment is concatenated with the representation to predict the factual outcome as while the reverse treatment assignment is concatenated with the representation to predict the counterfactual outcome. Compared to their work, we add a residual block to estimate the individual treatment effect based on the representation. [17, 1] proposed random causal forests (RCF) which is built upon the idea of random forests to estimate the heterogeneous treatment effect.

5. EXPERIMENTS

5.1 Evaluation Metrics

To compare among various models, we report the RMSE of estimated individual treatment effect, denoted

$$\epsilon_{ITE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((Y_1(x_i) - Y_0(x_i)) - I\hat{TE}(x_i))^2},$$

and the absolute error in average treatment effect

$$\epsilon_{ATE} = \left|\frac{1}{n}\sum_{i=1}^{n}(f_t(x_i) - f_s(x_i)) - \frac{1}{n}\sum_{i=1}^{n}(Y_1(x_i) - Y_0(x_i))\right|.$$

Following [4, 5], we report the Precision in Estimation of Heterogeneous Effect (PEHE),

$$PEHE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((Y_1(x_i) - Y_0(x_i)) - (\hat{y_1}(x_i) - \hat{y_0}(x_i))^2}.$$

Compared to the fact that achieving a small RMSE of estimated ITE needs the accurate estimation of counterfactual responses, a good (small) PEHE requires the accurate estimation of both factual and counterfactual responses.

However, calculating ϵ_{ITE}, ϵ_{ATE}, and PEHE requires the "ground truth" of the ITE for each participant in the experiment. We cannot gather the counterfactual outcomes from RCTs and observational studies, and thus do not have the ITE of each participant. We cannot evaluate ϵ_{ITE} and PEHE on these datasets. In order to evaluate the performance on these datasets across various models, we use a measure, called policy risk, introduced by [13]. Given a model f, the participant x is assigned to the treatment $\pi_f(x) = 1$ if $f(x,1) - f(x,0) > \lambda$ (in the case of RCN, $\Delta f > \lambda$), where λ is the treatment threshold, and to the control $\pi_f(x) = 0$ otherwise. The risk policy is defined as:

$$R_{Pol}(\pi_f) = 1 - (\mathbb{E}[Y_1|\pi_f(x) = 1] \cdot p(\pi_f = 1) + \mathbb{E}[Y_0|\pi_f(x) = 0] \cdot p(\pi_f = 0)).$$

The empirical estimator of the risk policy on a dataset is calculated by:

$$\hat{R}_{Pol}(\pi_f) = 1 - (\mathbb{E}[Y_1|\pi_f(x) = 1, t = 1] \cdot p(\pi_f = 1) + \mathbb{E}[Y_0|\pi_f(x) = 0, t = 0] \cdot p(\pi_f = 0)).$$

To obtain the policy risk, we use the method introduced by [16]. We select a subset of participants in the dataset where the treatment recommendation inferred by the model is the same as the treatment assignment in the experiment and then calculate the average loss from the subset of the data (see Table 1 for illustrative data).

For the datasets without the "ground truth" on ITE, we also calculate the average treatment effect on the treated by $\text{ATT} = \frac{1}{n_t}\sum_{i=1}^{n_t} y_i^{(1)} - \frac{1}{n_s}\sum_{i=1}^{n_s} y_i^{(0)}$, and report the error on ATT as $\epsilon_{ATT} = \left|\text{ATT} - \frac{1}{n_t}\sum_{i=1}^{n_t}(f_t(x_i) - f_s(x_i))\right|$.

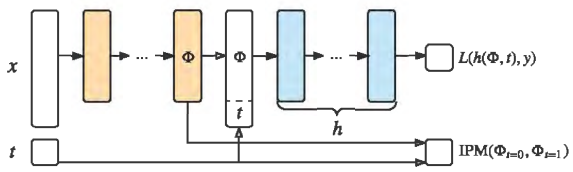


Figure 2: CFR for ITE estimation. L is a loss function, IPM is an integral probability metric

5.2 Baselines

Balancing Neural Networks (BNN) is a neural networks-based model for counterfactual inference. Compared to RCN, it has exactly the same fc1 and fc2 layers with IPM regularizer to learn the representation $\Phi(x)$ of the participant x. However, instead of using residual block to estimate treatment effect, it concatenates the treatment assignment t_i to the output of fc2 layer $\Phi(x)$ and feeds $[\Phi(x_i), t_i]$ to another two fully connected layers to generate the predicted outcome. We refer to this particular structure of BNN as BNN-2-2, following [5].

The Counterfactual Regression (CFR) [13] is built on the BNN. The important difference between these two models is that the CFR uses a more powerful distribution metric in the form of IPMs to learn a balancing representation. We compare our model with BNN-2-2 and CFR to verify the efficacy of residual block in terms of estimating individual treatment effect.

We introduce a simple neural networks baseline model to evaluate the efficacy of the IPM regularizer and residual mapping. This baseline model is a feed-forward neural networks model with four hidden layers, trained to predict the factual outcome based on X and t, without the IPM regularizer and the residual block. We refer to this as NN-4.

Table 1: Hypothetical data for some example students. The predicted outcome is the probability that the student would complete the assignment. Students in bold are those whose randomized treatment assignment is congruent with the recommendation of the counterfactual inference model. Data from these students would be used to calculate the policy risk.

ID	Group	Completion	Predicted outcome if treated	Predicted outcome if not treated	Treatment effect	Treat?
1	Control	1	0.8	0.75	0.05	1
2	**Control**	**0**	**0.3**	**0.45**	**-0.15**	**0**
3	**Treatment**	**0**	**0.50**	**0.38**	**0.12**	**1**
4	Treament	1	0.91	0.99	-0.08	0

5.3 Simulation based on real data - IHDP

The Infant Health and Development Program (IHDP) dataset was a semi-simulated dataset introduced by [4]. The dataset consists of a number of covariates from a real randomized experiment. The goal of the experiment is to study the impact of superior child care and home visits on future cognitive test scores. [4] discarded a biased subset of the treated population in order to introduce imbalance between treated and control subjects and used a simulated counterfactual outcome. Eventually, there are 747 subjects (139 treated, 608 control), each represented by 25 covariates assessing the attributes of the children and their mothers.

5.4 ASSISTments dataset

The ASSISTments online learning platform [3] is a free web-based platform utilized by a large user-base of teachers and students. The platform has been the subject of a recent study within the state of Maine [9], demonstrating significant learning gains for students using the platform. The dataset used in this work comes from one of 22 randomized controlled experiments [12] collected within the platform. This experiment was run in assignment types known as "skill builders" in which students are given problems until a threshold of understanding is reached; within ASSISTments, this threshold is traditionally three consecutive correct responses. Reaching this threshold denotes sufficient performance and completion of the assignment. In addition to this experimental data, information of the students prior to condition assignment is also provided in the form of problem-level log data providing a breadth of student information at fine levels of granularity.

In this experiment, there are two kinds of hints (video versus text) available for each problem from the assignment when students answer the problem incorrectly. The assignment to the video hint and the text video was random. Video content was designed to mirror text hint in an attempt to provide identical assistance. There are 147 students who received the video hint and 237 students who received the text hint. The dataset includes 15 covariates such as student past-performance history, class-past performance history. We solve a binary classification task which is to predict the completion of the assignment for each student.

6. RESULTS

The results of IHDP is presented in Table 2 when the treatment threshold $\lambda = 0$. We see that our proposed RCN performs the best on the dataset in terms of estimating ITE, ATE and PEHE. There is an especially large improvement on estimating ITE. These results indicate that the residual block $\Delta f(x)$ helps accurately predict the value of ITE based on the feature representation $\Phi(x)$ for a given participant x.

Table 2: Results of IHDP

Model	ϵ_{ITE}	ϵ_{ATE}	PEHE
NN-4	2.0	0.5	1.9
BNN-2-2	1.7	0.3	1.6
CFR	1.4	0.2	1.6
RCN	1.1	0.05	1.4

The results of ASSISTments dataset are the interest of our work since we hope to apply the RCN to educational experiments in order to support decision making in terms of personalized learning. The results in terms of policy risk and the average treatment effect on the treated are shown in Table 3 when the treatment threshold $\lambda = 0$. The model TA means "Treated All" where all students are assigned to the treatment while the model NT means "Not Treated" where all students are assigned to the control. Without considering that the effects of an intervention may differ for individual students, the model with the better performance out of these two models would be adopted when a choice must be made between these two interventions. The RCN, which considers the individual treatment effect, outperforms the TA and the NT. This indicates that taking the individual effect into account helps make a better choice of interventions. The comparison between the CFR and the RCN suggests that the RCN performs better than the CFR does in terms of risk policy and ATT.

To investigate the correlation between policy risk and treatment threshold λ, we plot the value of policy risk as a function of treatment threshold λ in Figure 3. For the results of the ASSISTments dataset from the CFR, the maximum predicted ITE in the dataset is 0.44. Once the threshold λ is larger than 0.44, the CFR is converted to "Not Treated" where all students are assigned to the control. Since the maximum predicted ITE in the ASSISTments dataset from the CFR is 0.18, the CFR is converted to "Not Treated" once the treatment threshold λ is larger than 0.18.

7. CONCLUSION

As online educational experiments become popular and easy to conduct, and machine learning becomes a major tool for researchers, counterfactual inference gains a lot of interest for the purpose of personalized learning. In this paper we

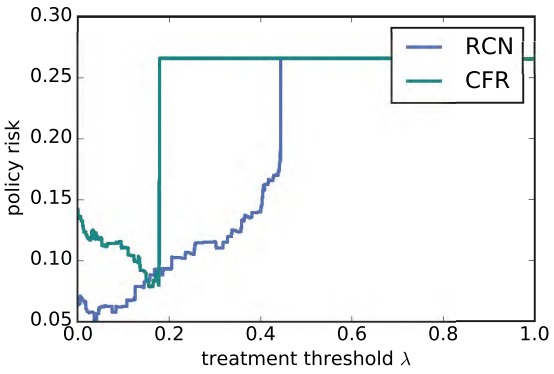


Figure 3: Treatment threshold versus policy risk on ASSISTments dataset. The lower policy risk is the better.

Table 3: Results of the ASSISTments Dataset

Model	$R_{\mathbf{POL}}$	ϵ_{ATT}
TA	0.14	-
NT	0.27	-
CFR	0.14	0.08
RCN	0.08	0.03

propose the Residual Counterfactual Networks (RCN) to estimate the individual treatment effect. Because of the dissimilarity between the distributions of the control and the treated populations, the RCN uses IPMs, such as Wasserstein and MMD distance, to learn balancing deep features from the data. A residual block is adopted on the deep features to learn the individual treatment effect (ITE) so that estimation of the ITE is dependent on the deep features. We apply our model to both synthetic datasets and real-world datasets from online educational experiment, indicating that our model achieves the state-of-the-art.

One open question for the future work is how to generalize our model for the situations where there is more than one treatment in the experiment. Integral Probability Metric (IPM) can only measure the distance between two distributions. We could use pair-wised IPM if there are more than two distributions. But this would be computationally time-consuming if the number of distributions increases. Since running experiments is expensive and collecting enough data for the model to make a reliable prediction is difficult, we need a better optimization algorithm which allows us to train the model efficiently.

8. ACKNOWLEDGMENTS

We acknowledge funding from multiple NSF grants (ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736 & DRL-1031398), the U.S. Department of Education (IES R305A120125 & R305C100024 and GAANN), the ONR, and the Gates Foundation.

9. REFERENCES

[1] S. Athey and G. Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, 5 July 2016.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. pages 770–778, 2016.

[3] 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.

[4] J. L. Hill. Bayesian nonparametric modeling for causal inference. *J. Comput. Graph. Stat.*, 20(1):217–240, 2011.

[5] F. Johansson, U. Shalit, and D. Sontag. Learning representations for counterfactual inference. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 3020–3029, 2016.

[6] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems 29*, pages 136–144. Curran Associates, Inc., 2016.

[7] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation: Learning bounds and algorithms. 2009.

[8] J. Pearl. Causal inference in statistics: An overview. *Stat. Surv.*, 3(0):96–146, 2009.

[9] J. Roschelle, M. Feng, R. F. Murphy, and C. A. Mason. Online mathematics homework increases student achievement. *AERA Open*, 1 Oct. 2016.

[10] D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *J. Educ. Psychol.*, 66(5):688, Oct. 1974.

[11] D. B. Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *J. Am. Stat. Assoc.*, 2005.

[12] D. Selent, T. Patikorn, and N. Heffernan. ASSISTments dataset from multiple randomized controlled experiments. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale*, L@S '16, pages 181–184, New York, NY, USA, 2016. ACM.

[13] U. Shalit, F. Johansson, and D. Sontag. Estimating individual treatment effect: generalization bounds and algorithms. 13 June 2016.

[14] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, G. R. Lanckriet, et al. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599, 2012.

[15] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, and G. R. G. Lanckriet. On integral probability metrics, φ-divergences and binary classification. 18 Jan. 2009.

[16] A. J. Vickers, M. W. Kattan, and S. Daniel. Method for evaluating prediction models that apply the results of randomized trials to individual patients. *Trials*, 8(1):14, 5 June 2007.

[17] S. Wager and S. Athey. Estimation and inference of heterogeneous treatment effects using random forests. 14 Oct. 2015.

Online Learning Persistence and Academic Achievement

Ying Fang
University of Memphis
365 Innovation Drive
Memphis, TN 38152
yfang2@memphis.edu

Benjamin Nye
University of Southern California
12015 Waterfront Drive
Playa Vista, CA 90094
nye@ict.usc.edu

Philip Pavlik
University of Memphis
365 Innovation Drive
Memphis, TN 38152
ppavlik@memphis.edu

Yonghong Jade Xu
University of Memphis
3720 Alumni Avenue
Memphis, TN 38152
yxu@memphis.edu

Arthur Graesser
University of Memphis
365 Innovation Drive
Memphis, TN 38152
graesser@memphis.edu

Xiangen Hu
University of Memphis
365 Innovation Drive
Memphis, TN 38152
xhu@memphis.edu

ABSTRACT

Student persistence in online learning environments has typically been studied at the macro-level (e.g., completion of an online course, number of academic terms completed, etc.). The current examines student persistence in an adaptive learning environment, ALEKS (Assessment and LEarning in Knowledge Spaces). Specifically, the study explores the relationship between students' academic achievement and their persistence during learning. By using archived data that included their math learning log data and performance on two standardized tests, we first explored student learning behavior patterns with regard to their persistence during learning. Clustering analysis identified three distinctive patterns of persistence-related learning behaviors: (1) High persistence and rare topic shifting; (2) Low persistence and frequent topic shifting; and (3) Moderate persistence and moderate topic shifting. We further explored the association between persistence and academic achievement. No significant differences were observed between academic achievement and the different learning patterns. We interpret this result in addition to a preliminary exploration of topic mastery trends, to suggest that "wheel-spinning" behaviors coexist with persistence, and is ultimately not beneficial to learning.

Keywords

ALEKS, persistence, academic achievement

1. INTRODUCTION

Assessment of LEarning in Knowledge Space (ALEKS) is an online adaptive learning system built based on Knowledge Space Theory [8]. According to Knowledge Space Theory, a knowledge domain is represented by a finite set of concepts. The knowledge state of a student in a domain can be represented by a particular subset of concepts that the student is capable of mastering. By gauging learner's knowledge state, ALEKS determines what a student knows and is ready to learn, and provides personalized learning paths that are ideal for each student [3]. When a learner first use ALEKS, the system starts with an individualized initial assessment to find the student's knowledge state. The assessment usually consists of 20 to 30 problems (out of more than 600 problems). After the initial assessment, the student receives a report in a color-keyed pie chart (as shown in Fig. 1). Each "slice" of the pie chart corresponds to a particular area of the syllabus, and the darker shades of color indicating how much the student has mastered in that area [1]. After the first assessment, ALEKS identifies the student's knowledge state and generates a list of topics the student is ready to learn in each area. Once a student chooses the area and topic he/she wants to work on, ALEKS will provide a set of problems, and the student learns by solving problems under a specific topic. After successfully solving problems covering the same topic, the system will determine a student's mastery of the topic and the add the topic to the student's knowledge pie, and the student can then move onto a new topic [2].

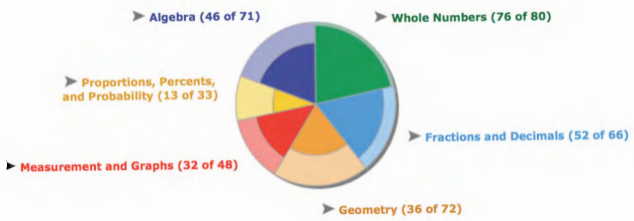


Figure 1: ALEKS knowledge pie showing number of concepts learner has learned and needs to learn

As one of the popular adaptive learning systems, ALEKS was evaluated in some empirical studies which were carried out in different settings, and was observed to be effective in most of the studies [6, 9, 12, 13, 16, 19]. These studies generally measured ALEKS students' learning gains or academic achievements; however, none of them looked at students' learning process, or online learning behaviors. In this study, we explored students' offline learning outcomes and online learning behavior patterns, and investigated whether persistence was associated with academic achievement in an individualized online learning environment. We further examined students' wheel-spinning behaviors [5] in order to understand the association.

2. RELATED WORK

In this section, we will introduce how persistence has been studied in different learning contexts--traditional classroom environment and online learning environment, and how the relationship between persistence and academic achievement has been investigated. Persistence is "the quality that allows someone to continue doing something or trying to do something even though it is difficult or opposed by other people" [15]. According to Rovai, persistence is the behavior of continuing action despite the

presence of obstacles [22]. Persistence in the face of adversity is often described as a result of high motivation. For instance, in the literature investigating classroom learning, persistence was typically examined as an outcome factor of motivation. Elliot and his colleagues [7] found mastery goals and performance approach goals were positive predictors of persistence; Vansteenkiste et al. [24] found intrinsic motivation improved student persistence; Multon et al. [18] proved that self-efficacy facilitated persistence. Although the concept of persistence was studied in different literature, it was operationalized in various ways. For example, in the meta-analysis by Multon and his colleagues [18], they summarized three ways of operationalizing persistence after viewing eighteen studies-- time spent on task, number of items or tasks attempted or completed, and number of academic terms completed. Apart from these three commonly used measures, persistence was also frequently measured with self-reports [4, 7, 27].

In the context of online learning environment, persistence was usually defined as the completion of an online course, or an antonym of attrition [10, 14, 20, 22]. Persistent learners, who were referred to as "completers", were the learners who successfully completed an online course. Non-persistent learners, who were referred to "dropouts", were the learners who did not finish a course [10, 14]. Persistence was mainly explored as a dependent variable affected by psychological and social factors, such as self-motivation, engagement, economic support, etc. [14]. Persistence was also investigated as a consequence correlated with online behaviors such as participation, discussion, etc. [17, 21].

Despite various studies on persistence in learning, persistence was rarely studied as a predictive factor. Stekel and Tobias [23] hypothesized a curvilinear relationship between self-estimated persistence and achievement. They predicted a moderate amount of persistence would lead to the highest achievement. They also hypothesized that persistence would be positively related to achievement in lecture-related instructional environment, but unrelated in the individualized instructional environment. However, they failed to prove their hypotheses. While examining the mediation effect of persistence on the relationship between goals and academic achievement, Elliot *et al.* [7] found self-reported persistence was a positive predictor of exam performance in lecture-based classroom setting. This proved one of Stekel and Tobias' hypotheses. For online learning system like ALEKS, the instructional context could be considered individualized because ALEKS models student's knowledge state and always provides the concepts students are ready to learn. Therefore, we wonder whether persistence is unrelated to academic achievement in the individualized learning environment like ALEKS.

3. METHODS

3.1 DATA SETS

The data sets used for this study were collected from Jackson-Madison Intelligent Tutoring System Evaluation (JMITSE) program. JMITSE was an after-school program applied in five middle schools in Jackson-Madison County School System of Tennessee from 2009 to 2012. The goal of JMITSE program was to investigate whether technology outperformed human teachers in math teaching. There were two experimental conditions: teacher condition and technology condition. In the teacher condition, students learned math with math teachers in the after-school program. In the technology condition, students learned math with ALEKS. For this study, we only used data from the ALEKS condition. The program lasted for three academic years and 366 sixth-graders were assigned to the ALEKS condition altogether. Participants were supposed to study for two one-hour sessions every week, for twenty-five weeks. Logs of all students' online learning activities were recorded by the system. The ALEKS log file included students' online ID, the topics (i.e. concepts) students attempted, learning mode (i.e. learning, review), time elapsed and the result of each attempt. For each attempt, there are five possible results: correct, wrong, explain, added to pie and failed. "Correct" is shown after a learner attempts a task and gets the correct answer. "Wrong" is shown after a learner attempts a task and gets a wrong answer. After a learner gets a wrong answer, two buttons "Try" and "Explain" will be shown to the learner. If the learner hits the "Try" button, he/she will be given another problem to work on. If the learner hits the "Explain" button, a worked example of that problem will be provided (as shown in Fig. 2). Reading an explanation is regarded as an attempt and the result is recorded as "Explain." "Added to Pie" is shown after learner attempts a problem correctly. The difference between "Added to Pie" and "Correct" is that "Correct" is based on one single attempt, but "Added to Pie" is based on multiple correct attempts. When a learner can correctly answer problems under a concept consistently, ALEKS decides the learner has mastered the concept and adds the concept to the learner's knowledge pie. After being added to the knowledge pie, that topic will not be given to the learner again, except for reviewing. "Failed" is shown after a learner attempts a task and answers incorrectly. Similar to "Added to Pie", it is not merely based on one single attempt, instead, it happens when there are multiple unsuccessful attempts and the system decides that the learner failed to learn that topic.

The participants of JMITSE took the Tennessee Comprehensive Assessment Program (TCAP), which is a standardized test, twice. Before entering the program, the students took TCAP5, which was TCAP for 5th graders. After finishing the program, the students took TCAP6, which was TCAP for 6th graders. The two tests were used as pretest and posttest in the analysis.

3.2 DATA PROCESS

The log file used in this study contains 366 students' 330,319 lines of online learning sequence. Each line represents an attempt from a student on one topic. Most students attempted multiple topics, and most topics were attempted multiple times. Therefore, for each student, there were multiple rows of data. Firstly, the data was aggregated at topic level. After aggregation, the number of observations for each individual student equaled to the number of topics they attempted. For each topic attempted by a student, we computed the number of attempts and amount of time spent on the topic, as well as whether it was mastered. We named the variables "Attempt", "Time" and "Master". Pearson product-moment correlation coefficient indicated that "Attempt" and "Time" were highly correlated (r=.98). To determine which variable to use as the measure of effort, we further examined the distribution of the two variables. The distribution of the two variables revealed that neither of them were normally distributed. However, after log transformation, "Attempts" became approximately normally distributed, but "Time" was still skewed (as shown in Fig. 2). Therefore, "Attempts" was chosen to measure student's effort on task. Next, we created three variables as measures of persistence and dummy coded them. They were "High persistence", "Moderate persistence" and "Switch". While "High persistence" and "Moderate persistence" were used to describe different levels

of persistent learning behaviors, "Switch" was used to describe non-persistent behaviors when a student gave up a topic quickly and switched to a new topic before mastery. For a topic, if its log-

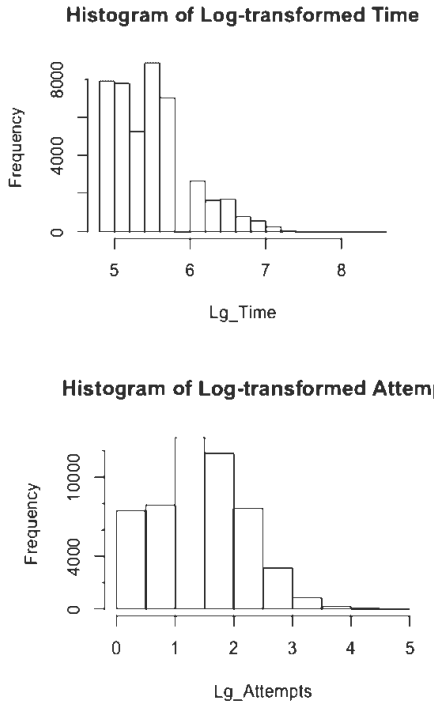


Figure 2: Distribution of log-transformed attempts and log-transformed time on each topic

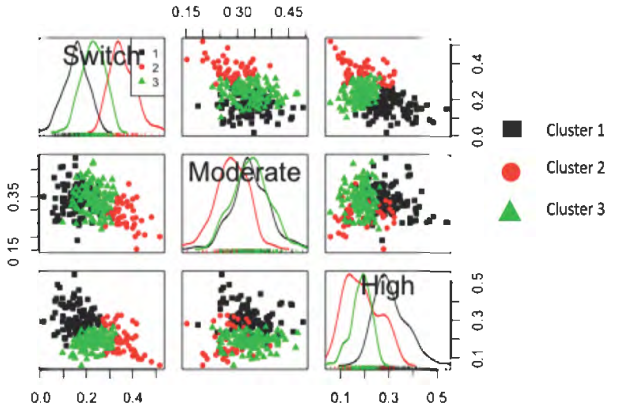
transformed attempts were in the fourth quartile of the distribution, "High persistence" was coded 1, otherwise it was coded 0. If its log-transformed attempts fell into the second or third quartile of the distribution, "Moderate persistence" was coded as 1, otherwise it was coded 0. For "Switch", both attempts and the result were taken into account. If a topic's log-transformed attempts was in the first quartile of the distribution, and the topic was not mastered, "switch" was coded 1, otherwise it was coded 0. After the new variables were created and coded, the 51,982 rows of data were aggregated to student level by averaging the persistence variables, and we got 366 observations. After second aggregation, the three persistence variables became continuous rather than binary. These variables represent the percentage of topics that a student persisted at each level. For instance, if a student gets 0.2 in "high persistence", it means that the student attempted twenty percent of the topics with high persistence. Lastly, we computed the number of topics each student attempted for data screening. The three persistence variables were percentages, which represented the percentage of topics attempted with some type of behaviors. If the total number of topics attempted by the were too small, it did not necessarily imply certain behavior patterns, even if the percentage for that behavior was high. Therefore, we decided to screen the students who only attempted a small number of topics. Based on the distribution of topics attempted by each student, the students whose attempted number of topics fell within the first quantile (Topics<=61) were screened from further analysis. There were 275 observations after screening.

After data process, we conducted cluster analysis to explore students' persistence learning patterns. We performed analysis of covariance to compare academic achievements of students from different groups to explore the association between online behavior and academic achievement. We also conducted analysis of variance to compare the mastery topics between groups to better understand the association.

4. RESULTS

4.1 CLUSTER ANALYSIS

There is no strictly defined sample size for cluster analysis. According to the suggestion of Formann [11], the minimal sample size should be no less than 2^k cases (k = number of variables), preferably $5*2^k$. The study examined the clustering of 275 observations across three variables, which fell comfortably within the accepted range. Ward's [25] hierarchical clustering technique was applied and the squared Euclidean distance was used to calculate the distance between clusters. A scree plot was used to determine the optimum number of clusters, where the levelling-off point indicated a reduced variability between clusters after it [26]. Examination of scree plot revealed flattening between three and four clusters, indicating that a three-cluster solution best captured the similarities and differences between students on the three variables. The cluster membership did not change by repeating the analysis, and significant differences were found by conducting ANOVAs for the clustering variables, which further confirmed the quality of the solution. The three-cluster solution is shown in Fig. 3. The scales are the percentage of topics students attempted with a specific behavior. The scales are the percentage of topics students attempted with a specific behavior. For example, the y axis of the top row is the percentage of switch behavior. The x axis of the top middle block is the percentage of moderate persistent learning behavior, and x axis of the top right block is the percentage of high persistent learning behavior. From the top middle block, we can find the clusters are more distinct on switch behavior (i.e. y axis), whereas on the moderate persistence behavior (i.e. x axis) there is more overlap between the student clusters. From the top right block, we can find the black cluster has more high persistent learning behavior, and the green and red clusters have more overlap. The descriptive statistics on the grouping variables and the academic achievement variables, that



we further explored, are shown in Table 1.

Figure 3: Scatterplot matrices of three-level persistence of three clusters

Cluster 1: High persistence, low switch

Cluster 1 (i.e. the black cluster in Fig. 3) accounts for 37.5% of the study sample (n=103). The students in this cluster switched topics less than members of other two clusters. The switching ratio of cluster 1 is 0.16, which indicates that students quickly gave up or switched to other topics before mastery for 16% of the tasks they attempted. For 34% of the tasks, the students worked with moderate persistence (i.e. attempted the task for 3-7 times). And for 31% of the tasks, the students worked with high persistence (i.e. attempted the task for 8 or more times). These students did not easily give up on tasks, and put a large amount of effort on one third of the tasks they got, which indicated that they were persistent learners.

Table 1: Mean scores and standard deviations for each variable by cluster

	Cluster 1 (n = 103)	Cluster 2 (n = 54)	Cluster 3 (n = 118)
Switch	0.16 (σ=0.05)	0.36 (σ=0.05)	0.23 (σ=0.05)
Moderate persistence	0.34 (σ=0.05)	0.28 (σ=0 .05)	0.34 (σ=0.05)
High persistence	0.31 (σ=0.07)	0.19 (σ=0.07)	0.18 (σ=0.04)
TCAP5	46.72 (σ=18.25)	39.37 (σ=17.60)	47.28 (σ=17.23)
TCAP6	43.23 (σ=20.89)	32.69 (σ=18.44)	40.49 (σ=21.63)

Cluster 2: Low persistence, high switch

Cluster 2 (i.e. the red cluster in Fig. 3) is a comparatively smaller cluster including 19.6% (n=54) of the study sample. The distinctive characteristics of this cluster is their high switching ratio. For 36% of the tasks they were given, the learners quickly gave up or switched to new tasks before mastering them. The students worked with moderate persistence (i.e. attempted the task for 3-7 times) on 28% of the tasks. And worked with high persistence for 19% of the tasks (i.e. attempted the task for 8 or more times). Compared with the other two clusters, the students in this cluster were not very persistent. Although they worked on some tasks with multiple attempts, they gave up on a large percentage of the tasks, and they were not willing to put too much effort on a task.

Cluster 3: Moderate persistence, moderate switch

Cluster 3 (i.e. the green cluster in Fig. 3) is the largest cluster with 118 students representing 42.8% of the study sample. The student in this cluster switched topics on 23% of the tasks, which is higher than that of Cluster 1 but lower than that of Cluster 2. They worked with moderate persistence on 34% of the tasks and with high persistence on 18% of the tasks. Compared to the other two clusters, this cluster does not distinctively stand out in any type of behavior. The students gave up a medium portion of topics and worked with high effort on a comparatively low portion of topics. They worked on the tasks with mostly moderate persistence. It seems they were regulating their learning in a rational way in the self-regulated learning environment.

4.2 ANALYSIS OF COVARIANCE (ANCOVA)

In order to investigate the association between persistence and academic performance, a one-way analysis of covariance (ANCOVA) was conducted to determine a statistically significant difference between three clusters on posttest scores controlling for pretest scores. The effect of cluster on posttest scores after controlling for pretest scores was not statistically significant, $F(2,212) = 1.25$, $p = .29$, which means the academic achievement of the three clusters with different behavior patterns were not significantly different from each other.

4.3 ANALYSIS OF VARIANCE (ANOVA) AND POST HOC TESTS

In order to understand why persistence was not related to academic achievement, we further examined the percentage of topics attempted with moderate persistence and high persistence. For clusters one, two and three, the percentages of tasks attempted with moderate persistence without mastery were 0.11 (σ = 0.05), 0.08 (σ = 0.04) and 0.07 (σ = 0.03), respectively. The percentages of tasks attempted with high persistence without mastery were 0.21(σ = 0.08), 0.17 (σ = 0.06) and 0.16 (σ = 0.06). Analysis of variance (ANOVA) indicated a significant difference of the unmastered topics attempted with moderate ($F (2, 272) = 30.3$, $p < .001$) and high persistence ($F(2,272) = 14.3$, $p < .001$) among the three clusters. Post-hoc tests indicated Cluster 1 was significantly

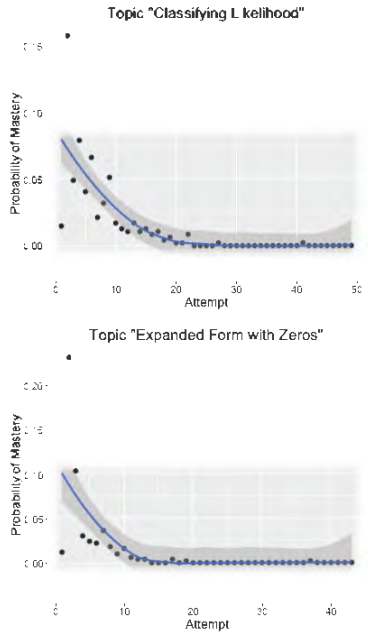


Figure 4: Mastery probability over attempts for topic "Classifying likelihood" and topic "Expanded form with zeros"

higher than both Cluster 2 and Cluster 3 in unmastered topics with both moderate and high persistence. This provides some insight as to why persistence did not make a difference in learning: the students were wheel-spinning [5]. We explored two highly attempted topics in our data sets and found the probability of mastering those topics got close to zero after a certain number of attempts (as shown in Fig. 4). This indicates the existence of wheel-spinning.

Another one-way analysis of variance (ANOVA) was conducted to determine a statistically significant difference between three clusters on the number of mastered topics at different difficulty levels. The topics were divided into three levels based on the percentage of students who mastered them. The topics in the first quartile had the highest mastery percentage, which we defined as easy topics. The topics in the second and third quartiles had the medium mastery percentage, and were defined as medium topics. The topics in the fourth quartile, had the lowest mastery percentage, and were defined as hard topics. The numbers of mastered easy topics were not found to be significantly different among three clusters, $F(2,272) = 2.56$, $p = .08$. However, the numbers of mastered medium ($F(2,272) = 9.98$, $p = 0$) and hard topics ($F(2,251) = 8.92$, $p = 0$) were found to be significantly different between clusters. Post-hoc tests indicated that cluster one and three mastered significantly more medium and hard topics than cluster two, but there was no statistically significant difference between cluster one and three. The means and standard deviations of the number of topics mastered by each cluster are shown in table 2.

Table 2: Means and standard deviations of the number of topics mastered by each cluster

	Cluster 1	Cluster 2	Cluster 3
Easy topics	24.06 (σ=12.1)	22.7 (σ=12.88)	27.21 (σ=15.08)
Medium topics	49.46 (σ=26.85)	33.56 (σ=18.74)	51.75 (σ=26.99)
Hard topics	16.11 (σ=13.93)	6.86 (σ=6.05)	14.27 (σ=12.13)

5. DISCUSSION AND CONCLUSION

In previous research, student persistence has only been measured by macro-level data (e.g., completion of an entire course). This study took a different approach by examining persistence at a more micro-level; specifically, we looked at student persistence within specific tasks in the ALEKS learning system. We were able to extract three distinct clusters of persistence related student behaviors through cluster analysis. The students in the high persistence cluster put medium to high effort in most of the topics they attempted, and they rarely switched to a new topic before mastery. The students in the moderate persistence cluster put medium effort in most topics they attempted and they did not easily give up topics before mastery. The students in the low persistence cluster frequently switched to new topics before mastery, often giving up tasks after one or two attempts. The comparison of students' academic achievement in the three clusters did not reveal any significant difference. This result is consistent with the hypothesis proposed by Stekel and Tobias [23], who suggested that persistence and achievement are unrelated within individual learning contexts. Although learning gains were not different between clusters in standardized tests, the mastery of topics was found to be different. The more persistent clusters--cluster one and cluster three-- mastered more medium and hard topics than the non-persistent cluster--cluster two. This suggests persistence was associated with learning in ALEKS, especially for more difficult topics. The inconsistency between learning gain in ALEKS and TCAPs might be related to different topics covered in ALEKS and TCAPs.

It is worth noting that the pretest and posttest assessments present a limitation to the current analysis. The TCAP5 and TCAP6 were used as pretest and posttest measures, and may cover different concepts that are not well aligned. However, a further look at the possible reasons behind non-productive persistence suggested wheel-spinning might relate to ineffective learning. That is, even though students were persistently working on a single topic, they appeared to be at an impasse. These impasses were not resolved with more attempts, which ultimately resulted in the student never mastering the topic. Although ALEKS has a system that can detect ineffective learning and provide feedback, like "Failed", to learners, the percentage of "Failed" was very low (i.e., 1%). In many cases, learners were struggling and wheel-spinning, but the system did not stop them with a "Failed" indicator, or any other type of intervention. Therefore, we suggest ALEKS to improve the mechanism to detect wheel-spinning and provide intervention in a timely manner.

REFERENCES

[1] ALEKS. (2016). "Assessment". Retrieved from https://www.aleks.com/about_aleks/assessment

[2] ALEKS. (2016). "Learning mode". Retrieved from https://www.aleks.com/about_aleks/learning_mode

[3] Albert, D., & Lukas, J. (Eds.). (1999). Knowledge spaces: Theories, empirical research, and applications. Psychology Press.

[4] Agbuga, B., & Xiang, P. (2008). Achievement goals and their relations to self-reported persistence/effort in secondary physical education: A trichotomous achievement goal framework. *Journal of Teaching in Physical Education*, *27*(2), 179.

[5] Beck, J. E., & Gong, Y. (2013). Wheel-spinning: Students who fail to master a skill. In *Artificial Intelligence in Education* (pp. 431-440). Springer Berlin Heidelberg.

[6] 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.

[7] Elliot, A. J., McGregor, H. A., & Gable, S. (1999). Achievement goals, study strategies, and exam performance: A mediational analysis. *Journal of educational psychology*, *91*(3), 549.

[8] Falmagne, J. C., Koppen, M., Villano, M., Doignon, J. P., & Johannesen, L. (1990). Introduction to knowledge spaces:

How to build, test, and search them. *Psychological Review*, *97*(2), 201.

[9] Fanusi, A., D. (2015). *The effect of ALEKS math support on standardized math test scores in middle school* (Doctoral dissertation). *ProQuest LLC.*

[10] Finnegan, C., Morris, L. V., & Lee, K. (2009). Differences by course discipline on student behavior, persistence, and achievement in online courses of undergraduate general education. *Journal of College Student Retention: Research, Theory & Practice*, *10*(1), 39-54.

[11] Formann, A. K. (1984). Latent Class Analysis. *Wiley StatsRef: Statistics Reference Online*.

[12] Fullmer, P. (2012). Assessment of tutoring laboratories in a learning assistance center. *Journal of College Reading and Learning, 42(2)*, 67-89.

[13] Hagerty, G., & Smith, S. (2005). Using the web-based interactive software ALEKS to enhance college algebra. *Mathematics and Computer Education, 39(3)*, 183.

[14] Hart, C. (2012). Factors associated with student persistence in an online program of study: A review of the literature. *Journal of Interactive Online Learning*, *11*(1), 19-42.

[15] *Merriam-Webster's collegiate dictionary* (10th ed.). (2003). Springfield, MA: Merriam-Webster Incorporated.

[16] Mertes, E. S. (2013). A Mathematics Education Comparative Analysis of ALEKS Technology and Direct Classroom Instruction (Doctoral dissertation). ProQuest LLC.

[17] 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.

[18] Multon, K. D., Brown, S. D., & Lent, R. W. (1991). Relation of self-efficacy beliefs to academic outcomes: A meta-analytic investigation. *Journal of counseling psychology*, *38*(1), 30.

[19] Nwaogu, E. (2012). The effect of ALEKS on students' mathematics achievement in an online learning environment and the cognitive complexity of the initial and final assessment. *Middle-secondary Education and Instructional Technology Dissertations*. Paper 94.

[20] Park, J. H., & Choi, H. J. (2009). Factors Influencing Adult Learners' Decision to Drop Out or Persist in Online Learning. *Educational Technology & Society*, *12*(4), 207-217.

[21] Rafaeli, S., & Ravid, G. (1997). Online, web-based learning environment for an information systems course: Access logs, linearity and performance. In *ISECON* , 97, 92-99.

[22] Rovai, A. P. (2003). In search of higher persistence rates in distance education online programs. *The Internet and Higher Education*, *6*(1), 1-16.

[23] Stekel, Karen W., & Sigmund Tobias. "Persistence and Achievement." (1977).

[24] Vansteenkiste, M., Simons, J., Lens, W., Sheldon, K. M., & Deci, E. L. (2004). Motivating learning, performance, and persistence: the synergistic effects of intrinsic goal contents and autonomy-supportive contexts. *Journal of personality and social psychology*, *87*(2), 246.

[25] Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, *58*(301), 236-244.

[26] 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.

[27] Xiang, P., & Lee, A. (2002). Achievement goals, perceived motivational climate, and students' self-reported mastery behaviors. *Research Quarterly for Exercise and Sport*, *73*(1), 58-65.

Using Temporal Association Rule Mining to Predict Dyadic Rapport in Peer Tutoring

Michael Madaio
Carnegie Mellon University
Pittsburgh, Pennsylvania
mmadaio@cs.cmu.edu

Rae Lasko
Carnegie Mellon University
Pittsburgh, Pennsylvania
rlasko@andrew.cmu.edu

Amy Ogan
Carnegie Mellon University
Pittsburgh, Pennsylvania
aeo@cs.cmu.edu

Justine Cassell
Carnegie Mellon University
Pittsburgh, Pennsylvania
justine@cs.cmu.edu

ABSTRACT

Social relationships, such as interpersonal closeness or rapport, can lead to improved student learning, but such dynamic, interpersonal phenomena can be difficult for educational support technologies to detect. In this paper, we describe an approach for rapport detection in peer tutoring, using temporal association rules learned from nonverbal, social, and on-task verbal behaviors. From a corpus of 60 hours of annotated multimodal peer tutoring data, we learn the temporal association between behaviors and the rapport score for each 30-second "thin-slice". We then train a stacked ensemble classification model on those association rules and evaluate our ability to reliably predict rapport using multimodal behavioral data. We find that our approach allows us to predict rapport well above chance, and more accurately than two baseline models. We are able to predict high rapport more accurately for strangers and low rapport more accurately for friends, which we believe holds promise for the integration of rapport detection into collaborative learning supports and intelligent tutoring systems.

Keywords

rapport, association rule mining, peer tutoring, social states

1. INTRODUCTION

Social relationships, such as the long-term closeness of friends or the short-term rapport built while getting to know someone, have been shown to result in benefits for student learning, such as increased help-seeking, productive cognitive conflict, and elaborated reasoning [2]. In collaborative learning settings, higher interpersonal rapport between students is associated with productive educational processes such as instances of transactive reasoning [13] and greater learning gains over time [18]. Educational technologies, such as intelligent tutoring systems (ITS) and pedagogical agents, increasingly attempt to reap the benefits of interpersonal closeness and rapport between humans and agents to improve engagement, motivation, or trust in the pedagogical agent [19]. However, before educational technologies can respond appropriately to the rapport between collaborating students, or build rapport between students and a pedagogical agent, they must first model that rapport as it changes over time, given the available behavioral data. The educational data mining community has developed, over the last several decades, detectors of individual student phenomena such as frustration, boredom, engagement, carelessness, and many others [3, 17], but it has developed relatively fewer methods for modeling inter-personal social phenomena such as the rapport between members of a collaborative group or peer tutoring dyad.

This paper is intended to contribute to the detection of interpersonal social states, such as rapport, through nonverbal, task (verbal) and social (verbal) channels, captured through audio and video input. In this paper, we describe a process for using temporal association rule mining to learn patterns of behaviors from an annotated corpus of nearly 60 hours of dyadic peer tutoring interactions. We then use those temporal association rules to predict the "thin-slice" dyadic rapport level for every 30-second time-slice, via a stacked ensemble model. We find that temporal rules generated from annotations of students' nonverbal, on-task, and off-task social behaviors were overall able to predict rapport at levels well above chance, and at nearly double the prediction performance (AUC) of a baseline approach. We found that this approach allows us to predict high rapport significantly better than low rapport overall, while predicting high rapport for strangers more accurately than for friends.

This paper contributes to the Educational Data Mining (EDM) community in several ways: (1) We describe a process for automatically learning temporal association rules from annotations of nonverbal, and social and on-task verbal behaviors, and using those rules to predict rapport in a stacked ensemble model, compared to two baseline approaches. (2) We describe the variation in the number of high-confidence rules learned for each of the behavioral channels, to inform future developers of rapport detectors of the

data sources that may be most fruitful to capture. (3) We evaluate the predictive performance of those temporal rules in predicting rapport for both friends and strangers, thereby addressing both short- and long-term rapport.

2. RELATED WORK

In order to choose the behaviors used to predict rapport, we draw on a framework of rapport-building proposed by [22]. In this theoretical model, rapport is a dyadic phenomenon, co-constructed over time by both members of the dyad. According to [22], rapport is developed through nonverbal behaviors and verbal social conversational strategies that serve various social functions and sub-goals in rapport development, such as face management, mutual attentiveness, and coordination [22]. Our work extends [22]'s approach by also incorporating the task-related verbal strategies from both tutor and tutee, such as feedback, instructions, and task-related questions which are essential for the tutoring process, and which we hypothesize will impact, and be impacted by the rapport between members of a peer tutoring dyad [9].

Prior researchers in discourse analysis, multi-modal interaction, and dialogue systems have developed detectors for various aspects of interpersonal relationship development, such as Yu et al.'s friendship prediction for peer tutoring dyads, which found that dyadic features such as mutual gaze and smile behaviors were predictive of friendship [21]. In prior EDM work, some [15] have used the temporal co-occurrence of nonverbal behaviors (operationalized as Facial Action Units) to capture "behavioral synchronicity" in collaborative problem-solving dyads. Others have developed automatic classifiers of on-task-related interpersonal behaviors, such as [14]'s method for classifying socio-cognitive conflict in collaborative learning within an intelligent tutoring system. Others, such as [20], have developed automatic classifiers of dyadic impoliteness and positivity, work that we build on here with the social conversational strategies we incorporate into the association rules. Prior work has demonstrated the effectiveness of out-of-domain social talk in pedagogical agents, such as [8]'s social pedagogical agent used in collaborative learning.

2.1 Temporal Patterns in Behavior

As rapport-building is a dynamic phenomena, it is impacted by the contingent patterns of verbal and nonverbal behavior. Ohlssen et al. describe how popular methods for discourse analysis that use a "code-and-count" method [12] collapse the temporal dimension and are thus unable to understand the rich patterns of interaction likely to impact learning, or rapport. To address this gap, we draw on the Temporal Interval Tree Association Rule Learning (Titarl) framework [7] to discover temporal patterns of verbal and nonverbal behavior and their association with the dyadic rapport between members of a tutoring dyad for every 30-second time slice. The Titarl framework has been previously used to analyze medical patients' vital sign data [7], and in our lab, [24] have used Titarl to identify patterns of social conversational strategies and nonverbal behaviors predictive of levels of rapport. Crucially, however, [24] did not include the tutoring and learning behaviors that are the heart of the task component of the peer tutoring interactions, and which are likely to impact rapport through, for example, the face-threatening nature of providing feedback or instructions [9]. Therefore, in order to more effectively predict the rapport between members of a peer tutoring dyad, we include rules learned from the nonverbal, social verbal, and tutoring-related verbal behaviors.

3. METHODS

3.1 Research Questions

RQ1: Can temporal association rules learned from social conversational strategies, task, and nonverbal behaviors in peer tutoring be used to predict rapport at levels above chance? From [7] and [24], we believe that they can, and that we can improve the predictive performance by adding the task-related verbal behaviors.

RQ2: Is a classifier trained on temporal association rules better able to predict rapport (a) for some relationship types than others or (b) at some levels than others? Following [24], we believe we will be better able to predict high rapport among strangers than among friends.

RQ3: Are temporal association rules (TAR) generated using all three channels of on-task (verbal), social (verbal), and nonverbal behavioral better able to predict rapport than rules generated from any one or two of those behavior types? From [9, 21, 24], we believe that including task, social, and nonverbal together will perform the best.

3.2 Data Collection and Dialogue Corpus

The dialogue corpus described here was collected as part of a larger study on the effects of rapport-building on reciprocal peer tutoring [9, 10, 18, 22]. The participants were assigned to 12 dyads that alternated tutoring each other in Algebra for 5 weekly hour-long sessions, for a total of 60 hours. Half were male and half were female, assigned to same-gender dyads. To investigate how the impact of various task, social, and nonverbal behaviors on rapport differs between dyads with varying degrees of interpersonal closeness, we used friendship as a proxy for long-term rapport and thus asked half of the participants to bring a same-age, same-gender friend to the session with them, and for the other half of the dyads, we paired them with a stranger, using the 5 weeks to capture short-term rapport-building. Audio and video data were recorded, transcribed, and segmented for clause-level dialogue annotation.

3.3 Thin-Slice Rapport Ratings

The rapport between the participants, was evaluated using a 'thin-slice' approach [1]. First, the corpus was divided into 30-second video slices, then shuffled (so the raters did not inadvertently rate the change in rapport), and provided to naive, third-party raters. Three such raters rated the rapport present in each slice on a Likert scale from 1-7, from lowest possible rapport to highest possible rapport. A single rating was then chosen for each slice using an inverse bias-corrected weighted majority vote approach, described in [18], to account for potential over-use or under-use of certain labels by the raters. The final consensus measure of inter-rater reliability, or Cronbach's α, was .86, justifying the use of this rating selection method [18]. This rating was used when learning the associations between the task, social, and nonverbal behaviors and the rapport level.

Table 1: Annotation Types, Labels, Definitions, and Examples

Type	Label	Definition	Example
Task	Knowledge-telling	Stating procedures or the answer	Divide it by 9.
Task	Knowledge-building	Providing explanations	That's because it can be reduced.
Task	Correct or Incorrect Feedback	Evaluating their partner's correctness	No, that's not quite it.
Task	Shallow Question	Asking about procedures or answers	Is that right?
Task	Deep Question	Asking about reasoning or concepts	Why would you do that?
Social	Self-Disclosure	Sharing personal information about oneself	I suck at negative numbers.
Social	Refer to Shared Experience	Discussing an experience they had together	Remember that soccer game?
Social	Violation of Social Norms	Statements that break social conventions	It's a zero, dummy.
Social	Praise	Positive acknowledgment of the other	You're so smart!
Social	Reciprocation	Responding to a conversational move with the same conversational move.	Tutor self-discloses, then the tutee self-discloses

3.4 Dialogue Annotation

To investigate the impact of rapport-building verbal (social and task) and nonverbal behaviors, we annotated our dataset for 3 types of nonverbal behaviors, 5 types of social conversational strategies, and 5 types of tutoring and learning behaviors, as shown in Table 1, all annotated with $> .7$ Krippendorff's α. The nonverbal behaviors annotated were head nods, smiles, and shifts in eye gaze from the partner, to the Algebra worksheets, to anywhere else, similar to [21]. The social verbal behaviors were chosen according to [22]'s theory of rapport-building, behaviors such as self-disclosure, reference to shared experiences, violation of social norms, and others. The on-task verbal behaviors annotated are based in part on [16]'s work on knowledge-telling and knowledge-building, as well as [6] work with procedural and conceptual questions, described in more detail in [10].

3.5 Temporal Association Rule Mining

To investigate the impact that these nonverbal, task, and social behaviors had on rapport at a 30-second thin-slice level, we adopted a temporal association rule mining approach, following [23]. The framework we use, the "Temporal Interval Tree Association Rule Learning" (Titarl) algorithm [7], allows us to identify temporal patterns of behaviors within each time slice that are probabilistically associated with the value of rapport for that slice. For each 30-second time window, a rule is learned much like the generic rule below.

"If event A happens at time t, there is 50% chance of event B happening between time t+3 to t+5".

Our data is comprised of both multivariate symbolic time sequences (the nonverbal, task, and social behaviors) and multivariate scalar time series (the rapport value for each slice). The Titarl algorithm will learn a large set of rules on a subset of our data (the training set), filter those rules based on a set of parameter thresholds, fuse similar simple rules into more complex rules, which we then use in predicting rapport on a held-out test set. Because we believed that the ways that friends and strangers build rapport with each other over 5 weeks are likely to differ following [23], we ran the Titarl algorithm on sets of friend dyads and sets of stranger dyads separately.

3.6 Rapport Detection Process

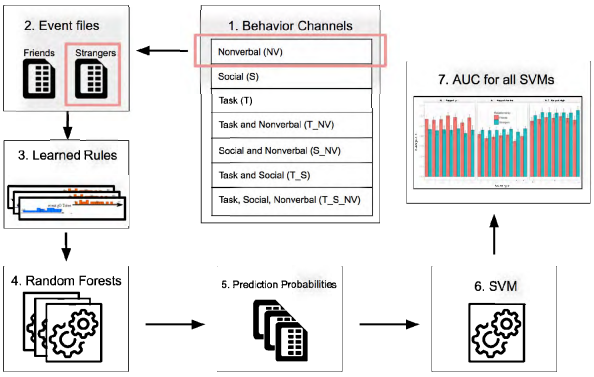


Figure 1: Multi-step process for prediction of rapport using temporal association rule mining and a stacked classifier ensemble.

We describe here an approach laid out in Figure 1. We first divided our 6 friend dyads and 6 stranger dyads, with 5 sessions per dyad, into a training set of 4 dyads (20 total sessions) and a held-out test set of 2 dyads (10 total sessions) for both relationship types. Then, (Step 1) we created seven combinations of the Social, Task, and Nonverbal annotations described in Table 1, to identify differences in prediction performance for the different behavior types (RQ3). Next (Step 2), for each of those behavior combinations, we created a matrix M with $n+1$ columns, with $n =$ the total number of annotation types (used by the tutor and the tutee), described in Table 1, with the first column in M being the start time, in seconds, of each behavior. Each row in M was an event, or the start of an annotated verbal or nonverbal behavior. From each matrix M, we generated an "event file" which included the behavior sequence as well as the scalar time series of the rapport value for the 30-second time slice within which those behaviors occurred.

Then, using these files, we (Step 3) learned a set of association rules R for each training set, using the Titarl algorithm [7]. These rules contain a head, which is the scalar output value of rapport (an integer from 1-7), and a body, which is the ordered set of annotated behaviors used to predict the rapport in each slice. Prior to learning, we specified the minimum confidence (the probability of the prediction of the rule to be true) at 50%, the minimum support (the per-

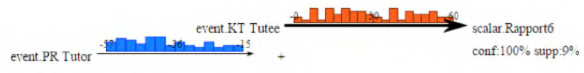


Figure 2: Example temporal association rule for strangers with high rapport, with 100% confidence, 9% support, and 44 uses.

centage of events explained by the rule) at 5%, and the minimum number of uses for each rule at 10, following [24]. An example of a rule can be seen in Figure 2, where a Tutor's use of Praise (PR), followed by the Tutee's "Knowledge-telling" (KT), or self-explanation, is associated with a rapport value of 6 (high), with confidence of 100%, support of 9%, and 44 uses in that model. This rule was learned from a Task and Social behavioral model, for a dyad of Strangers. The nature of these data can be further illustrated with another example, from a highly confident association rule learned from the Task and Social model, for dyads of Friends. The following high-confidence rule is associated with Rapport of 1 (low): a Tutee asks a Shallow Question, receiving four "Knowledge-telling" utterances in a row from their Tutor, to which the Tutee responds with a "Social Norm Violation". In other words, the tutee asks about the procedure, the tutor tells him what to do in multiple utterances, and the tutee responds with some norm violation, perhaps rudeness. To ensure that the rules learned from each set of dyads were not overfit to the particular training set of dyads used to learn them, we learned a rule set (i.e. repeated Steps 1-3) for all possible combinations of the 6 friend and 6 stranger dyads, resulting in 15 "folds" for friend dyads and 15 for strangers (i.e. choosing all possible sets of 4 dyads to use as training sets from the 6 total dyads). Each fold had several hundred association rules learned above our threshold for confidence, support, and usage. In Figure 3, we show the mean number of rules learned, showing only those with confidence, support, and usage above the median for ease of visualization.

After learning the rules, in Step 4 we use the rules to train random forest classifiers to predict the rapport level for each 30-second slice. To do this, we first generated a matrix *N* for each rule set in each of the 30 training sets, with a row for each rule event in that set, and *n+1* columns, where *n* is the number of rules in that train set, and the final column was a binary indicator of the rapport value for that time slice. We ran 7 random forest classifiers (one for each rapport level) for each matrix *N*, for each of the 15 folds of friends and 15 folds of stranger training sets, giving us (in Step 5) a prediction probability estimate for each of the 7 rapport values, for each event in every fold, for each of the 7 behavioral channels (from Step 1). Finally, we wanted to evaluate the relative impact of those 7 behavior types, and so we composed different combinations of nonverbal, social, and task behavior. We then, in Step 6, use the prediction probability output by the random forest classifiers as the input features in training a single multi-class Support Vector Machine (SVM) classification model for each of the 30 folds to predict the overall rapport level for each time slice in that fold. In the following section, we discuss the performance of this final classification step in predicting rapport for each relationship type and evaluate its performance against two baselines from earlier steps in the process.

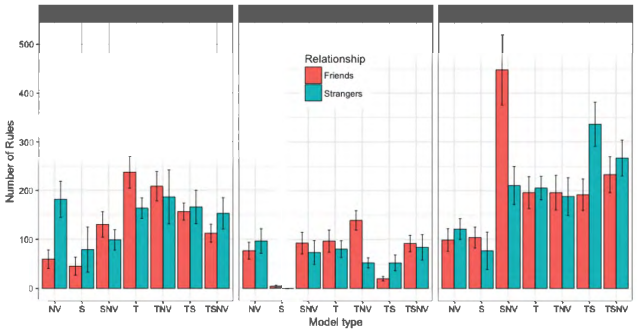


Figure 3: Mean number (and standard error) of rules learned from 7 behavioral channels, with high confidence, support, and usage, for friends and strangers with low, neutral, and high rapport.

4. RESULTS

First, before investigating our first research question about the performance of our approach in predicting rapport, we wanted to inspect the total number of rules learned from each behavioral channel with high confidence, support, and usage, to better understand the extent to which the number of highly confident temporal rules varied for each behavioral type. See Figure 3 for the mean number and standard errors for rules learned above the median confidence, support, and usage for low, neutral, and high rapport for friends and strangers. Based on the distribution of slices at each level, we converted the 7 scalar rapport values to the low (1-3), neutral (4), and high (5-7) rapport levels.

We see that friends had significantly more ($t(20.8)=2.7$, $p=.01$) high-confidence Social and Nonverbal ("SNV") rules learned in High Rapport slices than the next largest behavioral channel, the "TSNV" channel, combining Task, Social, and Nonverbal. This suggests that a method for detecting high rapport between friends that uses Social and Nonverbal behaviors will have many more high-confidence, frequently occurring rules with which to predict rapport than using other sets of behavior types. Conversely, for rules learned from Friend dyads for Low Rapport slices, there is a significantly ($t(26.6)=2.6$, $p<.05$) greater number of high-confidence, frequently occurring Task ("T") rules than rules learned from the Social and Nonverbal (SNV) behaviors. That is, there are substantially more high-confidence, high-support, and frequently occurring ways in which Friends displayed Low Rapport through their on-task behavior (and on-task combined with nonverbal, "TNV") than through other available channels. This suggests that a method for detecting students' low rapport, for a dyad of friends, may benefit from incorporating the task-related behaviors such as instructions, explanations, questions, and provision of feedback in addition to purely social behaviors, as in [23]. Similarly, for Strangers, their Task and Social ("TS") channel had the largest number of rules learned associated with High Rapport slices, significantly more than the "SNV" behaviors ($t(26.8)=1.2$, $p<.05$), though not significantly more than the TSNV behaviors. This suggests that a detector of high rapport that leverages Task and Social behaviors may have more high-confidence association rules from which to draw for its

Table 2: Average PR-AUC (and standard deviation) of 3 rapport prediction models

Model	PR-AUC
IF Baseline	.42 (.07)
RF Baseline	.33 (.03)
TAR Ensemble	.60 (.08)

classification of high rapport for students without a prior friendship relationship (i.e. "strangers") than one relying solely on Strangers' social and nonverbal behaviors.

Then, to evaluate the overall performance of our approach in predicting low, neutral, and high rapport, we used the prediction probability from the 7 binary random forest classifiers (from Step 5) as the input into a 3-way one-vs-rest SVM classifier, for every behavioral channel model (Step 6). We first ran a 10-fold cross-validated grid search on our training set to discover the optimal set of parameters to use for the SVM model, using an RBF kernel, with C=10 and $\gamma = 1$. From the SVM, we use the average area under the Precision-Recall curve (PR-AUC) for each of the 7 behavioral models as our performance measure, following [5].

First, for RQ1, to validate the appropriateness of our stacked ensemble approach ("TAR Ensemble"), we compare its prediction performance to two baseline approaches. We compare first to a baseline that treats the annotated behaviors in each slice as independent features in an SVM using the same parameters ("IF Baseline"). The TAR Ensemble significantly ($t(413) = 24.4$, $p<.001$) outperforms the IF Baseline with a mean AUC of .60 (sd = 08) for the TAR Ensembles, compared to a mean AUC of .42 (sd = .07) for the IF Baseline. We then compare the TAR Ensemble to another baseline ("RF Baseline") that simply takes the largest prediction probability from the 7 random forests (Step 5 in Figure 2) as the predicted class value, using random selection for ties. The TAR Ensemble significantly ($t(256) = 46$, $p<.001$) outperforms the RF Baseline by nearly 2 to 1, with a mean AUC of .60 (sd = 08) for our approach and a mean AUC of .33 (sd = .03) for the RF Baseline. See Table 2 for a summary of the PR-AUC values for each model.

For RQ2a, we find that the Stacked Ensemble is better able to predict High Rapport than Low ($t(417)=5.9$, $p<.005$). For RQ2b, we are better able to predict Low Rapport for Friends than Strangers ($t(197) = 5.8$, $p<.001$). Conversely, we are better able to predict the rapport among Strangers than among Friends for both Neutral ($t(206.5) = 5.5$, $p<.001$) and High rapport levels ($t(207) = 2.7$, $p<.01$). For RQ3, no single set of behavioral channels significantly outperformed the others, in an ANOVA of the PR-AUC measure with each relationship type (Friend/Stranger), rapport level (Low/Neutral/High), and behavioral type (TS, TSNV, etc).

5. DISCUSSION AND CONCLUSION

Interpersonal social dynamics provide the grounding for learning interactions, whether students are learning collaboratively, in peer tutoring, or working with their classroom teacher or even a virtual agent. However, technological supports for learning often focus on detecting and modeling individual, intra-personal states such as students' affect or engagement, without considering the latent social state underpinning their interactions with others. In this work, we present one method for detecting the latent social state of interpersonal rapport in learning interactions, using a temporal association rule mining approach to learn patterns of nonverbal and verbal (social and task) behaviors, as input in predicting the rapport level in a stacked ensemble model. Our ensemble approach outperforms two baselines, (1) the independent behaviors as features, and (2) the random forests trained on the temporal association rules.

We find that, overall, our approach is better able to predict high rapport than low rapport, and it predicts high and neutral rapport more accurately for Strangers than for Friends, while predicting low rapport more accurately for Friends than for Strangers. This is good news for designers of virtual agents that want to detect and build rapport with a new student, or designers of computer-supported collaborative learning technologies that want to detect rapport in learning. However, contrary to our expectations (for RQ3), we saw no significant difference in prediction performance across the models generated from different combinations of behavior types (e.g. SNV, TSNV, etc). We did see a significant difference in the total number of association rules learned from those behavior types, however, suggesting that rapport detectors will be better able to predict rapport if they use the behavior types that occur more frequently in learning. For instance, a rapport detection method for strangers that incorporates Task and Social behavior will have significantly more high-confidence, high-support association rules with which to detect the rapport between them.

One of the limitations of this current approach is that, while it may reach quite good levels of performance in detecting rapport, the large number of rules learned make it difficult to identify the specific rules that are most predictive of rapport, in addition to concerns about dimensionality. This work is limited by the small sample size, and by being restricted to same-gender dyads; using a larger set of dyads to conduct these analyses may reveal differences in prediction performance for different behavioral types (social, task, nonverbal), if they exist. We have currently finished collecting 22 dyads' worth of interactions among strangers (over 40 hours), and we will be conducting a similar set of annotations and analyses on them. In this paper, the thin-slice rapport ratings and annotations were hand-annotated from a corpus of audio/video data, limiting the automaticity of this approach. However, we are in the process of moving to a crowd-sourced method for obtaining the ground truth rapport ratings for each 30-second slice. Preliminary experiments for crowd-sourcing the thin-slice rapport annotation using Amazon Mechanical Turk have yielded a Krippendorff's α of 0.69 across 3 raters for each thin-slice.

In future work, we intend to use this rapport estimation method for a rapport-building virtual agent in an intelligent tutoring system. We have developed automatic classifiers for the three types of nonverbal behaviors described here, using the OpenFace system [4], and social conversational strategy classifiers, such as those described by [23], classifiers which have already been integrated into a "socially aware robot assistant" (SARA), as described by [11]. Our next step is to develop a task-related classifier, perhaps similar to that

used in [14], to recognize students' task-related utterances as part of the rapport estimation and reasoning about natural language response generation. We believe that this paper contributes to the larger goal of educational data mining by demonstrating one approach to using multimodal data to model latent social phenomena important to learning, in this case the interpersonal rapport in peer tutoring.

6. ACKNOWLEDGMENTS

The research reported here was supported, in whole or in part, by the National Science Foundation Cyberlearning Award No.1523162, and the Institute of Education Sciences, U.S. Department of Education, through Grant R305B150008 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent the views of the Institute or the U.S. Department of Education.

7. REFERENCES

[1] N. Ambady and R. Rosenthal. Thin slices of expressive behavior as predictors of interpersonal consequences: A meta-analysis. *Psychological Bulletin*, 111(2):256–274, 1992.

[2] M. Azmitia and R. Montgomery. Friendship, transactive dialogues, and the development of scientific reasoning. *Social development*, 2(3):202–221, 1993.

[3] R. S. Baker, S. K. D'Mello, M. M. T. Rodrigo, and A. C. Graesser. 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, 2010.

[4] T. Baltrušaitis, P. Robinson, and L.-P. Morency. Openface: an open source facial behavior analysis toolkit. In *Applications of Computer Vision (WACV), year=2016, organization=IEEE*.

[5] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd ICML*, pages 233–240. ACM, 2006.

[6] A. C. Graesser, N. K. Person, and J. P. Magliano. Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied cognitive psychology*, 9(6):495–522, 1995.

[7] M. Guillame-Bert and A. Dubrawski. Learning temporal rules to forecast events in multivariate time sequences. In *2nd Workshop on Machine Learning for Clinical Data Analysis, Healthcare and Genomics. NIPS*, 2014.

[8] R. Kumar, H. Ai, J. L. Beuth, and C. P. Rosé. Socially capable conversational tutors can be effective in collaborative learning situations. In *International Conference on Intelligent Tutoring Systems*, pages 156–164. Springer, 2010.

[9] M. Madaio, J. Cassell, and A. Ogan. The impact of peer tutors ' use of indirect feedback and instructions. In *Computer-Supported Collaborative Learning Conference*, 2017.

[10] M. A. Madaio, A. Ogan, and J. Cassell. The effect of friendship and tutoring roles on reciprocal peer tutoring strategies. In *International Conference on Intelligent Tutoring Systems*. Springer, 2016.

[11] Y. Matsuyama, A. Bhardwaj, R. Zhao, O. J. Romero, S. A. Akoju, and J. Cassell. Socially-aware animated intelligent personal assistant agent. In *17th Annual Meeting of SIGDIAL*, page 224, 2016.

[12] S. Ohlsson, B. D. Eugenio, B. Chow, D. Fossati, X. Lu, and T. C. Kershaw. Beyond the code-and-count analysis of tutoring dialogues. *AIED: Building technology rich learning contexts that work*, 158:349, 2007.

[13] J. Olsen and S. Finkelstein. Through the (thin-slice) looking glass : An initial look at rapport and co-construction within peer collaboration. In *Computer-Supported Collaborative Learning Conference*, in press.

[14] D. Prata, R. Baker, E. Costa, C. Rose, and Y. Cui. Detecting and understanding the impact of cognitive and interpersonal conflict in computer supported collaborative learning environments. In *Educational Data Mining 2009*, 2009.

[15] V. Ramanarayanan and S. Khan. Novel features for capturing cooccurrence behavior in dyadic collaborative problem solving tasks. In *Educational Data Mining 2009*, 2016.

[16] R. D. Roscoe and M. T. Chi. Understanding tutor learning: Knowledge-building and knowledge-telling in peer tutors' explanations and questions. *RER*, 77(4):534–574, 2007.

[17] M. O. Z. San Pedro, R. S. d Baker, and M. M. T. Rodrigo. Carelessness and affect in an intelligent tutoring system for mathematics. *IJAIED*, 24(2):189–210, 2014.

[18] T. Sinha and J. Cassell. We click, we align, we learn: Impact of influence and convergence processes on student learning and rapport building. In *Proceedings of the 1st Workshop on Modeling INTERPERsonal SynchrONy And infLuence*, pages 13–20. ACM, 2015.

[19] E. Walker and A. Ogan. We're in this together: Intentional design of social relationships with aied systems. *IJAIED*, 26(2):713–729, 2016.

[20] W. Y. Wang, S. Finkelstein, A. Ogan, A. W. Black, and J. Cassell. Love ya, jerkface: using sparse log-linear models to build positive (and impolite) relationships with teens. In *Proceedings of SIGDIAL*, pages 20–29. Association for Computational Linguistics, 2012.

[21] Z. Yu, D. Gerritsen, A. Ogan, A. W. Black, and J. Cassell. Automatic prediction of friendship via multi-model dyadic features. In *Proceedings of SIGDIAL*, pages 51–60, 2013.

[22] R. Zhao, A. Papangelis, and J. Cassell. Towards a dyadic computational model of rapport management for human-virtual agent interaction. In *IVA*, pages 514–527. Springer, 2014.

[23] R. Zhao, T. Sinha, A. W. Black, and J. Cassell. Automatic recognition of conversational strategies in the service of a socially-aware dialog system. In *17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 381, 2016.

[24] R. Zhao, T. Sinha, A. W. Black, and J. Cassell. Socially-aware virtual agents: Automatically assessing dyadic rapport from temporal patterns of behavior. In *International Conference on Intelligent Virtual Agents*, pages 218–233. Springer, 2016.

Learning to Represent Student Knowledge on Programming Exercises Using Deep Learning

Lisa Wang, Angela Sy, Larry Liu, Chris Piech
Stanford University
{lisa1010@cs, angelasy, hrlarry, piech@cs}.stanford.edu

ABSTRACT

Modeling student knowledge while students are acquiring new concepts is a crucial stepping stone towards providing personalized automated feedback at scale. We believe that rich information about a student's learning is captured within her responses to open-ended problems with unbounded solution spaces, such as programming exercises. In addition, sequential snapshots of a student's progress while she is solving a single exercise can provide valuable insights into her learning behavior. Creating representations for a student's knowledge state is a challenging task, but with recent advances in machine learning, there are more promising techniques to learn representations for complex entities. In our work, we feed the embedded program submission sequence into a recurrent neural network and train it on two tasks of predicting the student's future performance. By training on these tasks, the model learns nuanced representations of a student's knowledge, exposes patterns about a student's learning behavior, and reliably predicts future student performance. Even more importantly, the model differentiates within a pool of poorly performing students and picks out students who have true knowledge gaps, giving teachers early warnings to provide assistance.

Keywords

Educational data mining; Online education; Personalized learning; Knowledge tracing; Machine learning; Representation learning; Sequential modeling.

1. INTRODUCTION

With the inception of online learning platforms, educators around the world can reach millions of students by disseminating course content through virtual classrooms. However, in these online environments, teachers' ability to observe students is lost. Understanding a student's incremental progress is invaluable. For instance, if a teacher watches a student struggle with an exercise, they see the student's strengths as well as their knowledge gaps. The process by which the student reaches the final solution is equally as important as the solution itself. We attempt to encode these markers of progress. We performed representation learning with recurrent neural networks to understand a student's learning trajectory as they solve open-ended programming exercises from the *Hour of Code* course, a MOOC on *Code.org*. The deep learning model trains on a student's history of past code submissions and predicts the student's future performance on the current or the next exercise. The model is able to learn meaningful feature representations for a student's series of submissions and hence does not require manual feature selection, which would be very difficult for open-ended exercises. Furthermore, the learned representations can be used for other related tasks, such as predicting an intervention.

1.1 Motivation: Instructional Scaffolding

The widely used pedagogical concept of the zone of proximal development (ZPD) suggests that ideal learning objectives are in a sweet spot of difficulty called the ZPD: more difficult than what the student can accomplish on their own, but not so difficult that they cannot succeed even with guidance [3, 24]. The guidance for accomplishing such challenging-but-achievable objectives is called instructional scaffolding, and it is most effective when personalized to each student's mastery of the material [18].

Scaffolding is particularly difficult in MOOCs–it is hard to personalize instruction to thousands of students at once. While some research has explored the merits of academic habit scaffolding [6] or reciprocal scaffolding with peer collaboration [19] in MOOCs, the most promising work lies in expert scaffolding, which involves an expert, usually a teacher, in the relevant domain of knowledge providing guidance to help students acquire knowledge [**?**]. Effective teachers possess pedagogical content knowledge (PCK), or expertise about not only the domain of knowledge, but also how to best teach that material to learners [21]. Most importantly, PCK helps anticipate where students will struggle.

In existing MOOC research, the expert scaffolding usually takes the form of feedback to students' responses on assignments. Yet, many current systems for automating feedback in MOOCs relies on time-consuming and potentially arbitrary tasks of feature engineering [20] or defining rulesets [22] applicable only to single exercises. This manual encoding of PCK is task-specific and not a generalizable unsupervised process. A more generalizable signal of student failing learning objectives is student attrition from MOOCs. Limited work exists exploring correlations between attrition

and student engagement with MOOC materials [27] or other students (e.g. on discussion forums) [17, 26]. However, to the authors' knowledge, existing MOOC attrition research does not control for student achievement. Often, attrition is merely a downstream symptom of struggling with learning objectives. When students underachieve, their self-concept of themselves as learners may be threatened, which recursively reinforces lower achievement and disengagement [4]. In general, anticipating common domain-specific mistakes with PCK can help preempt them and mitigate subsequent disengagement, and thus the unsupervised *anticipation* of student mistakes is a worthwhile objective for automated systems that can ultimately improve learning.

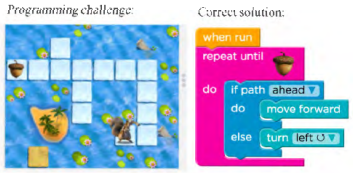


Figure 1: Exercise 18 in the Code.org Hour of Code. Left, the programming challenge. Right, the solution. The challenge is to program the squirrel to reach the acorn, while using as few coding blocks as possible. `https://studio.code.org/hoc/18`.

2. RELATED WORK

Representation Learning with Neural Networks

In the field of machine learning, representation learning is the task of learning a model to create meaningful representations from low-level raw data inputs. The goal of representation learning is to reduce the amount of human input and expert knowledge needed to preprocess data before feeding it into machine learning algorithms [1]. In contrast to manually selecting high-level features, representation learning algorithms are trained to extract features directly from raw input, e.g. from words in a document. The combination of linear functions and nonlinearities stacked in layers allows deep neural networks (DNNs) to learn abstract representations in an efficient manner [1]. Empirically, DNNs do particularly well when the data has high semantic complexity and manually choosing features is not only tedious, but often insufficient. Once the representations are trained on one task, they can be used for other related tasks as well. E.g. In *word2vec* [12], word representations were trained on predicting context words but were then used for document classification and translation. Empirically, DNNs do particularly well when the raw data has high semantic complexity and manually choosing features is not only tedious, but often insufficient. Recurrent neural networks (RNNs) are a subtype of neural networks which take inputs over multiple timesteps and are therefore well-suited for learning representations on sequential data with temporal relationships.

2.1 Program Code Embeddings

In order to expand DKT to understand students as they produce rich responses over time within an exercise, a necessary step is to create meaningful embeddings of their program submissions. Piech et al. proposed to use recursive neural networks to create program embeddings for student code[15]. Recursive neural networks that learn embeddings on syntax trees were first developed by the NLP community to vectorize sentence parse trees [23], but are even more applicable to computer programs due to their inherent tree structure, since any program can be represented as an Abstract Syntax Tree (AST).

2.2 Knowledge Tracing (KT) and Deep KT

The task of knowledge tracing can be formalized as: given observations of interactions $x_0 \dots x_t$ taken by a student on a particular learning task, predict aspects of their next interaction x_{t+1} [5]. Piech et al. applied RNNs to data from Khan Academy's online courses to perform knowledge tracing by predicting student performance [14]. The authors found that RNNs can robustly predict whether or not a student will solve a particular problem correctly given their performance on prior problems. Other models that are designed to take low dimensional inputs, such as IRT and modifications of Bayesian Knowledge Tracing [28] [13], sometimes outperform the initial version of Deep Knowledge Tracing (DKT) [25] [10]. However, DKT does not require student interactions to be manually labeled with relevant concepts and the RNN paradigm was designed to take vectorized inputs, hence it can utilize inputs that extend beyond the discrete inputs of traditional models [7]. These properties make the model an appropriate fit to understand trajectories of open-ended student responses, which have unbounded input spaces.

A limitation with the work of Piech et al. is that it does not fully leverage the promise of using neural networks to trace knowledge. The dataset they used only contained binary information about a student's final answer (i.e. correct or incorrect). In contrast, the *Hour of Code* dataset comprises program submissions that each have a boundless solution space. These infinite variations represent richly structured data which we can encode as program embeddings. The ideas presented in this paper work towards a model with the representative capacity to tackle **open-ended** knowledge tracing [9]. In addition, previous work in deep knowledge tracing has looked at student responses over multiple exercises, but not within an exercise. Our method focuses on a student's *sequence of submissions* within a *single* programming exercise to predict future achievement. We model student learning and progress by capturing representations of the current state of a student's knowledge as they work through the exercise and incrementally submit programs. When focusing exclusively on the final submission, these incremental steps are ignored.

3. EXPERIMENTS: TASK DEFINITIONS

In order to create representations of a student's current state of knowledge, we chose the two following training tasks:

- **Task A**:
  Based on a student's sequence of k code submission attempts ***over time*** *(hereby, their "trajectory")* $T = [AST_1, AST_2, ..., AST_k]$ *on a programming exercise, predict at the end of the sequence whether the student will successfully complete or fail the* ***next*** *programming exercise within the same course.*

- **Task B**:
  At each $t \leq k$*, given a student's sequence thus far of t code submission attempts* $T = [AST_1, AST_2, ..., AST_t]$ *on a programming exercise, predict whether the student*

*will successfully complete or fail the **current** programming exercise.*

Task A is pedagogically comparable to predicting whether or not a student will be able to learn a new concept given the way they did or did not learn previous concepts. Phrased differently, a student who quickly (e.g. in few time steps) demonstrates some level of mastery of material (i.e. the goodness of their final submission) should be considered more likely to outperform a student who took a long time and may have struggled before eventually demonstrating the same level of mastery. Meanwhile, **Task B** is pedagogically comparable to detecting whether or not a student is struggling to acquire the present concept as they incrementally engage with the learning objective. In other words, teachers can get real-time information about the learning of the students. We expect **Task B** to be more difficult but also more pedagogically powerful. **Task B** is unlike **Task A** in that **Task B** does not use the full trajectory of a problem, *which would contain post-hoc knowledge of whether or not a student gave up in an earlier learning interaction*, for prediction. All of the students used as inputs in **Task B** can be considered not attrited at least at *some* point in the prediction task. Critically, success on **Task B** would enable teachers to predict at-risk students who may *eventually* give up and not complete the exercise but have not yet given up, where in **Task A**, by the time a teacher knows a student has given up one exercise (the inputs of **A**) as you are trying to guess their success on the next exercise, it may be too late to get the attrited student to rejoin in the learning environment (e.g. re-enroll after dropping out).

4. DATASET: HOUR OF CODE EXERCISE

The *Hour of Code* course consists of twenty introductory programming exercises aimed at teaching beginners fundamental concepts in programming. Students build their programs in a drag-and-drop interface that pieces together blocks of code. The number of possible programs a student can write is infinite since submissions can include any number of block types in any combination. A student can run their code multiple times for any exercise. These submissions provide temporal snapshots to track the student's learning progress. The student submission data for Exercises 4 and 18 from this course are publicly available on *code.org/research*. For our experiments, we focus on the sequences of intermediate submissions on Exercise 18. We chose Exercise 18 (over Exercise 4) because it covers multiple concepts such as loops, if-else statements, and nested statements, resulting in more complex and varied code submissions. This Exercise 18 data set contains 1,263,360 code submissions, and, in turn, more varied trajectories of student learning, of which 79,553 are unique, made by 263,569 students. 81.0% of these students arrived at the correct solution in their last submission. In comparison, there were 1,138,506 code submissions, of which only 10,293 were unique. The 509,405 students who attempted Exercise 4 succeeded at a 97.8% rate.

Since the *Hour of Code* exercises do not have a bounded solution space, students could produce arbitrarily long trajectories. We noted that the accuracy of student submissions have a high correlation with trajectory lengths. For instance, the vast majority of students with trajectory length 1 solved the problem with their very first submission. Hence, for both tasks **A** and **B**, we chose to only include trajectories of length 3 or above. Pedagogically, we are also more interested in students who don't get the answer right away, and we speculate that longer trajectories should roughly correlate with greater struggling with the learning objective.

5. MODELS

5.1 Recurrent Neural Network Model for Student Trajectories

Since we would like to capture aspects of a student's learning behavior over time, RNN's are a suitable neural network architecture for our experiments, as RNN's have empirically performed well on sequence modeling tasks in other domains. For both tasks **A** and **B**, we used a Long Short Term Memory (LSTM) RNN architecture, which is a popular extension to plain RNNs since it reduces the effect of vanishing gradients [8]. A student's trajectory consists of k program submissions, which are represented as ASTs. Note that an AST contains all the information about a program and can be mapped back into a program. These ASTs are converted into program embeddings using a recursive neural network similar to the one described in [15]. The program embedding is a more compact representation of the original AST, which captures aspects of the program; in particular its functionality. This sequence of program embeddings gets fed into an RNN, as illustrated in Figure 2.

For **task A**, we used a three layer deep LSTM. To make the prediction at the end of the sequence, we pass the hidden state at the last timestep through a fully connected layer and a subsequent softmax layer. The output $\hat{y}$ of the softmax layer is an estimated probability distribution over two binary classes, indicating whether the student successfully solved the next exercise. For task **B**, we built a dynamic three layer LSTM, which makes a prediction at every timestep t based on the hidden state at t. Hence, if a student submits three times, we will use the sequence thus far to make three predictions.

5.2 Baselines

Task A: The goal here is to show that our model can learn from the program embeddings alone whether a student is likely to succeed on the subsequent exercise and contrast its performance against the state of the art baseline using handpicked features. For the baseline, we chose the following two features for a student's trajectory T, which have been shown to be highly correlated with learning outcome and performance on the next exercise and trained a logistic regression model.

1. The Poisson path score of the trajectory T as defined in [16]. Intuitively, the path score is an estimate of the time it will take a student to complete the trajectory series. The path score of a student trajectory has previously been related to student retention in sequential challenges [16]. $pathScore(T) = \sum_{x \in T} \frac{1}{\lambda_x}$ where λ_x is the number of times AST x appears in student submissions.

2. Indicator feature of student success on current exercise 18. A student succeeded if they ended the trajectory with the solution AST.

Task B: Here, we would like to demonstrate that an LSTM is able to capture more information about a student's trajectory and capture the temporal relationships within the sequence. Hence, we picked logistic regression on program

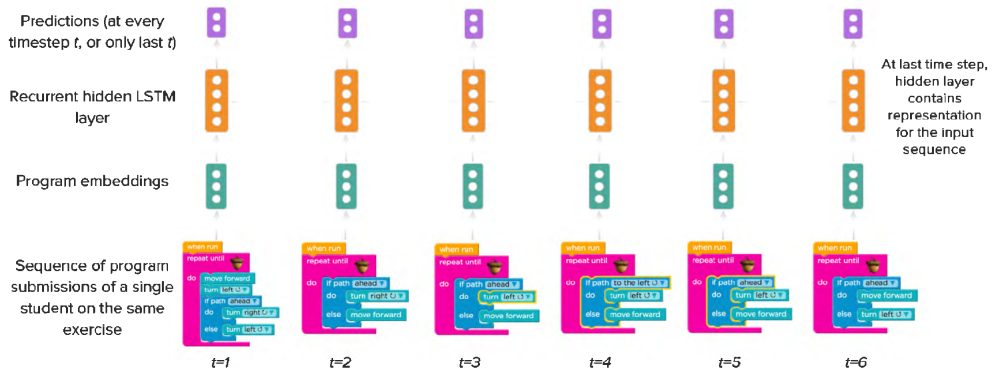


*Figure 2: Simplified Sequential RNN Model. For Experiment **A**, the model only predicts at the last timestep. For Experiment **B**, the model predicts at every timestep. Note that the RNN can be unrolled any number of times, since the parameter weights are shared across timesteps. Note that our models for both tasks stack multiple LSTM layers to increase expressivity.*

embeddings as our baseline. Since logistic regression cannot take in a sequence of embeddings, we consider every embedding within a trajectory sequence as a separate sample that we pass into the logistic regression model. Hence, this model ignores any previous temporal information; e.g. at timestep t, it ignores all embeddings from timestep 1 to $t-1$. Note that this is fairly high baseline, since we feed in program embeddings which are learned using neural networks. We also included a random baseline as a sanity check.

6. RESULTS

6.1 Quantitative Results

Task A: For both the pathscore baseline model and the LSTM model, we used 90% of the data set to perform training and validation and the remaining 10% for testing. The LSTM model consistently outperforms the path score baseline by around 5% on test accuracy at every trajectory length. This result is significant since the input we feed into the LSTM model consists of program embeddings, and not hand-picked features like success on current problem. Our model identified trajectories that show more promise. The ability to understand trajectories suggests that the representations used for the programs within the trajectories were also meaningful. The program embeddings were trained to predict the output of any given student program. Our program embedding model was able to correctly predict the output for 96% of the programs in a hold out set, compared to a 54% accuracy from always predicting the most common output.

Task B: We trained on trajectories of variable lengths 5 to 15, using 90% for training/validation and 10% for testing. At every timestep, we perform a binary prediction. Let's call these two classes "success" and"failure". Since the "failure" class is pedagogically more important, we reported recall, precision and F1 score for the "failure" class at each timestep for our LSTM model as well as for the logistic regression baseline and the random baseline (see Figure 3). We can observe that logistic regression on program embeddings appears to be a very strong baseline. This is potentially due to a high correlation between certain ASTs and the "success" or "failure" classes. Our model does particularly well on recall on the "failure" class, which is pedagogically more important than precision. In education settings, it is much worse to miss students who will fail then giving superfluous support to students who would be successful anyway. It is also worth noting that with increasing number of timesteps, the gaps between the LSTM model and the logistic regression baseline on recall and F1 are increasing. In particular, while recall and precision roughly remain constant after timestep 5 for logistic regression, recall is improving significantly for the LSTM while precision stays roughly constant.

6.2 Analysis of Trajectory Representations

The hidden layer outputs of the trained neural net can be interpreted as the learned feature representations. Input samples that share patterns in the context of the learned task should ideally be mapped close to each other in the representation space.

Visualizing the learned representations of a neural net is an empiric method to explore what the neural net has learned. t-Stochastic Neighbor Embedding (t-SNE) [11] is particularly suited for visualization of high-dimensional data, as it can uncover structures at different scales. Figure 4 shows a t-SNE visualization of student trajectory embeddings for trajectories of length 6. We can observe five distinct clusters, labeled **A** through **E**, which we were also able to identify using the K-Means clustering algorithm with number of centroids set to 5. Each cluster contains trajectories sharing some high-level properties. Some statistics for the clusters are summarized in table 1.

Within these clusters of student trajectories, qualitative analysis found 3 distinct learning groups. **Cluster A** contains the best students who make consistent progress, showing logical debugging steps to apply programming concepts. Each step fixes an existing error and moves towards the correct final solution. A notable differentiator for **Cluster A** students is that they did not return to sections of their solution that they had already corrected. This demonstrates comprehension of the error and that they have digested the concept.

Students in **Clusters C** and **E** make inconsistent progress and show signs of random guessing. Some students methodically test combinations of elements to engineer a passing solution. This behavior likely represents uncertain or distrusted knowledge. This kind of behavior is overlooked by the current grading system as *Code.org* only considers cor-

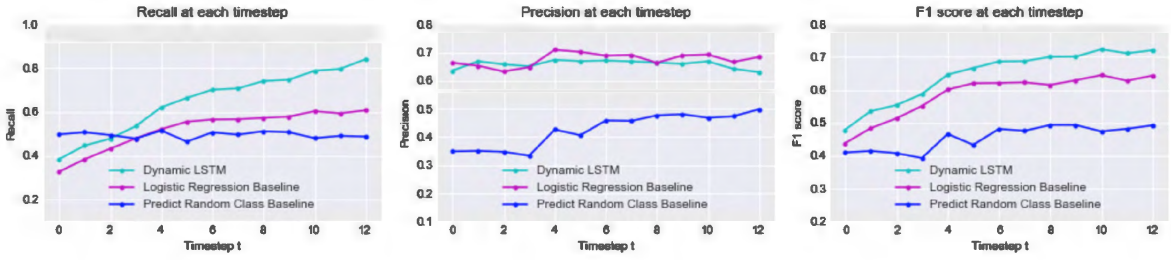


*Figure 3: Recall, Precision and F1 Score at each timestep on task **B**, for the "failure" class.*

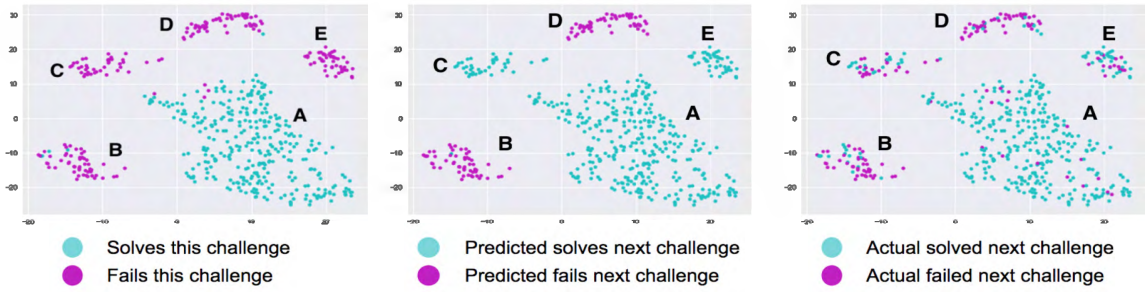


Figure 4: t-SNE visualization of trajectory representations. Left: Ground truth student success for current challenge. Center: Predictions for next challenge given student trajectories on this challenge. Right: Ground Truth student success for next challenge.

rectness of the final submission when scoring, a "number-right scoring" policy. The alternative is a "negative marking" policy, which would penalize students for wrong submissions along with the final answer. Educators have found that number-right scoring is a less reliable grading policy that overestimates student achievement particularly for students with more distrusted knowledge because it obscures whether responses represent true understanding or a lucky guess [2]. Anecdotally, we speculate that the students' success in the next problem may come from being able to reverse-engineer conceptual knowledge through repeated guessing.

Students in **Clusters B** and **D** appear to miss important concepts tested in this exercise. Students in **Cluster D** used an average of 9.21 blocks for every solution (see Table 1), almost twice as many total blocks as other clusters. Rather than solving the challenge with one generalized program, they break the challenge down into segments, hardcoding steps to pass each segment. Students in **Cluster D** have the highest usage of *move forward* blocks and *turn* blocks since students rely on these simple elements rather than the more complex *if-else* and *while* statements, both crucial learning components of this challenge. An ideal solution would include one *if-else* statement and one *while* loop. Students in **Cluster D** used the *if* statement only an average of 0.87 times and the *while* statement 0.60 times. Inspection of their programs show that students in **Cluster B** and **Cluster D** often disregard the *while* statement completely, unlike other clusters where students' solutions consistently contain the *while* loop) even if used incorrectly or inefficiently.

In summary, this analysis shows that our model can create more nuanced representations that lead to better predictions than a model that only looks at binary success indicators. Given that all students in **Clusters B, C, D,** and **E** performed poorly on the current exercise, a binary input model analyzing student success on Exercise 18 could not have distinguished between these poorly performing students. However, our model predicted that students in **Clusters C** and **E**, despite getting an incorrect answer for Exercise 18, would be successful on the next exercise. See Figure 4 *Left* and Figure 4 *Center*. Students in **Cluster C** went from a success rate of 0% in the current problem to a success rate of 48% in the next problem. Students in **Cluster E** went from 15% to 71%. This high success rate for **Clusters C** and **E** is visually noticeable in Figure 4 *Right*. The students' learning trajectories provided our model information to understand the students' learning at a deeper level and make these nuanced predictions, validating the claim that analyzing student trajectories provides richer data for the model.

Table 1: Statistics on student clusters (K-means)

Cluster	**A**	**B**	**C**	**D**	**E**
# students	316	51	44	58	62
Avg # total blocks	4.42	5.10	5.45	9.21	5.41
Avg # *if* statements	0.95	0.95	1.03	0.87	0.97
Avg # *while* blocks	0.89	0.83	0.97	0.60	0.92
Avg # *move forward* blocks	1.38	1.20	2.00	5.81	2.12
Avg # *turn* blocks	1.20	1.32	1.44	1.94	1.40
Success rate on current problem	99.7%	1.7%	0.0%	1.7%	14.5%
Success rate on next problem	95.3%	25.5%	47.7%	17.2%	71.0%

7. CONCLUSION

Our work focuses on multi-step exercises with unbounded solution spaces. While open-ended exercises encourage more flexible problem solving (e.g. in comparison to multiple-choice questions), understanding a student's progress is more challenging due to unbounded variations in student submissions. Given that digital learning platforms can easily archive the temporal dimension of student submissions, we proposed a new approach for learning representations of student knowledge by using program embeddings of student code submissions *over time* instead of hand-picked features. We showed that the trajectories of these representations produce distinct clusters of different student learning behaviors not picked up by a model that only observes binary success outcomes. We also showed that these representations can predict future student performance. We envision creating automated hint systems, where deep knowledge tracing has the potential to identify weaknesses and provide personalized feedback. By being able to anticipate student struggles in particular, we are in essence *capturing pedagogical content knowledge in an unsupervised fashion*. These applications could help improve and personalize the learning experience of students both in the classroom and on online education platforms.

8. ACKNOWLEDGMENTS

The authors would like to thank Code.org for providing the *Hour of Code* data set and Mehran Sahami, Nishith Khandwala and Daniel Guo for constructive feedback and help.

9. REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[2] R. F. Burton. Misinformation, partial knowledge and guessing in true/false tests. *Medical Education*, 36(9):805–811, 2002.

[3] S. Chaiklin. The zone of proximal development in vygotsky's analysis of learning and instruction. *Vygotsky's educational theory in cultural context*, 1:39–64, 2003.

[4] G. L. Cohen, J. Garcia, V. Purdie-Vaughns, N. Apfel, and P. Brzustoski. Recursive processes in self-affirmation: Intervening to close the minority achievement gap. *science*, 324(5925):400–403, 2009.

[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, 1994.

[6] I. Gutiérrez-Rojas, C. Alario-Hoyos, M. Pérez-Sanagustín, D. Leony, and C. Delgado-Kloos. Scaffolding self-learning in moocs. *Proceedings of the Second MOOC European Stakeholders Summit, EMOOCs*, pages 43–49, 2014.

[7] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[9] T. Jenkins. On the difficulty of learning to program. In *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences*, volume 4, pages 53–58. Citeseer, 2002.

[10] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416*, 2016.

[11] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[13] Z. A. Pardos and N. T. Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 255–266. Springer, 2010.

[14] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.

[15] C. Piech, J. Huang, A. Nguyen, M. Phulsuksombati, M. Sahami, and L. J. Guibas. Learning program embeddings to propagate feedback on student code. *CoRR abs/1505.05969*, 2015.

[16] C. Piech, M. Sahami, J. Huang, and L. Guibas. Autonomously generating hints by inferring problem solving policies. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 195–204. ACM, 2015.

[17] 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 *Proceedings of the first ACM conference on Learning@ scale conference*, pages 197–198. ACM, 2014.

[18] R. K. Sawyer. *The Cambridge handbook of the learning sciences*. Cambridge University Press, 2005.

[19] A. Sharif and B. Magrill. Discussion forums in moocs. *International Journal of Learning, Teaching and Educational Research*, 12(1), 2015.

[20] S. Shatnawi, M. M. Gaber, and M. Cocea. Automatic content related feedback for moocs based on course domain ontology. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 27–35. Springer, 2014.

[21] L. S. Shulman. Those who understand: Knowledge growth in teaching. *Educational researcher*, 15(2):4–14, 1986.

[22] R. Singh, S. Gulwani, and A. Solar-Lezama. Automated feedback generation for introductory programming assignments. *ACM SIGPLAN Notices*, 48(6):15–26, 2013.

[23] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[24] L. Vygotsky. Interaction between learning and development. *Readings on the development of children*, 23(3):34–41, 1978.

[25] K. H. Wilson, Y. Karklin, B. Han, and C. Ekanadham. Back to the basics: Bayesian extensions of irt outperform neural networks for proficiency estimation. *arXiv preprint arXiv:1604.02336*, 2016.

[26] 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*, volume 11, page 14, 2013.

[27] C. Ye and G. Biswas. Early prediction of student dropout and performance in moocs using higher granularity temporal information. *Journal of Learning Analytics*, 1(3):169–172, 2014.

[28] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180. Springer, 2013.

Development of a Trajectory Model for Visualizing Teacher ICT Usage Based on Event Segmentation Data

Longwei Zheng
lwzheng@dec.ecnu.edu.cn

Rui Shi
yyshirui@163.com

Bingcong Wu
648780962@qq.com

Xiaoqing Gu
xqgu@ses.ecnu.edu.cn

Yuanyuan Feng
yyfeng@fl.ecnu.edu.cn

East China Normal University
Shanghai, China, 200062

ABSTRACT

The adoption of educational technologies such as e-textbook has offered a new opportunity to gain insight into teachers' usage of ICT (Information and Communication Technologies). In the e-textbook platform, customized digital products and the learning activities organized in digital environment require teachers to make greater efforts in planning lessons and producing resources. In addition, usage of technology can vary greatly from one group of teachers from another in various contexts. In this study, we demonstrate how computations like event segmentation and contextual numbers can be exploited in visualizing trajectories of teacher's ICT usage. We also study with the experience structure via the implicit patterns within the raw data of an e-textbook platform. Such automated visual characterization might be helpful to the wide and scalable application of teaching analytics to represent teacher's ICT usage.

Keywords

Visual analytics; teaching analytics; contextual numbers; ICT

1. INTRODUCTION

Information and Communication Technologies (ICT) are becoming increasingly pervasive in education [12] and are making a difference in the ways teacher plan lesson and organize activities [13]. It is also well documented that teachers need support to make effective use of information technology in their teaching, because the incorporation of ICT is not easy process which involves many technical complexities [10]. With a goal of better use of ICT, teaching analytics is conceived as an analytics approach that focuses on the design, development, evaluation of visual analytics methods and tools for teachers [20].

However, the crucial step of supporting teacher interventions based on learning analytics insights remains under-supported [17]. As it often happens elsewhere in learning analytics, most learning environments are not designed for data analysis and mining [8], even if they do analysis, they are designed to focus on analyzing student learning or behavior and provide feedback to the teacher [1,20], not to analyze and represent the teacher's data they store. Therefore, many studies depict learning analytics for teachers rather than analytics about teaching [17].

In addition, although much work has been done on visualizing analytics result, their design and use is less understood, which can lead to the weak implementation as a result of promoting ineffective feedback [21,19,3]. In many cases, however, it is not easy to compare the complex objects over high dimension visualization which requires users to understand the semantics of visual representation and feature that are assumed by model and algorithm. Besides that, some visualization approaches present the narrow scope of the representation, as focused on one snapshot of a certain topic of data for a certain period time. It usually did represent several aspects of dataset that occurring within the environments but did not represent the nature of connections inside the datasets and provide a global view of usage [2]. As a consequence, the application of dashboards requires additional information processing in various work.

The purpose of this study is to design a computational procedure based on behavior data with the intent to create a visualization of trajectory that will help describe teacher ICT usage.

To explore these issues, we make a case study in which the data is gathered from an e-textbook server without any additional sensor or APIs. In previous study [23], we found that a segmentation method is effective in effort to provide features distilled for predicting e-textbook adoption in early days. In this study, we bring together event segmentation and one-dimension *Self-organizing map* to integrate an authentic teaching experience involving digital environment with embedded robust and continuous characterizing of ICT usage trajectories. The raw data records which were created in a e-textbook platform will be computationally transformed and displayed, so that teachers and other stakeholders can utilize the information of result of contextual visualization to get insights and improve dynamic and diagnostic decision-making.

2. DATA

We investigated issues within the context of data from an e-textbook platform named *ZoomClass*. *ZoomClass* includes a web-based authoring environment and an iPad application for teachers. Teachers were given access to customize all digital content for specific teaching objectives. They typically create courses, upload media resources and products which are mostly customized by themselves in other tools (such as PowerPoint), design tasks, assign activities, and insert quizzes on the web-based environment before class. Also, they can record and upload photos and videos by iPad application. The users of *ZoomClass* are teachers and students at a primary school of Shanghai. We obtained data on teacher authoring action records and student response action records, for 110 teachers enrolled in this e-textbook platform, observed over more than 5 semesters since 2014 October. Until January 2017, the teachers have performed a total of 117,324 actions, created 4,653 courses, uploaded 16, 901 digital resource included almost 9,000 image products and get 3,364,533 responses from students.

Figure 1. The iPad application *ZoomClass*

3. METHOD

In this study, we bring together an event segmentation algorithm and a nonparametric mapping which is called contextual numbers, to integrate an authentic teaching experience involving e-textbook platform with embedded continuous characterizing of ICT usage trajectories. In general, the intent of event segmentation is to determine how a threshold should be set automatically when partitioning action streams into usage feature spaces. And the approach of contextual numbers is used to map the high dimensional space of usage to a continuous one-dimensional numerical field, which are ordered in the given context, similar numbers refer to similar high dimensional states of usage. Figure 2 shows the computational procedure and associated steps, which will be discussed in detail in this section.

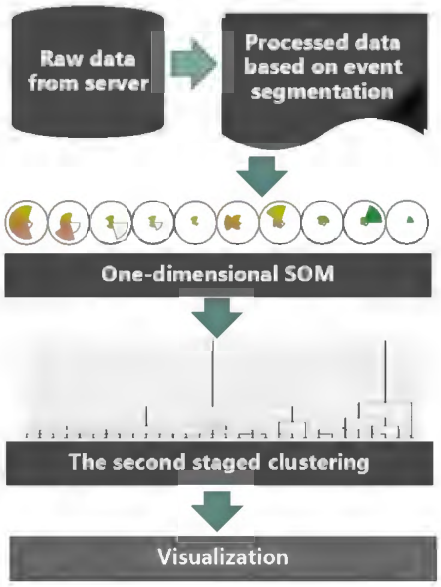


Figure 2. The computational procedure and associated steps

3.1 Event Segmentation

In our study, data comes from the raw records of an e-textbook platform. Two characteristics of this data are contained: 1. Data only recorded by the back-end server without any sensor embedded in front-end, that means the grain size of our data is much bigger when comparing with the sensor data (such as clickstream); 2. Multi-platform operation, which would cause the break off of data capturing when teacher transfer to another platform. Thus, these two problems lead to an amount of missing action data among our data set. In considering of this issue, an event segmentation method is introduced to transform action records to event dataset.

Event segmentation is a method means dividing a given number of observation into subsets with statistical characteristics that are similar within each subsets and different between subsets [4]. In this study, the goal of event segmentation is to automatically partition teacher actions into separate events, the segmentation method is only based the date time information of server log records. We consider action records in chronological order such that

$$R = \{R_1, \dots, R_m\} \tag{1}$$

where R_i is the ith action record in data set R with length m. A event segment $e_{i,j}$ which is a subset of R can be given as

$$e_{i,j} = \{R_i, \dots, R_j\}, \ 1 \le i \le j \le m \tag{2}$$

Intuitively, the time differences between inter-action records in an event are typically smaller than time differences between inter-action records from separate events, so the time intervals between observations are often considered as a criterion to judge partitioning [11].

With respect to the fact that teachers with various contexts have different usage of e-textbook, it is very likely that teachers perform diverse action frequencies during different period. Zheng and colleagues [24] developed an analysis method to discover the user water behavioral habits, in their invention, a novel continuous event segmentation algorithm based on threshold optimization was created to automatically separate the water usage records into multiple individual bath events for each user, this study employed a similar method to create features from teachers' action record data sets. In the event segmentation algorithm created by Zheng et al., a threshold of time difference has been used to determine whether consecutive action records are in a same event. The algorithm consists of following steps: 1. Compute inter-action intervals; 2. Compare every interval to the threshold of time difference. In step 2, If the interval is smaller, these two inter-actions are considered in a same event, if the interval is greater, they are divided into two different events. The algorithm will run through all of inter-action intervals, then we can obtain individual events from action log sets. An automatic threshold optimization model was developed to search the optimal threshold value to segment event.

The threshold optimization of each teacher in one week consists of following steps: 1. Segment events with successively varying thresholds, a fixed time delta d is set between two successive thresholds, we consider this threshold set in chronological order such that

$$TS = \{ts_1, ts_2, ts_3 \ \dots\} \tag{3}$$

2. Compute event number y for each threshold ts; 3. Specify minimum rate of event numbers' change for optimal threshold detection. In step 3, optimization algorithm uses a sliding window with a fixed size. The window can only contain n points, beginning at the current point and ending right before the next identified point. The optimization tries to find a possible starting point which is followed with a sequence of almost unchanged points. Suppose the threshold of the current point is ts_j, the average rate of event numbers' change cr is defined as follows:

$$cr(ts_j) = \frac{1}{n}\sum_{i=j}^{n}\left|\frac{y_i - y_{i+1}}{d}\right| \tag{4}$$

the final optimal threshold can be selected from given threshold set as follows:

$$ot(TS) = \text{Argmin}_i(cr(ts_i)) \quad (5)$$

Figure 3 shows an example of an event segmentation with varying thresholds. Here, the number of events declines rapidly when threshold is smaller than 10 minutes, it implies most inter-action intervals of the teacher are smaller than 10 minutes. And there is a significant possibility to separate an individual event into two or more sub-events if a small value is determined as threshold. Therefore, an interval value is more rational to determined as threshold until the number of individual events touches down and levels off at almost zero. The slopes of inter-thresholds are used to detect the signal of change rate. When the average of *n* (In this case, *n* is set to 8) consecutive slopes of inter-threshold are closet to zero, the first threshold point in sliding window is flagged as optimal threshold value of an individual teacher's inter-event interval in a week. In Figure 2, the point of 26 minute is possible the optimal threshold.

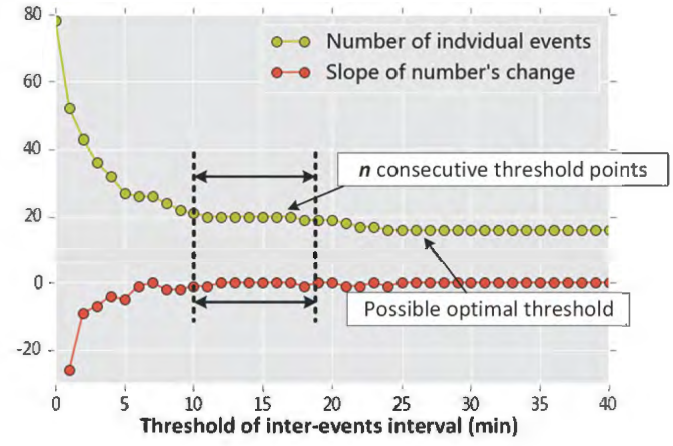


Figure 3. A sliding window which searches an optimal threshold point. Suppose that *n* = 8, the point of 26 minute is possible the optimal threshold

3.2 Creation of Features

We employed event segmentation algorithm in both teachers' and students' action records. The resulting segmented event dataset consists of 10,146 total event rows from 117,324 teachers' authoring action records, and 23, 203 total teaching activity rows from 3,364,533 students' response records. With the respect to trajectory visualization, a process of aggregation is performed in these events for frequency conversion and resampling by a week to generate time series data set. Eight features were distilled from processed dataset:

The total duration of producing event (DE) – Event transformed from teachers' action data which is about producing indicates the fact that they create new media resources and upload files with the authoring platform. The total duration of producing event allows us to know how long would teachers spend on preparing their lessons on the learning platform in a week.

The number of long producing events (LE) – In order to minimize noise in the segmentation, we discretize events (exclusive of single-action events) into three buckets based on quartiles of durations of every events. The producing event with a duration longer than upper quartile is considered as long producing event.

The number of middle producing events (ME) – The producing event with a duration longer than lower quartile and shorter than the upper quartile is considered as middle producing event.

The number of short producing events (SE) – The producing event with a duration shorter than the lower quartile is considered as short producing event.

The number of single-action producing events (SPE) – The events with only one single action are special in this case. A single-action event could be created in the situation where a resource producing last for a long time without any other neighboring action or just a testing action is performed. Therefore, we separate the single-action events into two groups by its action type.

The number of single-action common events (SCE) – The event consists of only one single action which has not explicit relation with producing, such as creating a virtual folder with a default name, are considered as a common event with a single action.

The number of teaching activities (TA) – Teaching activity in this study is about 'consuming' which indicates the evidence that teachers utilize the resources they've uploaded to the learning platform before class. With event segmentation, teaching activity is transformed from students' concurrent response records which include answer submitting, media file uploading and help requesting. The tasks assigned inside e-textbook application by teachers are also considered as the teaching activity even they are mostly finished after class.

The number of engaging days (ED) – The day that teacher is active in authoring platform is considered as an engaging day. However, the single-action common events are omitted when determining whether a teacher is active in a day.

3.3 Contextual Numbers

Self-organizing map is a nonlinearly projecting mapping algorithm which is introduced by Kohonen [7]. The earliest applications were in engineering tasks, later the algorithm has become a generic methodology， which has been applied in clustering, visualization, data organization, characterization, and exploration [6]. Self-organizing map consists of organized nodes that include a N-dimensional weight vector. In regard to the observations $X = \{x_1, x_2, \ldots, x_n\}$ in N-dimensional space $x_i \in R^n$, the procedure can be summarized in three processes: competition, cooperation and self-adaptation. The SOM training algorithm can be thought of as a net which is spread to the data cloud. In general, it moves the weight vectors to make them span across the data cloud, so that the neighboring nodes get similar weight vector [7].

Traditionally, most applications of SOM algorithm were organized in a two-dimensional coordinate system (such as [2], [18]). In these applications, after projecting the data to SOM grid, the indexes of nodes as single values are able to create a new contextual order, which can be used to transform each high-dimensional point to a new computational space. The close points are similar in this context, however, this similarity is not interpretable in a single dimensional arrow comparing with classic number space [15].

In this regard, a one-dimensional SOM called *contextual numbers* was introduced by Moosavi [14], this method can be seen as a sequence of ordered numbers pointing to a high-dimensional space, these numbers are ordered according to their similarities within the selected high-dimensional state space or context. In contextual numbers, *K* nodes will be produced in one-dimension

after the mapping of SOM with X, and each node with an attached high-dimensional weight vector represents the original information. Instead of using the values within the nodes, a series new contextual orders were created. It can be summarized in the following steps: 1) Calculate the posterior probability of assigning contextual number; 2) Select the corresponding number when the posterior probability reaches the dominant peak as the node index. The difference between the two-dimensional and the one-dimensional can be reflected in the relation of indices and the weight vector. In a two dimensional grid, the neighborhood similarity expands in two directions. Therefore, there is no direct correlation between the numerical values of indices and the similarity of their weight vectors. But in one dimensional grid, valuable property of contextual numbers is that there is a direct correlation between indices [14]. As in the most two-dimensional cases, the final index of trained SOM will not be used directly as a numerical value but instead of assigned weight vector, contextual numbers allow us to create a continuous number space converted from a high dimension space, which can fit completely to a univariate space [15]. In terms of usage time series analysis in this study, we can have a univariate usage time series for each teacher along the week by conversion of contextual numbers.

It should be noted that the index we mapped to each node is not the classic numbers. The value of these indices are not means the performance grades, but the similarity of two or more nodes. If two index have close values (e.g. node number 1 and node number 2 in SOM network. Numbering is arbitrary, but we usually start from upper-left corner and go row by row) they are similar in this context [15].

3.4 The second staged clustering

With the indices (contextual numbers), hierarchical clustering is performed in this part. One advantage of hierarchical clustering algorithms is that it can help with the interpretation of the results by creating meaningful taxonomies. On account of these numbers implicate contextual information which is difficult to interpret, a common two-staged clustering is employed to combine most similar indexes, as what the previous applications did to the nodes of two-dimensional SOM grid (such as [22,16]). Then a typology from clustering results is developed, which is also proven to make it more accessible when stockholders are involved in exploration of data using visual inspection [5].

In order to get good performance of clustering, first we employ the k-means and the intrinsic metrics—within-cluster Sum of Square for Error (SSE) to compare the performance of different number of clusters. Based on the within-cluster SSE, the elbow method is used to estimate the optimal number of clusters k for a given task. In this study, the elbow is located at k = 5, thus we choose it as the number of clustering. Finally, we perform hierarchy clustering on the contextual numbers.

4. RESULT

This section presents the two stages of our research: in the first part the high dimensional observations from the processed time series data are converted to corresponding contextual numbers, a series of continuous indices and a specific typology which is built for interpretation; in the second part, we apply this to produce visualizations on teacher ICT use trajectory.

4.1 Usage Typology

Firstly, a SOM network is trained on a single dimension network with the eight-dimensions usage data, and the range of indexes is set from 0 to 29. Therefore, each index node has two neighbors except the first and the last. In this regard, we apply the second staged clustering to discretize the contextual number indexes into groups for interpretation, and it is determined that there are five groups to be discovered in our study. The details of the groups are shown in Table 1.

As can be seen in Table 1, Group A characterizes the *Limited use* pattern. Teachers in this group have spent very few time on using the authoring platform. Few product indicates that they never upload media resources; Meanwhile, they organize a few activities once a week, which illustrates the technology is seldom used in their classes; The usage of this group usually is performed at the beginning or the end of semesters.

Group B characterizes the *Early use* pattern. The teachers in this group organize even fewer activities than the teachers in Group A. But they have at least a middle or a short producing event a week, which means some resources were produced to prepare for the lesson., they try to use the platform to prepare lessons. We find out usage of that this group is the mainstream during the first three semesters.

Group C characterizes the *Consuming use* pattern. Teachers in this group begin to use the learning platform more frequently than Group A and Group B. They are very willing to implement this application to organize teaching activities and usually have plenty of responses on the e-textbook, but they only produce at most once a week. We can also find that they have highest single-action common actions than teachers who are in other groups, since they tend to consume the resources rather than produce.

Group D characterizes the *Moderate use* pattern. The teachers in this Group begin to frequently produce resources on the platform, many of them would use the learning platform three out of five working days for every week. Compared to those three groups we mentioned before, teachers in this group are actually using this technology to plan lessons with the resources which are built by themselves. As they are producing frequently, we find that they have highest mid-events. But compared to teachers in Group C, they have slightly less activities which means they are not relying on the e-textbook to teach in classes like teachers in Group C do.

Group E characterizes the *Intensive use* pattern. The teachers in this Group usually heavily produce resources during a long time, they produce many resources on the platform. Among the five working days each week, they almost produce everyday, they also organized numerous activities that means they are actually use the application a lot in class.

Therefore, we can build some meaningful names and stories for every group and create fictitious typology labels to the contextual number indexes, in order to provide an easy way to understand the contextual meaning of indexes. As shown in Table 2, we summarize each group, giving the key characteristics and the indexes belong to.

Table 1. Grouping results showing the mean value for each feature and cluster

	Group				
	A	B	C	D	E
Name	Limited use	Early use	Consuming use	Moderate use	Intensive use
Index	0~5	6~14	15~19	20~25	26~29
DE	0.258	34.161	35.389	78.920	257.665
LE	0.000	0.285	0.288	0.522	2.156
ME	0.000	0.692	0.742	2.597	2.012
SE	0.029	0.371	1.000	0.827	0.514
SPE	0.206	0.432	0.327	0.931	0.452
SCE	0.531	0.532	2.336	1.743	2.218
TA	6.396	2.883	21.107	8.560	32.863
ED	0.025	1.065	1.408	2.866	4.174

Table 2. The user typology derived from two-stage clustering

Group	Indexes	Name	Typology Label
A	0-5	Limited use	Almost no product
			A few activities once a week
			Centralized in the beginning or end of semesters
B	6-14	Early use	Few teaching activities
			At least a middle/short event a week
			The mainstream of the earlier stage
C	15-19	Consuming use	Plenty of activities
			Producing at most once a week
			More independent actions
D	20-25	Moderate use	Frequently producing
			Highest middle-event rates
			Slightly less activities
E	26-29	Intensive use	Heavily producing during a long time
			Almost producing everyday
			Organizing numerous activities

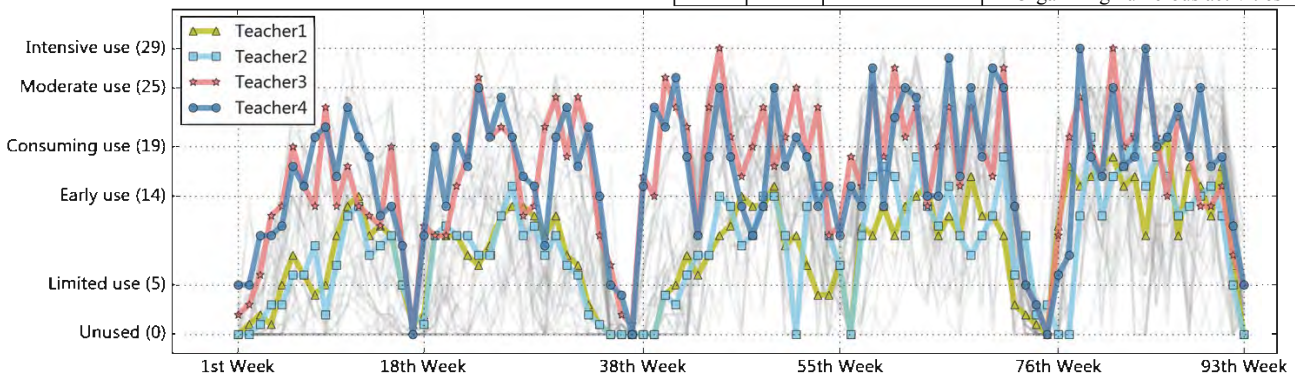


Figure 4. Sample trajectory visualization

4.2 Usage Trajectory

Finally, we can explore visual trajectory with the typology to identify the implicit patterns and hypothesis. This visualization provides the capability to trace states and discovery patterns without reducing the information to simple statistics, it illustrates the teacher usage trajectories which is helpful as teachers and stockholders rarely trace the process of how they use the ICT in teaching.

As shown in Figure 4, Y axis is the index of one dimension SOM and X axis shows the week which is the length of time to be observed in this case. The figure shows the states and trajectories of each teacher over the time. Therefore, similar teacher has similar index number during the time. It allowed us to identify how a teacher uses this technology by comparing the trajectories and pattern of each teacher in relation to the others using the contextual numbers of SOM. If we are familiar to a few teachers' usage, we can consider these teachers as contexts for relative positioning when identifying a new teacher usage, even we don't know the interpretation of the contextual numbers. As shown in Figure 4, we can consider Teacher 3 as a template if we were familiar with the his or her usage or performance, then the usage of Teacher 4 is easy to be identified by comparing their similar trajectories. The result of our statistical analysis on index set shows that the Teacher 3 and Teacher 4 have a lowest Euclidean distance. On the other hand, we can also automatically find similar teachers based on distance calculation between each trajectories.

As the use of an "adopted" technology can vary greatly from one group of teachers to another [9], this figure provide an easy way to partition the teachers in terms of the variations along the two dimensions of contextual index and time of usage. In this case, as shown among the intense user group, Teacher 3 and Teacher 4, the contextual numbers indexes mostly ranged from 10 to 29, which were almost consistently higher than the the indexes of moderate user group, Teacher 1 and Teacher 2, whose usage was mostly labeled as early use or consuming use in the first three semesters. Apparently, Teacher 1 and Teacher 2 adopted this tool for teaching, but did not rely on the tool in the same way that Teacher 3 and Teacher 4 did. However, it is not rational to evaluate the performance of teachers' ICT with the number of index, because the SOM indexes are used as computable numbers to represent the state based on the contexts, but the values of indexes don't follow the concept of natural numbers which can be interpreted as ordered grades. Therefore, the higher index does not always indicate better performance, even though it seems that higher contextual number index is labeled with more intensive use in this case.

This method is also able to indicate potential patterns from trajectories of contextual numbers. As shown in Figure 4, the state of teacher's usage fluctuates visibly over each semester. More specifically, as we can see Teacher 4 in the last semester, at beginning of this semester the number of state stands at a limited usage index. Then, the number shoots up over the next two weeks, peaking at 29, which means a state of intensive use. After that, the contextual number declines rapidly for two weeks, bottoming out at 16 which is labeled as a consuming using index. The next week experiences a very sharp rise, reaching the intensive use area again. According the indexes of usage in the following weeks, a total of 5 peaks can be respectively detected. The peak pattern

discovered from trajectory plotting describes a behavior that teacher tends to produce the teaching resources intensively in first one week, then consume them in this week and the following one to two weeks. We apply frequent sequence mining to segmented trajectory data of active teachers to explore this idea, the result shows that the sequences of peak pattern (such as Sequence [Consuming use, Intensive use, Consuming use]) all get highest frequency in the group of their week length.

5. CONCLUSION

This paper introduced a computation procedure for visualizing the trajectories of teacher's ICT usage based on the resource producing process and the experience structure via the implicit patterns within the raw data by event segmentation and contextual numbers. The resulting visualization provides the capability to trace states and discovery patterns without reducing the information to simple statistics, such automated visual characterization might be helpful to the wide and scalable application of teaching analytics to represent teacher's ICT usage. Our future work will be oriented to the *spatiotemporal dynamic* in education, especially the application of ICT, in which the knowledge extraction of web-based education system can be viewed as a formative evaluation technique. In this condition, high-dimensional time series with different features can be replaced by a series of contextual numbers, where this numerical numbers can be embedded in any data driven analysis and prediction [14].

6. ACKNOWLEDGMENTS

We would like thank Liangjun Zhang and colleagues for the tutorial of segmentation algorithm in their practice book.

7. REFERENCES

[1] Dyckhoff, A. L., Zielke, D., Bültmann, M., Chatti, M. A., & Schroeder, U. 2012. Design and implementation of a learning analytics toolkit for teachers. *Educational Technology & Society*, *15*(3), 58-76.

[2] Fournier, H., Kop, R., & Sitlia, H. 2011. The value of learning analytics to networked learning on a personal learning environment. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge* (pp. 104-109). ACM.

[3] Gašević, D., Dawson, S., & Siemens, G. 2015. Let's not forget: Learning analytics are about learning. *TechTrends*, *59*(1), 64-71.

[4] Gedikli, A., Aksoy, H., and Unal, N. E. 2008. Segmentation algorithm for long time series analysis. *Stochastic Environmental Research and Risk Assessment*, *22*(3), 291-302.

[5] Gibson, D., & de Freitas, S. 2016. Exploratory analysis in learning analytics. *Technology, Knowledge and Learning*, *21*(1), 5-19.

[6] Kaski, S., Kangas, J., & Kohonen, T. 1998. Bibliography of self-organizing map (SOM) papers: 1981–1997. *Neural computing surveys*, *1*(3&4), 1-176.

[7] Kohonen, T. 2013. Essentials of the self-organizing map. *Neural Networks*, *37*, 52-65.

[8] Krüger, A., Merceron, A., & Wolf, B. 2010. A data model to ease analysis and mining of educational data. In *Educational Data Mining 2010*.

[9] Leary, H., Lee, V. R., & Recker, M. 2014. More than just plain old technology adoption: Understanding variations in teachers' use of an online planning tool. *ICLS 2014 Proceedings*, 110.

[10] Levin, T., & Wadmany, R. 2008. Teachers' views on factors affecting effective integration of information technology in the classroom: Developmental scenery. *Journal of Technology and Teacher Education*, *16*(2), 233.

[11] Loui, A. C., & Savakis, A. E. 2000. Automatic image event segmentation and quality screening for albuming applications. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on* (Vol. 2, pp. 1125-1128). IEEE.

[12] Martinovic, D., & Zhang, Z. 2012. Situating ICT in the teacher education program: Overcoming challenges, fulfilling expectations. *Teaching and Teacher Education, 28*(3), 461-469.

[13] Maull, K. E., Saldivar, M. G., & Sumner, T. 2010. Online curriculum planning behavior of teachers. In *Educational Data Mining 2010*.

[14] Moosavi, V. 2014. Computing With Contextual Numbers. *arXiv preprint arXiv:1408.0889*.

[15] Moosavi, V. 2014. Toward Engendering Contextual Numbers:Self Organizing Maps in Coexistence with Data-Driven Logic. *http://nbviewer.jupyter.org/gist/sevamoo/9543236*.

[16] Ong, S. H., Yeo, N. C., Lee, K. H., Venkatesh, Y. V., & Cao, D. M. 2002. Segmentation of color images using a two-stage self-organizing network. *Image and vision computing*, *20*(4), 279-289.

[17] Prieto, L. P., Sharma, K., Dillenbourg, P., & Jesús, M. 2016. Teaching analytics: towards automatic extraction of orchestration graphs using wearable sensors. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge* (pp. 148-157). ACM.

[18] Schreck, T., Bernard, J., Von Landesberger, T., & Kohlhammer, J. 2009. Visual cluster analysis of trajectory data with interactive kohonen maps. *Information Visualization*, *8*(1), 14-29.

[19] Tanes, Z., Arnold, K. E., King, A. S., & Remnet, M. A. 2011. Using Signals for appropriate feedback: Perceptions and practices. *Computers & Education*, *57*(4), 2414-2422.

[20] Vatrapu, R., Reimann, P., Hussain, A., & und Beratung, M. P. F. 2012. Towards teaching analytics: Repertory grids for formative assessment. In *Proc. International Conference of the Learning Sciences (ICLS) 2012*.

[21] Verbert, K., Duval, E., Klerkx, J., Govaerts, S., & Santos, J. L. 2013. Learning analytics dashboard applications. *American Behavioral Scientist*, *57*(10), 1500-1509.

[22] Vijayakumar, C., Damayanti, G., Pant, R., & Sreedhar, C. M. (2007). Segmentation and grading of brain tumors on apparent diffusion coefficient images using self-organizing maps. *Computerized Medical Imaging and Graphics*, *31*(7), 473-484.

[23] Zheng, L., Gong, W., & Gu, X. 2017. Predicting e-textbook adoption based on event segmentation of teachers' usage. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference* (pp. 560-561). ACM.

[24] Zheng, Q., Liang, G., Quan, Y., Gao, W. and Wang, S. 2015. Analysis method, apparatus and system for user water bath behavioral habits. CN105115164A. 2015(in Chinese).

Posters

Modeling Network Dynamics of MOOC Discussion Interactions at Scale

Jingjing Zhang
Beijing Normal University
Beijing, China
jingjing.zhang@bnu.edu.cn

Maxim Skryabin
Stepik
Saint Petersburg, Russia
ms@stepik.org

ABSTRACT
This paper attempts to model network dynamics of MOOC discussion interactions. It contributes to providing alternatives to conducting null hypothesis significance testing in educational studies. Using data collected from two successive psychology MOOCs in 2014 and 2015, the probabilistic longitudinal network analysis was performed by employing stochastic actor-based models with statistical accuracy. Understanding the mechanisms that drive the dynamics of discussions shed light on the design of a self-generated and learner-supported learning environment to meet the challenges of accommodating a massive and global student body.

1. Author Keywords

interactions; SIENA; probabilistic longitudinal network analysis; network dynamics; peer-supported learning.

2. ACM Classification Keywords

I.6.4 Simulation and Modelling: Model Validation and Analysis.

INTRODUCTION

Understanding learning at scale is a challenging task. As stated earlier, particular concerns are the extremely high rates of attrition and the pattern of steeply unequal participation in MOOCs. Using traditional educational methods fail to link the observed behavioral patterns within a network to the underlying the effects of network structure and the role of the participants that may explain why these patterns emerge. This study is an empirical investigation of the network dynamics of MOOC discussions, and attempts to make a contribution to providing alternatives to conducting null hypothesis significance testing in educational studies. Understanding the mechanisms that drive the dynamics of discussions shed light on the design of a self-generated and learner-supported learning environment to meet the challenges of accommodating a massive and global student body.

Using data collected from two successive psychology MOOCs in 2014 and 2015 and applying probabilistic longitudinal network analysis, this study seeks to rigorously measure the dynamic mechanisms that drive discussion change over time. The probabilistic analysis was performed by employing stochastic actor-based models with statistical accuracy.

METHODS

The probabilistic longitudinal network analysis was performed by employing stochastic actor-based models defined and evaluated with the program Simulation Investigation for Empirical Network Analysis. Four hypotheses are proposed to test the network dynamics of MOOC discussions.

Hypothesis 1 (H1): There is a tendency towards reciprocation in studied discussion networks (i→j and j→i). (Dyadic Level)

Hypothesis 2 (H2): There is a tendency towards transitivity (i.e. increasing transitivity and reducing distance between actors; i→j, j→k and i→k). (Triadic Level)

Hypothesis 3 (H3): There is a tendency towards the increasing volume of interactions between learners themselves.

Hypothesis 4 (H4): There is a tendency towards preferential attachment within the studied networks.

PRELIMINARY RESULTS

Descriptive statistics of the discussion network

In 2014 MOOC, 1915 participants posted 5251 messages in total, of which 217 are threads, 5034 are replies and comments, while in 2015 psychology MOOC, 962 threads were provided, and 3097 are replies and comments.

In 2014 Psychology MOOC, there are topics initiated by TAs to collect feedbacks for individual sections and to answer content-related Q&A for each section. As shown in Figure 1, the number of the postings falling into the discussing categories initiated by TAs is relatively larger than the number of the same topics which are initiated by learners themselves. The category "content-related Q&A initiated by TAs for individual sessions" seems to attract a good number of replies and comments over time. Interestingly, as shown in Figure 1, the discussions of exercises share a similar quantitative pattern of content-related discussions; while the enquiries about the logistics of the course follow a similar pattern of technical discussions in both two offerings of psychology MOOCs. In

2015 Psychology MOOC, technical problems occurred during the mid-examination, showing as a peak in Figure 1.

Network Dynamics

Table 2 and 3 present the results of SIENA estimation. As shown in Table 2 and 3, the results of Model 0 (network effects: reciprocity; transitivity) indicate a tendency for participants to create mutual relationships at both dyadic and triadic levels, which leads to cohesiveness in the studied networks. This confirms that hypothesis H1 and H2 are accepted. The exceptional case is the transitivity effect identified in the category of "feedback" (i.e. general feedbacks to instructors and TAs initiated by learners), where there is no tendency for participants to create mutual relationship at triadic levels. This deserves a detailed examination in the future analysis. Interestingly, under the topic categories of "feedback" and "TA about" (i.e. enquiries about the logistics of the course initiated by TAs), when same role is used as a control variable, the transitivity effect is significant with a negative coefficient. Compared to discussions in other categories, it is less likely to create cohesive subgroups when learners provide feedbacks to the course and enquiries about course logistics.

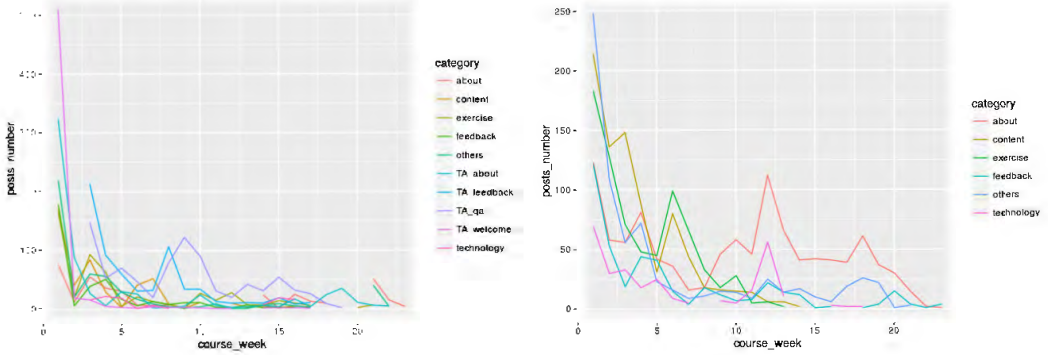


Figure 1. The number of postings within different discussion topics over time (2014 left & 2015 right).

In both courses, same role is a significant covariate effect with a negative coefficient. Thus, H3 (Model 1: reciprocity; transitivity; same role) is rejected, indicating that there is no tendency towards an increasing volume of interactions between learners.

H4 (Model 2: reciprocity; transitivity; Activity of alter) states that there is a tendency towards preferential attachment within the studied networks. The preferential attachment effect is not consistent among discussions of different topics. In most discussions, there is a tendency for participants who are actively involved in forum discussions in the early stages to become even more engaged over time. Nevertheless, when discussing exercises in 2014 Psychology MOOC, there is no preferential attachment effect, which deserves a future examination.

Category	Model 0	Model 1	Model 2
about	3.49* (0.37) 1.06* (0.29)	3.27* (0.40) 0.82* (0.32) -2.25* (0.24)	3.89* (0.35) 1.32* (0.31) -0.51 (0.34)
content	4.53* (0.32) 0.76* (0.23)	4.48* (0.32) 0.84* (0.23) -1.56* (0.28)	4.27* (0.29) 0.63* (0.23) 0.20* (0.09)
exercise	4.27* (0.35) 0.35 (0.34)	4.17* (0.34) 0.33 (0.36) -2.34* (0.35)	5.44* (0.46) 0.97 (0.43) -1.26* (0.60)
feedback	3.36* (0.50) -0.92* (0.41)	3.33* (0.50) -0.96* (0.40) -1.16* (0.44)	4.03* (0.78) -0.38 (0.74) -0.89 (0.93)
technology	3.26* (0.61) 0.03 (0.57)	3.02* (0.67) -0.15 (0.59) -2.18* (0.30)	3.63* (0.54) 0.36 (0.64)
			-0.53 (0.52)
TA about	5.13* (0.33) 0.16 (0.12)	3.67* (1.02) -0.41* (0.13) -5.91* (0.10)	4.71* (0.78) -0.81* (0.15) 0.20* (0.01)
TA feedback	3.30* (0.33) 0.87* (0.18)	0.64 (0.35) 0.08 (0.09) -4.98* (0.11)	0.67 (0.44) 0.28 (0.16) 0.12* (0.004)
TA Q&A	1.56* (0.44) 1.35* (0.17)	0.37 (0.48) 0.50 (0.08) -4.05* (0.10)	0.42 (0.46) 0.89* (0.17) 0.19* (0.01)

Table 1. Estimation results of network effects with standard errors in parentheses (2014 Psychology)

Category	Model 0	Model 1	Model 2
about	3.63* (0.21) 1.61* (0.17)	3.39* (0.23) 1.21* (0.17) -3.02* (0.19)	3.08* (0.22) 1.02* (0.20) 0.16* (0.01)
content	4.23* (0.23) 1.37* (0.20)	4.26* (0.22) 1.35* (0.20) -1.39* (0.49)	3.89* (0.26) 0.71* (0.21) 0.09* (0.01)
exercise	3.46* (0.27) 1.23* (0.23)	3.52* (0.26) 1.23* (0.24) -0.02 (1.18)	3.28* (0.25) 1.04* (0.23) 0.11* (0.04)
feedback	3.68* (0.37) 1.03* (0.33)	3.50* (0.38) 1.01* (0.34) -2.69* (0.28)	3.33* (0.35) 0.82* (0.37) 0.28* (0.10)

Table 2. Estimation results of network effects with standard errors in parentheses (2015 Psychology)

Studying MOOC Completion at Scale Using the MOOC Replication Framework

Juan Miguel L. Andres
Ryan S. Baker
University of Pennsylvania
Philadelphia, PA 19104
+1 (877) 736-6473
andresju@gse.upenn.edu,
rybaker@upenn.edu

George Siemens
Catherine A. Spann
University of Texas Arlington
Arlington, TX 76019
+1 (817) 272-2011
gsiemens@gmail.com,
caspann17@gmail.com

Dragan Gašević
University of Edinburgh
Edinburgh EH89YL, UK
+44 (131) 650-1000
dragan.gasevic@ed.ac.uk

Scott Crossley
Georgia State University
Atlanta, GA 30303
+1 (404) 413-5000
sacrossley@gmail.com

ABSTRACT

Research on learner behaviors and course completion within Massive Open Online Courses (MOOCs) has been mostly confined to single courses, making the findings difficult to generalize across different data sets and to assess which contexts and types of courses these findings apply to. This paper reports on the development of the MOOC Replication Framework (MORF), a framework that facilitates the replication of previously published findings across multiple data sets and the seamless integration of new findings as new research is conducted or new hypotheses are generated. MORF enables larger-scale analysis of MOOC research questions than previously feasible, and enables researchers around the world to conduct analyses on huge multi-MOOC data sets without having to negotiate access to data.

Keywords

MOOC, MORF, replication, meta-analysis.

1. INTRODUCTION

Massive Open Online Courses (MOOCs) have created new opportunities to study learning at scale, with millions of users registered, thousands of courses offered, and billions of student-platform interactions [1]. Both the popularity of MOOCs among students [2] and their benefits to those who complete them [3] suggest that MOOCs present a new, easily scalable, and easily accessible opportunity for learning. A major criticism of MOOC platforms, however, is their frequently high attrition rates [4], with only 10% or fewer learners completing many popular MOOC courses [1, 5]. As such, a majority of research on MOOCs in the past 3 years has been geared towards increasing student completion. Researchers have investigated features of individual courses, universities, platforms, and students [2] as possible explanations of why students complete or fail to complete.

A majority of this research, however, has been limited to single courses, often taught by the researchers themselves, which is due in most part to the lack of access to other data. In order to increase access to data and make analysis easier, researchers at UC Berkley developed an open-source repository and analytics tool for MOOC data [6]. Their tool allows for the implementation of several analytic models, facilitating the re-use and replication of an analysis in a new MOOC.

Running analyses on single data sets, however, still limits the generalizability of findings, and leads to inconsistency between published reports [7]. In the context of MOOCs, for example, one study investigated the possibility of predicting course completion based on forum posting behavior in a 3D graphics course [8]. They found that starting threads more frequently than average was predictive of completion. Another study investigating the relationship between forum posting behaviors, confusion, and completion in two courses on Algebra and Microeconomics found the opposite to be true; participants that started threads more frequently were *less* likely to complete [9].

The current limited scope of much of the current research within MOOCs has led to several contradictory findings of this nature, duplicating the "crisis of replication" seen in the social psychology community [10]. The ability to determine which findings generalize across MOOCs, and what contexts findings stabilize, will lead to knowledge that can more effectively drive the design of MOOCs and enhance practical outcomes for learners.

2. MORF: GOALS AND ARCHITECTURE

To address this limitation, we have developed MORF, the **MO**OC **R**eplication **F**ramework, a framework for investigating research questions in MOOCs within data from multiple MOOC data sets. Our goal is to determine which relationships (particularly, previously published findings) hold across different courses and iterations of those courses, and which findings are unique to specific kinds of courses and/or kinds of participants. In our first report of MORF [11], we discussed the MORF architecture and attempted to replicate 21 published findings in the context of a single MOOC.

MORF represents findings as production rules, a simple formalism previously used in work to develop human-understandable computational theory in psychology and education [14]. This approach allows findings to be represented in a fashion that human researchers and practitioners can easily understand, but which can be parametrically adapted to different contexts, where slightly different variations of the same findings may hold.

The production rule system was built using Jess, an expert system programming language [15]. All findings were programmed into if-else production rules following the format, "If a student who is <attribute> does <operator>, then <outcome>." Attributes are pieces of information about a student, such as whether a student reports a certain goal on a pre-course questionnaire. Operators are actions a student does within the MOOC. Outcomes are, in the case

of the current study, whether or not the student in question completed the MOOC (but could represent other outcomes, such as watching more than half of the videos). Not all production rules need to have both attributes and operators. For example, production rules that look at time spent in specific course pages may have only operators (e.g., spending more time in the forums than the average student) and outcomes (i.e., whether or not the participant completed the MOOC).

Each production rule returns two counts: 1) the confidence [16], or the number of participants who fit the rule, i.e., meets both the if and the then statements, and 2) the conviction [17], the production rule's counterfactual, i.e., the number of participants who match the rule's then statement but not the rule's if statement. For example, in the production rule, "If a student posts more frequently to the discussion forum than the average student, then they are more likely to complete the MOOC," the two counts returned are the number of participants that posted more than the average student and completed the MOOC, and the number of participants who posted less than the average, *but still* completed the MOOC. As a result, for each MOOC, a confidence and a conviction for each production rule can be generated.

A chi-square test of independence can then be calculated comparing each confidence to each conviction. The chi-square test can determine whether the two values are significantly different from each other, and in doing so, determine whether the production rule or its counterfactual significantly generalized to the data set. Odds ratio and risk ratio effect sizes per production rule are also calculated. Stouffer's [18] Z-score method can be used in order to combine the results per finding across multiple MOOC data sets, to obtain a single statistical significance.

Currently, 40 MOOC data sets and 21 production rules related to pre-course survey responses, time spent in course pages, forum posting behaviors, forum post linguistic features, and completion are incorporated in the framework.

3. FUTURE WORK

First, we plan to expand the current set of variables being modeled in MORF, both in terms of predictor (independent) variables and outcome (dependent) variables. This will enable us to replicate a broader range of published findings. Our first efforts do not yet include findings involving data from performance on assignments or behavior during video-watching, two essential activities in MOOCs.

Second, we intend to add to MORF a characterization of the features of the MOOCs themselves, towards studying whether some findings fail to replicate in specific MOOCs due to the differences in design, domain, or audience between MOOCs. Understanding how the features of a MOOC itself can explain differences in which results replicate may help us to explain some of the contradictory findings previously reported in single-MOOC research. Doing so will help us to understand which findings apply in which contexts, towards understanding how the different design of different MOOCs drive differences in the factors associated with student success.

4. REFERENCES

[1] Jordan, K. (2014). Initial trends in enrolment and completion of massive open online courses. *The International Review of Research in Open and Distributed Learning*, *15*(1).

[2] Adamopoulos, P. (2013). What makes a great MOOC? An interdisciplinary analysis of student retention in online courses.

[3] Zhenghao, C., Alcorn, B., Christensen, G., Eriksson, N., Koller, D., & Emanuel, E. (2015). Who's Benefiting from MOOCs, and Why. *Harvard Business Review*

[4] Clow, D. (2013). MOOCs and the funnel of participation. In Proceedings of the Third International Conference on Learning Analytics and Knowledge (pp. 185-189). ACM

[5] Yang, D., Sinha, T., Adamson, D., & Rose, C. P. (2013). Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In Proceedings of the 2013 NIPS Data-driven education Workshop (Vol. 11, p. 14)

[6] Pardos, Z. A., & Kao, K. (2015, March). moocRP: An open-source analytics platform. In *Proceedings of the Second (2015) ACM conference on learning@ scale* (pp. 103-110). ACM.

[7] Łukasz, K., Sharma, K., Shirvani Boroujeni, M., & Dillenbourg, P. (2016). On generalizability of MOOC models. In *Proceedings of the 9th International Conference on Educational Data Mining* (No. EPFL-CONF-223613, pp. 406-411).

[8] Andersson, U., Arvemo, T., & Gellerstedt, M. (2016). How well can completion of online courses be predicted using binary logistic regression?. In *IRIS39-The 39th Information Systems Research Conference in Scandinavia, Ljungskile, Sweden, 7-10 August 2016*.

[9] Yang, D., Wen, M., Howley, I., Kraut, R., & Rose, C. (2015). Exploring the effect of confusion in discussion forums of massive open online courses. In Proceedings of the Second (2015) ACM Conference on Learning@ Scale (pp. 121-130). ACM.

[10] Makel, M. C., & Plucker, J. A. (2014). Facts are more important than novelty replication in the education sciences. *Educational Researcher*, 0013189X14545513.

[11] Andres, J.M.L., Baker, R.S., Siemens, G., Gašević, D., & Spann, C.A. (in press). Replicating 21 Findings on Student Success in Online Learning. *Technology, Instruction, Cognition, & Learning*.

[12] Schmidt, F. L., & Hunter, J. E. (2014). *Methods of meta-analysis: Correcting error and bias in research findings*. Sage publications.

[13] Koedinger, K. R., Baker, R. S., Cunningham, K., Skogsholm, A., Leber, B., & Stamper, J. (2010). A data repository for the EDM community: The PSLC DataShop. *Handbook of educational data mining*, *43*.

[14] Anderson, J. R., Matessa, M., & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, *12*(4), 439-462.

[15] Friedman-Hill, E. (2002). Jess, the expert system shell for the java platform.*USA: Distributed Computing Systems*.

[16] Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining Associations between Sets of Items in Massive Databases. In *Proceedings of the ACM-SIGMOD Int'l Conference on Management of Data* (pp. 207-216).

[17] Brin, S., Motwani, R., Ullman, J. D., & Tsur, S. (1997, June). Dynamic itemset counting and implication rules for market basket data. In *ACM SIGMOD Record* (Vol. 26, No. 2, pp. 255-264). ACM.

[18] Stouffer, S.A., Suchman, E.A., DeVinney, L.C., Star, S.A. & Williams, R.M. Jr. (1949). *The American Soldier, Vol. 1: Adjustment during Army Life*. Princeton University Press, Princeton.

Clustering Students in ASSISTments: Exploring System- and School-Level Traits to Advance Personalization

Seth Adjei, Korinn Ostrow, Erik Erickson, Neil Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609
{saadjei, ksostrow, eerickson, nth}@wpi.edu

ABSTRACT
Few attempts have been made to create student models that cluster student and school level traits as a means to design personalized learning interventions. In the present work, data from ASSISTments was enriched with publicly available school level data and K-Means clustering was employed. Results revealed the importance of school locale, measures of district wealth, and system interaction patterns as potential foci for personalization. Clusters were then applied to a test set of held out data and cluster assignments were used to help predict end-of-year standardized mathematics test scores. Findings suggest that while cluster interpretations were not generalizable to held out data, clustering was generally helpful in predicting standardized test scores.

Keywords
K-Means Clustering, Student-System Interactions, School Level Characteristics, Standardized Tests, Ensembled Prediction Model.

1. INTRODUCTION
The focus of research using vast educational data often lends itself to the development of learner models, or various sophisticated predictive models that help to pinpoint when and how learning occurs on a personalized level. Popular approaches include Bayesian Networks (i.e., Bayesian Knowledge Tracing) [3], Performance Factors Analysis [6], and Neural Networks (i.e., Deep Learning) [4]. However, it is valuable to ask if simpler models built to leverage student, school, and district level data can be useful in establishing learner profiles.

The use of clustering to group similar students within various types of online learning environments has typically been a successful endeavor [1, 2, 7, 8]. The present work seeks to balance the complexity of working with high volumes of educational data and building simple predictive learner models through clustering by answering the following research questions:

1. Are there distinct types of learners within ASSISTments [5] that can be identified by clustering student, school, and district level characteristics and measures of student/system interaction?
2. What student types are defined via cluster interpretation? Do interpretations generalize to unseen data?
3. Can clusters help predict significant differences in end-of-year test scores?

2. METHODOLOGY
The present work assessed log files from students in the state of Maine working in ASSISTments [5], an online learning system focused on middle school mathematics, during the 2014-2015 academic year. This data was extended by merging additional school and district level data from the Common Core of Data supported by the NCES and IES (https://nces.ed.gov/ccd/). Students' scores on the standardized, end-of-year TerraNova mathematics test were also included in the dataset.

For each student, the dataset contained averages for the following student/system interaction features: problem count, time spent on problems, percent correct across assignments, hints used per problem, number of problems per assignment for which hints were used, and assignment completion rate. Additionally, each student's data included continuous measures retrieved from the NCES/IES data (i.e., the percentage of students in the school eligible for free or reduced lunch) as well as one-hot encoded forms of categorical features like school locale. The cleaned dataset represented 1,557 unique students from 21 schools, with 171,983 unique student/assignment pairs stemming from 35,127 assignments. Each observation or row represented the overall performance and characteristics of a single student and their school or district. De-identified data is available at tiny.cc/EDM2017Clustering for further reference.

The modeling approach used in the present work was adapted from that in [1]. An initial 70% of the data was randomly selected to form the training set. The training set was used for initial K-Means clustering and cluster interpretation. The K-Means algorithm was sourced from R's statistics package, implementing Euclidean distance as the default distance measure. The remaining 30% of the data was used to form the test set. The test set was used to build models predicting TerraNova scores. First, predictions were made to assign students in the test set to a cluster. Following student assignment, clusters were reinterpreted to verify whether trained interpretations generalized to unseen data. Cluster membership was then used to help predict TerraNova scores alongside student-system interaction features using cluster-specific stepwise linear regressions. These regression models were then ensembled and measures of model accuracy were compared to a traditional approach where K = 1.

3. TRAINING
In order to determine the optimal value for K, 10-fold cross validation was implemented on the training set to build scree plots. To determine the most appropriate value from this set, the mean and median of optimal K values across folds were considered (M = 4.1, Med. = 4). As such, four clusters were forced using K-Means on the training data. The four resulting clusters were characteristic of unique types of students, ultimately labeled as "proficient," "struggling," "learning," and "gaming." Graphics and additional information on cluster characteristics are available at tiny.cc/EDM2017Clustering for further reference.

Table 1. Coefficients, Standard Errors, and Model Statistics per cluster on test set data when K=1 and K=4.

	K = 1		*K = 4*							
	1 (n = 442)		*1 (n=127)*		*2 (n=160)*		*3 (n=124)*		*4 (n=31)*	
IVs	*b*	*SE*	*b*	*SE*	*b*	*SE*	*b*	*SE*	*b*	*SE*
Intercept	631.94***	20.37	712.95***	51.78	504.41***	30.36	567.63***	34.66	*680.14****	*63.13*
Percent Correct	110.66***	22.76	81.95	61.70	268.30***	33.92	131.02***	35.16	18.73	68.74
Ave. Time	-0.08**	0.03	-0.10	0.07	0.01	0.04	0.09	0.06	-0.09	0.09
Completed	0.35	12.13	-63.05	39.89	8.47	15.55	22.10	18.68	-18.80	34.25
Total Hints	1.73	2.69	7.01	6.08	8.00*	3.66	-38.84***	8.02	-52.73*	19.80
Hint Instances	-0.11	3.53	-9.34	11.25	-4.13	4.13	49.75***	9.68	71.09**	24.23
Model Stats										
F (DF)	17.55*** (5, 436)		1.30 (5, 121)		22.87*** (5, 154)		8.18*** (5, 118)		2.00 (5, 25)	
R^2 (Adj. R^2)	0.168 (0.158)		0.051 (0.012)		0.426 (0.408)		0.257 (0.226)		0.286 (0.143)	

4. TESTING & MODEL EVALUATION

Using the remaining 30% of the data that had been held out from the training set, student, school, and district level features (excluding TerraNova test score) were used to predict student assignment to one of the four clusters developed in training. Following student assignment, clusters were interpreted to verify whether initial cluster labels generalized to this unseen data. Cluster characteristics varied for the test set, suggesting that cluster interpretations did not generalize. Graphics and additional information on cluster characteristics are available at tiny.cc/EDM2017Clustering for further reference.

Cluster membership was then used to help predict TerraNova scores alongside student/system interaction features using cluster-specific stepwise linear regressions. Following the ensembling approach used in [7], separate regression models were built for each cluster before being ensembled to form a prediction model. Cluster models helped to depict the relative importance of student/system interaction features in the prediction of TerraNova scores for each value of K, as shown in Table 1. Variability in feature significance was observed across clusters. An alternative prediction model was constructed using the full dataset (essentially, K=1) in order to compare the accuracy of ensembled cluster models to an unclustered baseline. Table 1 presents unstandardized beta coefficients, standard errors, significance values, and overall model statistics across clusters and values of K, and reveals that cluster assignment was sometimes significant in predicting TerraNova scores.

In terms of prediction model accuracy, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) were both lowest when K=4 (23.27 and 30.32, respectively, compared to 25.88 and 33.44 when K=1). Additionally, the difference between MAE and RMSE was lower when K=4 (7.05 compared to 7.56), suggesting that the variance in individual prediction errors decreases as K increases. Variance explained, as measured by R^2, was also higher when K=4, suggesting that the ensembled model was a stronger option than grouping all data together into a single cluster.

5. DISCUSSION

Results of our clustering exploration revealed that there are distinct types of learners within ASSISTments that can be identified by using K-Means to cluster student, school, and district level characteristics and measures of student/system interaction. Results suggested that clusters contained identifiably different patterns of student behavior. However, applying these clusters to a test set revealed that cluster interpretations did not generalize well to held out data. The results of subsequent linear regression models suggested that if clustering could be reliably linked to student features, the approach could potentially be used to help drive personalization within the ASSISTments platform.

Limitations of this work include being bound by the hierarchical nature of the data, assumptions inherent to K-Means analysis, and the potential for artificial inflation of model accuracy due to regression to the mean. As it stands, clustering does not necessarily fail as a method of personalization. Understanding the features that are important to each cluster, as well as the overall accuracy of ensembled cluster models and how such accuracy differs with varying values of K, could help to guide the design of learning interventions specific to particular students. However, the reliability of the approach may be extremely sensitive to the quantity and quality of available data, making clustering a difficult approach for personalized learning.

6. ACKNOWLEDGMENTS

Thanks to NSF (1440753, 1252297, 1109483, 1316736 & 1031398), U.S. D.O.E. (IES R305A120125 & R305C100024 and GAANN), ONR, and the Gates Foundation.

7. REFERENCES

[1] Amershi, S. & Conati, C. 2007. Unsupervised and supervised machine learning in user modeling for intelligent learning environments. Proc 12th Int Conf on Int UI. ACM, 72-81.

[2] Bouchet, F., Harley, J.M., Trevors, G.J., & Azevedo, R. 2013. Clustering and Profiling Students According to their Interactions with an Intelligent Tutoring System Fostering Self-Regulated Learning. *JEDM*. 5(1): 104-146.

[3] Corbett, A.T. & Anderson, J.R. 1995. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*. 4: 253-278.

[4] Deng, L. & Yu, D. 2014. Deep Learning: Methods and Applications. *Found and Trends in Sig Proc.* 7(3-4): 1–199.

[5] 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. *Int. J AIED*. 24(4): 470-497.

[6] Pavlik, P.I., Cen, H., & Koedinger, K.R. 2009. Performance Factors Analysis: A New Alternative to Knowledge Tracing. AIED. 531-538.

[7] Trivedi S., Pardos Z.A., & Heffernan N.T. 2011. Clustering Students to Generate an Ensemble to Improve Standard Test Score Predictions. Proc 15th Int Conf on AIED. 377-384.

[8] Zakrzewska, D. 2008. Using Clustering Technique for Students' Grouping in Intelligent E-Learning Systems. In A. Holzinger (Ed.): USAB 2008, LNCS 5298, 403–410.

Application of the Dynamic Time Warping Distance for the Student Drop-out Prediction on Time Series Data

Alexander Askinadze
Institute of Computer Science
Heinrich Heine University Düsseldorf
askinadze@cs.uni-duesseldorf.de

Stefan Conrad
Institute of Computer Science
Heinrich Heine University Düsseldorf
conrad@cs.uni-duesseldorf.de

ABSTRACT

It is reported by different universities that over 40% of students do not complete their studies within 6 years. Especially in technical courses, the drop-out rate is already very high at the beginning. Therefore an automatic drop-out prediction is useful for a monitoring system. Since the study progress data can be sorted by time, we show how they can be transformed into a multivariate time series. Then we examine the dynamic time warping (DTW) distance in conjunction with the k-nn classifier and show how DTW can be used as an SVM kernel for drop-out prediction on the time-series data. With this approach, we are able to recognize about 67% of the drop outs from the course of study after the first semester and about 60% after the second semester.

1. INTRODUCTION

The number of drop out is a big problem for many universities. Over 40% of students do not complete their studies within 6 years [1]. Especially in technical courses, the number of drop outs in the first semesters is high. So in [5] it is reported that in the Electrical Engineering course the drop-out rate of beginners is about 40%. Human monitoring is used to solve this problem [5]. With a large number of students, this can lead to a huge manual effort, so that a machine-made pre-selection could facilitate the work of a human decision-maker. Most students fail in the first semesters, which requires an early prediction. The quality of the available data is very important for automatic drop-out prediction. However, due to data protection laws, often little data are available for use. The data is often restricted to only a small amount of private data and the study progress data, so that only examinations, their corresponding grades, and the number of attempts per semester are given. Because of the dearth of data, it is important to obtain as much semantics as possible from the data, such as temporal aspects. The study progress data can be viewed as a multivariate time series. In this paper, we will investigate methods that can perform drop-out predictions on time-series data.

2. RELATED WORK

Many studies have been published on student drop-out prediction like [1], [5], [6]. The data mining methods used include SVM, decision trees, k-nn, and neural networks. Studies were also made in the field of time series analysis. In [6] the authors investigated time series clustering to identify at-risk online students. Several studies, for example [7], have used DTW for time series clustering to identify distinct activity patterns among students. The results of the individual publications are difficult to compare with each other because the data used and the goals are very different. While some seek to prevent drop outs from a study subject, others seek to prevent drop outs from the whole study. We also use DTW, but not for clustering, but as distance for classifiers.

3. METHOD

If only the data of the study progress are available per semester, as much semantic information as possible must be collected from the data. Assuming that the study progress of a student S consists of n semesters $s = \{sem_1, ..., sem_n\}$, a function $\Phi : s \to T^S_{n,m}$ with $\Phi(s) = \Phi(\{sem_1, ..., sem_n\}) = \{\phi(sem_1)^\top, ..., \phi(sem_n)^\top\} = \{s_{1 \le k \le n} = [s_{1,k}, ..., s_{m,k}] \in \mathbb{R}^m\} = T^S_{n,m}$ which transforms the ordered set s into a multivariate time series is needed.

In each semester the students have the possibility to take q courses. The results of each course can be expressed by a number p of properties such as the final score or the number of trials. All information of a semester can thus be represented in a vector of size $m = q \times p$. If, for example, the 3 properties were: 1) achieved grade (numeric), 2) passed (binary), and 3) number of attempts (numeric), and in a certain semester, a student had taken the first and last course from the list of all possible courses then the resulting vector for a semester could look like the one shown below.

$$\phi(sem_i) = [\underbrace{5\ \ 1\ \ 1}_{course\ 1}\ \ \underbrace{0\ \ 0\ \ 0}_{...}\ \ \underbrace{1\ \ 0\ \ 2}_{course\ q}]$$

Thus, we can represent a student as a temporal sequence of his completed semesters. To compare these two sequences, we need a distance for multivariate time series. A well-researched distance for time series is the d_{DTW} distance.

Dynamic Time Warping (DTW) [3] is an algorithm from the domain of time series. It is generally defined for univariate time series and can be used to calculate a distance of the two time series $a = (a_1, ..., a_n), a_i \in \mathbb{R}$ and $b = (b_1, ..., b_m), b_j \in \mathbb{R}$ with different length. To extend the DTW distance for multivariate time series, various methods have been proposed in the literature like DTW_D [8]. DTW_D is calculated just as in the one-dimensional case, except that the pairwise distance $d(a_i, b_j)$ is calculated with the Euclidean distance.

The drop-out prediction is a binary problem. One of the most popular binary classifiers is the *support vector machine* (SVM) [4] because it can separate linear separable sets optimally from each other. If the training dataset is not linearly

separable, a kernel trick is used to solve the problem. An often used kernel is the Gaussian kernel. In [2], an adaptation of the Gaussian kernel to the Gaussian DTW (GDTW) kernel was made for sequential data. The GDTW kernel K_{GDTW} can be defined by $K_{GDTW}(x, y) = e^{-\gamma d_{DTW}(x,y)}$.

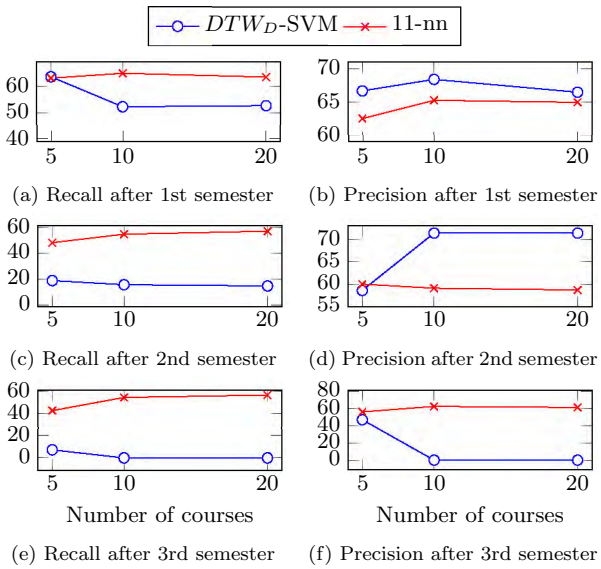
4. EVALUATION

We have a data set with 704 students of which 310 did not successfully complete their studies within 10 semesters. For each student the following information per semester is available: idCourse, number of attempts, examination status (passed, failed), recognized exam (true, false), reached grade, and semester. We use recall and precision as evaluation measures. The evaluation is performed 3 times for all parameters with a 10-fold cross-validation for the two approaches DTW_D-SVM and DTW_D-k-nn (ordinary k-nn classifier that uses the DTW_D distance). It is examined per semester how good the prediction is at the end of the semester. For example, the students who have studied at least 2 semesters are considered for the training and prediction of the drop out after the second semester. The length of the resulting multivariate time series vectors depends strongly on the number of courses used. Therefore, we will examine the influence of the number of courses used to create the vectors. In the dataset there are more than 100 courses. Because most students of our dataset drop out after a few examinations, we sort all courses according to the number of students who have enrolled in them. Then the 5, 10 and 20 courses with the highest enrollment will be used for further study. After the first investigation, we have found that the k-nn parameter $k = 11$ is comparatively well suited and is therefore used for the evaluation.

We first consider the prediction after the first semester. The recall and precision results are shown in Figure 1. In the second semester, 609 students are still active, of whom 215 will be leaving in the future. After the last examination of the second semester, almost 60% of these students can be recognized with 11-nn. The precision of 11-nn is also about 60%. DTW_D-SVM achieves a 10% higher precision when using more than 10 courses to create the multivariate time series vectors. However, the recall value of DTW_D-SVM is significantly smaller. At the end of the 3rd semester, the limits of DTW_D-SVM are recognizable. Both the recall and the precision values are smaller for 5 courses, and decrease to 0 for more used courses. 11-nn remains stable and provides similar results as after the second semester. In the third semester, 542 students are still active, of whom 143 will be leaving. 11-nn can recognize about 84 of these 143 students.

5. CONCLUSION

We have shown how a study progress can be transformed into a multivariate time series. Then we demonstrated that the DTW_D distance can be used within an SVM kernel to make an SVM usable for student time series data. We compared the DTW_D-SVM with the 11-nn classifier, which also uses the DTW_D distance on a dataset with 704 students and found that the k-nn classifier is better suited to achieve higher recall values in the drop-out prediction. The DTW_D-SVM is only suitable until the second semester and provides better precision results. In the later semesters, the values become worse due to most of the students in the first semester fail because of a few specific courses. For the students from the technical courses, it is usually the first mathematics courses. In the later semesters, the reasons cannot be stated so easily. Generally this approach is only for the prediction and not to determine the reasons. In future work we want additionally determine the reasons for drop outs.



(a) Recall after 1st semester (b) Precision after 1st semester
(c) Recall after 2nd semester (d) Precision after 2nd semester
(e) Recall after 3rd semester (f) Precision after 3rd semester

Figure 1: Recall and Precision results

6. REFERENCES

[1] L. Aulck, N. Velagapudi, J. Blumenstock, and J. West. Predicting student dropout in higher education. *arXiv preprint arXiv:1606.06364*, 2016.

[2] C. Bahlmann, B. Haasdonk, and H. Burkhardt. Online handwriting recognition with support vector machines-a kernel approach. In *Frontiers in handwriting recognition, 2002. proceedings. eighth international workshop on*, pages 49–54. IEEE, 2002.

[3] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[4] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[5] G. Dekker, M. Pechenizkiy, and J. Vleeshouwers. Predicting students drop out: A case study. In *Educational Data Mining 2009*, 2009.

[6] J.-L. Hung, M. Wang, S. Wang, M. Abdelrasoul, W. He, et al. Identifying at-risk students for early interventions? a time-series clustering approach. *IEEE Transactions on Emerging Topics in Computing*, 2015.

[7] E. Młynarska, D. Greene, and P. Cunningham. Time series clustering of moodle activity data. In *24th Irish Conference on Artificial Intelligence and Cognitive Science (AICS'16), University College Dublin, Dublin, Ireland, 20-21 September 2016*, 2016.

[8] M. Shokoohi-Yekta, B. Hu, H. Jin, J. Wang, and E. Keogh. Generalizing dtw to the multi-dimensional case requires an adaptive approach. *Data Mining and Knowledge Discovery*, 31(1):1–31, 2017.

Student Use of Scaffolded Inquiry Simulations in Middle School Science

Elizabeth McBride
University of California
Berkeley
bethmcbride@berkeley.edu

Jonathan Vitale
University of California
Berkeley
jonvitale@berkeley.edu

Marcia Linn
University of California
Berkeley
mclinn@berkeley.edu

ABSTRACT
Interactive simulations can help students make sense of complex phenomena in which multiple variables are at play. To succeed, these simulations benefit from scaffolds that guide students to keep track of their investigations and reach meaningful insights. In this research, we designed an interactive simulation of a solar oven design and explored how students utilized the simulation during learning and how scaffolds functioned to alter the learning experience. We used a table for recording trials and guiding questions to scaffold students' interactions with the simulation. We employed data mining techniques to analyze student interactions for use of the control of variables strategy and other approaches. We found that the control of variables strategy may not be as beneficial for learning as an exploratory strategy.

Keywords
Interactive Simulations, Science Education, Inquiry, Log Data

1. INTRODUCTION
Simulations can be powerful tools for allowing students to engage in inquiry, especially in science disciplines. To succeed, these simulations generally benefit from scaffolds that guide students to keep track of their investigations and reach meaningful insights [6]. In this study, we examine guiding questions and recording of trials in a table as scaffolds. We use a simulation of a solar oven that allows students to investigate the multiple variables at play in energy transformation and gives representation to invisible phenomena.

We used the knowledge integration framework to create the curriculum about solar ovens, because the framework focuses on building coherent understanding [4]. This framework offers instructional design principles to enhance connections between design decisions and scientific principles. The knowledge integration framework has proven useful for design of instruction featuring dynamic visualizations [8] and engineering design [1, 6].

Various scaffolding methods are often used with interactive simulations. Often, these scaffolds are implicit, or built into the system with the simulation [7]. For example, guiding questions are used with inquiry simulations to direct students' attention toward certain features of simulations [2]. Other tools, like concept maps and note-taking spaces can also assist students in making sense of inquiry simulations [3].

Using log files from student interactions with the curriculum and output from the automatically generated tables (simulation scaffolding), we use feature engineering to identify how students use the model and whether these uses have an impact on learning.

2. CURRICULUM
This research focuses on a curriculum about solar ovens that is run using the Web-based Inquiry Science Environment (WISE). During this curriculum, students design, build, and test a solar oven. Students use an interactive computer simulation to test the different materials in their oven during the design process.

This curriculum takes between 10-15 hours, and students complete the project in groups of 2 or 3. Students also complete individual pretests and posttests.

2.1 Interactive Computer Simulation
The scaffolds we developed for the interactive simulation are twofold; short response style questions direct students to investigate capabilities and limitations of the simulation and an automatically generated table helps students to keep track of trials they have run. The table includes information about all of the settings used in that trial, as well as the results of the trial at certain time points.

3. DATA
This data comes from 635 students across three schools and five teachers. These students formed 255 teams. After dropping students who did not complete significant portions of the curriculum, there were 558 students and 246 groups or partial groups remaining.

4. DESCRIPTIVE STATISTICS
Of the 246 groups who participated in the curriculum, 216 (87.80%) of the students used the computer model to pro-

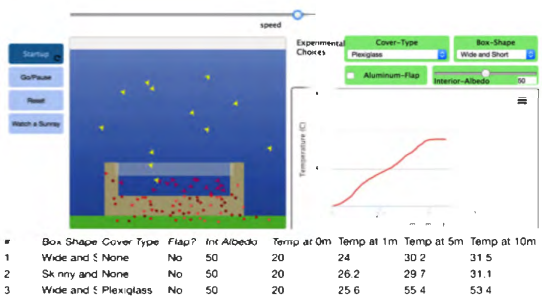


Figure 1: The interactive simulation used by students to test solar ovens and visualize energy transformation; below the table simulation is output from the automatically generated table

duce at least one row of data during the first design iteration. We consider each row of data produced to be a trial. As seen in figure 2, many groups do not use the simulation scaffolds at all and produce zero rows in the automatically generated table. Still more students produce only 1 row in the table, which may mean they are confirming their ideas for a solar oven that they have already discussed and planned prior to using the simulation and without any evidence outside of their intuitions.

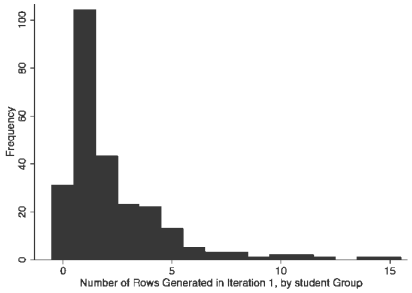


Figure 2: Histogram depicting the frequency of the number of trials run by a group of students during the first iteration of using the simulation (Mean: 2.27)

5. CONTROLLING VARIABLES

We define a control of variables strategy as changing a single variable at a time. We use feature engineering to develop a variable, *COV Trials*, that represents the number of trials a student ran using the control of variables strategy. Overall, 137 (55.69%) of the 246 groups employed a control of variables strategy. There were 216 groups that used the table scaffolds to generate at least one row of data. Of the groups that generated at least two rows in the table (115), 103 of them (89.56%) employed a control of variables strategy.

6. EFFECT ON LEARNING

Using pretest and posttest scores we aimed to understand the effect of actions with the simulation on learning. We found that the number of rows generated during the simulation was a significant predictor of learning (b = 0.10, t(546) = 2.68, $p < 0.01$). However, simply employing a control of variables strategy was not a significant predictor of learning. There were also two short response scaffolding questions. We generated a variable based on the number of questions students answered (0, 1, or 2). This was predictive of learning (b = 0.10, t(546) = 2.56, $p = 0.011$).

Overall, evidence suggests that students should be encouraged to experiment with the model and guided to produce at least two rows of data in the table to improve learning outcomes and use the short response questions. Perhaps changing more than one variable at a time in this type of environment indicates that students are spending more time thinking about possible outcomes.

7. LIMITATIONS

While we have found simulations to be beneficial for student learning in previous work [5], it is important to note that not all student learning is due to interactions with the simulation. While there is likely some difference between students who generated one row versus those who generated two or more rows, it is difficult to understand the differences between using a control of variables strategy and generating multiple rows of data in the table.

8. REFERENCES

[1] J. Chiu, P. Malcolm, D. Hecht, C. DeJaegher, E. Pan, M. Bradley, and M. Burghardt. Wisengineering: Supporting precollege engineering design and mathematical understanding. *Computers & Education*, 67:142–155, 2013.

[2] C. Hmelo and R. Day. Contextualized questioning to scaffold learning from simulations. *Computers & Education*, 32(2):151–164, 1999.

[3] Y. Kali and M. Linn. Technology-enhanced support strategies for inquiry learning. *Handbook of research on educational communications and technology*, pages 145–161, 2008.

[4] M. Linn and B. Eylon. *Science learning and instruction: Taking advantage of technology to promote knowledge integration.* Routledge, 2011.

[5] E. McBride, J. Vitale, L. Applebaum, and M. Linn. Use of interactive computer models to promote integration of science concepts through the engineering design process. In *Proceedings of the 12th International Conference of the Learning Sciences*, Singapore, Singapore, June 2016 2016.

[6] K. McElhaney and M. Linn. Investigations of a complex, realistic task: Intentional, unsystematic, and exhaustive experimenters. *Journal of Research in Science Teaching*, 48(7):745–770, 2011.

[7] N. Podolefsky, E. Moore, and K. Perkins. Implicit scaffolding in interactive simulations: Design strategies to support multiple educational goals. *Chemistry Education Research and Practice*, 14(3):257–268, 2013.

[8] K. Ryoo and M. Linn. Can dynamic visualizations improve middle school students' understanding of energy in photosynthesis? *Journal of Research in Science Teaching*, 49(2):218–243, 2012.

Modeling Dormitory Occupancy Using Markov Chains

David D. Pokrajac
Delaware State University
1200 N DuPont Hwy
Dover, DE 19901
+1-302-857-7614
dpokrajac@desu.edu

Kimberley Sudler
Delaware State University
1200 N DuPont Hwy
Dover, DE 19901
+1-302-857-7036
krsudler@desu.edu

Diana Yankovich
Delaware State University
1200 N DuPont Hwy
Dover, DE 19901
+1-302-857-6308
dyankovich@desu.edu

Teresa Hardee
Delaware State University
1200 N DuPont Hwy
Dover, DE 19901
+1-302-857-7837
thardee@desu.edu

ABSTRACT
We introduce a Markov chain based model that quantifies university dormitory occupancy as a function of parameters related to university housing policies, students' success and academic progress, and customer satisfaction/dorm availability. The model provides sensitivity of university housing occupancy on change of the parameters. We demonstrated functionality of the model on several case scenarios from a public university.

Keywords
Modeling, dormitory occupancy, university housing, Markov chains, sensitivity, students' success, Banner.

1. INTRODUCTION
In this study, we introduce a housing occupancy model based on Markov chains [e.g., 1]. The model determines relationship between the number of students in dormitories, number of students in incoming class and probabilities quantifying students' retention, advancement between ranks (freshmen, sophomores, etc.), customer satisfaction and availability of housing. The model provides an opportunity for what-if analysis and assessment of change in housing occupancy due to variation of model parameters. The values of model parameters are learned from a transactional database.

We provide a case study based on three years data from Delaware State University, a public comprehensive historically black college/university in Delaware and demonstrate quantitative change of housing occupancy as results of possible changes in housing policy, housing demand and retention. The proposed technique is applicable to universities offering predominantly undergraduate programs and can be easily adapted for universities with substantial graduate programs and participation of international students.

2. METHODOLOGY
2.1 Problem
We consider a university offering undergraduate programs. The students at the university may be of in-state or out-of-state domicile (in-state students are the students whose residence is in the same state as the university). During the course of study, out-of-state students may convert to in-state or vice versa. A new student at the university can be enrolled as a new freshman (NF) or a new transfer (NT). For a student retained at the university, a rank depends on the cumulative number of credits (earned at the university + transferred). The ranks satisfy partial order. Thus, a NF or NT, if retained, may continue as returning freshmen (RF), sophomore (SO), junior (JR) or senior (SR). Retained RF may continue as RF or progress into SO, JR or SR. Retained SO may continue as SO, or progress as JR or SR. Each student in a particular year can be a dorm resident. If retained, a student may change dorm residency status, i.e., a dorm non-resident may become dorm resident or vice versa.

Our goal is to determine the relationship between various parameters characterizing students' population and academic progress and the total number of dorm residents in a particular year.

2.2 Markov Chain Model
We model the considered problem with a time-homogeneous Markov chain [1]. A student at the university can be described by a state $s_{(i,j,k)}$ determined by an ordered triple of indices i, j, and k indicating domicile, rank and dorm residence: $i \in \{InState, OutOfState\}$, $j \in \{NF, NT, RF, SO, JR, SR\}$ and $k \in \{DormResident, NotDormResident\}$. The starting states correspond to $i \in \{InState, OutOfState\}$, $j \in \{NF, NT\}$, $k \in \{DormResident, NotDormResident\}$. The total number of non-absorbing states is 24. In addition, a student can graduate or leave the university, corresponding to an absorbing state, denoted with s_a. The transition between states $s_{(i,j,k)}$ and $s_{(i',j',k')}$ is uniquely determined by transition probability that, under the assumption of time homogeneity is denoted by $p_{(i,j,k),(i',j',k')}$. In addition, the model includes transition probabilities $p_{(i,j,k),a}$ from states $s_{(i,j,k)}$ to the absorbing state.

2.3 Model Implementation

To operationalize the model, we introduce the following assumptions and simplifications:

1) Students can transition only from out-of-state to in-state status;

2) For in-state students who continue to stay in dorms, the transition probability can be expressed as product of probabilities that a student is retained, that a student advanced from rank *j* to *j'* and the probability that a student stayed in dorm;

3) For out-of-state students who continue to stay in dorms as out-of-state, the transition probability is expressed as a product of probabilities that a student is retained, that a student does not change out-of-state status, that a student advanced from rank *j* to *j'* and the probability that a student stayed in dorm;

4) For out-of-state students who continue to stay in dorms as in-state, the transition probability is expressed as a product of probabilities that a student is retained, that a student changes out-of-state status to instate, that a student advanced from rank *j* to *j'* and the probability that a student stayed in dorm;

5) We compute probabilities that a dorm resident with domicile *i'* and rank *j'* was a dorm resident in the previous year.

2.4 Model Sensitivity

After the parameter values are estimated, the sensitivity s_l of the number of students in dorms on a particular parameter π_l can be determined as: $s(\pi_l) = \frac{\Delta N^y}{\Delta \pi_l}$, where ΔN^y is change of number of students in dorms, due to change $\Delta\pi_l = \pi_l^{new} - \pi_l$ of a parameter. Subsequently, the influence of change of particular model parameters on the model output—the number of students in dorms can be linearized such that: $\Delta N^y = \sum_l s(\pi_l)\Delta\pi_l$.

3. RESULTS

3.1 Data Set

We estimated the model discussed in Section 2 on data from Delaware State University (DSU), a historically black college/university (HBCU) located in Dover, DE, USA. DSU utilizes Banner® Version 8 (Ellucian, Fairwax, VA, USA) as a higher education enterprise resource planning (ERP) system. The dataset contained the total of 13,709 records from years 2013/14—2015/16. Each record had the values of attributes: StudentID, Year, Rank, DormResidence, Domicile. StudentID is a unique identifier of a student and together with Year comprise the primary key of the extracted table.

3.2 What-if Analyses

In this section we analyze realistic cases for changes of some of the model parameters and their influence on the change of number of students in dormitories.

Case 1. Due to policy change, *all* new freshmen and new transfers are expected to stay at university housing *regardless* whether they are in-state or out-of-state. We can easily obtain the increase of the number of students in dormitories of ΔN^y=467.

Case 2. Due to implementation of initiatives to address needs of incoming and returning freshmen, the retentions of in-dorm new and returning freshmen increase to 80%. This leads to the increase of ΔN^y=175 students in dorms.

Case 3. Owing to improvement of dorm facilities, the demand for dorm housing for upper rank students increases. This can, thus, be considered as a result of increased customer satisfaction. As a consequence, this leads to the increase of ΔN^y=83.

4. DISCUSSION

The proposed model makes it possible to account for retention that is frequently a key performance indicator related to university strategic plans and one of common quantitative measures of students' success. Further, the model involves parameters related to academic progress of students. Also, we can indirectly model housing satisfaction and availability. The model makes it possible to consider in-state and out-of-state students separately, as the two groups of students that may have different demography, socio-economical conditions and academic success. Also, it is possible to evaluate the relationship between the size of the incoming class (new freshmen and transfers) and the housing occupancy.

The model considers only two categories of students: in-state and out-of-state students. For universities with substantial numbers of international students, they can be added as an additional category and treated similarly as out-of-state students. The model assumes that in-state students cannot become out-of-state. However, the assumption can be relaxed by introducing a non-zero probability that in-state students of rank *j* become out-of-state.The assumptions 2—4 (probability independencies) may be contingent on university policies (distribution of students within dorms and on-campus housing allocation across student classes/ranks). Hence, they should be validated prior to the application of the proposed models at another institution of higher education. The current model assumes that the students who leave the university without graduating do not come on a later date. In reality, some students may leave the university temporarily and return ("stop-outs"). Note that we utilized point estimates, hence the accuracy of parameter estimates (e.g., standard deviation) has not been addressed. Future work will include the development of interval estimates for model parameters as well as an application of validation techniques (e.g., leave-one-out cross-validation) to more strictly justify predictive ability of the model.

5. CONCLUSION

We proposed a Markov chain-based model of university housing occupancy and demonstrated it in a case study of a public university. We have shown that the proposed model can be useful in quantifying what-if scenarios related to changes in housing policy, retention and customer satisfaction. The model is developed for a university offering primarily undergraduate programs. It can be extended to graduate program offering institutions, with a challenge that graduate (especially PhD) programs are typically less structured (as evidenced in lack of ranks corresponding to sophomores, juniors, seniors in undergraduate programs). We demonstrated the use of a model with parameters estimated from data readily available on an industry-standard ERP system (Banner). As such, the model can be easily deployed at an institution of higher education that utilizes this or similar technology.

6. ACKNOWLEDGMENTS

This work has been supported through a grant from the Bill and Melinda Gates foundation.

7. REFERENCE

[1] Grinstead, C.M. 1997. *Introduction to Probability*, 2nd edn. American Mathematical Society, Providence, RI.

Improving Models of Peer Grading in SPOC *

Yong Han, Wenjun Wu, Xuan Zhou
State Key Laboratory of Software Development Environment,
School of Computer Science, Beihang University, China
{hanyong, wwj, zhouxuan}@nlsde.buaa.edu.cn

ABSTRACT

Peer-grading is commonly used to allow students to work as graders to evaluate their peer's open-ended assignments in MOOC courses. As a variant of MOOCs, SPOC (Small Private online course) adopt the peer-grading method to grade a number of student submissions. We propose a new ability-aware peer-grading model for SPOC courses by introducing prior knowledge level of each student grader as their grading ability in the process of calculating grading score.

1. INTRODUCTION

Small Private online course (SPOC) is a version of MOOCs used locally with on-campus students. It often has the relatively smaller number of students than a MOOCs course. SPOC students may come from the same classroom and know each other. Previous research efforts on peer-grading suggest that there is great disparity between the observed scores presented by student graders and the true scores (the instructor-given scores). Therefore, it is a major challenge on how to correctly aggregate peer assessment results to generate a fair score for every homework submission.

To solve the problem, we propose a group of new peer-grading models by considering the student mastery of knowledge level as a major factor for estimating final scores. Throughout the paper, we call the mastery of knowledge level as the students' grading ability. Based on every student's learning behavior and quiz-answering outcomes, we design a two-stage individualized knowledge tracing model to accurately assess their grading ability. Moreover, we introduce the new peer-grading models by integrating every student's grading ability into the factor of reliability. Experimental results in our SPOC course verify the effectiveness of our new models.

2. RELATED WORK

Many research efforts have been made to investigate the factors that can affect the grader bias and reliability.

*The accompanying appendix at: http://admire.nlsde.buaa.edu.cn/paper/2017-3.pdf

Goldin et al. [1] used the Bayesian models for peer grading in the setting of traditional classrooms. They explored the major factors including grader bias, and the rubric biases in their models. Walsh introduced a new algorithm named by PeerRank[4] based on the assumptions that the ability of student graders can be measured by the grades they received in the process of peer grading. Our models are inspired from the previous research work done in [3, 2]. We introduce the grading ability of students in their models and develop an individualized knowledge tracing model to estimate such ability.

3. DATASETS

The data sets in our experiments were collected in a SPOC course named by "The Experiment of Computer Network" that is hosted on our MOOC platform. The course is designed to teach both 4th grade CS undergraduate and the first-year graduate students about basic knowledge and skills on designing networking plans and configuring networking devices at the multiple levels of link protocol, TCP/IP protocol and network applications.

The course comprises of 10 chapters, each of which has 8-14 problems as homework assignment for students. The course also includes two open-ended assignments in graduate courses and three open-ended assignments in undergraduate courses. Preliminary statistical analysis of the dataset reveals that most peer-graded score tend to be higher than instructor-given scores for the same submissions.

4. PROBABILISTIC MODELS OF PEER GRADING IN SPOC

In this paper, we first establish a two-stage model to assess student mastery level of each knowledge skill, which can be used for estimating the graders' reliability. And then, we present three probabilistic graph models for peer grading by extending the models PG4 and PG5 of [3].

4.1 Individualized Knowledge-Tracing model for Ability Estimation

At the first stage, we extract interpretive quantities to predict the probability that a student has mastered the knowledge of that certain chapter in which the logistic regression method is used to fit these features and predict the engagement level of every student[5]. At the second stage, our work adopts the knowledge tracing model and ameliorates it by combining the prediction results obtained in the first stage. The sequence of the exercises in each unit is modeled by H-

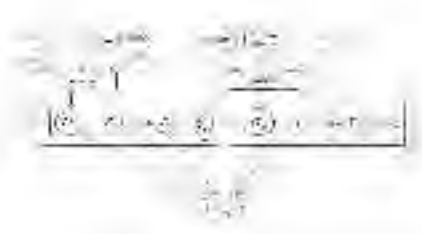


Figure 1: The relationship of the factors used in our models.

MM named as PPS (the Prior Per Student Model). We refer to the results that the HMM generated as a_v, which denotes the ability of graders prior to the peer-grading tasks. We train the model of HMM by using a_v as the initial element of the sequence and then introduce it and the true score as the parameters to model the reliability of a grader by a distribution of Gamma or Gaussian.
Our Experiments show that our estimated ability has relevance with the true score and can be used to estimate the grader reliability. Thus it is reasonable to use grader ability to estimate the reliability.

4.2 Peer-Grading models

We represent a_v as the prior distribution of estimating every grader's mastery of preparatory knowledge, τ_v as the reliability of the student grader v, b_v as the bias of the student grader v, s_u as the true score of a submission, and z_u^v as observed score for the submission.
Model PG6

$$\tau_v \sim \mathcal{G}(a_v, \beta_v)$$
$$b_v \sim \mathcal{N}(0, 1/\eta)$$
$$s_u \sim \mathcal{N}(\mu_0, 1/\gamma_0)$$
$$z_u^v \sim \mathcal{N}(s_u + b_v, 1/\tau_v)$$

We refer to our first model as PG6: the reliability variable τ_v follows the Gamma distribution with a_v as the shape parameter instead of the true score in PG4 in [2] and utilize the student's performance on multiple-choice exercises to estimate his reliability in the process of peer-grading tasks.
Based on Model PG6, we introduce the Model PG7 by remodeling the reliability variable τ_v ($\tau_v \sim \mathcal{N}(a_v, \beta_v)$) with the Gaussian distribution instead of the Gamma distribution. The mean value of the Gaussian distribution in PG7 is still a_v. We also make further extension on Model PG7 by adding the true score s_v with the a_v to calculate the mean of the reliability variable τ_v ($\tau_v \sim \mathcal{N}(\theta_1 a_v + \theta_2 s_v, 1/\beta_v)$) and introduce the parameter λ to re-model the observed variable z_u^v ($z_u^v \sim \mathcal{N}(s_u + b_v, \lambda/\tau_v)$). This extended model is named as Model PG8.
In the above three models (PG6-PG8), we assume the overall bias random variable b_v follows the Gaussian distribution with the mean value at zero. The true score s_u follows the Gaussian distribution with the mean value at μ_0. Moreover, the hyper-parameters β_0, η_0, μ_0, γ_0, θ_1, θ_2, λ are the priors. For the observed scores z_u^v in the PG8, the parameter λ is similar to β_0 in PG6 and PG7, whose function is to scale the variance of its Gaussian.

4.3 Inference and evaluation

The details of the model inference procedures for PG6, PG7 and PG8 are described in the appendix. Our experiments are all based on Gibbs sampling. At the beginning of the Gibbs sampling process, the values of these parameters β_0, η_0, μ_0, γ_0 and λ are initialized to empirical values. We run our experiments by running for 400 iterations with the first 50 burn-in samples eliminated.

5. EXPERIMENTAL RESULTS

We compare our models PG6-PG8 with the baseline model based on simple median value, the models of PG1-PG3 proposed in [3], and the models of PG4-PG5 defined in [2]. The evaluation metric is the root-mean-score-error (RMSE), which is computed as the deviation between the estimated score and the true score assigned by the course staff.
Compared to PG1-3 and PG4-5, our models PG6 and PG7 demonstrate the same level of RMSE in most cases. The model PG8 has more obvious improvement than PG6-7, achieving the lowest RMSE. Therefore, it confirms that PG8 demonstrates the best performance among all the models on average. By combining the grader ability and the true score, the model PG8 is the best approach among all the models for estimating the peer-grading scores in SPOC courses.

6. CONCLUSIONS

In this paper, we first introduce a two-stage individualized knowledge tracing model to estimate each grader's level of knowledge mastery as their grading ability. And then, we propose three new probability graph models by introducing the grading ability as the major parameter for the latent variable of grader reliability. The experiments based on the dataset of our SPOC course demonstrate that our models can be effectively applied to aggregate the peer grades in SPOC courses.

7. ACKNOWLEDGMENTS

This work was supported by grant from State Key Laboratory of Software Development Environment of Beihang university of China (Funding No. SKLSDE-2015ZX-03) and NSFC (Grant No. 61532004).

8. REFERENCES

[1] Ilya M Goldin. Accounting for peer reviewer bias with bayesian models. In *Proceedings of the Workshop on Intelligent Support for Learning Groups at the 11th International Conference on Intelligent Tutoring Systems*, 2012.
[2] Fei Mi and Dit-Yan Yeung. Probabilistic graphical models for boosting cardinal and ordinal peer grading in moocs. In *AAAI*, pages 454–460, 2015.
[3] Chris Piech, Jonathan Huang, Zhenghao Chen, Chuong Do, Andrew Ng, and Daphne Koller. Tuned models of peer assessment in moocs. *arXiv preprint arXiv:1307.2579*, 2013.
[4] Toby Walsh. The peerrank method for peer assessment. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence*, pages 909–914. IOS Press, 2014.
[5] Hsiang-Fu Yu, Hung-Yi Lo, Hsun-Ping Hsieh, Jing-Kai Lou, Todd G McKenzie, Jung-Wei Chou, Po-Han Chung, Chia-Hua Ho, Chun-Fu Chang, Yin-Hsuan Wei, et al. Feature engineering and classifier ensemble for kdd cup 2010. In *KDD Cup*, 2010.

Personalized Feedback for Open-Response Mathematical Questions using Long Short-Term Memory Networks

Joshua J. Michalenko
Rice University
jjm7@rice.edu

Andrew S. Lan
Princeton University
andrew.lan@princeton.edu

Richard G. Baraniuk
Rice University
richb@rice.edu

ABSTRACT

In this paper, we explore the problem of automatic grading and feedback generation for open-response mathematical questions. We resort to the long short-term memory (LSTM) network to learn the simple task of polynomial factorization and use the trained network for grading and feedback. We use Wolfram Alpha to synthetically generate a training dataset that consists of step-by-step responses to polynomial factorization questions to train the LSTM network. Preliminary results validate the efficacy of LSTMs in learning to factor low-order polynomials; we also demonstrate how to leverage the trained network for automatic grading and personalized feedback generation.

Keywords

Automatic grading, Feedback generation, Long short-term memory networks, Mathematical expressions

1. INTRODUCTION

In spite of tremendous advances in technology for education, learning today largely remains a "one-size-fits-all" approach. Personalized learning is the manifestation of *differentiation*, the idea that all students access content and develop mastery differently. The personalized learning experience necessitates a scalable approach since the number of students is much larger than the number of teachers. Many recent advances focus on using machine learning algorithms to analyze student data, but mostly resort to limited utility multiple-choice questions for grading a feedback [5].

The mathematical language processing (MLP) framework proposed in [4] is the first automatic grading and feedback generation tool for open-response mathematical questions. MLP is capable of automatically grading a large number of student responses requiring minimal human effort, but lacks an effective feedback mechanism because it not capable of truly understanding mathematics, and is therefore unable to provide informative feedback. A series of recent tools based on recurrent neural networks (RNNs) [3] have found great success in various NLP tasks (e.g., machine translation, image captioning, etc.) and predicting the output of simple computer code [7]. Natural language processing for the purposes of grading and feedback has also made substantial progress in several restricted domains including essay evaluation and mathematical proof verification [2, 6]. These successes inspires us to use RNNs to analyze responses to mathematical questions due to their sequential, step-by-step format and their algorithmic nature. They support our belief that LSTMs have the ability to learn simple mathematical operations such as factoring polynomials from data and providing relevant feedback.

1.1 Contributions

In this paper, we apply the LSTM network [3], a type of RNN, to try to understand simple mathematics for automatic grading and feedback generation for open-response mathematical questions. In particular, we study the simple problem of *polynomial factorization* due to the fact that responses to polynomial factorization questions are typically short and require only simple mathematical operations. We first generate a synthetic dataset using the Wolfram Alpha API consisting of responses (step-by-step solutions with mathematical expressions and text explaining the mathematical operations performed) to polynomial factorization questions. We then train multiple LSTM networks on the dataset and evaluate their performance on factoring previously unseen polynomials. Preliminary results show that the trained character level networks can factor previously unseen polynomials up to the second order with sufficient accuracy, after training on enough examples. More importantly, we showcase how the trained networks have the potential for automatic grading and feedback generation for open-response mathematical questions.

We emphasize that our proposed method has the capability to go beyond Wolfram Alpha. First, the ability of the trained LSTM networks to generalize to previously unseen examples enables *transfer* between domains, i.e., these networks have the capability of learning a rule in a certain context and apply it in another context. This property enables a LSTM network to build on its own knowledge as more and more training data becomes available, which is a much more scalable approach than the rules-based Wolfram Alpha system, which requires new rules to be manually coded for every new domain.

2. EXPERIMENTS

Experimental setup. We generate factorable polynomials that are subsequently used by the Wolfram Alpha API to produce responses on how to fully factor these polynomials. The responses include step-by-step solutions that consist of a series of mathematical expressions that end up in a fully-factored final form, together with concise text describing the mathematical operations involved. The data generation process is limited to polynomials with a single variable, co-

	Character Level % Error			Expression Level % Error		
# units	1 Layer	2 Layer	3 Layer	1 Layer	2 Layer	3 Layer
50	31.11	20.98	20.40	87.93	80.76	78.28
200	11.79	10.68	10.12	68.55	59.39	56.80
512	12.94	**8.21**	10.32	42.38	39.94	**38.95**

Table 1: Character and expression level misclassification errors on the test set. Performance of the best models are highlighted in bold.

efficients that are less than 10 and up to the third order. We construct a training dataset including 200,000 responses to various factoring questions this way. A test dataset is constructed with 20 first, 20 second, and 20 third order polynomials to be factored. We emphasize that, while for the simple task of polynomial factorization, Wolfram Alpha is able to generate the correct response, our aim is to develop a method that can generalize to more complicated mathematical operations that are too complicated for a rules-based system like Wolfram Alpha to cover. We train our LSTM networks to operate on a character-by-character level, i.e., use each character in a response as input and output data at each time instant. We train 9 different LSTM networks with varying number of hidden units ($N \in \{50, 200, 512\}$) and layers (1, 2, and 3). We use 95% of the generated training dataset for training and 5% as the validation dataset; We train the LSTM networks for a total of 50-150 epochs or terminate the training process early if the validation error shows minimal change across 10 epochs. In order to achieve faster training, we apply the curriculum learning approach [1], i.e., we start by training the LSTM networks on factorizations of first order polynomials until the validation error cannot be further reduced, and then proceed to train on responses factoring second order polynomials and beyond.

Results and discussion. We evaluate the performance of our trained LSTM networks on factoring previously unseen polynomials using two metrics. The first metric computes the character-level misclassification error rate by comparing every character in the correct factorization to the maximum-likelihood predicted character by the trained LSTM network. The second metric computes the expression-level misclassification error rate by comparing every full mathematical expression in the correct factorization to the full predicted expression by the trained LSTM network; a successful classification means that the entire expression is correctly predicted.

Experimental results for all 9 LSTM networks on both metrics are shown in Table 1. In general, LSTM networks with more hidden units and layers achieves lower misclassification error rates. We note that the expression-level misclassification rate is much higher (the best model achieves an error rate of 38.95%) than the character-level misclassification rate (the best model achieves an error rate of 8.21%). This observation is not surprising since correctly predicting the entire expression is much more difficult than successfully predicting a character. Moreover, we observe that the best model achieves error rates of 0% and 15%, respectively, on factoring first and second order polynomials but a 100% error rate on third order polynomials. This result is due to the fact that factoring third order polynomials is hard since it requires first factoring out a second order polynomial as an intermediate step.

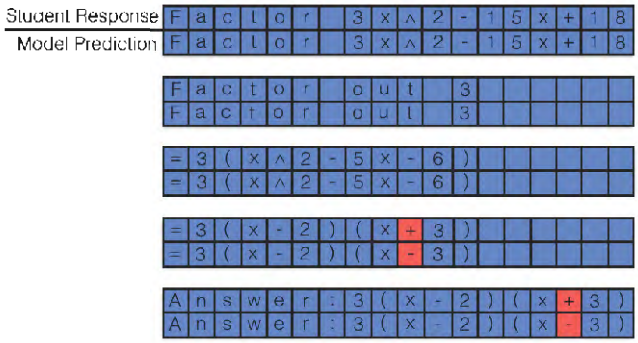


Figure 1: Illustration of how to use of a trained LSTM network to detect when a student's response deviates from the correct response.

Using trained LSTM networks for grading and feedback. We now illustrate how the trained LSTM networks can be used for automatic grading and feedback generation. Figure 1 shows a typical use case with an actual student response and a direct comparison to the maximum-likelihood character the trained LSTM network predicts given the previous characters as input. For automatic grading, we can calculate the predictive likelihood of every character in a student's response using a trained LSTM network. We can then assign a grade to a response by its total predictive likelihood; since our LSTM networks are trained on correct responses, a correct response will have a higher predictive likelihood than an incorrect one. For personalized feedback generation, we can automatically alert a student that they might have made an error if the predictive likelihood of the next input character is lower than a certain threshold. In Figure 1, such an error is shown in red where the student response contains a character that the trained LSTM network predicts as highly unlikely. Using these predictive probabilities, we can also automatically provide hints to a student about the most likely next expression in case they get stuck.

3. REFERENCES

[1] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proc. 26th Intl. Conf. Mach. Learn.*, pages 41–48, June 2009.

[2] M. Cramer, B. Fisseni, P. Koepke, and D. Kühlwein. The naproche project controlled natural language proof checking of mathematical texts. In *Cont. Nat. Lang.*, pages 170–186, 2009.

[3] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[4] A. S. Lan, D. Vats, A. E. Waters, and R. G. Baraniuk. Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In *Proc. 2nd ACM Conf. Learn. at Scale*, pages 167–176, Mar. 2015.

[5] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *J. Mach. Learn. Res.*, 15:1959–2008, June 2014.

[6] A. Naumowicz and A. Kornilowicz. A brief overview of mizar. In *In Proc. 22nd Intl. Conf Theorem Proving in Higher Order Logics*, pages 67–72, 2009.

[7] W. Zaremba and I. Sutskever. Learning To Execute. *arXiv preprint arXiv:1410.4615*, pages 1–25, Feb. 2015.

Intelligent Composition of Test Papers based on MOOC Learning Data*

Lin Ma
Department of Computer Science and Technology, Tsinghua University
Beijing, China 100084
ml16@mails.tsinghua.edu.cn

Yuchun Ma
Department of Computer Science and Technology, Tsinghua University
Beijing, China 100084
myc@mail.tsinghua.edu.cn

ABSTRACT

In recent years, most of the studies related to MOOC are mainly about prediction and data analysis, while how to evaluate the learning performance is still based on the experience of teachers. Especially, how to compose a proper exam paper is still a tedious work. In this paper, we use genetic algorithm to compose test papers with the support of MOOC learning data considering various constraints and objectives. The experimental results based on a MOOC course show that the mean absolute error of prediction model is roughly around 12 points on 100 points scale and we can successfully achieve the intelligent composition of test papers with various objectives optimized.

Keywords

MOOC(Massive Open Online Course); Machine Learning; Performance Prediction; Genetic Algorithm; Automatic Composition of Test Paper

1. INTRODUCTION

In this paper, we focus on how to evaluate MOOC learners' learning performance. Traditional written test's high dependence on the teacher and neglect of the learners make it ineffective in the MOOC learning environment. So in this paper, we provide a novel approach that the final exam papers could be automatically composed with the support of MOOC learning data considering various constraints and objectives. In our approach, different machine learning techniques are employed to construct a prediction model of learning performance based on MOOC learning data. With the prediction model of the learning performance, an intelligent composition approach is proposed with various objectives and constraints considered.

2. RELATED WORK

From 2012 to now, more and more people start to study MOOC, such as [2, 1]. Common algorithms of automatically generating test papers mainly include stochastic selection with approximate matching[6], backtracking and genetic algorithm[4, 5].

3. MODEL AND OVERALL FRAMEWORK

3.1 Model

Figure 1 shows the whole process of using MOOC learning data to intelligently auto-generate test paper. The input is MOOC learners' learning data, and the output is a test paper. Here we use the scores of usual quiz and homeworks as learning data, and use the score of final exam to represent learning performance. The whole process is composed of two important phases, performance prediction and test paper's composition. In the first phase, we use machine learning techniques to train the performance prediction model. And in the second phase, we use genetic algorithm to generate test paper.

3.2 Classified Performance Prediction Model for Different Levels of Learners

Performance prediction is a very common and simple regression problem. However, if model is constructed simply for all learners, the prediction results are always not very satisfactory because of the complexity and diversity of learners. Intuitively, we know that students with different learning levels will have different learning patterns [2]. Therefore, the features which are useful and contribute to the prediction results are obviously different for different levels of learners. Hence, the performance prediction of massive learners should be based on the level of learners, rater than treating them as a whole. Different levels of learners should have different prediction model.

3.3 Intelligent Composition of Test Papers Based on Genetic Algorithm

The goal of this section is to generate a test paper that meets all constraints as much as possible. The constraints include total score, difficulty, question types and knowledge points. We need to format all constraints to a argument matrix as the input of the composition of test papers[6]. For question types and knowledge points, it can be obtained by multiplying distribution matrix by total scores. For difficulty, most of the statistical analysis show that a good test has a normal distribution of scores, so we can generate it according to the

*This paper is supported by Online Education Fund of Quan Tong Education (2016ZD304).

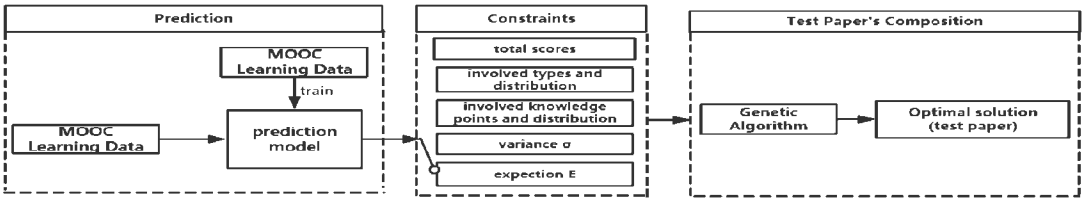


Figure 1: The model framework of intelligent composition of test paper based on MOOC learning data

Table 1: Prediction Error of Machine Learning Algorithms

Model	M5rules	SMOreg	LWR	LR	BP
Overall	21.103	21.423	21.657	21.132	34.006
Classified	12.069	12.82	11.127	13.026	15.058

expected scores E and variance σ. The expected score is exactly our predicted results in the last phase. The proportion of a certain difficulty level can be derived from the proportion of students in the corresponding scores. For instance, the proportion of "easy" level is equal to the proportion of students in scores 80-100 if there are a total of 5 levels. The design of the genetic algorithm can be obtained from [4] and [6].

4. EXPERIMENTAL RESULTS

4.1 Data Description

Our data comes from *Combinatorial Mathematics*, a math class opened for graduates majored in computer science and technology, Tsinghua University. It has been opened in both EdX and xuetangX. We can get a total of 35 features, including 25 quiz scores, 8 homework scores and 1 final exam score. And the feature need to be predicted is final exam score since we use it to represent learner's learning performance.

4.2 Prediction Experiment and Results

This experiment is a comparative experiment of the classified prediction model and the overall prediction model. We adopt machine learning algorithms used in [3]. In classified model, we divided the learners into two groups according to their academic performance, passing the exam as a group and the rest as a group. The final prediction results are shown in table 1. Note that here we adopt mean absolute error as our prediction error and all of the scores appearing in this paper are converted to percentile scores. From the results, we find classified model for different levels of learners can greatly reduce the prediction error by around 10 points.

4.3 The Composition of Test Paper Based on MOOC Learning Data

This experiment is conducted to verify the performance of the composition algorithm. In this experiment, we first randomly select n testers from 17 testers. And then generating a test paper according to the average performance of all selected testers to test them. From the experimental results shown in table 2, we find that predicted scores(performance)

Table 2: Examination Results

number of testers	predicted scores (performance)	real exam scores
17	77.59	75.08
16	79.75	71.37
13	73.91	69.23
12	75.94	61.14
6	82.48	69.63

are very close to their real exam scores and the error decreases as the number of testers increases, which indicates that our model is effective for evaluation of a group of MOOC learners' learning performance.

5. CONCLUSION

The general idea of this paper is automatically generating personalized papers under the guidance of MOOC learners' usual performance, so as to guide their further study. But there are still many details need to be further refined, such as prediction accuracy, efficiency of the composition algorithm, and so on. Therefore, it's just a first step in integrating machine learning, MOOCs, and test development. Our future work will continue to focus on these details to make it better.

6. REFERENCES

[1] G. Balakrishnan and D. 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] 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 edx's first mooc. *Research & Practice in Assessment*, 8, 2013.

[3] S. B. Kotsiantis and P. E. Pintelas. Predicting students marks in hellenic open university. In *Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05)*, pages 664–668. IEEE, 2005.

[4] Y. Ou-Yang and H.-F. Luo. Design of personalized test paper generating system of educational telenet based on genetic algorithm. In *Computer Science & Education, 2009. ICCSE'09. 4th International Conference on*, pages 170–173. IEEE, 2009.

[5] Y. Qing. Research on auto-generating test paper based on genetic algorithm. *JOURNAL OF JINAN UNIVERSITY(SCIENCE AND TECHNOLOGY)*, 18(3):228–231, 2004.

[6] G. M. Wang Yuying, Hou Shuang. Algorithm for automatic test paper generation. *JOURNAL OF HARBIN INSTITUTE OF TECHNOLOGY*, 35(3):342–346, 2003.

Toward Replicable Predictive Model Evaluation in MOOCs

Josh Gardner, Christopher Brooks
School of Information
University of Michigan
{jpgard, brooksch}@umich.edu

ABSTRACT

In this paper, we present and apply a procedure for evaluating predictive models in MOOCs. First, we expand upon a procedure to statistically test hypotheses about model performance which goes beyond the state-of-the-practice in the community and covers the full scope of predictive model-building in MOOCs. Second, we apply this method to a series of algorithms and feature sets derived from a large and diverse sample of MOOCs ($N = 31$), concluding that several models built with simple clickstream-based feature extraction methods outperform those built from forum- and assignment-based feature extraction methods.

1. INTRODUCTION AND RELATED WORK

Building predictive models of student success has emerged as a core task in the fields of learning analytics and educational data mining.[1] The process of building such models in MOOCs involves at least three key stages: (1) extracting structured data and informative features from raw platform data (clickstream server logs, database tables, etc.); (2) selecting algorithms and models; and (3) tuning hyperparameters. Together, these stages profoundly influence the performance of predictive models. We identify at least two methodological gaps in current educational data mining research as it relates to this task: (1) current research typically isolates these steps, e.g., evaluating different approaches to feature extraction or algorithm selection separately without considering their relation to each other; and (2) procedures for rigorous and reproducible statistical inference about the relative performance of these models, and accounting for the many model specifications considered in the course of an experiment, are often not followed.

Previous predictive modeling research in MOOCs has evaluated features derived from clickstreams, discussion fora, assignments, and surveys, among other sources. In addition, this research has applied a variety of algorithms to such data for dropout prediction, including linear and logistic regression, support vector machines, tree-based methods, ensemble methods, neural networks, and deep learning. However, a literature survey by the authors indicated that accepted statistical practices for evaluating these models are often neglected by such research[2] In particular, more than half of surveyed research did not utilize any statistical testing for evaluating model performance, despite obtaining estimates directly on the training set through cross-validation for multiple models. These methods are susceptible to spurious results and low replicability due to multiple comparisons, biased performance estimates, and random variation from resampling schemes [3, 4, 7, 11]. Recent research has provided evidence that some MOOC research may not be replicable when applied to new or different courses [1]; at the very least, this highlights the importance of adopting reproducible and statistically valid methods for model evaluation in MOOCs [8]. An extensive literature exists on statistically reliable methods for model evaluation [4, 6, 11].

2. METHODOLOGY

We implement a testing and inference procedure from [3] for selecting the best of $k > 2$ models across $N > 1$ datasets (in this experiment, a *model* is a *feature set-algorithm-hyperparameter combination*), which consists of two steps. First, a Friedman test is used to test the null hypothesis that the performance of all models is equivalent [5]. The Friedman statistic

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \quad (1)$$

where R_i^j is the rank of the jth of k algorithms on N datasets and the statistic is χ_{k-1}^2 distributed, is compared to a critical value at the selected significance level ($\alpha = 0.05$ in this experiment). If H_0 is rejected, then we proceed to the second stage, the post-hoc Nemenyi test, where

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \quad (2)$$

is used to determine whether the performance between any two classifiers is significantly different, where q_α is based on the Studentized range statistic divided by $\sqrt{2}$.

This two-stage procedure allows us to conduct comparisons across multiple models and datasets to draw inferences about

[1]The current work evaluates models of student dropout in MOOCs, but this methodology applies to any supervised predictive modeling task.

[2]This survey reviewed the 2014-2016 International Society for Educational Data Mining (EDM) and the International Learning Analytics and Knowledge (LAK) conference proceedings, and included research which attempted to predict completion or performance using behavioral or academic features with features derived from MOOC platform data; a full survey is forthcoming in a future work.

whether true performance differences exist, accounting for the number of comparisons k and datasets N. Unlike using simple average cross-validated training performance, this procedure uses statistical testing to evaluate whether the observed difference is statistically significant or may be merely spurious, based on the available data. In applying this method to a *feature set + algorithm + hyperparameter* combination, we can (1) evaluate feature extraction as a testable modeling component; (2) capture and evaluate the synergy between feature extraction, algorithm, and hyperparameters; and (3) draw inferences which fully account for the number of comparisons across all of these elements. [3]

3. EXPERIMENT AND RESULTS

As an illustrative example, we compare a series of models using three feature sets and two predictive algorithms on a set of 31 offerings of 5 unique courses offered by the University of Michigan on Coursera, with 298,909 total learners. From the raw clickstream files and database tables, we extracted a series of features intended to replicate (with some additions) features shown to be effective dropout predictors, with each utilizing information from a different raw data source: *clickstream* [10], *assignment* [9], and *forum* features [1].

We train two classifiers – standard classification trees and adaptive boosted trees – on various combinations of the three feature sets, performing no hyperparameter tuning (to limit the number of comparisons, k). Figure 1 presents the results of our analysis.

Results from dropout prediction after course week 2 are shown in Figure 1, but our findings were consistent across all four weeks examined. We find that models utilizing clickstream features consistently outperform those using forum and quiz features. This difference was statistically significant for all model configurations tested. Changing the classification algorithm had little effect on the performance of quiz- and forum-featured models, which were statistically indistinguishable from each other in every week evaluated. When the clickstream features are combined with forum and quiz features to form a "full" model, this model achieves better performance than the clickstream features alone, but this improvement is never statistically significant over the best clickstream-only model. This suggests that the forum and quiz features contain useful structure which may require powerful, flexible classification algorithms to capture. Our conclusion – that the highest-performing model is statistically indistinguishable from other models in this analysis – stands in contrast to the practice of much of the prior research surveyed, which often concludes that the best average performance is the "best" model; this is intended to serve as an example for inferential language in future research.

4. FUTURE RESEARCH

Future research should utilize this or other methods for statistically evaluating performance comparisons of predictive models. In particular, it should explore Bayesian methods for model evaluation, which allow the direct estimation of probabilities of hypotheses, avoid concerns about multiple comparisons, and have other additional advantages [2].

[3]There are clear advantages to adopting this specific procedure over other testing approaches such as ANOVA, or other nonparametric approaches; see §3.2.1 of [3] for detailed discussion of these benefits.

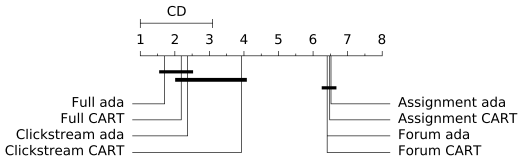


Figure 1: Critical Difference (CD) diagram of week 2 dropout prediction models. Models are plotted by average rank, with bold CD lines indicating statistically indistinguishable models (at $\alpha = 0.05$). We reject H_0 of equivalent performance for models not connected by CD lines. These results show a statistically significant performance gap between clickstream features and assignment or forum features.

5. REFERENCES

[1] J. M. L. Andres, R. S. Baker, G. Siemens, D. GAŠEVIĆ, and C. A. Spann. Replicating 21 findings on student success in online learning.

[2] A. Benavoli, G. Corani, J. Demsar, and M. Zaffalon. Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. 14 June 2016.

[3] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7(Jan):1–30, 2006.

[4] T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.*, 10(7):1895–1923, 15 Sept. 1998.

[5] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Stat.*, 11(1):86–92, 1940.

[6] S. Garcia and F. Herrera. An extension on"statistical comparisons of classifiers over multiple data sets"for all pairwise comparisons. *J. Mach. Learn. Res.*, 9(Dec):2677–2694, 2008.

[7] C. Nadeau and Y. Bengio. Inference for the generalization error. *Mach. Learn.*, 52(3):239–281, 2003.

[8] F. van der Sluis, T. van der Zee, and J. Ginn. Learning about learning at scale: Methodological challenges and recommendations. In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale*, L@S '17, pages 131–140, New York, NY, USA, 2017. ACM.

[9] K. Veeramachaneni, U.-M. O'Reilly, and C. Taylor. Towards feature engineering at scale for data from massive open online courses. 20 July 2014.

[10] W. Xing, X. Chen, J. Stein, and M. Marcinkowski. Temporal predication of dropouts in MOOCs: Reaching the low hanging fruit through stacking generalization. *Comput. Human Behav.*, 58:119–129, 2016.

[11] O. T. Yildiz, E. Alpaydin, and Senior Member. Ordering and finding the best of k>2 supervised learning algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(3), 2006.

Modeling the Zone of Proximal Development with a Computational Approach

Irene-Angelica Chounta, Bruce M. McLaren
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh PA, 15213, USA
{ichounta,bmclaren}@cs.cmu.edu

Patricia Albacete, Pamela Jordan, Sandra Katz
Learning Research and Development Center
University of Pittsburgh
Pittsburgh PA, 15260, USA
{palbacet, pjordan, katz}@pitt.edu

ABSTRACT

In this paper, we propose a computational approach to modeling the Zone of Proximal Development of students who learn using a natural-language tutoring system for physics. We employ a student model to predict students' performance based on their prior knowledge and activity when using a dialogue tutor to practice with conceptual, reflection questions about high-school level physics. Furthermore, we introduce the concept of the "Grey Area", the area in which the student model cannot predict with acceptable accuracy whether a student has mastered the knowledge components or skills present in a particular step.

Keywords

Natural-language tutoring systems, intelligent tutoring systems, student modeling, zone of proximal development

1. INTRODUCTION

Intelligent Tutoring Systems (ITSs) support students in grasping concepts, applying them during problem-solving activities, addressing misconceptions and in general improving students' proficiency in science, math and other areas [6]. ITS researchers have been studying the use of simulated tutorial dialogues that aim to engage students in reflective discussions about scientific concepts [4]. However, to a large extent, these systems lack the ability to gauge students' level of mastery over the curriculum that the tutoring system was designed to support. This is also challenging for human tutors, who do gauge the level of knowledge and understanding of their tutees to some degree, although they are poor at diagnosing the causes of student errors [3]. We argue that in order to provide meaningful instruction and scaffolding to students, a tutoring system should appropriately adapt the learning material with respect to both content and presentation. A way to achieve this is to dynamically assess students' knowledge state and needs. Human tutors use their assessment of student ability to adapt the level of discussion to the student's "zone of proximal development" (ZPD)—that is, "*the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers*" [7]. Deriving ways to identify and formally describe the ZPD is an important step towards understanding the mechanisms that drive learning and development, gaining insights about learners' needs, and providing appropriate pedagogical interventions [2]. Following the practice of human tutors, we propose a computational approach to model the ZPD of students who carry out learning activities using a dialogue-based intelligent tutoring system. We employ a student model to assess students' changing knowledge as they engage in a dialogue with the system. Based on the model's predictions, we define the concept of the "Grey Area", a probabilistic region in which the model's predictive accuracy is low. We argue that this region can be used to indicate whether a student is in the ZPD. Our research hypothesis is that we can use the outcome of the student model (i.e., the fitted probabilities that predict students' performance) to model students' ZPD. To the best of our knowledge, this is a novel approach to modeling the ZPD. Even though we focus on dialogue-based tutoring systems, we expect that our approach can be generalized and extended to other kinds of ITSs.

2. METHODOLOGY

In this study, we used data collected during three previous studies with the Rimac system to train a student model and frame the proposed approach. Rimac is a web-based natural-language tutoring system that engages students in conceptual discussions after they solve quantitative physics problems [5]. Rimac's dialogues present a directed line of reasoning (DLR) where knowledge components (KCs) relate to tutor question/student response pairings. To model students' knowledge we used an Additive Factor Model (AFM) [1]. The model predicts the probability of a student completing a step correctly as a linear function of student parameters, knowledge components and learning parameters. AFM takes into account the frequency of prior practice and exposure to skills but not the correctness of responses. The dataset consists of training sessions of 291 students over a period of 4 years (2011-2015). Students worked on physics problems that explore motion laws and address 88 knowledge components (KCs). The dataset contains in total 15,644 student responses that were classified as correct or incorrect using the AFM student model.

Our research hypothesis is that we can use the fitted probabilities, as predicted by the student model, to model the ZPD. The core rationale is that if the student model cannot predict with high accuracy whether a student will answer a tutor's question correctly, then it might be the case that the student is in the ZPD. The student model provides predictions at the step level: each step consists of one question/answer exchange from the tutorial dialogue. A step may involve one or more KCs. The classification threshold (i.e., the cutoff determining whether a response is

classified as correct or incorrect) is 0.5 and it was validated by the ROC curve for the binary classifier. We expect that the closer the prediction is to the classification threshold, the higher the uncertainty of the model and thus, the higher the prediction error. Based on our hypothesis, this window of uncertainty can be used to approximately model the student's zone of proximal development. We refer to this window as the "Grey Area".

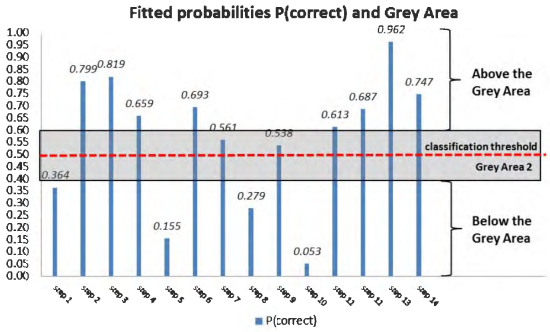


Figure 1. The Grey Area concept with respect to the fitted probabilities as predicted by the student model for a random student and for the various steps of a learning activity. Here we depict the example of a symmetrical Grey Area extending on both sides of the classification threshold.

The concept of the Grey Area is depicted in Figure 1. The space "Above the Grey Area" denotes the area where the student is predicted to answer correctly and consequently may indicate the area above the ZPD; that is, the area in which the student is able to carry out a task without any assistance. Accordingly, the space "Below the Grey Area" denotes the area where the student is predicted to answer incorrectly and consequently may indicate the area below the ZPD; that is, the area in which the student is not able to carry out the task either with or without assistance. In this paper, we model the grey area symmetrically around the classification threshold for simplicity and because the binary classifier was set to 0.5. However, the symmetry of the Grey Area is something that could change depending on the classification threshold and the learning objectives. Furthermore, we do not propose a specific size for the Grey Area. We believe that the decision about the appropriate size (or shape) of the Grey Area is not only a modeling issue but mainly a pedagogical one since it relies on the importance of the concepts taught, the teaching strategy and the learning objectives.

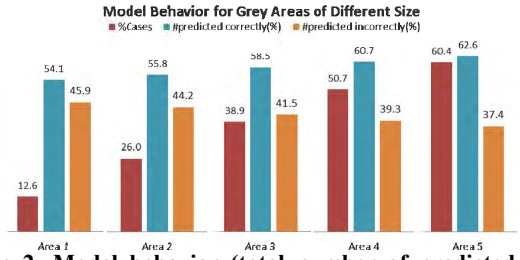


Figure 2. Model behavior (total number of predicted cases, cases predicted correctly and cases predicted incorrectly) within five grey areas of different sizes. The areas are ordered from the most narrow (Area 1) to the widest (Area 5).

Figure 2 presents an analysis of the cases that are contained in the Grey Area. In this preliminary analysis, we examined five Grey Areas of different size. On one hand, choosing a narrow grey area to model the ZPD would limit the number of cases we scaffold since fewer cases would fall within the area. On the other hand, choosing a wide grey area would affect the accuracy; that is, some cases that could be predicted correctly would be falsely labeled as "grey". However this work does not aim to define the appropriate size for the Grey Area but rather to study how the model's behavior may change for areas of different size.

3. DISCUSSION

In this paper, we present a computational approach that aims to model the Zone of Proximal Development in ITSs. To that end, we introduce the concept of the "Grey Area". Our proposal is that if the model cannot predict the state of a student's knowledge, it may be that the student is in the ZPD. We envision that the contribution of the proposed approach, besides its novelty (to the best of our knowledge there is no quantified operationalization of the ZPD) will be in defining and perhaps revising instructional methods to be implemented by ITSs. Choosing the "next step" is a prominent issue in the case of dialogue-based intelligent tutors. Not only should the task be appropriate with respect to the background knowledge of the student, but it should also be presented in an appropriate manner so that the student will not be overwhelmed and discouraged. To address this issue, we need an assessment of the knowledge state of each student and insight into the appropriate level of support the student needs to achieve the learning goals. This is described by the notion of ZPD. It is evident that if we can model the ZPD then we can adapt our instructional strategy accordingly. A limitation of our work is that we have not yet been able to conduct a rigorous evaluation of our approach; however, plans to validate our modeling methods are being developed. Our immediate plan is to carry out extensive studies to explore the proposed approach to modeling the ZPD further, as well as to better understand the strengths and limitations of using a student model to guide students through adaptive lines of reasoning.

4. REFERENCES

[1] Cen, H., Koedinger, K., and Junker, B. 2008. Comparing two IRT models for conjunctive skills. In *International Conference on Intelligent Tutoring Systems*, 796–798.

[2] Chaiklin, S. 2003. The zone of proximal development in Vygotsky's analysis of learning and instruction. *Vygotsky's educational theory in cultural context.* 1: 39–64.

[3] Chi, M.T., Siler, S.A., and Jeong, H. 2004. Can tutors monitor students' understanding accurately? *Cognition Instruct*. 22, 3: 363–387.

[4] Di Eugenio, B., Glass, M., and Trolio, M.J. 2002. The DIAG experiments: Natural language generation for intelligent tutoring systems. In *INLG02, The Third International Natural Language Generation Conference*, 120–127.

[5] Katz, S., and Albacete, P.L. 2013. A tutoring system that simulates the highly interactive nature of human tutoring. *J. Educ. Psychol.* 105, 4: 1126.

[6] VanLehn, K. 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*. 46, 4: 197–221.

[7] Vygotsky, L. 1978. Interaction between learning and development. *Readings on the development of children.* 23, 3: 34–41.

A Prediction and Early Alert Model Using Learning Management System Data and Grounded in Learning Science Theory

Wonjoon Hong
University of Nevada, Las Vegas
4505 S Maryland Pkwy Las Vegas, NV 89154, USA
+001(702)895-3253
hongw1@unlv.nevada.edu

Matthew L. Bernacki
University of Nevada, Las Vegas
4505 S Maryland Pkwy Las Vegas, NV 89154, USA
+001(702)895-4013
matt.bernacki@unlv.edu

ABSTRACT

Students experience considerable challenge in STEM coursework and many struggle to earn the grades needed to move forward in their majors. Interventions informed by prediction models can support learners to ensure successful completion of STEM courses and entry into the STEM workforce. In order to accurately target intervention efforts, we developed a prediction model based on log data generated by student use of content hosted on a learning management system (LMS; Blackboard Learn) course site in the first weeks of the course. The prediction model employed a forward selection logistic regression algorithm (with 10-fold cross validation) trained on four semesters of data, and provided instructors the opportunity to message students and provide learning support before the first major exam, potentially intervening before onset of poor performance. The best fitting model was used to identify students unlikely to obtain the required grade (B or better) in the course. Among 106 students predicted to perform poorly, 63 received a message from the instructor's account that referenced an upcoming exam and linked students to supportive materials. Messaged students who accessed learning supports outperformed non-messaged but eligible students (n = 43) on each of five subsequent exams throughout the semester (ds = .64 - .88). Fifty-eight percent earned a B or better, compared to 25% of non-messaged peers predicted to earn a C or worse. This study affirms that data-driven early alert messages can provide targeted support and boost achievement in challenging STEM courses.

Keywords

Learning management system, Prediction modeling, Early warning system, STEM learning, learning sciences

1. INTRODUCTION

Learning management system (LMS) have become a central tool in higher education. Logs of learning events can be combined with achievement data in order to identify (un)productive patterns of events and predict the achievement of future students based on their behavioral match to prior students who achieved certain levels of performance [1].

2. METHODS

The university LMS, Blackboard Learn, captures and records student use of materials hosted on course sites. Student activity and achievement data (N=510) from 4 semesters of an undergraduate calculus course taught by two instructors (identical content, assessments) from fall 2014 to spring 2016 informed prediction modeling (Table 1).

Table 1. Training and testing data

Section	Training set	Testing set
Instructor A	Fa 2014 & Sp 2015 (n=167)	Fa 2015 (n=96)
Instructor B	Fa 2014 & 2015 (n=161)	Sp 2016 (n=86)
Both	Instructor A (Fa 2014 & Sp 2015) Instructor B (Fa 2014 & 2015) (n=328)	Instructor A (Fall 2015) Instructor B (Spring 2016) (n=182)

Developing the prediction model went through two main phases, training and testing process. In the training phase, logistic regression with forward selection was used to build the prediction model, and the problem of overfitting was examined through 10-fold cross-validation. In the testing phase, the most accurate prediction model developed in the training phase was applied to the testing data set to assess potential overfitting and ensure generalizability to future students' data [2].

Based on the Kappa (κ) and recall, the best 3-week prediction model developed through the training and testing phases was then applied to data from fall 2016 Calculus students to identify students in need of an early alert message that provides learning support.

In order to investigate the effect of messaging identified students, those identified as likely to perform poorly by the prediction model were randomly divided into two groups, a "Message" group who would receive a message that focused attention on an upcoming exam and some useful learning resources (Figure 1) and a "No Message" group who would not.

Hi [Name]!
Our first course exam is coming up on Friday…
1. The first is a one-page summary of advice from students who have completed the course with an excellent grade in the past.…
2. A set of learning modules called "The Science of Learning to Learn." These modules describe learning strategies you can use with our course materials…

Figure 1. Message to students

3. RESULTS

Among three models, the prediction model based on Instructor B's students produced the best Kappa ($\kappa = 0.26$) and recall (73%) values. The model accurately identified ≥ 7 in 10 students who would ultimately earn less than 80% of points (i.e., a C or Worse). We thus moved forward to the testing phase using the Instructor B model (Table 2) and for the prediction and messaging phase.

Table 2. Prediction models in the training and testing phase

	True: Predicted				K	Accuracy (%)	Precision (%)	Recall (%)
	1;1	1;0	0;1	0;0				
Training set								
Instructor A (Fall 2014 & Sp 2015)	79	12	46	27	.25	65	87	37
Instructor B (Fa 2014 & 2015)	**39**	**36**	**23**	**63**	**.26**	**63**	**52**	**73**
Both	97	69	63	96	.19	59	59	60
Testing set								
Instructor A (Fa 2015)	16	25	9	46	.24	65	65	84
Instructor B (Sp 2016)	19	23	11	33	.20	61	59	75
Both	35	48	20	79	.21	63	62	80

In the testing phase, attributes and their weights achieved from the training phase were applied to the testing data to examine risk of overfitting. The prediction model resulted in the Kappa value of .20 or more for all testing sets. In addition, values of recall were 84, 75, and 80 respectively, all of which were greater than result in the training phase. We thus retain the Instructor B model for the prediction and messaging phase.

Upon sending the message four days prior to the first exam, student access of recommended resources and performance on exams were tracked throughout the remainder of the semester. For all exams throughout the semester, the students in treatment group (i.e., Message & Access) performed better than those without any treatment (No Message, No Access; $p < .05$). In addition, effect sizes for all exams were more than "medium" ($d > .5$) (Table 3).

Table 4. Contingency Table

		Predicted C or Worse		Total
		Messaged	Control	
True	B or Better	11 (58%)	7 (25%)	18
	C or Worse	8 (42%)	21 (75%)	29
Total		19	28	47

Table 4 shows the proportion of students who performed better than (i.e., B or Better) vs. as projected (i.e., C or Worse). A Chi-square analysis indicated that a significantly greater proportion of students (58%) in the Message and Access group earned a final grade of B or better, $\chi^2 (47) = 5.18$, $p = .02$. Only 25% of students predicted to earn a C or worse outperformed their prediction in the No Message, No Access control group.

4. DISCUSSION

In this study, those who received a brief email message from a course instructor and accessed a learning resource outperformed non-messaged students on all exams. Results thus indicate that data-driven interventions can be provided relatively early in the semester – six weeks earlier than the typical data-driven indicator of poor future outcome: a week 9 response to midterm grades. The >200-word message required only a minute or two of a typical student's time, and a visit to the advice page – the common material accessed – required only slightly more time investment from messaged students (~900 words).

The benefits of receiving a message and accessing the resources it recommends were substantial: 12% on all exams, or a full letter grade. Surprisingly, few students heeded the early alert as intended; 30% of messaged students accessed supportive materials, confirming that obtaining students' attention is a clear challenge to realization of the benefits messaging can provide. Messaging efforts thus clearly require improvement. We must also consider how to provide more adaptive message contents based on students' likelihoods of poor performance, or different supports based on the maladaptive practices summarized by features present in students' prediction models. More specific feedback about the kinds of learning behaviors that require adjustment may further increase messages' effects.

5. ACKNOWLEDGMENTS

This project was supported by National Science Foundation Award number #1420491 and Office of Information Technology.

6. REFERENCES

[1] Arnold, K. E., & Pistilli, M. D. 2012. Course signals at Purdue: Using learning analytics to increase student success. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (Apr. 2012). ACM, New York, NY, 267-270.

[2] Hämäläinen, W. and Vinni, M. 2010. Classifiers for educational data mining. In *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizky, and R. Baker, Eds. Data Mining and Knowledge Discovery Series. CRC Press, Boca Raton, FL, 57-7.

Table 3. Result of t-test of scores for all exams

	No Message & No Access			Message & Access			t	df	Sig.	Mean difference	Cohen's d
	N	Mean	SD	N	Mean	SD					
Exam 1	24	77.0	11.0	17	85.5	8.3	2.701	39	0.010	8.51	0.877
Exam 2	23	73.7	19.0	17	85.7	10.4	2.349	38	0.024	12.01	0.783
Exam 3	22	59.5	14.8	18	71.5	22.2	2.047	38	0.048	12.00	0.637
Exam 4	22	58.9	15.9	19	71.0	20.3	2.136	39	0.039	12.09	0.663
Final	22	55.7	23.8	19	70.9	23.6	2.043	39	0.048	15.17	0.640

Cluster Analysis of Real Time Location Data - An Application of Gaussian Mixture Models

Alvaro Ortiz-Vazquez
EdLab
Teachers College Columbia University
New York, New York USA
ao2444@columbia.edu

Xiang Liu
EdLab
Teachers College Columbia University
New York, New York USA
xl2438@tc.columbia.edu

Ching-Fu Lan
EdLab
Teachers College Columbia University
New York, New York USA
cl2483@tc.columbia.edu

Hui Soo Chae
EdLab
Teachers College Columbia University
New York, New York USA
hsc2001@tc.columbia.edu

Gary Natriello
EdLab
Teachers College Columbia University
New York, New York USA
gjn6@tc.columbia.edu

ABSTRACT
Clustering analysis in the context of education is important for determining the effectiveness of group activities especially when participants freely rotate between groups such as in a gallery exhibit or other informal learning space or set-up. In this paper, we cover a method of applying Gaussian Mixture Models to two-dimensional data. We further describe the analysis procedure, and the success of implementing this analysis using simulated data and real data. Finally, we discuss some educational applications as well as future directions for this research.

Keywords
Gaussian Mixture Models, MCMC, Gibbs Sampling, Real-Time Location System, Informal Learning Spaces, Learning Analytics, Dynamic Mixture Model

1. INTRODUCTION

Real-time locating systems have become increasingly popular and are predicted to be more widely adopted in informal learning institutions such as libraries, museums, and after school spaces in the next few years [2] [4]. Location intelligence and contextually relevant information can inform dynamically customized information and meaningful learning analytics for both learners and educators based on visitors and/or learners' location [3]. Such data are especially useful to understand social interactions in informal learning events. Therefore, it is essential for researchers to develop data mining methods to more efficiently and effectively explore real-time location data of learners.

Gaussian Mixture Models (GMM) are very useful for analyzing two-dimensional data which may be clustered into groups such as that collected by a real-time locating system in an informal learning space. To estimate the parameters of the GMM we employ a Markov Chain Monte Carlo method of Gibbs sampling [1] whose stationary state is the posterior distribution of the mixture model. This method applied to a frozen snapshot of the two-dimensional real-time location tracking data allows us to gain information about the groups, such as group membership, group location, and internal group dispersion, based only on the tag position data. Other algorithms such as k-means clustering may similarly cluster two-dimensional data but are non-parametric whereas Gibbs sampling is parametric.

2. DATA ANALYSIS

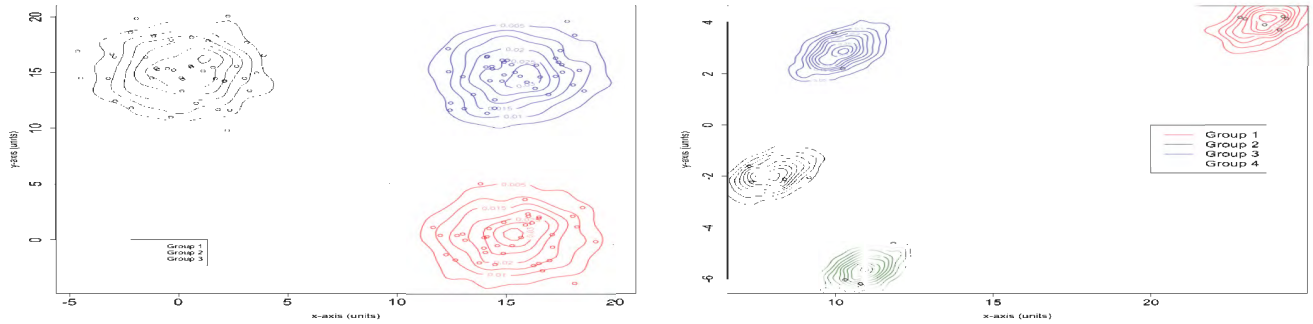
2.1 Simulations

To test the Gibbs sampling process and our R code we have drawn a set of location data points from bivariate normal distributions centered around three different centers ($\boldsymbol{\mu}_1 = (15, 15)$, $\boldsymbol{\mu}_2 = (15, 0)$,$\boldsymbol{\mu}_3 = (0, 15)$) with a common covariance. We observed the latent parameters of our Gibbs sampler reaching a stationary state in less than 100 iterations. In Figure 1a we generate estimated points using the estimated group centers and covariance and perform kernal density estimates to generate the coverage contours plotted over the original generated data. The percentage of estimated points outside the contours is marked on the contour lines. In this case we see that for 120 data points, a small number of the data lie outside of the 99.5% percent coverage contours. We can also verify the results by comparing the generating values for the centers and covariance with the estimated values.

2.2 Applications on Real Data

The real data were collected at an Edlab meeting at an innovative learning space: the Smith Learning Theater at Teachers College Columbia University. The Smith Learning Theater features technologies such as the Quuppa ^{TM} real-time locating system, installed to return measurable results and provide feedback to organizers and facilitators. In this meeting, 15 EdLab members wore Quuppa real-time locating tags and freely explored four stations of augmented/virtual reality apps in order to provide reviews for a national edtech competition. Applying the Gibbs Sampling method over the real data we again obvserved convergence within just 100 iterations. Again the coverage contours are drawn onto the plot of the positions in Figure 1b. In this analysis we did not have previous knowledge on the station device locations likely to be correlated with the group centers. However, we can still verify the success of the algorithm by noting that the data points are largely within the ninety-five percent coverage region. As such, our method returns accurate group information even with a small dataset.

3. DISCUSSION



(a) (b)

Figure 1: Kernal Density Estimate Contour Plots Over Simulation Data (a) and Real Data (b)

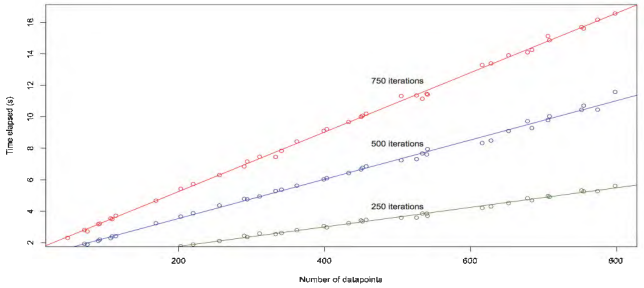


Figure 2: Linear Correlation Between Computational Time and the Number of Data Points

3.1 Educational Research and Applications

Our method has the limitation that the expected number of groups must be specified prior to performing the Gibbs sampling. This quantity can be available for events where group work takes place, or participants move around through different stations. In such an event our analysis can be implemented repeatedly over a series of consecutive discrete snapshots covering a period of time. By observing the group membership at each snapshot, the educator can determine information about who moved together as a group, or who moved mostly independently. Common group membership can be denoted in an adjacency matrix for the tags where the value for each index (i, j) is the number of snapshots in which two locating tags y_i, y_j shared the same group assignment. This approach has the potential to provide information about whether the learning space or activity was better suited for group learning or independent learning and the preferences of each participant to remain with the same group of people or move about with different people. In other events where group work may be taking place one can easily determine the amount of cross-group collaboration during a period of time by again looking at the cumulative group assignment data.

3.1.1 *Feasibility Analysis*

The implementation of the Gibbs Sampling algorithm takes linear $\mathcal{O}(N)$ time where N is the number of position data points in a single snapshot. We can generate N position data points and record the time elapsed for M iterations and visualize the linear relationship in Figure 2. Given an hour long event with 500 participants, covered by 360 snapshots, the linear model suggests that one could perform 250 iterations of the sampler over every snapshot in under twenty minutes. As such implementation of our method is feasable for most educational contexts.

3.2 Future Work

While our model is useful to see the group information within a snapshot of real-time location data, we believe that more important data will arise from extending our current mixture model to a Dynamic Mixture Model (DMM) [5]. In such a DMM, the group distribution of each snapshot would be dependent on the previous one. According to Wei et al. (2007) the assumption that two consecutive snapshots are dependent can allow us to analyze important patterns that would otherwise be missed in discrete snapshot analysis. By incorporating the temporal component, we expect to more accurately model transitions between groups. The application of our method is especially valuable in informal learning spaces as many learning events in these spaces encourage free exploration and group interactions, and evaluating learners' engagement and social group dynamics is challenging using other traditional research methods.

References

[1] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE transactions on pattern analysis and machine intelligence*, 6(6):721–41, jun 1984.

[2] B. Herr-Stephenson, D. Rhoten, D. Perkel, C. Sims, A. Balsamo, M. Klosterman, and S. S. Bautista. *Digital Media and Technology in Afterschool Programs , Libraries , and Museums*. 2011.

[3] K. Jaebker and G. Bowman. Context is king: Using indoor-location technology for new visitor experiences | MW2015: Museums and the Web 2015, 2017.

[4] L. Johnson, S. Adams Becker, M. Cummins, V. Estrada, A. Freeman, and C. Hall. *Horizon Report: 2016 Higher Education Edition*. The New Media Consortium, Austin, Texas, museum edi edition, 2016.

[5] X. Wei, J. Sun, and X. Wang. Dynamic mixture models for multiple time series. *Proceedings of the 20th international joint conference on Artifical intelligence*, (Dmm):2909–2914, 2007.

An LDA Topic Model and Social Network Analysis of a School Blogging Platform

Xiaoting Kuang
EdLab
Dept. of Human Development
Teachers College
Columbia University
xk2120@columbia.edu

Hui Soo Chae
EdLab
Teachers College
Columbia University
hsc2001@columbia.edu

Brian Hughes
EdLab
Teachers College
Columbia University
bsh2001@columbia.edu

Gary Natriello
EdLab
Teachers College
Columbia University
gjn6@columbia.edu

ABSTRACT
Pressible is a school blogging and content management system developed by EdLab at Teachers College Columbia University. In this paper, social network analysis and natural language processing with Latent Dirichlet Allocation topic model approaches were utilized to gain insights into Pressible, to explore four developmental stages of a college-wide social network and their associations with blog content. The results showed that professors who developed courses became the most influential persons in the network. Students extended the online discussion topics beyond the scope of course topic set by professors.

Keywords
SNA, NLP, Topic Model, LDA

1. INTRODUCTION
EdLab adapted the Wordpress Content management systems (CMS) framework and developed Pressible for the Teachers College (TC) community in 2008. It was designed for fast content delivery, minimization of users' time spent managing technology, and developing connections between users (Zhou, 2013). From the perspective of social constructivist theory, people communicate, contribute and acquire knowledge through social engagement and discussion of topics (Vygotsky, 1978). People also gain knowledge online via connecting information (Siemens, 2004). Massive Open Online Courses (MOOCs) provide more opportunities for people to study for personal intellectual growth (Kizilcec et al., 2017). Social factors from online discussion forums (Rose, et al., 2014) and engaging in higher order thinking behaviors enhanced learning in MOOCs (Wang, et al., 2016). Higher Education utilizes academic blogging to facilitate social networking, self-directed learning, and collaboration. Simulation studies on the blogosphere indicate that improved management facilities on course blogs positively affect the density and connectedness in learning networks (Wild & Sigurðarson, 2011). This study utilized social network analysis (SNA) to investigate human-human interaction and the development of social connections on this blogging platform. Next, Latent Dirichlet Allocation (LDA) topic model method was applied to understand human-information interaction during different developmental stages of Pressible. This study provides an exploratory examination of four developmental stages of an online learning community in a school blogging system.

2. METHODOLOGY
2.1. Participants and Data Collection
The data were collected from the entire Pressible database and contained 3598 users and 594 sites, with 50422 posts in total. The specific aim of this study was to explore the social network and its association with content creation. Only the interactions between registered IDs were counted as valid connections. After the reconstruction of the database for SNA, there were 172 blogs with data on a total of 11146 connections and 429 interactive users.

2.2. Social Network Analysis
SNA is a method to analyze the connections, relationships, and interactions between individuals and communities in the collaborative social network, expressed as the node and edge diagrams (Wild, 2016; Slater et al., 2017). In this study, R package igraph (Csardi & Nepusz, 2006) constructs, modifies and calculates the social networks. Density measures the proportion of contacts observed between pairs of nodes in the network; Eigen centrality measures the importance of a node's network by weighting its top connecting nodes' indegree and outdegree centrality (Daniel, et al., 2010).

2.3. Latent Dirichlet Allocation Topic Modeling
To analyze the content of comments and posts in the blogs, LDA topic modeling was utilized to discover and infer the general topics by scanning the words and their distribution probabilities within documents (Blei, et al., 2003). The R package *tm* was used to construct the corpus for text mining. The *tm* package removes spaces, stop words, numbers, spaces, and punctuation, converting the words to lower case and roots to construct a term-document matrix, which allows analysis of individual words in the corpus (Feinerer & Hornik, 2015; Lang, 2017). The R packages *topicmodels and tidytext* were utilized to calculate the term frequency, construct the inverse document matrix, remove the uncommon terms, find the most common words for individual topics and group the documents by generated topics (Grün & Hornik, 2011; Lang, 2017; Silge & Robinson, 2017).

3. RESULTS AND DISCUSSIONS
3.1. Social Network Development
Descriptive statistics analysis on yearly data was conducted to show the general social network activity in Pressible by developmental stages (Tables 1). The results indicate that this blogging system shifted from a development stage (beginning to 2010 Summer), to a stable growth stage (2010 Fall to 2012 Summer), a rapid growth stage (2012 Fall to 2015 Summer), into a decline stage (2015 Fall until now). The active member numbers increased from the development stage to rapid growth stage and decreased in the decline stage. Their engagement rates as average connection numbers increased from development to the rapid growth stage, which also dropped at the decline stage. Therefore, the number of active members and their engagement rate determine the growth of this online social learning community. The density of the social network among active members decreased while the network was growing from 2011 to 2015 (Fig. 1), indicating that the network became decentralized as more active members joined. Most of the participants were students. They became less active in interactions on Pressible after graduation. New students joined the social network and formed new social centers. Thereby, the global social density decreased because of the dynamic student community (Fig. 1). As more professors built their courses on Pressible, more active students joined this online learning community for discussions and made meaningful connections. Recruiting more professors to take

advantages of Pressible for its online course creation features is a key to maintaining the rapid growth of this social network.

Table 1. Descriptive Statistics by Developmental Stage

Stage	Ave. conn. (/year)	Ave. active IDs (/year)	Ave. conn. per IDs (/year)	Top popular topic of the stage
Development	215.5	36.5	5.9	video game
Stable Growth	1333	89.5	14.9	teach and learn
Rapid Growth	2021	118.7	17.0	think and know
Decline	993	104	9.6	music performance

3.2. Most Influential Members and Topic Interaction Analysis by Developmental Stage

To determine the optimal number of topics of the whole Pressible database, the perplexity values of models were calculated. The LDA topic training model was constructed based on 10000 documents with the range of 2 to 50 topic numbers. The other 1146 documents are used to test the model with the calculation of perplexity and entropy. Based on the perplexity of testing data, 30 is the optimal topic number for this dataset.

During the developmental stage, the library staff was the most active members in the network. Their online discussions focused on the topics: "video game, education", indicating that library staff was using Pressible as a communication tool to share thoughts and discuss education media.

During the stable growth stage, a TC professor (ID: 1490) from the music education program built his courses on Pressible for three years (2011 to 2013), and he continuously received the highest eigen centrality score for three years. During the stable growth stage, the popular topics became focused on education. People who talked about "think and know" were also interested in "video game" at this stage.

During the rapid growth stage, the professor with ID 3132 brought new students into this blogging system though his courses *Creativity & Problem Solving in Music Education*. It was a course extended from the materials developed by the professor with ID 1490, with the same topic "read" and high-frequency words "music, read" for most of the posts. This was the pedagogy course to meet the New York State and national teacher preparation standards.

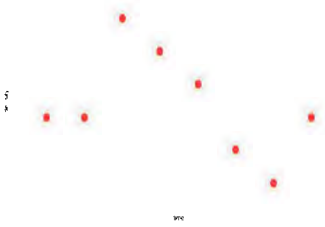


Figure 1. Social Network Density by Year.

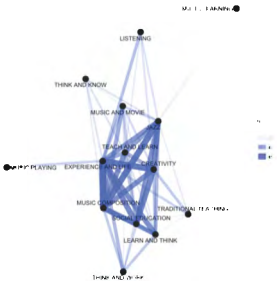


Figure 2. Topic Co-occurrence frequency in the rapid growth stage

Individuals' topic co-occurrence indicated a robust network in the rapid growth stage (Fig. 2). People talked about the topics of "creativity", "music composition", "Jazz", "social education", "learn and think", "experience and life" and "teach and learn" at high co-occurrence frequencies (above 30). In the decline stage, the topic co-occurrence network dropped in topic connection intensity which might be due to less active members in the overall network (Table 1). This finding indicated that more active members encouraged online discussions with more diverse topics. In course blogs, students extended discussion topics to the perspectives that they care about: "music learning, music playing, social education, creativity and experience and life", beyond the scope of the professor's set topic "read".

4. IMPLICATIONS

This study identifies and explores four developmental stages of the social network: development, stable growth, rapid growth, and decline. The SNA and topic model analysis results imply that the influential people will bring new communities into the social network by sharing the content of the hottest topics. Deliberately recruiting more influential people into the social network would accelerate its transition from the stable growth stage to the rapid growth stage.

5. REFERENCES

[1] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. Journal of Machine Learning Research, 3(Jan), 993-1022

[2] Csardi, G., & Nepusz, T. (2006) The igraph software package for complex network research, InterJournal, Complex Systems 1695. 2006.

[3] Daniel, M., Messing, S., Nowak, M., & Westwood, S. J. (2010) Social Network Analysis Labs in R. Stanford University

[4] Feinerer, I., & Hornik, K. (2015). tm: Text Mining Package. R package version 0.6-2.

[5] Grün, B., & Hornik, K. (2011). "topicmodels: An R Package for Fitting Topic Models." Journal of Statistical Software, 40(13), pp. 1-30

[6] Kizilcec, R. F., Pérez-Sanagustín, M., & Maldonado, J. J. (2017). Self-regulated learning strategies predict learner behavior and goal attainment in Massive Open Online Courses. *Computers & Education*, *104*, 18-33.

[7] Lang, C. (2017) HUDK 4051: Learning Analytics: Process and Theory. Columbia University. New York

[8] Rosé, C. P., Carlson, R., Yang, D., Wen, M., Resnick, L., Goldman, P., & Sherer, J. (2014). Social factors that contribute to attrition in MOOCs. In Proceedings of the first ACM conference on Learning (pp. 197-198). ACM

[9] Siemens, G. (2004). Connectivism: A learning theory for the digital age. elearnspace. Retrieved December 12, 2007, CHI '00. ACM, New York, NY, 526-531

[10] Silge, J., and Robinson, D. (2017) "Text Mining with R: A Tidy Approach" O'Reilly Media

[11] Slater, S., Joksimovic, S., Kovanovic, V., Baker, R., & Gasevic, D. (2017) Tools for Educational Data Mining: A Review. Journal of Educational and Behavioral Statistics. 2017, Vol. 42, No. 1 p85-106

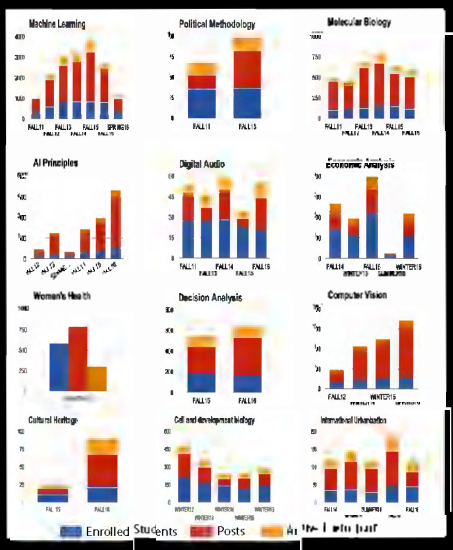
[12] Vygotsky, L. (1978). Mind and society: The development of higher psychological processes. Cambridge, MA: Harvard University Press.

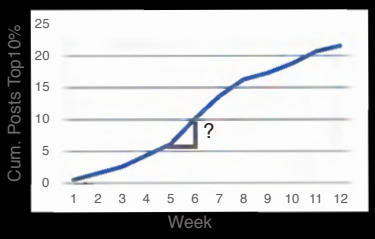
[13] Wang, X., Wen, M., & Rosé, C. P. (2016, April). Towards triggering higher-order thinking behaviors in MOOCs. In Proceedings of the Sixth International Conference on Learning Analytics & Knowledge (pp. 398-407). ACM

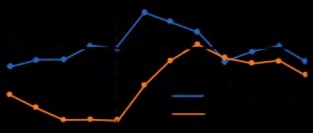
[14] Wild, F., & Sigurðarson, S. E. (2011). Simulating learning networks in a higher education blogosphere–at scale. In European Conference on Technology Enhanced Learning (pp. 412-423). Springer Berlin Heidelberg

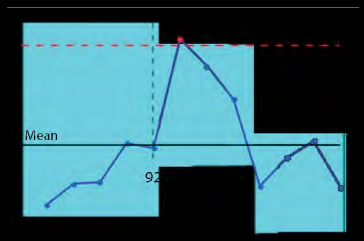
[15] Wild, F. (2016). Learning analytics in R with SNA, LSA, and MPIA. Springer.

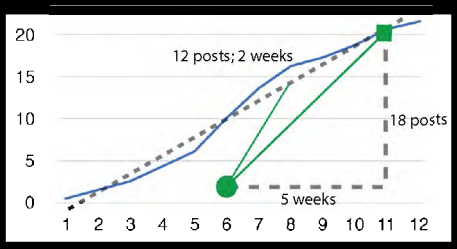
[16] Zhou Z. (2013) Connecting Teacher Bloggers: Unleashing the Educational Power of Wordpress

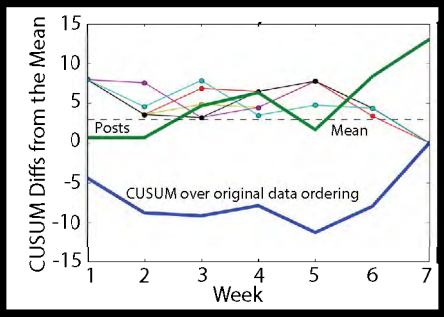

Machine Learning
Political Methodology
Molecular Biology
AI Principles
Digital Audio
Economic Analysis
Women's Health
Decision Analysis
Computer Vision
Cultural Heritage
Cell and developmental biology
Enrolled Students
Posts


Cum. Posts Top10%
25
20
15
10
5
0
?
1 2 3 4 5 6 7 8 9 10 11 12
Week




Mean
92


20
15
10
5
0
12 posts; 2 weeks
18 posts
5 weeks
1 2 3 4 5 6 7 8 9 10 11 12


CUSUM Diffs from the Mean
15
10
5
0
-5
-10
-15
Posts
Mean
CUSUM over original data ordering
1 2 3 4 5 6 7
Week

Untangling The Program Name Versus The Curriculum: An Investigation of Titles and Curriculum Content

R. Wes Crues
University of Illinois
Dept. of Educational Psychology
1310 South Sixth Street
Champaign, Illinois
crues2@illinois.edu

ABSTRACT

This investigation focuses on the relationship between skills taught during business programs and whether the skills taught relate to the title of the program, as deemed by subject-matter experts. We hone-in on formal degree and non-degree programs in small business education, entrepreneurship education, or a blend of these two to determine if the *name* of the program is related to the *skills taught* in said program. We use a collection of excerpts from college catalogs, which are all descriptions of the formal academic programs. We then use $k-$means clustering to group program descriptions into interpretable clusters. We discuss the findings from the cluster analysis.

Keywords

text mining, clustering, higher education, business education

1. INTRODUCTION

Major academic disciplines are typically collections of finer-grained specialties; for example, a computer science department might consist of experts in human-computer interaction, artificial intelligence, algorithm design, among others. Colleges likely have departments with similar names, but we want to understand if similarly named degree programs at different universities equip students with similar skills. To discern whether or not this task is tractable, we used a collection of program descriptions from college catalogs about programs claiming to teach students entrepreneurship, small business, or a blend between these two curriculum areas. These definitions are used throughout:

- A *program description* is at least one, but often composes a few paragraphs, which delineates skills taught in programs, and might provide some learning goals and a listing of courses;
- *Entrepreneurship* is defined as " trying to identify opportunities and putting useful ideas into practice" [1] (p. 6);
- and, *small business management* is "the ongoing process of owning and operating an established business" [3] (p. 28).

Our study explores whether we can use text clustering to identify a clear distinction between these two areas of business education, determine if there are differences between two-year and four-year programs, and whether there are differences between degree and non-degree programs.

2. METHOD

A research team manually assembled a collection of 697 program descriptions from college catalogs for institutions located in the United States. Research assistants went to college websites and manually extracted text from published college catalogs online. The initial list of programs was derived from the 2013 Integrated Postsecondary Education Data System (IPEDS) maintained by the United States Department of Education. After filtering institutions which did not have any business programs, a random sample of programs arrived at the collection used.

Program descriptions spanned programs focusing in entrepreneurship, small business management, or a blend of the two. Additional program descriptions were collected which were considered special focus programs; these were programs which teach a specific skill set on operating a business (examples include funeral home management to hair weaving and braiding entrepreneur). We also considered formal degree (e.g., associates and bachelor degrees) or non-degree programs (e.g, certificates or specializations), and whether the home institution is public or private, for-profit or not-for-profit, and whether the institution is a 2-year, 4-year, or 4-year and beyond institution [5]. Table 1 presents the distribution of program labels and whether the program is a degree or non-degree program.

Table 1: Distribution of Program Descriptions

Program Label	N	Degree/Non-Degree
Entrepreneurship	444	247/197
Small Business & E-ship	82	42/40
Small Business	79	20/59
Special Focus	92	34/58

2.1 Preprocessing Program Descriptions

Program descriptions were transformed into raw text format, tokenized into unigrams, except for a few words. A few bigrams and trigrams were specified using knowledge from a domain-expert, for example, business plan(s), social entrepreneurship, home based business, and venture capital. Punctation, numbers, and top words were removed using the pre-defined English stop word list in the "tm" package in R [2]. We used stemmed words by using the Porter stemming algorithm [6]. We used binary indicators to determine whether a term was present in each program description when constructing the document-term matrix [4].

2.2 Corpus Statistics

Our initial document-term matrix contained 7799 unique terms with a sparsity of 99%. We removed very frequent terms deemed to have no substantive value by a domain expert. Due to the nature of the corpus (i.e., program descriptions), words such as catalog, college, semester, requirements, and introduction, among others, were excluded. Eventually, we used the "removeSparseTerms" function in the "tm" package in R [2], which resulted in a document-term matrix with 16 unique terms, however, still 70% sparse.

2.3 Program Description Clustering

We utilized $k-$means because this clustering technique was favored in prior studies [7]. We experimented with various numbers of centroids, and after discussions with domain experts, we determined $k = 10$ was an optimal solution. The domain expert believed this solution provided an interpretable and reasonable grouping of programs. Specifically, the distribution of whether the program was an entrepreneurship, small business, a blend of these, or a special focus program, coupled with their expectations of distribution of formal degree programs versus non-degree programs. More than ten centroids resulted in clusters containing less than five documents, while less than ten resulted in a solution which did not provide what domain experts believed to be the most interpretable.

3. RESULTS

Five of the clusters exhibited a focus on teaching entrepreneurship in the context of having an idea, creating a start-up, with the intention of scaling the business into a large enterprise. Within these clusters, two clusters had words indicating programs might teach entrepreneurship to equip students to solve global problems and health concerns. Words indicating entrepreneurship might be taught to professionals in fields besides business (i.e., law and engineering) appeared in one cluster. One cluster appeared to teach general business skills, without a clear focus on entrepreneurship or small business. Another cluster contained special focus programs, which seek to prepare students for a specialized, technical career, such as a travel agent or carpenter. Two clusters contained small business programs, where one focused on keenly on running ones' own business, while the other included this while teaching students to innovative. One cluster contained very detailed program descriptions from one institution.

4. DISCUSSION & CONCLUSIONS

We found the definition of entrepreneurship which pertains to creating and expanding new enterprise appeared to be almost exclusively in four-year colleges, especially research universities. In contrast, small business management and operating a small business were taught almost exclusively at two-year colleges. A few of the two-year colleges also had many specialized programs in applied fields, such as the cosmetology; these types of programs were nearly exclusive to two-year colleges. Another element of entrepreneurship is creativity and innovation. These skills, specifically innovation, seemed to be taught primarily in the four-year sector. The programs that considered themselves a blend tend to focus more on small businesses than entrepreneurship. We found innovation and these skills to be taught more in degree. On the other hand, skills related to managing a small business were in non-degree programs.

From our findings about entrepreneurship and small business education, we generally found labels of programs match the skills one would expect to learn given the name of the program. However, one cluster in our analyses did not indicate skills in the targeted areas were being specifically taught. A limitation of our study is program descriptions vary in length and detail, which might be problematic for clustering. Our further work plans to consider whether skills taught have changed over time; for example, are skills being taught today the same skills taught a decade ago?

5. ACKNOWLEDGMENTS

The author would like to acknowledge two domain experts, Dr. Cindy Kehoe and the late Dr. Paul Magelli for their expertise in entrepreneurship education. Their advice about contextual meaning of results was invaluable in interpreting these analyses. The author would also like to acknowledge the Ewing Marion Kauffman foundation, which funded this work through a grant to inventory and interpret entrepreneurship education in higher education in the United States.

6. REFERENCES

[1] B. R. Barringer and R. D. Ireland. *Entrepreneurship: Successfully launching new ventures*. Pearson, Upper Saddle River, New Jersey, fourth edition, 2012.

[2] I. Feinerer, K. Hornik, and D. Meyer. Text mining infrastructure in r. *Journal of Statistical Software*, 25(5):1–54, 2008.

[3] T. S. Hatten. *Small business management: Entrepreneurship and beyond*. Houghton Mifflin Company, Boston, Massachusetts, fourth edition, 2009.

[4] P. Howland and H. Park. Cluster-preserving dimension reduction methods for efficient classification of text data. In M. W. Berry, editor, *Survey of Text Mining*, pages 3–24. Springer Science+Business Media, 2004.

[5] National Center for Education Statistics. *IPEDS Glossary*, 2017.

[6] M. F. Porter. An algorithm for suffix stripping. *Program: Electronic Library and Information Systems*, 40(3):211–218, 2006.

[7] M. Steinbach, G. Karypis, V. Kumar, et al. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, volume 400, pages 525–526. Boston, 2000.

Emerging Patterns in Student's Learning Attributes through Text Mining

Kejkaew Thanasuan
Learning Institute
King Mongkut's University of Technology, Thonburi[1]
(+662) 470-8395
kejkaew.tha@kmutt.ac.th

Warasinee Chaisangmongkon
Institute of Field Robotics
King Mongkut's University of Technology, Thonburi[1]
(+662) 470-9716
warasinee.cha@kmutt.ac.th

Chanikarn Wongviriyawong
Institute of Field Robotics
King Mongkut's University of Technology, Thonburi[1]
(+662) 470-9717
chanikarn@fibo.kmutt.ac.th

ABSTRACT
Text mining has been used in various fields including education. Using unsupervised sentiment analysis combined with a clustering algorithm, we discovered 2 emerging clusters of learning characteristics (traditional (T) and experiential (E)), and correlations among learning attitudes such as motivation, peer relationship and positive attitude. We found a positive correlation between social learning and peer relationship ($p<0.005$), but negative between social learning and negative attitude ($p<0.05$) in E. Social learning was positively correlated with positive attitude ($p<0.001$) in T.

Keywords
Text mining, clustering algorithm, sentiment analysis, motivation, engagement

1. INTRODUCTION
Studies have shown that attitudes are related to motivation, engagement and outcome in learning. When learners have positive attitude, they would spend more time engaging in learning [5, 9]. Difference in students with positive attitude and motivation in e-learning settings was observed [6]. Students with boredom have poorer learning outcome than those with frustration [1]. Hence, sentiment analysis could be used to harness learning attitudes.

Recently, machine learning methods in natural language processing have become prevalent, while there are many training datasets for supervised learning algorithms. However, the task of opinion mining without such dataset can be a challenge. We combined one symbolic technique for an unsupervised machine learning with clustering algorithm to discover emerging patterns among texts written in *Thai* that could reflect student's learning attitudes. Our findings demonstrated how such approach could be useful in exploring and understanding relationships among learning attitudes.

2. METHODS
2.1. Data Acquisition
Our subjects were 83 freshman undergraduate students (M:F = 62:21) (average age = 17.2) in Robotics and Automation Engineering, at King Mongkut's University of Technology Thonburi. They consented to participate in the study. This data set was collected while students were taking same classes. Students wrote in *Thai* about what they learned each week for all 14 weeks.

2.2. Data Analyses
We used an open source Lexitron dictionary (NECTEC, 2006) as word database in *Thai* and an open source algorithm Lexto (NECTEC, 1994) to tokenize texts into longest words possible. We had 383 entries. On average, each entry had 124.3 words.

Word frequency was calculated for each student as the ratio of the number of times each unique word appeared in any learning journal and the total number of words appeared. Irrelevant words (prepositions, conjunctions, and generic verbs and nouns) or words that appeared less than 20 times in all entries were filtered out. Negation and irrealis phenomena, out-of-topic sentences, or irony and sarcasm were not treated in our analysis. We performed several clustering algorithms on the distance matrix with various initial conditions and different number of clusters (2, 3, or 4) to determine if any pattern of word clusters could emerge.

Among frequently-used words, instructors chose words that represented these six attitudes: 1) positive relationship with others (Peer relationship), 2) desire to improve oneself (Motivation), 3) positive emotion (Positive attitude), 4) negative emotion (Negative attitude), 5) engagement in learning on one's own (Solitary learning), and 6) engagement in learning that involves others (Social learning). The associated words were also evaluated by another group of students to indicate levels of congruity of each attitude[2]. The results are shown in Table 1. We calculated a student's attitude score to be the sum of percentage of word frequency for each word associated with each of the 6 attitudes. Pearson correlation coefficient and p-value of the correlation were computed between any two attitudes. Correlation analyses were performed independently for each cluster.

3. RESULTS AND DISCUSSION
We found that 2 clusters emerged, yielding the most consistent set of words. The first cluster contained words such as take exams, read books, problem sets, formula, lessons, math, writing, calculus, physics, language, etc. The second contained words such as human being, people, work, see, team, fun, talk, play, like,

[1] King Mongkut's University of Technology, Thonburi's address: 126 Pracha Uthit Rd, Bang Mot, Thung Khru, Bangkok 10140 Thailand

[2] The data were collected from 28 native Thai speakers (average age = 20.18). They were asked to rate how each pair of words and an attitude was meaningfully or semantically related (e.g. Peer Relationship vs. Group) in a 5-point Likert scale.

group, together, etc. The first cluster was labelled T for traditional and the second, E for experiential. Although initial conditions and clustering algorithms were varied, these two clusters emerged.

Table 1. Words Associated with 6 attitudes and their rating[3] (mean score and standard deviation in parentheses)

Attitude	Associated Words	Rating
Peer Relationship	group, talk, help, together, team, help each other, we, everyone, etc.	3.87 (0.48)
Motivation	improve, practice, better, goals, development, improvement, etc.	3.76 (0.4)
Positive Attitude	fun, enjoy, like, happy, funny, good, excited, etc.	3.64 (0.37)
Negative Attitude	stressed, confused, sleepy, slow, difficult, do not understand, etc.	3.01 (0.45)
Solitary Learning	exams, formula, scores, books, grades, study, responsibility, etc.	3.34 (0.77)
Social Learning	hands on, experiment, project, communication, participate, etc.	3.71 (0.31)

Motivation was positively correlated with solitary learning ($R=0.4$ (T) and 0.55 (E); $p<0.05$). It could mean that for T, when one desires to improve oneself, one engages in learning even on one's own. Our result supports a previous finding that motivation and engagement were correlated [3, 10]. Such correlation for E might be because when one enjoys learning with others, their motivation increases. Previous studies showed that people who reported feeling happy were engaged in social activities more often and that sociability was a strong predictor of life satisfaction [2, 7].

Additionally, for E, motivation was positively correlated with social learning ($R=0.42$, $p<0.05$); social learning was positively correlated with peer relationship ($R=0.6$, $p<0.005$), but negatively correlated with negative attitude ($R=-0.44$, $p<0.05$). For T, social learning was positively correlated with positive attitude ($R=0.55$, $p<0.001$). Relationships with peers are very important in helping learners become adaptive in different learning environments [8]. Previous studies showed that students with positive peer relationship were likely to be engaged in academic tasks and perform better in school than students without positive peer relationships [11, 12, 13]. Our finding supports existing literature that learning abilities are related to attitude of learners [5].

However, our approach has some limitations. Our algorithm is a simple frequency counting. However, since less frequently used words have been filtered out, we expected that our results would still be robust even with different weighting methods. Moreover, no sarcasm, negation or irrealis phenomena were considered. This might have a slight effect on our results.

Future work involves testing robustness of our approach with more data. To explore additional emergence, we could also apply adjustments to various clustering algorithms [4]. We are developing a platform to help teachers quantify student's attitudes.

4. REFERENCES

[1] Baker, R. S., 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.

[2] Costa, P. T. and McCrae, R. R. 1980. Influence of extraversion and neuroticism on subjective well-being: happy and unhappy people. *Journal of personality and social psychology*. 38, 4 (Apr. 1980), 668-678.

[3] Hsieh, T. L. 2014. Motivation matters? The relationship among different types of learning motivation, engagement behaviors and learning outcomes of undergraduate students in Taiwan. *Higher Education*. 68, 3, 417-433.

[4] Li, G. and Liu, F. 2012. Application of a clustering method on sentiment analysis. *Journal of Information Science*. 38, 2, 127-139.

[5] McMillan, J. H. 1977. The effect of effort and feedback on the formation of student attitudes. *American educational research journal*. 14, 3, 317-330.

[6] Moshinskie, J. 2001. How to keep e-learners from escaping. *Performance Improvement*. 40, 6, 30-37.

[7] Robinson, J. P. and Martin, S. 2008. What do happy people do?. *Social Indicators Research*. 89,3, 565-571.

[8] Rubin, K. H., Bukowski, W., and Parker, J. G. 1998. Peer interactions, relationships, and groups. *Handbook of child psychology*. 3, 5, 619-700.

[9] Sanderson, H. W. 1976. Student attitudes and willingness to spend time in unit mastery learning. *Research in the Teaching of English*. 10,2, 191-198.

[10] Walker, C. O., Greene, B. A., and Mansell, R. A. 2006. Identification with academics, intrinsic/extrinsic motivation, and self-efficacy as predictors of cognitive engagement. *Learning and Individual Differences*. 16, 1, 1-12.

[11] Wentzel, K. R. 2005. Peer relationships, motivation, and academic performance at school. In *Handbook of competence and motivation,* A. J. Elliot and C. S. Dweck, Eds. Guilford Press, New York, 279-296.

[12] Wentzel, K. R., Barry, C. M., and Caldwell, K. A. 2004. Friendships in Middle School: Influences on Motivation and School Adjustment. *Journal of educational psychology*. 96, 2 (Jun. 2004), 195-203.

[13] Wentzel, K. R. and McNamara, C. C. 1999. Interpersonal relationships, emotional distress, and prosocial behavior in middle school. *The Journal of Early Adolescence*, 19, 1, 114-125.

[3] For rating, a five-point score means strongly agree and an one-point score means strongly disagree.

A Neural Network Approach to Estimate Student Skill Mastery in Cognitive Diagnostic Assessments

Qi Guo, Maria Cutumisu, Ying Cui
Department of Educational Psychology, University of Alberta
{qig, cutumisu, yc}@ualberta.ca

ABSTRACT

In computer-based tutoring systems, it is important to assess students' mastery of different skills and provide remediation. In this study, we propose a novel neural network approach to estimate students' skill mastery patterns. We conducted a simulation to evaluate the proposed neural network approach and we compared the neural network approach with one of the most widely used cognitive diagnostic algorithm, the DINA model, in terms of skill estimation accuracy and the ability to recover skill prerequisite relations. Results suggest that, while the neural network method is comparable in skill estimation accuracy to the DINA model, the former can recover skill prerequisite relations more accurately than the DINA model.

Keywords

prerequisite discovery, skills, neural network, student modeling, cognitive diagnosis model

1. INTRODUCTION

In intelligent tutoring systems, assessing students' skill mastery patterns and determining skill prerequisite relationship are two important areas of research. Various approaches are proposed to solve these two problems, including Educational Data Mining (EDM) approaches, such as Bayesian Knowledge Tracing, Learning and Performance Factor Analysis (for a comparison see [5]), and psychometric approaches, such as Cognitive Diagnostic Models (CDMs) [2, 6]. Compared to CDMs, which assess student skill mastery based on their responses to a test administered at one time point (i.e., no learning occurs during the test), the EDM approaches have the advantage of assessing student learning dynamically. However, unlike CDMs, which estimate every item's psychometric properties, the EDM approaches often assume all test items that measure the same set of skills have the same psychometric properties (e.g., same guessing and slipping parameters). This assumption is unlikely to be tenable in practice, and it may lead to less accurate skill estimation and less efficient item selection. While both approaches have their strengths and weaknesses, this study will focus on developing a new CDM approach using the neural networks, and evaluate the proposed approach by comparing it with the current most popular CDM method, the DINA (deterministic inputs, noisy "and" gate) model [2] using simulated data.

2. A BRIEF INTRODUCTION TO NEURAL NETWORKS

A neural network is a supervised classification algorithm that consists of several layers of neurons (i.e., processing units) [4]. Each neuron linearly combines information from previous layers and applies a non-linear *activation* function. The most commonly-used activation function is the logistic/sigmoid function. A typical feedforward neural network consists of a layer of hidden units and a layer of output units. Mathematically, it can be represented as:

$$Y_{n,q} = sigmoid\left(\vec{1}_{n,1}\vec{b}'_{1,q} + sigmoid(\vec{1}_{n,1}\vec{b}'_{1,k} + X_{n,p}W_{p,k})W_{k,q}\right),$$

where $Y_{n,q}$ is the output matrix consisting of n subjects' values on q output variables, $X_{n,p}$ is the input matrix consisting of n subjects' values on p input variables, $\vec{b}'_{1,k}$ is a vector of intercept values for k hidden units, $W_{p,k}$ is the weight matrix between p input variables and k hidden units, $\vec{b}'_{1,q}$ is a vector of intercept values for q output units, and $W_{k,q}$ is the weight matrix between k hidden units and q output units.

One challenge in applying neural networks to estimate students' skill mastery patterns is that students' skill mastery patterns are unobserved. Thus, we only have observed values for the input variables (students' item response patterns) but not for the output variables (students' skill mastery pattern).

3. METHODOLOGY: THE PROPOSED NEURAL NETWORK APPROACH

To overcome the problem mentioned above, we propose a novel neural network model that has the same input and output (i.e., students' item response patterns). The core idea underlying our approach is to first reduce the input (student item response patterns) to a smaller number of hidden units representing students' latent skills and then use these hidden units to best reproduce student item response vectors (i.e., output) with the restriction of the Q-matrix, a matrix that specifies the set of skills measured by each item. A conceptual diagram of the proposed network is shown in **Figure 1**.

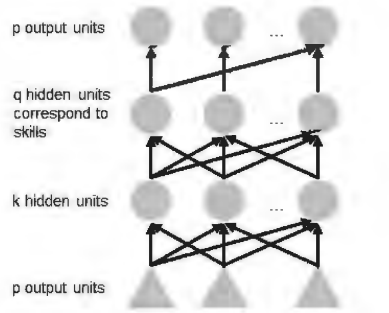


Figure 1. A diagram of the proposed neural network. Relations between skill hidden units and output units are specified based on the Q-matrix.

It is important to note that the relation between the second layer of hidden units and output units is specified based on the Q-matrix, which specifies which skills are required by each item. Intuitively, the network first extracts features from student item response patterns and then it dictates the relations between features and student item response patterns based on the Q-matrix. Mathematically, the model can be represented as follows:

$$Y_{n,p} = sigmoid\big(\vec{1}_{n,1}\vec{b}'_{1,p} + sigmoid\big(\vec{1}_{n,1}\vec{b}'_{1,q} + sigmoid(\vec{1}_{n,1}\vec{b}'_{1,k} + X_{n,p}W_{p,k})W_{k,q}\big)W_{q,p} \odot Q'_{q,p}\big),$$

where $\odot$ represents elementwise multiplication, and $Q'_{q,p}$ is the Q-matrix.

Similar to a regular neural network, the proposed model uses maximum likelihood to define the cost function and it can be optimized using some variants of gradient descent (e.g., rprop [4]). To speed up the optimization, it is important to choose meaningful starting values for the weight matrices. To initialize $W_{q,p}$, we can first train a multivariate logistic regression with all the theoretically possible skill patterns (i.e., expected theoretical plausible skill patterns) as input, and their corresponding expected item response patterns (i.e., item response pattern assuming no slips and guesses) as output, assuming slipping and guessing parameters are 0. Then, we use the weight matrix from this multivariate logistic regression as the starting values of the proposed neural network.

4. EVALUATION

In order to demonstrate the accuracy of the proposed neural network, we conducted a preliminary simulation study. Five thousand students' responses (correct/incorrect) to 28 test items were generated based on a skill prerequisite model shown in Figure 2. Skill prerequisite relations, true model used in the simulation (left); recovered using DINA skill estimates (middle) and neural network skill estimates (right)

To evaluate the recovered prerequisite relationship, we counted the number of estimated causal links that were not in the true model, and the number of missing causal links that were in the true model.

and a Q-matrix (available upon request). The guessing and slipping parameters for all items were set to 0.1. We compared the proposed method with the DINA model in terms of accuracy of 1) student skill pattern estimates and 2) skill prerequisite relation recovery. Accuracy of skill pattern estimates is defined as:

$$accuracy = 1 - \frac{|estimated\ skill\ pattern\ matrix - true\ skill\ pattern\ matrix|}{n * q},$$

where n is the sample size, and q is the number of skills in the Q-matrix. The skill prerequisite relations were recovered by using a Bayesian network to model the relations among estimated student skills. The causal direction in the Bayesian network is determined by the following heuristic [1]:

If P(skill1=0)<P(skill2=0), then skill1 is the prerequisite of skill2.

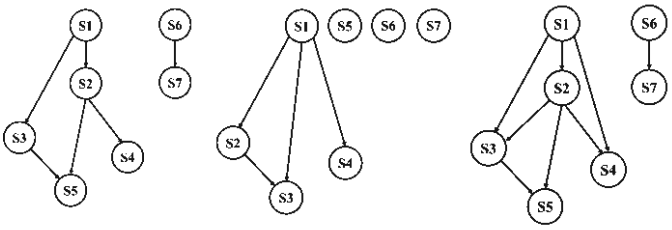


Figure 2. Skill prerequisite relations, true model used in the simulation (left); recovered using DINA skill estimates (middle) and neural network skill estimates (right)

To evaluate the recovered prerequisite relationship, we counted the number of estimated causal links that were not in the true model, and the number of missing causal links that were in the true model.

We programed our proposed neural network using Python. The number of hidden units in the first layer was set to 56. The number of hidden units in the second layer was set to seven, corresponding to seven skills in the Q-matrix. The Rprop algorithm was used to optimize the neural network. For the DINA analysis, we used the *CDM* R package [6]. For the Bayesian network analysis, we used the *bnlearn* R package's mmhc algorithm [7] and *Rgraphviz* R package [3].

The results suggested that the proposed method had similar or slightly better accuracy (89.2%) at estimating skill patterns than the DINA model (87.9%). Moreover, the proposed method was better at recovering the skill prerequisite relations. The recovered skill prerequisite relations by the DINA model and the proposed method are shown in Figure . The prerequisite relations recovered based on the DINA skill estimates only contained two arcs from the true model (i.e., S1 to S2, S1 to S3), and they contained two arcs that were not in the true model (S1 to S4, S2 to S3). The prerequisite relations recovered based on the neural network skill estimates contained all the arcs from the original model, as well as two arcs that were not in the true model (S1 to S4, S2 to S3). Overall, the results suggested that the proposed network had slightly better skill estimation accuracy than the DINA model and it was more accurate at recovering skill prerequisite relations than the DINA model.

5. CONCLUSIONS AND DISCUSSION

This study proposed a novel neural network approach to estimate student skill mastery patterns in CDM. Traditionally, parameter estimation of models with latent variables usually depends on Expectation Maximization or Markov Chain Monte Carlo methods. The proposed neural network approach frames the latent variable model problem as a supervised problem and it solves it using the gradient descent method. Initial evidence suggests that the proposed method has comparable skill estimation accuracy as the DINA model, but it can recover skill prerequisite relations better than the DINA model. Further research is needed to rigorously evaluate this method.

6. REFERENCES

[1] Chen, Y., Gonzlez-Brenes, J. and Tian, J. 2016. Joint Discovery of Skill Prerequisite Graphs and Student Models. In *International Conference on Educational Data Mining*. Raleigh, NC, IEDMS, 46-53.

[2] de La Torre, J. 2009. DINA Model and Parameter Estimation: A Didactic. *Journal of Educational and Behavioral Statistics, 34*, 1, 115-130.

[3] Gentry, J., Long, L., Gentleman, R., Falcon, S., Hahne, F., Sarkar, D. and Rgraphviz, K. H. 2009. Provides plotting capabilities for R graph objects. *R package version, 2*, 0.

[4] Goodfellow, I., Bengio, Y., Courville, A. 2016. *Deep Learning*. MIT Press.

[5] Pavlik, P. I., Cen, H., Koedinger, K.R. 2009. Performance factors analysis - A new alternative to knowledge tracing. *Proceedings of the 2009 conference on artificial intelligence in education: building learning systems that care: from knowledge representation to affective modelling* (Brighton, 2009), 531-538.

[6] Robitzsch, A., Kiefer, T., George, A. and Uenlue, A. 2014. *CDM: Cognitive diagnosis modeling.R package version 3.1-14*, .

[7] Scutari, M. bnlearn: Bayesian Network Structure Learning, R package version 2.7 (2011). *URL* http://www.bnlearn.com.

Automatic Peer Tutor Matching: Data-Driven Methods to Enable New Opportunities for Help

Nicholas Diana
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
ndiana@cmu.edu

Michael Eagle
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
meagle@cs.cmu.edu

John Stamper
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
john@stamper.org

Shuchi Grover
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
shuchi.grover@sri.com

Marie Bienkowski
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
marie.bienkowski@sri.com

Satabdi Basu
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
satabdi.basu@sri.com

ABSTRACT

The number of students that can be helped in a given class period is limited by the time constraints of the class and the number of agents available for providing help. We use a classroom-replay of previously collected data to evaluate a data-driven method for increasing the number of students that can be helped. We use a machine learning model to identify students who need help in real-time, and an interaction network to group students who need similar help together using approach maps. By assigning these groups of struggling students to peer tutors (as well the instructor), we were able to more than double the number of students helped.

Keywords

Introductory Programming; Learning Analytics; Machine Learning; Peer Tutors; Educational Data Mining

1. INTRODUCTION

While a typical classroom may be full of students experiencing the same problem and students who have solved that problem, this expertise is rarely utilized. Instead, often the only source of help is the instructor, who is most likely unable to help all the students who need help within the time constraints of the class period. To address this problem, we propose and evaluate several methods for improving the efficiency of student assistance using machine learning.

Diana et al. [1] showed that low-level log data from the Alice introductory programming environment can be used to accurately predict student grades, and that they could increase the number of students helped by matching struggling students to a peer tutor based on the similarity of their code. A subsequent study [2] found that the accuracy and interpretability of the previously reported predictive model could be improved by increasing the grain size of the features from a vocabulary of terms derived through natural language processing (NLP) to small snippets of code. We explore how this improvement impacts peer tutor matching and the efficiency of providing help more generally. Additionally, we use an interaction network graph to test if students who may benefit from the same kind of help can be grouped together, increasing the efficiency of the instructor or peer tutor.

2. METHODS

The data used in the current study were originally collected by Werner et al. [3] as part of a two year project exploring the impact of game design and programming on the development of computer science skills. The students were asked to complete an assessment task called the *Fairy Assessment*. The current experiment closely follows the data transformation methodology reported in [1] to convert raw log data into program representations called *code-states* and the code-state complexity reduction methodology reported in [2] to reduce code-states to smaller, *code-chunks*.

We used ridge regression to predict students' grades. We compared two methods for generating the features inputted into the regression. In the first method, features were a vocabulary of NLP terms generated from the students' code-states. In the second method, each code-state was first converted into a list of code-chunks, and then into a *chunk-frequency vector*. A chunk-frequency vector is a vector whose length is equal to the total number of features being considered in the model. Each value in the vector corresponds to the frequency of the respective code-chunk.

The predicted grades were also used to estimate which students need help and which students may be able to provide help. We call the students classified as needing help using their actual grades *low-performing students*. This classification serves as the ground-truth that we use to evaluate our predictive model. In a real world implementation, we would not have access to the actual grades, so we must estimate them and use those estimates to classify students as need-

ing help. If a student's predicted grade was in the bottom quartile, and they have not been helped or are not currently being helped ("helped" status persists across time), then that student was added to the group of students who still need to be helped, which we call the *Help Pool*. If a student's predicted grade was in the top quartile, and they are not currently helping a student, then that student was added to the group of students who may be able to help other students, which we call the *Tutor Pool*. For each student in the *Help Pool*, we first checked to see if the instructor was available to help. If so, the instructor was assigned to that student. If the instructor was unavailable (i.e., helping another student), then we searched for a peer tutor. We used a network graph of each code-state (or code-chunk frequencies) for each user to match tutees to tutors. We searched for tutors who shared a common ancestor node (i.e., shared a previous program state) with the tutee. These tutors were added to a pool of potential tutors. From that pool we selected the tutor with the common ancestor node that was closest (i.e., least number of steps away) to the tutee's current node. The same method applied if segmenting was used, except that instead of matching the instructor or peer tutor to one student, the instructor or tutor was matched to a segment of students with a similar problem.

2.1 Efficiency Index

While the primary goal of our previous work [1] was to evaluate how well our model could correctly classify students who would go on to have a low final grade (low-performing students), the primary goal of the current experiment is to evaluate how efficient this intervention would be. That is, we were interested in what percentage of those low-performing students could be helped, and how we can maximize that percentage. We call this ratio the *Efficiency Index* (EI), and define it formally as:

$$EI = \frac{LowPerformingStudentsHelped/BeingHelped}{LowPerformingStudents} \quad (1)$$

The EI can be further broken down into the percentage of low-performing students helped by the instructor (EI_I) and the percentage of low-performing students helped by peer tutors (EI_{PT}).

3. RESULTS

We compared models using a linear mixed model with the measure of interest as the dependent variable, model as a fixed effect, and time bin as a random effect.

We hypothesized that we can use low-level programming data to group similar low-performing students together so that they can be helped as a group. To test this, we first replicated our previously reported model to use as a baseline measure. Then, we generated a new model that incorporated segmenting. Both models used NLP features in a ridge regression and an interaction network graph built using code-states as nodes. We found that the EI (M=0.467, SD=0.210) of the model that incorporated segmenting was significantly higher ($p<.001$) than the baseline model (M=0.305, SD=0.190).

We also hypothesized that using the presence or absence of code-chunks as model features would improve the performance of the model. To test this, we generated a model using a sample of the code-chunks from our previous work that were shown to be good predictors of learning outcomes [2]. We generated a model using these 16 code-chunk features (rather than the NLP-derived terms used in the baseline model), and found that this code-chunk model had a significantly lower ($p<.001$) RMSE (M=0.246, SD=0.064) than the baseline model (M=0.263, SD=0.073).

Finally, we hypothesized that a network graph generated using code-chunks as nodes would lead to greater coverage and a higher EI. To test this, we generated a model using the same 16 code-chunks described above as features in the regression. A network graph was also generated to incorporate segmenting. However, instead of each node corresponding to a code-state, each node corresponded to a chunk-frequency vector. Representing nodes as chunk-frequency vectors more than doubled the coverage (coverage=0.924) compared to the network graph generated using code-states (coverage=0.374). The EI of the model using chunk-frequency vectors to generate the network graph (M=0.813, SD=0.128) also had a significantly higher ($p<.001$) EI than the model using code-states (M=0.428, SD=0.217).

4. CONCLUSIONS

In this paper, we explored a method for increasing the amount of help given in a typical class period. Our previous work demonstrated that we can use a predictive model to accurately identify students who may need help. We built off of this work in two ways. First, we improved the accuracy of the predictive model by using more relevant features. Second, we drastically increased the number of students able to be helped from, on average, 3.72 to 9.92 by grouping low-performing students together to be helped as a group (in combination with better model features). These results suggest that using low-level log data to group and match low-performing students to peer tutors may be an effective way to increase the amount of help given in a classroom.

5. REFERENCES

[1] N. Diana, M. Eagle, J. Stamper, S. Grover, M. Bienkowski, and S. Basu. An instructor dashboard for real-time analytics in interactive programming assignments. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*, LAK '17, pages 272–279, New York, NY, USA, 2017. ACM.

[2] N. Diana, M. Eagle, J. Stamper, S. Grover, M. Bienkowski, and S. Basu. Data-driven generation of rubric parameters from an educational programming environment. Submitted.

[3] L. Werner, J. Denner, and S. Campe. The Fairy Performance Assessment : Measuring Computational Thinking in Middle School. *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education - SIGCSE '12*, pages 215–220, 2012.

Short-Answer Responses to STEM Exercises: Measuring Response Validity and Its Impact on Learning

Andrew Waters
OpenStax, Houston, TX
aew2@rice.edu

Phillip Grimaldi
OpenStax, Houston, TX
pjg3@rice.edu

Andrew Lan
Princeton University,
Princeton, NJ
andrew.lan@princeton.edu

Richard Baraniuk
Rice University, Houston, TX
richb@rice.edu

ABSTRACT

Educational technology commonly leverages multiple-choice questions for student practice, but short-answer questions hold the potential to provide better learning outcomes. Unfortunately, students in online settings often exhibit little effort when crafting short-answer responses, instead often produce off-topic (or invalid) responses that are off-topic and do not relate to the question being answered. In this study, we consider the effect of entering on-topic short-answer response on student learning and retention. To do this, we first develop a machine learning method to automatically label student open-form responses as either valid or invalid using a small amount of hand-labeled training data. Then, using data from several high school AP Biology and Physics classes, we present evidence that providing valid short-answer responses creates a positive educational benefit on later practice.

Keywords

Best educational practices, Cognitive psychology, Machine learning, Natural language processing, Mixed effect modeling

1. INTRODUCTION

An important part of the learning process is recalling learned information from memory [3]. In most educational situations, this practice is accomplished by asking students practice questions related to the learning material. In online learning, multiple-choice questions are by far the most common, following by short-answer questions. While multiple choice questions are attractive due to the ease of machine scoring, it is worth asking whether is is the best option for improving learning. Indeed, multiple-choice questions are oft-criticized because they are perceived to require only shallow recognition processes to complete [7]. Short-answer responses, by contrast, are generally believed to have a stronger learning benefit to students as they afford more difficult reconstructive cognitive processes.

Prior experiments examining the relative benefits of multiple-choice and short-answer have been mixed, with short-answer questions generally found to improve learning only when subsequent feedback is provided [2, 4]. One factor that has not been examined in prior research, however, is how the quality of short-answer responses provided by students contribute to learning. In online educational settings where students lack oversight, students do not always take the time to craft thoughtful short-answer responses. Instead, they often opt to to quickly enter an off-topic response to advance their progress or view feedback.

We hypothesize that students derive greater learning benefits when they produce valid short-answer responses than when they do not, even when those valid responses are incorrect. While it is possible to hand-label student responses as valid or invalid for a small number, it is not feasible to do this at large scale. To circumvent this scalability issue, we devise a machine-learning based classifier trained on a small number of hand-labeled exemplars. We then leverage this classifier to analyze the impact of entering valid responses on learning.

2. AUTOMATIC VALIDITY CLASSIFICATION

Due to the large number of words in student responses, our method for automatically classifying student short-answer responses as valid or invalid begins with parsing to reduce the overall size of the feature space. First, we attempt simple spelling correction for each word of a student's response. Following spelling correction, which strip common stopwords (e.g. of, as, is, etc) and replace any nonsensical words (e.g., random keyboard presses) with a specially defined tag, which has the effect of mapping all unknown words to the same label. Finally, we stem acceptable words in a student responses to further reduce the dimensionality of our feature space. Finally, we convert the parsed student response to a numerical feature vector using a bag-of-words model.

Following parsing, we employ a random forest [1] to classify each student response as either valid or invalid. We measured the performance of our method using 5-fold cross-validation on $20,000$ hand-labeled responses and found our accuracy to be 95%.

3. ANALYSIS OF VALID RESPONSES ON LEARNING

We now turn our attention to evaluating the impact of providing valid short-answer responses on future learning outcomes using real-world educational data.

Our dataset is taken from a pilot study of our online learning platform, OpenStax Tutor [6], which was conducted during the 2015–2016 academic year. OpenStax Tutor has two important features relevent to our discussion. First, it uses a hybrid answering format [7] that first requires students to enter a short-answer response to the question and requires the student to select the correct answer from a multiple-choice list. Second, OpenStax Tutor employs a concept known as spaced practice, which automatically assigns questions to students on material that they have learned in previous

assignments. The purpose of this feature is to ultimately improve long-term knowledge retention, but we leverage these spaced practice observations as an opportunity to observe the effects of entering valid short-answer responses on later practice.

The pilot consisted of two separate high school courses, AP Biology and standard (non-AP) Physics. A total of 207 students (74 AP Biology, 154 Physics) and 8 instructors (4 AP Biology, 4 Physics) participated in the pilot. There are roughly 100,000 short-answer responses on initial practice problems, and 20,000 of these answers were hand-labeled by subject matter experts as being valid or invalid responses to the given question. The average spaced practice problem occurs roughly 3 weeks after the initial practice on the topic is complete.

To analyze the impact of entering valid open-form responses we adopt a mixed effect logistic regression model [5]. Our binary outcome is whether or not the student answered the spaced practice question for a given topic correctly. Our random effects (R) are nuisance quantities for student ability, topic difficulty, and instructor quality. We examine two different fixed effects in our model: M, the number of multiple-choice questions that a student answered correctly on a given topic and V, the number of valid short-answer responses that a student provided on a given topic.

We consider four separate models for student success on spaced practice questions. Each model includes the random-effects R. We then separately consider the effects of the fixed effects M and V as well as considering both fixed effects jointly. We fit all four models to the AP Biology and Physics datasets separately. The results for AP Biology and Physics are shown on Table 1 and Table 2, respectively. In order to determine which model provided the best fit, we used the Akaike information criterion (AIC) metric, which imposes a penalty that penalizes modes with too many parameters to prevent overfitting. Models with lower AIC values are deemed better than models with higher AIC values.

For AP Biology, we found that the $R+V$ model achieved the lowest AIC implying that the number of valid responses provided a better predictor of success than the number of correct multiple-choice selections. The coefficient for the number of valid responses is positive and statistically significant, which matches our hypothesis that more valid responses improves student retention. For Physics, we note that $R+M+V$ provides the lowest AIC value, and is significantly better than considering $R+M$ alone. This implies that both factors together produce better modeling fitting.

Table 1: Summary of AP Biology Data Models

	Dependent variable:			
	Correct on Spaced Practice			
	(R)	$(R+M)$	$(R+V)$	$(R+M+V)$
Number Core Correct		0.030^{*}		-0.009
		(0.016)		(0.027)
Number Core Valid			0.034^{**}	0.040^{*}
			(0.013)	(0.023)
Constant	0.613^{***}	0.467^{***}	0.427^{***}	0.437^{***}
	(0.075)	(0.107)	(0.105)	(0.109)
Observations	1,987	1,987	1,987	1,987
Log Likelihood	$-1,278.010$	$-1,276.102$	$-1,274.653$	$-1,274.599$
Akaike Inf. Crit.	2,562.019	2,560.203	2,557.305	2,559.199
Note:				*p<0.1; **p<0.05; ***p<0.01

Table 2: Summary of Physics Data Models

	Dependent variable:			
	Correct on Spaced Practice			
	(R)	$(R+M)$	$(R+V)$	$(R+M+V)$
Number Core Correct		0.082^{***}		0.076^{***}
		(0.013)		(0.013)
Number Core Valid			0.097^{***}	0.078^{***}
			(0.023)	(0.022)
Constant	0.002	-0.316^{***}	-0.105	-0.377^{***}
	(0.074)	(0.087)	(0.079)	(0.089)
Observations	4,000	4,000	4,000	4,000
Log Likelihood	$-2,703.761$	$-2,682.312$	$-2,693.697$	$-2,675.836$
Akaike Inf. Crit.	5,413.522	5,372.623	5,395.394	5,361.672
Note:				*p<0.1; **p<0.05; ***p<0.01

4. CONCLUSIONS

We have developed a machine-learning based method for classifying student open-form responses to questions as being either valid (on-topic) or invalid (off-topic) using a combination of intelligent parsing and supervised classification. We have further presented evidence that students who spend time crafting thoughtful responses show improved learning outcomes when practicing earlier material.

The results that we have derived in this work are the result of searching for patterns in existing data and relied on students deciding of their own volition whether or not to enter a valid short-answer response. Future research in this area will involve more highly controlled study in which the opportunity to enter a short-answer response will be controlled by our learning system. This will allow us greater control over our experimental setup and aid in the interpretation of our final result.

5. ACKNOWLEDGMENTS

Thanks to the Art Ciocca, the Laura and John Arnold Foundation, and John and Ann Doerr for supporting this research. Thanks also to Micaela McGlone, Debshila Basu Malik, and Alicia Chang for their help in conducting the pilot studies, preparing the data, and many helpful discussions regarding this work.

6. REFERENCES

[1] T. K. Ho. Random decision forests. In *Proc. 3rd Intl. Conf. Document Analysis and Recognition*, volume 1, pages 278–282. IEEE, 1995.

[2] S. Kang, K. McDermott, and H. Roediger. Test format and corrective feedback modify the effects of testing on long-term retention. *European J. Cognitive Psychology*, 19:528–558, 2007.

[3] J. Karpicke and P. Grimaldi. Retrieval-based learning: A perspective for enhancing meaningful learning. *Educational Psychology Review*, 24:401–418, 2012.

[4] J. L. Little, E. L. Bjork, R. A. Bjork, and G. Angello. Multiple-choice tests exonerated, at least of some charges: Fostering test-induced learning and avoiding test-induced forgetting. *Psychological Science*, 23:1337–1344, 2012.

[5] C. E. McCulloch and J. M. Neuhaus. *Generalized Linear Mixed Models*. Wiley Online Library, 2001.

[6] OpenStaxTutor. https://openstaxtutor.org/, 2017.

[7] J. Park. Constructive multiple-choice testing system. *British Journal of Educational Technology*, 41(6):1054–1064, 2010.

Using an Additive Factor Model and Performance Factor Analysis to Assess Learning Gains in a Tutoring System to Help Adults with Reading Difficulties

Genghu Shi
University of Memphis
The Institute for Intelligent Systems
365 Innovation Dr.
Memphis, TN, 38152
1 001 901 438 8934
gshi@memphis.edu

Philip Pavlik, Jr
University of Memphis
The Institute for Intelligent Systems
365 Innovation Dr.
Memphis, TN, 38152
1 001 901 678 2326
ppavlik@memphis.edu

Arthur Graesser
University of Memphis
The Institute for Intelligent Systems
365 Innovation Dr.
Memphis, TN, 38152
1 001 901 240 4795
art.graesser@gmail.com

ABSTRACT
After developing an intelligent tutoring system (ITS), or any other class of learning environments, one of the first questions that should be asked is whether the system was effective in helping students learn the targeted skills or subject matter. In this study, we employed two educational data mining models (Additive Factor Model, AFM and Performance Factor Analysis, PFA) which are available in Datashop (LearnSphere) to assess the learning gains on 5 theoretical levels of adults. With AFM, for the KC models tested, the results showed positive learning gains for the Rhetorical Structure knowledge component in contrast, for the PFA model, adults did not learn from either successes or failures.

Keywords
Learning gains, Theoretical Levels, Additive Factor Model, Performance Factor Analysis, CSAL Autotutor

1. INTRODUCTION
One of the first questions that is asked after developing an intelligent tutoring system (ITS) is whether the system was effective in helping students learn the targeted skills or subject matter. Learning gains are based on the performance of the students as they work on the system over time with many opportunities for learning. These learning gains can be assessed at a fine-grained level by tracking the learning of specific knowledge components (KCs), which are particular skills, strategies, concepts, or facts, as articulated in the Knowledge-Learning-Instruction (KLI) framework [2]. In this paper, we analyze the learning of the theoretical components (KCs) which were based on models of comprehension that adopt a multilevel framework in our dialogue-based intelligent tutoring system, called CSAL AutoTutor, that was designed to help struggling adult readers learn reading comprehension strategies. The Graesser and McNamara framework identifies 5 levels [1]: words (W), syntax (S), the explicit textbase (TB), the referential situation model (SM), the discourse genre and rhetorical structure (RS, the type of discourse and its composition). And, the computational models used in the analysis were Additive Factor Model (AFM) and Performance Factor Analysis, both of which were from Datashop (LearnSphere) [3]. 3 questions will be addressed in this paper: 1. When training the adults to read, did the performance of the adults follow the levels of text difficulty? 2. Did adults' learning gains increase after using the Autotutor which just provided some instructions on reading comprehension strategies and some practice? 3. Did adults learn from successes or failures?

2. METHODOLOGY
The adult readers were 52 adults in Atlanta and Toronto who participated in a study of 100 hours of intervention that was conducted by the CSAL team, and they completed up to 30 lessons throughout the intervention. Each lesson had between 10 and 30 multiple choice questions to assess their performance When they answered a question incorrectly, they were given a hint to see whether they selected correctly among the two remaining options. However, in this analysis we only considered performance on their first type, not the follow-up.

The original measures in the AFM model included performance, practice opportunities (the number of questions they answered in a lesson), the knowledge components (KCs were the 5 theoretical components), and subject (participant). For model fitting, pre-test scores and text difficulty (easy, medium, and hard) were entered into the original models (Table 1). Ultimately, we ran 10 models (5 AFM models and 5 PFA models) for the KC approaches, and determined which AFM and PFA models had the best performance, based on AIC, BIC, and Loglikelihood.

Table 1. Models Construction by Adding New Variables

Models	Variables
Model 1	Pre-test score
Model 2	Pre-test score, Text Difficulty
Model 3	Pre-test score, Text Difficulty: KC Model
Model 4	Pre-test score, Practice Opportunity: KC Model
Model 5	Pre-test score, Text Difficulty: Practice Opportunity: KC Model

** These models are basically logit mixed effect models. The ":" refers to interactive effect.*

3. RESULTS AND DISCUSSION
Analyses of the 10 models consistently showed that model 3 was the best model, yielding the lowest AIC BIC and Loglikelihood scores.

Both Table 2 (AFM results) and Table 3 (PFA results) confirm the obvious expectation that pretest score is a strong predictor of adults' performance. Also, only for Rhetorical Structure, performance decreased as a function of text difficulty. This is consistent with the Graesser and McNamara's multilevel

theoretical framework that distinguishes the deeper discourse levels of processing (such as the Situation Model and Rhetorical Structure) from the basic reading levels (such as Words and Syntax) [1]. As shown in table 2, only for Rhetorical Structure, performance significantly got better as the practice opportunity increased, but the case of the other KCs was different. As shown in table 3, although cumulative correctness had significant interactions with Syntax and Situational Model, while cumulative incorrectness had significant interactions with Syntax and Textbase, the estimates of these interactions were all negative, which indicated that the performance got worse, no matter adults experienced more successes or failures on these KCs. And, for other KCs, the coefficients drifted to 0.

Table 2. AFM Output of Model 3 – Theoretical Levels

	Estimate	SE	Z Score	P-value	Sig.
Intercept	0.675	0.25	2.66	0.01	**
Pre-test Score	0.140	0.03	4.97	0.00	***
PO : RS	0.001	0.00	2.27	0.02	*
PO : S	-0.124	0.02	-5.16	0.00	***
PO : SM	-0.003	0.00	-3.69	0.00	***
PO : TB	-0.016	0.00	-4.98	0.00	***
PO : W	-0.004	0.00	-0.95	0.34	
RS : Hard	-1.805	0.19	-9.73	0.00	***
S : Hard	0.822	0.28	2.94	0.00	**
SM : Hard	-0.111	0.18	-0.62	0.54	
TB : Hard	0.014	0.19	0.07	0.94	
W : Hard	-0.204	0.30	-0.69	0.49	
RS : Medium	-1.241	0.18	-7.07	0.00	***
S : Medium	-0.078	0.26	-0.30	0.77	
SM : Medium	-0.035	0.18	-0.20	0.84	
TB : Medium	0.133	0.19	0.71	0.48	
W : Medium	0.529	0.29	1.84	0.07	.

**PO refers to practice opportunity. RS refers to Rhetorical Structure. S refers to Syntax. SM refers to Situational Model. TB refers to Textbase. W refers to Word. Easy, Medium, Hard are three levels of text difficulty.*

Table 3. PFA Output of Model 3 – Theoretical Levels

	Estimate	SE	Z Score	P-value	Sig.
Intercept	0.671	0.26	2.60	0.01	**
pretest	0.145	0.03	4.87	0.00	***
CC : RS	0.000	0.00	-0.12	0.91	
CC : S	-0.127	0.04	-3.47	0.00	***
CC : SM	-0.005	0.00	-2.32	0.02	*
CC : TB	-0.008	0.01	-1.30	0.19	
CC : W	-0.004	0.01	-0.69	0.49	
CI : RS	0.005	0.00	1.37	0.17	
CI : S	-0.123	0.04	-3.14	0.00	**
CI : SM	0.001	0.00	0.41	0.68	
CI : TB	-0.031	0.01	-2.77	0.01	**
CI : W	-0.002	0.02	-0.13	0.90	
RS : Hard	-1.808	0.19	-9.74	0.00	***
S : Hard	0.828	0.37	2.22	0.03	*
SM : Hard	-0.099	0.18	-0.55	0.58	
TB : Hard	-0.069	0.20	-0.35	0.73	
W : Hard	-0.209	0.30	-0.69	0.49	
RS : Medium	-1.248	0.18	-7.10	0.00	***
S : Medium	-0.079	0.27	-0.29	0.77	
SM : Medium	-0.023	0.18	-0.13	0.90	
TB : Medium	0.068	0.19	0.35	0.72	
W : Medium	0.524	0.30	1.77	0.08	.

**CC and CI refer to cumulative correctness and cumulative Incorrectness. Others are the same as Table 2.*

4. CONCLUSIONS

The model comparison revealed that practice opportunity, adults' prior literacy skills, KC model (theoretical levels) and text difficulty were factors influencing adults' performance. From the interactions between theoretical levels and text difficulty, we can draw the conclusion that adults' performance on Rhetorical Structure and Situational Model matched the difficulty levels of the texts used in the lessons of the two KCs, that is, they did better on easy texts and worse on medium and hard texts. But for the basic reading levels (Word, Syntax, and Textbase), situations were different. According to the results of AFM model, the learning gains on deeper discourse levels of processing (Rhetorical Structure) increased, because adults' performance became better when they continuously got practice opportunities. There were no learning gains observed on KCs like Situational Model, Syntax, Textbase, and Word. From results of PFA model, we didn't observe significant learning gains from either successes or failures.

5. ACKNOWLEDGMENTS

This research was supported by the National Center of Education Research (NCER) in the Institute of Education Sciences (IES) (R305C120001) and the National Science Foundation Data Infrastructure Building Blocks program under Grant No. (ACI-1443068).

6. REFERENCES

[1] Graesser AC, Mcnamara DS, Kulikowich JM (2011) Coh-Metrix providing multilevel analyses of text characteristics. Educational researcher 40:223-234

[2] Koedinger KR, Corbett AT, Perfetti C (2010) The knowledge-learning-instruction (KLI) framework: Toward bridging the science-practice chasm to enhance robust student learning. Cognitive Science

[3] Pavlik Jr PI, Cen H, Koedinger KR (2009) Performance Factors Analysis--A New Alternative to Knowledge Tracing. Online Submission

Identifying Student Communities in Blended Courses

Niki Gitinabard, Collin F. Lynch, Sarah Heckman, Tiffany Barnes
North Carolina State University
Computer Science Department
Raleigh, NC, US
{ngitina, cflycnh, sarah_heckman, tmbarnes}@ncsu.edu

ABSTRACT
Blended courses have become the norm in post-secondary education. Universities use large-scale learning management systems to manage class content. Instructors deliver readings, lectures, and office hours online; students use intelligent tutors, web forums, and online submission systems; and classes communicate via web forums. These online tools allow students to form new social networks or bring social relationships online. They also allow us to collect data on students' social relationships. In this paper we report on our research on community formation in blended courses based on online forum interactions. We found that it was possible to group students into communities using standard community detection algorithms via their posts and reply structure and that the students' grades are significantly correlated with their closest peers.

Keywords
Educational Data Mining, Graph data mining, Social Networks, Blended Courses

1. INTRODUCTION
Improvements in technology have facilitated new models of student and instructor engagement. Students now supplement the traditional course structure with online materials. Instructors can share class material online, have an online discussion forum, or make quizzes and homework submissions online. This in turn provides a wealth of new data on student behaviors that we can use to study students' social relationships. In particular it allows us to study the impact of these social ties on course outcomes.

In prior work Brown et. al. showed that students in MOOCs form pedagogically-relevant, and homogeneous social networks. Brown et. al. has shown that students can be clustered into stable communities based upon their pattern of online questions and replies [1]. They have also shown that students' final grades are significantly correlated with those of their closest peers and community group. They have also shown that these communities, while homogeneous in terms of performance, are not united by their incoming motivations for enrolling in the course nor for their prior experience level [2].

To date these results have only been found in MOOCs where the user forum represents students' primary connection to one-another, and almost all relevant course interactions occur online. Students in blended courses, by contrast, often have preexisting social ties that carry over from prior courses at the same institution. In this paper we show that while forum interactions are not the only means of communication between students, they still define the same communities as was found in MOOCs and that the students' final grades are significantly correlated with those of their community members.

2. DATASET INFORMATION
In this paper we report on studies of three distinct courses, "Discrete Math-2013", "Discrete Math-2015" and "Java Programming Concepts-2015". All three are undergraduate computer science courses, offered at NC State and include significant blended components. Discrete Math-2015 and Java Programming Concepts-2015 occurred contemporaneously during the Fall 2015 semester while Discrete Math-2013, a previous offering of Discrete Math-2015, was offered in Fall 2013.

3. METHODS
3.1 Defining Social Interactions
Each node in our social networks represents an individual participant in the class. In the first class anonymous posting was allowed, so we have an unknown user related to all the anonymous posts. Social relationships are represented as arcs. We define a social relationship based upon direct and indirect replies in the user forum. Our method was similar to that of Brown et. al. [2]. We defined an edge between A and B if B replied to a thread after A had done so. This interaction can include starting the original thread, replying with a follow-up, or posting a feedback on a reply. We then aggregate these edges to form a weighted graph containing arcs for all of the relations. We assume that anyone who posts on a thread has read the prior comments before doing so. Thus it defines a form of social interaction between the participants as the students are expressly choosing to make a public reply to one another. For the purposes of the present analysis we included only students in our network and thus

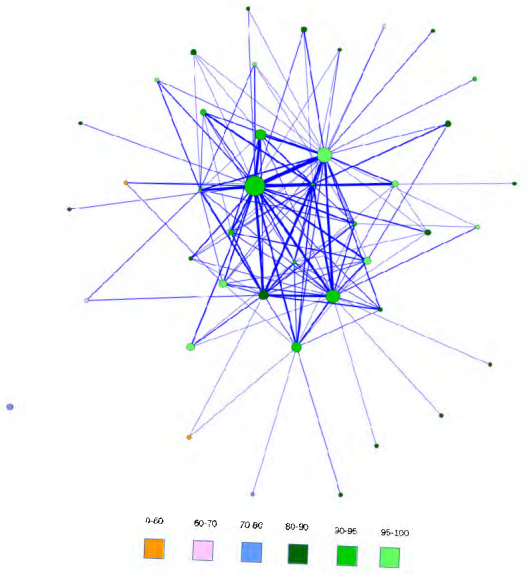


Figure 1: Communities generated on Discrete Math 2013 class

confined our social relationships to between-student connections.

3.2 Graph Analysis

For each of the graphs we generated, we removed the isolated vertices and performed clustering using the method described in [2, 1]. Our clustering method is an iterative process where we evaluate the modularity of graphs with an increasing number of clusters until we find a limit point where the modularity almost stops growing, which indicates the *natural cluster number*. After finding the natural number, on each iteration we generated the clusters via the Girvan-Newman edge-centrality algorithm[3]. On each iteration the algorithm removes the most central edge and and repeats until a set of k disjoint clusters has been produced. We then assessed whether or not the grade distributions in different clusters are significantly different by calculating the Kruskal-Wallis (KW) correlation between cluster assignment and grade. Kruskal-Wallis is a nonparametric analogue to the more common ANOVA test [4].

4. RESULTS

In graphs generated for Discrete Math 2014, we found that the graph reaches its natural cluster number at 42. We performed the Girvan Newman clustering and the resulting clusters can be seen in Figure 1. In this graph, each node represents a community, the size of the nodes shows the number of members and the color shows their average grade. We can observe that the KW correlation between cluster number and the grades is statistically significant ($p = 0.044 < 0.05$), which is similar to the results in MOOCs.

Our results show that, for Discrete Math 2015 ($p = 0.004 < 0.05$) and Java Programming Concepts 2015 ($p = 0.015 < 0.05$) graphs, there is a similar significant KW correlation between student grades and their communities.

5. DISCUSSION, CONCLUSIONS AND FUTURE WORK

In this paper, we generated a social graph between students in three different blended courses based on forum interactions. We found that similar to MOOCs, communities are formed in these graphs whose members tend to have similar grades. This is consistent with prior work which indicates that student communities on forum may be used to predict course outcomes [1, 2].

Having access to these social graphs can help instructors to identify the communities formed among students which can be used to find the students who need more help earlier. Our research does not show causality. Thus more research is needed to find out whether being in the communities makes their grades similar, or students are just likely to interact with others who are more like them. If we find out that the community membership has an effect on students' performance, we can use this information to identify isolated or poorly-performing groups early in the course and intervene by encouraging them to make contact with better students or seek help as a group.

There has been much work done on how forum interactions in MOOCs, being a hub in a social network or how being at the center of the graph could affect students' performance. We can use these graphs to conduct more research on which interaction levels will lead to better grades.

In further work we plan to address whether or not we can identify other types of social ties in blended courses, since the communications are more complicated.

6. ACKNOWLEDGMENTS

This work was supported by NSF grant #1418269: "Modeling Social Interaction & Performance in STEM Learning" Yoav Bergner, Ryan Baker, Danielle S. McNamara, & Tiffany Barnes Co-PIs.

7. REFERENCES

[1] R. Brown, C. Lynch, M. Eagle, J. Albert, T. Barnes, R. S. Baker, Y. Bergner, and D. S. McNamara. Good communities and bad communities: Does membership affect performance? In *Proceedings of the 8th International Conference on Educational Data Mining, EDM 2015, Madrid, Spain, June 26-29, 2015*, pages 612–613, 2015.

[2] R. Brown, C. Lynch, Y. Wang, M. Eagle, J. Albert, T. Barnes, R. S. Baker, Y. Bergner, and D. S. McNamara. Communities of performance & communities of preference. In *EDM (Workshops)*, 2015.

[3] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[4] W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

Automatic Scoring Method for Descriptive Test Using Recurrent Neural Network

Keiji Yasuda
KDDI Research
Garden Air Tower, 3-10-10,
Iidabashi, Chiyoda-ku, Tokyo
102-8460 Japan
ke-yasuda@kddi-research.jp

Izuru Nogaito
KDDI Research
2-1-15, Ohara, Fujimino city,
Saitama, 356-8502 Japan
iz-nogaito@kddi-research.jp

Hiroyuki Kawashima
KDDI Research
Garden Air Tower, 3-10-10,
Iidabashi, Chiyoda-ku, Tokyo
102-8460 Japan
hi-kawashima@kddi-research.jp

Hiroaki Kimura
KDDI Research
Garden Air Tower, 3-10-10,
Iidabashi, Chiyoda-ku, Tokyo
102-8460 Japan
ha-kimura@kddi-research.jp

Masayuki Hashimoto
KDDI CORPORATION
Garden Air Tower, 3-10-10,
Iidabashi, Chiyoda-ku, Tokyo
102-8460 Japan
mu-hashimoto@kddi.com

ABSTRACT

In this paper, we propose an automatic evaluation method for the descriptive type test. The method is based on Recurrent Neural Networks trained on a non-labeled language corpus and manually graded students' answers. The experimental results show that the proposed method is the second best result among five conventional methods, including BLEU, RIBES, and several sentence-embedding methods. And, the proposed method gives the best performance among several sentence embedding methods.

Keywords

RNN, LSTM, Language Model, Essay Scoring

1. INTRODUCTION

Twenty-first-century skills are advocated in the educational field. Compared to traditional knowledge-based education evaluated by multiple-choice tests, the evaluation of twenty-first-century skills is very difficult. A descriptive test is one solution to the problem, although the cost of scoring is prohibitive. In this paper, we propose a method to automatically score descriptive type tests to solve the problem stated above. The method uses long short-term memory (LSTM) recurrent neural networks (RNN) to score the answers written in natural language. The method requires two kinds of data sets.

One is a large language corpus used for pre-training of RNN. As pre-training, the RNN-based language model is trained using the corpus. A vector given by a hidden layer in the networks is thought to embed the meaning of processed sentences. Thus, the proposed method calculates the similarity between two vectors given by processing model answers and student answers on RNN. The other data set is a small labeled corpus that consists of model answers, student answers, and manually annotated scores of student answers. The labeled corpus is used for training of the RNN.

2. PROPOSED METHOD

The RNN framework used in the paper is shown in Fig. 1. As shown in the figure, the proposed method uses two kinds of corpora and two kinds of training parts. They are the pre-training of word embedding and the main training of the LSTM-type RNN [3].

Here, we express the sentence ($\mathbf{s}$) as the sequence of words $\mathbf{s} = w_1, \cdots, w_t, \cdots, w_T$. The word-embedding part projects the input word of time t (w_t) to high-dimension vector $x_{w_t} \in \mathbb{R}^{d_w}$ as follows:

$$\mathbf{x}_{w_t} = \mathbf{E}^{\mathrm{T}} \mathbf{w}_{w_t} \tag{1}$$

where $w_{w_t} \in \mathbb{R}^{|V|}$ is the one-hot vector of w_t and $\mathbf{E} \in \mathbb{R}^{|V| \times d_w}$ is the lookup table. $x_{w_t}$is used as the input for the LSMT part. The LSTM consists of four components: the forget gate ($\mathbf{f}_t$), input gate ($\mathbf{i}_t$) and output gate ($\mathbf{o}_t$), and the memory state ($\mathbf{c}_t$). These real-valued vectors are calculated by the following formulas:

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{x}_{w_t} + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f), \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_{w_t} + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i), \\
\mathbf{o}_t &= \sigma(\mathbf{W}_i \mathbf{x}_{w_t} + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o), \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_{\tilde{c}} \mathbf{x}_{w_t} + \mathbf{U}_{\tilde{c}} \mathbf{h}_{t-1} + \mathbf{b}_{\tilde{c}}), \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t
\end{aligned} \tag{2}$$

where $\mathbf{W}$ and $\mathbf{U}$ are weight matrices, and $\mathbf{b}$ is the bias vector. $\sigma(\cdot)$ and $\tanh(\cdot)$ are an element-wise sigmoid function

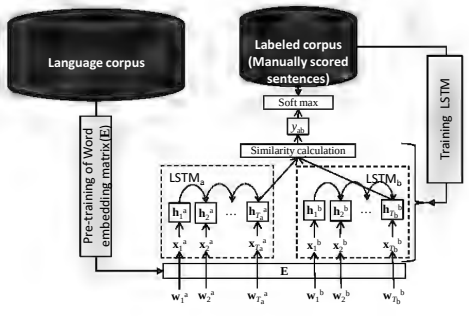


Figure 1: Framework of the proposed method.

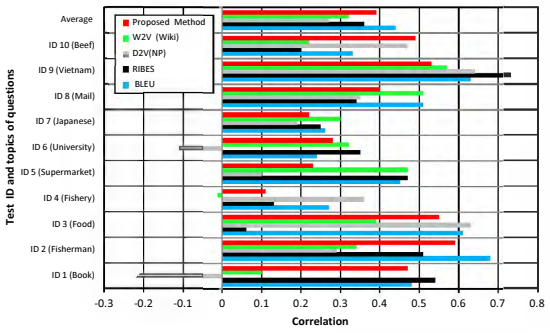


Figure 2: Experimental results.

and a hyperbolic tangent function, respectively. Using these vectors, hidden-layer vector ($\mathbf{h}_t \in \mathbb{R}^{d_s}$) is calculated as follows:

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{3}$$

where $\odot$ is element-wise multiplication. The main training part requires a labeled corpus that consists of model answers, the students' answers, and manually scored results of the students' answers. By using the labeled corpus, the second training part tunes the LSTM whose network configuration was proposed by Mueller et al. [1]. Using pre-trained word-embedding matrix $\mathbf{E}$ from the first training part, LSTM parameters are trained as follows.

First, randomly initialize LSTM parameters in Eq. 2. Then, duplicate the initialized LSTM ($\mathrm{LSTM_a}$ and $\mathrm{LSTM_b}$ in Fig. 1). One of them is used to process the student's answer and the other is used to process the model answer. We regard the hidden-layer vector of the sentence end as sentence embedding. To calculate the sentence similarity between the student's answer and the model answer, we add a new unit between the hidden layers. The unit calculates the L1 norm based on the similarity between the two sentence embeddings ($\mathbf{h}^{\mathrm{a}}_{T_{\mathrm{a}}}$ and $\mathbf{h}^{\mathrm{b}}_{T_{\mathrm{b}}}$ in Fig. 1) by using the following formula [1]:

$$\begin{aligned} g(\mathbf{h}^{\mathrm{a}}_{T_{\mathrm{a}}}, \mathbf{h}^{\mathrm{b}}_{T_{\mathrm{b}}}) &= \exp(-\|\mathbf{h}^{\mathrm{a}}_{T_{\mathrm{a}}} - \mathbf{h}^{\mathrm{b}}_{T_{\mathrm{b}}}\|_1) \\ &= \exp\left(-\sum_{i=1}^{d_s} \left|h^{\mathrm{a}}_{T_{\mathrm{a}}} - h^{\mathrm{b}}_{T_{\mathrm{b}}}\right|\right) \end{aligned} \tag{4}$$

The similarity calculation is performed only when both sentence pairs have been processed by the LSTM. Using the similarity calculated by Eq. 4 and the manually evaluated score, the deviation is back propagated to tune the LSTM weights. Here, we restrict the parameters of $\mathrm{LSTM_a}$ and $\mathrm{LSTM_b}$ to the same values.

3. EXPERIMENTS

The labeled corpus consists of 10 descriptive type questions and their answers. For each question, around 20 answers are manually scored. Additionally, there are also four model answers for each question. For the pre-training of the word-embedding matrix, we use a Mainichi newspaper corpus.

Since the size of the labeled corpus is very small, we carry out a leave-one-out cross-validation test for each question. The cross-validation is carried out only for student answers. The same model answers are used for training and evaluation. The LSTM in the paper can only process a pair of one student answer and one model answer at the same time. Thus, all combinations of student answers and model answers in the training set are used for training. For the scoring of the test set, we calculate the average score of several model answers. The evaluation measure is the correlation coefficients between the manual and the automatic scoring results.

Fig. 2 shows the experimental results. As baseline results, we show the results of BLEU, RIBES, and the Doc2Vec (D2V) cosine similarity method with the NewsPaper(NP) corpus and Wikipedia(Wiki) corpus by referring to the conventional research[2]. As shown in the figure, the proposed method never gives a negative correlation coefficient. Meanwhile the conventional sentence-embedding-based methods give negative correlation coefficients. Additionally, the proposed method gives the best results on average among sentence-embedding methods, which are two kinds of D2V and the proposed method. Compared to all methods, the proposed method offers the second-best performance.

4. CONCLUSIONS AND FUTURE WORKS

We proposed the LSTM-based automatic scoring method for descriptive tests. We carried out experiments using actual learning logs. According to the experimental results, the proposed method gives the best performance among several sentence-embedding methods, and the second-best results among five methods including BLEU and RIBES.

5. ACKNOWLEDGMENTS

This work used model answers, students' answers, and scoring data forms from the Lojim School. (http://lojim.jp/).

6. REFERENCES

[1] J. Mueller et al. Siamese recurrent architectures for learning sentence similarity. In Proc. of AAAI, pages 2786–2792, 2016.

[2] I. Nogaito et al. Study on automatic scoring of descriptive type tests using text similarity calculations. In Proc. of EDM, pages 616–617, 2016.

[3] M. Sundermeyer et al. LSTM neural networks for language modeling. In Proc. of Interspeech, pages 194–197, 2012.

Using Graph-based Modelling to explore changes in students' affective states during exploratory learning tasks

Beate Grawemeyer
Birkbeck, University of London
beate@dcs.bbk.ac.uk

Alex Wollenschlaeger
Birkbeck, University of London
awolle01@dcs.bbk.ac.uk

Sergio Gutierrez-Santos
Birkbeck, University of London
sergut@dcs.bbk.ac.uk

Wayne Holmes
The Open University, UK
wayne.holmes@open.ac.uk

Manolis Mavrikis
UCL Institute of Education
m.mavrikis@ucl.ac.uk

Alexandra Poulovassilis
Birkbeck, University of London
ap@dcs.bbk.ac.uk

ABSTRACT

We describe a graph-based modelling approach to exploring interactions associated with a change in students' affective state when they are working with an exploratory learning environment (ELE). Student-system interactions data collected during a user study was modelled, visualized and queried as a graph. Our findings provide new insights into how students are interacting with the ELE and the effects of the system's interventions on students' affective states.

1. INTRODUCTION

Much recent research has focussed on *Exploratory Learning Environments* (ELEs) which encourage students' open-ended interaction with a knowledge domain, combined with intelligent components that aim to provide pedagogical support to ensure students' productive interaction. The aim of this feedback is to balance students' freedom to explore alternative task solution approaches while at the same time providing sufficient support to ensure that the intended learning goals are being achieved [6]. Here we report on recent work into identifying interaction events that are associated with a change in students' affective state as they interact with an affect-aware ELE called *Fractions Lab*. We adopt a graph-based approach to modelling, querying and visualizing the student-system interactions data, extending preliminary work in this area reported in [8]. In our graphs, nodes represent occurrences of key indicators that are detected, inferred or generated by the ELE, and edges between such nodes represent the "next event" relationship. In contrast, recent work on interaction networks and hint generation (e.g. [4]) uses graphs whose nodes represent states within a problem-solving space and edges represent students' actions in transitioning between states. That work uses the graph-modelled data to automatically generate feedback for the student, whereas we use a graph-based modelling approach to investigate the effects of the system's interventions in order to better understand how students interact with the ELE with the aim of improving its support for students.

2. THE ELE AND USER STUDY

Fractions Lab is an ELE that is part of the iTalk2Learn learning platform targeted at children aged 8-12 years who are learning about fractions. As students interact with Fractions Lab they are asked to talk aloud about their reasoning process. This speech, together with their interactions, are used to detect students' affective states using a combination of Bayesian and rule-based reasoning [5]. Adaptive support is provided based on the student's performance and detected affective state. The affective states detected by Fractions Lab can be ranked according to their effect on learning, based on previous studies (e.g. [7, 3, 1]). For example, being in *flow* is a positive affective state as it indicates that the student is engaging with the learning task well. *Confusion* is mostly associated with realising misconceptions, which also contributes towards learning, while *frustration* and *boredom* are likely to have a negative effect on learning.

We conducted a user study in which iTalk2learn was used by students in a classroom setting. 41 students aged 8-10 took part, with parental consent, recruited from two schools in the UK. Students were given a short introduction to the system. They then engaged with the Fractions Lab ELE for 40 minutes. They then completed an online questionnaire that assessed their knowledge of fractions (the post-test).

The iTalk2Learn platform logged every student-system interaction, such as fractions being created or changed by students, buttons being clicked, feedback being provided by the system, feedback being viewed by students, and the system's detection of students' affective states. This data was then remodelled into a graph form, according to the graph data model shown in Figure 1. We see that the data model comprises two node types: Event nodes, that capture occurrences of key interactions, and EventType nodes, that hold additional metadata about each event. Edges labelled NEXT link together successive Event nodes, allowing us to build up a sequence of events that describe the history of student-system interactions as a student works on a task during a session. An edge labelled OCCURRENCE_OF links each Event node to an EventType node.

The data logged by iTalk2Learn was exported as text, parsed and pre-processed using Python and the Pandas and py2neo

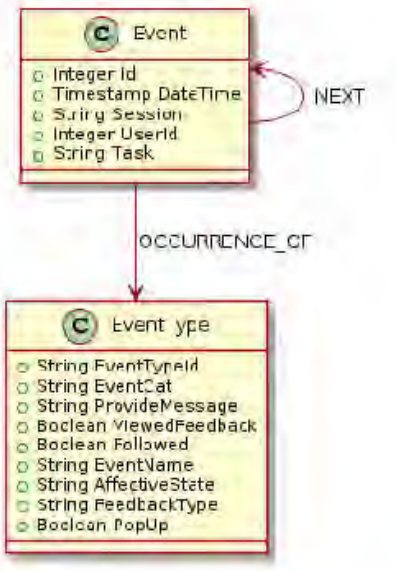


Figure 1: Graph data model for student-system interaction data.

libraries, and then loaded into the Neo4j graph database. To view the resulting data graph we developed a custom visualization tool in JavaScript using the Node.js library. Our tool allows viewing of large-scale changes in affective state as well as details of event sequences. Having interacted with these visualizations, we were interested to explore further the kinds of events that contribute towards changes in students' affective state as they work with Fractions Lab. To do this, we used Neo4j's graph query language, Cypher, to extract the metadata relating to pairs of consecutive events that exhibit a change in a student's affective state. The query below was used to find adjacent Event nodes connected by NEXT, and the EventType nodes they are connected to by OCCURRENCE_OF, such that the affective states associated with the EventType nodes are not equal:

```
MATCH (start_event: Event)-[:OCCURRENCE_OF]->(start_type: EventType),
      (end_event: Event)-[:OCCURRENCE_OF]->(end_type: EventType),
      p = (start_event)-[:NEXT]->(end_event)
WHERE start_type.affective_state in
    ["flow", "boredom", "confusion", "frustration"]
    AND end_type.affective_state in
    ["flow", "boredom", "confusion", "frustration"]
    AND NOT start_type.affective_state = end_type.affective_state
RETURN *
```

3. RESULTS AND CONCLUSIONS

We were interested to explore differences in students' affective states and interactions compared with their performance. Students' performance, based on the post-test score, was on average 3.83 (SD=1.46; min=0; max=6). A median split of students' scores resulted in a higher- and a lower-performing group (high: 27 students; low: 14 students). In order to investigate which interactions moved students into a different affective state we used association rule learning (c.f. [2]) over the data returned by the above Cypher query. We found that students are likely to move from *flow* to *frustration* when provided with reflective prompts in the low-performing group and with open-ended problem solving support in the high-performing group. This might imply that these types of support are imposing too high a cognitive demand on students. Additionally, certain interactions with their fractions may move both categories of student from *flow* to *frustration*. Viewing high-interruption or low-interruption feedback may move low or high performing students, respectively, from *flow* to *confusion*. Finally, we observed a positive effect of Affect Boost messages for both categories of student.

These findings extend earlier ones reported in [5] with a finer-grained analysis of students' affective state changes, identifying several situations where the system's support may need to be modified: (i) reviewing the content of both the high- and the low-interruption messages, to see if the incidences of confusion can be reduced; (ii) considering extending the provision of reflective prompts and open-ended support with additional affect boost messages and hints that students might also select to view, to mitigate against frustration; (iii) considering providing more scaffolds when students are manipulating their fractions, for example additional low-interruption feedback. Exploratory learning environments such as Fractions Lab can generate large volumes of student-system interactions data, making their interpretation a challenging task. We have seen here how modelling such data as a graph can open up new data visualization, querying and analysis opportunities, leading to new insights into how students are interacting with the ELE and the effects of the system's interventions, with the ultimate goal of designing improved support for students.

4. REFERENCES

[1] R. S. J. d. Baker et al. 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. *Int. J. Hum.-Comput. Stud.*, 68, 2010.

[2] D. L. Bazaldua, R. S. J. de Baker, and M. O. S. Pedro. Comparing expert and metric-based assessments of association rule interestingness. In *EDM*, 2014.

[3] S. K. D'Mello et al. Confusion can be beneficial for learning. *Learning & Instruction*, 29(1):153–170, 2014.

[4] M. Eagle, D. Hicks, B. Peddycord III, and T. Barnes. Exploring networks of problem-solving interactions. *LAK*, pages 21–30, 2015.

[5] B. Grawemeyer et al. Affective learning: Improving engagement and enhancing learning with affect-aware feedback. *User Modeling and User-Adapted Interaction - Special Issue on Impact of Learner Modeling*, 2017.

[6] S. Gutierrez-Santos, M. Mavrikis, and G. D. Magoulas. A Separation of Concerns for Engineering Intelligent Support for Exploratory Learning Environments. *J. Research and Practice in Inf. Tech.*, 44:347–360, 2013.

[7] R. Pekrun. The control-value theory of achievement emotions: Assumptions, corollaries, and implications for educational research and practice. *J. Edu. Psych. Rev.*, pages 315–341, 2006.

[8] A. Poulovassilis, S. Gutierrez-Santos, and M. Mavrikis. Graph-based modelling of students' interaction data from exploratory learning environments. In *Proceedings of G-EDM, at EDM*, 2015.

Predicting Performance in a Small Private Online Course

Wan Han, Ding Jun, Gao Xiaopeng, Yu Qiaoye, Liu Kangxu

School of Computer Science and Engineering
Beihang University
Beijing, China
+86-10-82338059
{wanhan, dingjun, gxp, yuqiaoye, liukangxu}@buaa.edu.cn

ABSTRACT

In this paper, we describe how we build accurate predictive models of students' performance in a SPOC (small private online course). We document a performance prediction methodology from raw logging data based on OpenEdX platform to model analysis. We attempted to predict students' performance of Computer Structure Lab Course (Fall 2016) offering at Beihang University. 28 predictive features extracted for 377 students, and our model achieved an AUC (area under curve) in the range of 0.62-0.83 when predicting one week in advance. This work would help to identify at-risk students in a SPOC.

Keywords

SPOC, student performance prediction, study behavior analysis, educational data mining, at-risk students

1. INTRODUCTION

EdX has designed and built an open-source online learning platform (OpenEdX) for online education. In addition to offering online courses, participating universities are also committed to researching how students learn and how technology can transform learning both on-campus and online throughout the world.

Some researches focus on how to predict students' performance by using study-related data. Stapel, M. [1] presented an ensemble method to predict students' performance, which includes six classification algorithms. Elbadrawy, A. [2] developed multi-regression models based on regression algorithms for predicting, and Ren, Z. [3] designed different kinds of features based on MOOC courses' characters, which improved the performance of their predictor. In addition to study-related data, social behavior data is helpful in predicting [4].

In this paper, we describe the performance prediction problem, and present models we built. A summary of which features played a role in gaining accurate predictions is presented. The most fundamental contribution is the design, development and demonstration of a performance prediction methodology, from raw logging data to model analysis, including data preprocessing, feature engineering, model evaluation and outcome analysis.

2. PREDICTION PROBLEM DEFINITION

Our SPOC was composed of 3 tutorials and 9 projects in Fall 2016, learners studied the tutorials from week 1 to week 6, and we released project 0 at week 7. We found it was important for learners to move on only after they'd mastered the core concept. Students started one project and as they mastered corresponding content, that they need to pass the test in class, and then they could be awarded to the next project.

Here our performance prediction is to predict whether the learner could pass their test at the end of each week according to their study behavior. We define time slices as weekly units. Time slices started the first week in which in class test was offered (week 7), and ended in the 16^{th} week, after the final test had closed.

So we could use the logging data from week 1 to week 6 to predict the learners' performance at week 7. Furthermore, we used 'lead' represents how many weeks in advance to predict performance. We assign the performance label (x_1, 0 for unpassed the test or 1 for passed the test) of the lead week as the predictive problem label. 'Lag' means use how many weeks of historical variables to classify.

3. PREDICTING WEEK PERFORMANCE

We did not use the non-behavioral attribute such as a leaner's age, gender and others. Instead, we used some features that would show different style of learning habits. One type of behavioral variables is based on the learner's interaction with the educational resources, including time spent on resources and problem / homework. As Colin Taylor described in [5], taking the effort to extract complex predictive features that require relative comparison or temporal trends, rather than using the direct covariates of behavior, is one important contributor to successful prediction. For instance, we create an average number of submissions per problem for each learner (x9). Then we compare a learner's x9 value to the distribution for that week. Feature x16 is the percentile over the distribution and x17 is the percent as compared to the max of the distribution. We also extracted features that related to learners' study habits. For instance, feature to describe whether learners begin doing the problem / homework soon after it was released, and features to characterize the learners that submit problem / homework in timely fashion or at last minute fashion.

To build predictive models, we utilize a common approach of flattening the data- assembling the features from different weeks as separate variables.

We first used logistic regression as our binary predictive model. It calculates a weighted average of a set of variables as an input to the logit function. There are different coefficients for the feature values. For the binary classification problem, the output of the logit function becomes the estimated probability of a positive training example.

When applying the logistic regression to learner week performance prediction. We used 28 features to form the feature vectors, and maintained the week performance value as the label.

3.1 Predicting Performance

When evaluating the classifier's performance. A testing set comprised of untrained covariates and labels evaluates the performance of the model as following steps:

The logistic function learned is applied to each data point and the estimated probability of a positive label is produced. And then a decision rule is applied to determine the class label for each probability estimate. Given the estimated labels for each data point and the true labels we calculate the confusion matrix, true positives and false positives and then obtain an operating point on the ROC curve. Then evaluate the area under the curve and report it as the performance of the model on the test data.

We need to present the results for multiple prediction problems for different week simultaneously. Here means for each week during our course, we want to predict the students' week performance using different historical data. The heat map of a lower right triangular matrix is assembled as shown in figure 1.

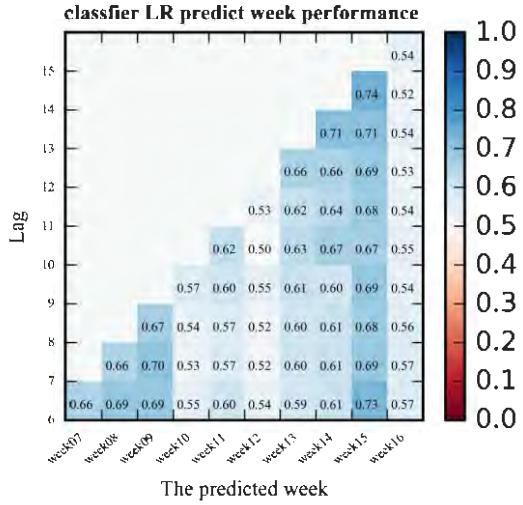


Figure 1. Logistic regression results

The x-axis of figure 1 is the week for which predictions are made in the experiment, while y-axis is the number of the how many week data we use for the prediction (lag). The color shown the area under the curve for the ROC the current model achieved.

We employed cross validation in all of our predictive modelling. Some partitions are used to construct a model, and others are used to evaluate the performance. Considering only 377 samples in our data set, we employed 3-fold cross validation and use the average of the ROC AUC over the folds as evaluation metric.

3.2 Feature Importance

We utilized randomized logistic regression methodology to identify the relative weighting of each features. As shown in figure 2, top features that had the most predictive power include whether learners interact with the resources more time (*max_observed_event_duration*), learners' interaction with the problems (*average_number_of_submissions_percentile*), study habits (*time_first_attempt, problem_finish_time_pre_start24h*, *problem_finish_time_pre_start48h*).

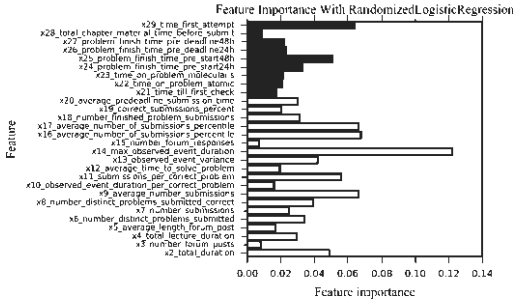


Figure 2. Relative importance of different features across all variants (lag / lead)

4. SUMMARY

We have taken an initial step towards identifying at-risk students in a SPOC, which could help instructors design interventions. Several prediction models are compared, with SVM preferred due to its good performance. The noteworthy accomplishments of our study when compared to other studies including: we extracted variable from the click stream logging data and then generate complex features which explain the learners' study behavior, especially how to describe the learners' study habits. We attributed SVM model to those variables as we achieve AUC in the range of 0.62-0.83 for one week ahead.

In the future, we will collaborate with course instructors to deploy our predictive models. And we will take more attention to why a student is failing, and what strategies make others' success in a SPOC or on-campus course.

5. ACKNOWLEDGMENTS

This research was supported by Teaching Research Funding in Honors College of Beihang University (2017) and Computer Information Specialty Construction Foundation Grant (No.201406025114).

6. REFERENCES

[1] Stapel, M., Zheng, Z., and Pinkwart, N. 2016. An ensemble method to predict student performance in an online math learning environment. In *Proceedings of the 9th International Conference on Educational Data Mining* (June 29 - July 2, 2016, Raleigh, NC, USA), 231-238.

[2] Elbadrawy, A., Studham, S., and Karypis, G. 2014. *Personalized multi-regression models for predicting students' performance in course activities*. Technical Report 14-011. University of Minnesota.

[3] Ren, Z., Rangwala, H., and Johri, A. 2016. Predicting Performance on MOOC Assessments using Multi-Regression Models. *arXiv preprint arXiv:1605.02269*.

[4] Bydžovská, H. 2016. A Comparative Analysis of Techniques for Predicting Student Performance. In *Proceedings of the 9th International Conference on Educational Data Mining* (June 29 - July 2, 2016, Raleigh, NC, USA), 306-311.

[5] Colin Taylor, Kalyan V., and Una-May O., 2014. Likely to stop? Predicting Stopout in Massive Open Online Courses. DOI = http://arxiv.org/pdf/1408.3382v1.pdf.

Social work in the classroom? A tool to evaluate topical relevance in student writing

Heeryung Choi
School of Information
University of Michigan
heeryung@umich.edu

Zijian Wang
Department of EECS
College of Engineering
University of Michigan
zijwang@umich.edu

Christopher Brooks
School of Information
University of Michigan
brooksch@umich.edu

Kevyn Collins-Thompson
School of Information
University of Michigan
kevynct@umich.edu

Beth Glover Reed
Social Work and Women's Studies
University of Michigan
bgr@umich.edu

Dale Fitch
School of Social Work
University of Missouri
fitchd@missouri.edu

ABSTRACT

In a climate where higher education institutions are actively aiming to increase inclusivity [2], we explore how a deep learning-based tool focused on text analysis is able to help assess how students think about issues of privilege, oppression, diversity and social justice (PODS). We created a vocabulary boosting and matching tool augmented with domain-specific corpora and relevance information. We find that the adoption of domain-specific corpora enhances model performance when identifying PODS-related words in short student-written responses to writing prompts, by building a more highly focused PODS vocabulary.

1. INTRODUCTION AND RELATED WORK

Universities are expanding their efforts toward creating more inclusive institutions of higher education [2]. One specific example is the principled blending of curricula with social justice and diversity issues in order to encourage PODS thinking (Privilege, Oppression, Diversity, Social justice) in the School of Social Work at the University of Michigan. PODS principles have been emphasized not only in individual courses but throughout the whole Social Work curriculum. Such a move naturally raises the question of scaled evaluation, both of individual students (e.g. formative or summative assessment) and programmatic evaluation.

In previous work, we explored mechanisms to detect elements of PODS thinking in student writing through semi-supervised machine learning [1]. We adopted the Empath tool [3] to generate an expanded vocabulary from a few seed words for PODS thinking detection, but were extremely limited in our ability to achieve accurate results. The first issue stems from the selection of large but general corpora which, while large in size and topic coverage, were not effective when we attempted to learn domain-specific bigrams. The other issue is how to filter less relevant words while boosting the size of the relevant lexicon. While generating a lexicon for Social Justice on Empath, we found that semantically irrelevant words like "therefore" and "yet" were in the output lexicon [1]. Thus, we expand on previous results and demonstrate a more robust and thorough treatment of the issues of detecting PODS thinking in student writing.

In this work, we consider the specific case of short student writings given in response to a writing prompt. Our goal is to build a technology solution that gives accurately coded responses and that enables instructors to identify quickly which students need elaborated feedback. The system will allow the instructors to focus remediation efforts on those who are of the highest need and to assess how well the overall curricula could increase PODS competency of students. Here we demonstrate the feasibility of using deep learning methods to detect evidence of PODS and apply these methods to a particular writing activity, innovating on the process used by others [3] to improve accuracy and reliability.

2. INSTRUMENTS

We created **Metapath**, a text analysis tool that allows users to use not only general corpora but also domain-specific corpora. Metapath is built on the ability of the Word2Vec model to calculate the similarity of concepts by mapping words and phrases to a vector space via a skip-gram model, and computing the cosine similarity of the corresponding vectors [4]. Given a word, the model gives users a 'most similar' word list ordered by the similarity score. In a preprocessing step, short words ($length \leq 2$), non-English terms, and most stopwords are considered as noise and removed from the corpora. After data cleaning, all words are stemmed using Porter stemming. Common phrases, i.e., multiword expressions, can be detected automatically by calculating mutual information gain within a threshold and minimum count. For example, the words 'Los Angeles' will become the phrase `los_angeles` after phrase detection while the model will return a list of high similarity words like `san_francisco` and `santa_barbara`. The judgment of whether the words are common phrases is based on the formula

$$\frac{cnt(a,b) - min\_count}{cnt(a) \cdot cnt(b)} \cdot N > threshold$$

where $cnt(a,b)$ means the frequency of word a and word b located together and N is the total vocabulary size.

We chose to use domain-specific corpora, i.e., MICUSP (Michigan Corpus of Upper-level Student Papers) and BAWE (British Academic Written English) [5], for detecting common phrases. The general Wikipedia corpus is used to train the model. In addition, considering the contextual nature of the PODS words, existing student responses gathered

from courses were included as a corpus. The domain-specific corpora are able to detect more related phrases on the topics of interest. For example, the proportions (10^{-3}%) of stemmed words like 'prejudic' and 'social_justic' in domain-specific corpora were relatively high (respectively 0.079 and 0.015), compared to the proportions of the same words in the general corpora, which were much lower (0.012 and 0).

3. EVALUATION

We conducted an evaluation to assess how well Metapath can assess PODS-related writing, using our domain-specific corpora, along two dimensions: comparing (1) inter-rater reliability (IRR) for PODS word annotation between human raters and Metapath and (2) IRR for quality evaluation between human raters and Metapath. The latter method is to include percentage of relevance of PODS words, which shows how semantically related each word is to seed words.

3.1 Data

The students' short written responses on PODS topic were used to evaluate Metapath, collected from four sections of a course offered in the School of Social Work ($n = 100$, word counts; $\bar{x} = 695.52$, $\sigma = 434.08$, $min = 115$, $max. = 2747$).

3.2 Approaches

For the evaluation, two expert human coders annotated PODS-related words in the student responses and evaluated overall PODS-relevance of each writing piece with three different marks: high, medium, and low. Their annotations and quality evaluation on student responses were compared with result of Metapath. To build a lexicon to evaluate PODS relevance of student writing, Metapath was boosted by essential PODS words, i.e., privilege, oppression, diversity, and social justice. Furthermore, two keywords from the writing prompt, i.e., "issues" and "actions", were also used to boost the PODS lexicon. After we boosted a lexicon (dim=500), the lexicon was used to calculate the IRR on annotations among two human raters and Metapath. The lexicon and its percentage of relevance were used to assess the overall PODS relevance of each response. After all the responses were ranked based on their percentage of relevance, they were categorized into high, medium, and low. The threshold of the each category was based on the proportion of each category decided by the human raters.

4. RESULTS AND DISCUSSION

We calculated group agreement among the two human raters and Metapath using Krippendorff's alpha (α). For the annotation comparison, IRR among two human raters alone is $\alpha = 0.4480$ ($n = 100$). When we added Metapath the overall group agreement dropped to $\alpha = 0.3804$ ($responses = 100$, $boosted\ words = 4300$, the maximum and minimum possible agreement the 3-rater scenario: $-0.4056 \leq \alpha \leq 0.6324$). IRRs between each human rater individual and Metapath were $\alpha = 0.1622$ and $\alpha = 0.1822$. For the quality evaluation, we achieved $\alpha = 0.3441$ ($responses = 100$, $boosted\ words = 660$) as the level of agreement between human raters and Metapath, which is close to the IRR between the two human raters ($\alpha = 0.4393$, the maximum and minimum possible agreement among 3-rater scenario: $-0.1875 \leq \alpha \leq 0.6223$). IRRs between each human rater individually and Metapath were $\alpha = 0.3702$ and $\alpha = 0.2234$. Overall, the evaluation showed that Metapath could identify PODS-related words and overall PODS relevance. The IRR that Metapath reached was close to those of human raters and not too low, considering the possible minimum and maximum agreement range.

It is worth pointing out that higher agreements in PODS word detection do not align with higher agreements in overall PODS relevance. We varied the size of Metapath's vocabulary by 500 words through setting *the number of boosted words* parameter. Even quite large vocabularies boosted the effectiveness of Metapath in the first task, declining only when values reached $n \approx 4000$. However, the IRR for quality analysis was the highest when $n = 660$.

Further research is needed to explore and improve the performance of Metapath. While identifying PODS-related words, there are still words and phrases in the field of social work that are not detected by Metapath, as noted by the experts. One way to address this is to focus on improved corpora, such as increasing the amount of response data generated by social work students and articles or books curated by PODS experts, or by using corpora based on accumulated Social Work student's writing. Finally, we note that this task is highly multifaceted, and here we have taken just a first pass at addressing it. Issues of personally-lived experiences, intersectionality of topics, and the nature of the writing prompt itself may require more traditional natural language processing techniques in order to capture deeper relationships in the text more fully.

5. ACKNOWLEDGEMENTS

The data used come from the British Academic Written English (BAWE) corpus, which was developed at the Universities of Warwick, Reading and Oxford Brookes, with funding from the ESRC (RES-000-23-0800). This study was funded in part with support from the Michigan Institute for Data Science (MIDAS).

6. REFERENCES

[1] H. Choi, C. Brooks, and K. Collins-Thompson. What does student writing tell us about their thinking on social justice? In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*, pages 594–595. ACM, 2017.

[2] E. DeRuy. The complicated process of adding diversity to the college syllabus. *The Atlantic*, Jul 2016.

[3] E. Fast, B. Chen, and M. S. Bernstein. Empath: Understanding topic signals in Large-Scale text. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4647–4657. ACM, 2016.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[5] M. B. O'Donnell and U. Römer. From student hard drive to web corpus (part 2): The annotation and online distribution of the Michigan Corpus of Upper-level Student Papers (MICUSP). *Corpora*, 7(1):1–18, 2012.

Causal Forest vs. Naïve Causal Forest in Detecting Personalization: An Empirical Study in ASSISTments

Biao Yin
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609
byin@wpi.edu

Anthony F. Botelho
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609
abotelho@wpi.edu

Thanaporn Patikorn
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609
tpatikorn@wpi.edu

Neil T. Heffernan
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609
nth@wpi.edu

Jian Zou
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609
jzou@wpi.edu

ABSTRACT

It is widely understood that students learn in a variety of different ways and what is beneficial for one student may not necessarily help another. This work observes the effectiveness of Causal Forests as they compare to a new method we present called Naïve Causal Forests. This new method, aimed to be a simpler, more intuitive approach to identifying heterogeneous effects, is developed to better understand the strengths and limitations of the Causal Forest method. We apply these techniques to real student data on three RCTs run within the ASSISTments online learning platform.

Keywords

Personalization, Heterogeneous Treatment Effects, Randomized Controlled Trials, Causal Forest, Random Forest

1. INTRODUCTION

The idea that students approach learning in differing ways is not a new concept to researchers in the field of education, but how to leverage these computer-based systems for individualized learning is not always clear. Individualization, also referred to as personalization, also exists outside the field of education as well. In other fields, this idea is described through heterogeneous treatment effects, as the effect of a particular treatment or intervention is not often homologous across all individuals. The introduction of computer-based systems in the classroom makes it feasible to supply aid to individuals allowing the teacher to focus on helping those students struggling most.

Recently, a technique known as a Causal Forest (CF) [8] has been developed, applying random forests to the task of identifying heterogeneous effects. This work explores a new, more intuitive method for identifying heterogeneity as it compares to the more complex CF method. This new method, called Naïve Causal Forest (NCF), attempts to employ a simpler approach based on the structure of CF to answer: 1. To what extent, if any, does the Causal Forest method outperform our simpler, more intuitive approach to identifying heterogeneous treatment effects in real student data? and 2. Do these models converge to large differences when compared using increasing sample sizes?

2. DATASET

The dataset used to build and evaluate our method is comprised of student information on 3 randomized control trials (RCTs) run within the ASSISTments online learning platform [2] from a previously published dataset [5]. ASSISTments is a free web-based platform where a recent efficacy trial found the system to be effective in improving student learning [4], motivating further study to better understand student behavior and measure effects within the platform.

After filtering the data to remove students with missing values, the Experiment 1 contains 519 students, the Experiment 2 contains 833 students, and Experiment 3 contains 1118 students.

3. METHODOLOGY

The Causal Forest (CF) method [8] has established itself as a viable model for identifying heterogeneous effects, for which we do not refute, but rather we wish to explore the benefits of this more complex method to a simpler, more intuitive approach. CF uses estimates of treatment effects within the splitting rule of a random forest algorithm; an "honest" variant uses a holdout set to estimate the effect for each split. Heterogeneous effects can be determined by observing students who then are grouped into different leaves of the generated trees. Our new method, which we have called Naïve Causal Forest, aims to implement a simpler approach that excludes the use of condition from the random forest until students are grouped into each leaf, where then an average treatment effect is calculated across each subgroup. In both methods, each tree has a "vote" as to what condition will benefit the students most.

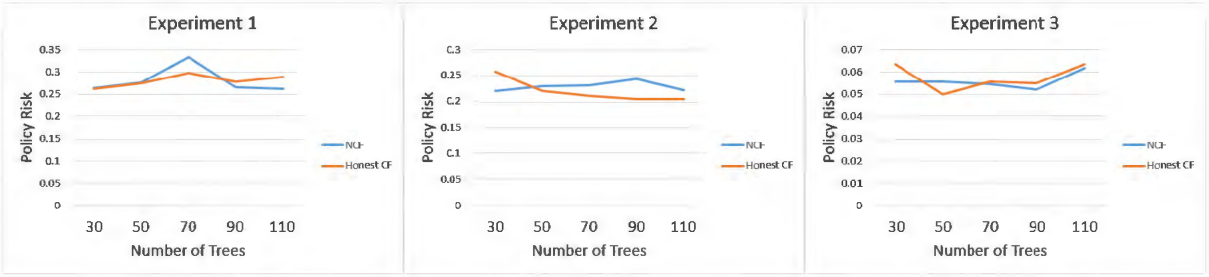


Figure 1: The 10-fold cross validation results for experiments 1 and 2 comparing NCF to an honest CF model. No reliable differences are found between the two methods, and both appear consistent with increases to the number of generated trees.

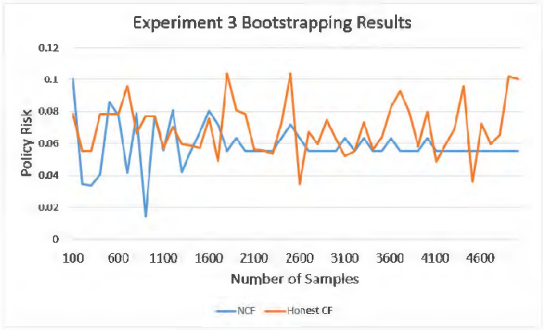


Figure 2: Experiment 3 bootstrapping results comparing NCF to two Causal Forest models.

We compare CF, implemented in R [3] using a Causal Tree package [1], and NCF in their ability to identify heterogeneous effects for the purpose of maximizing completion of the assignment. We calculate the Odds Ratio [7] within each leaf to identify which condition corresponds with the higher student completion rate within each leaf. We evaluate our models using a measure known as policy risk [6], where a lower value indicates better performance. This metric is used to compare the two methods for each experiment as the metric is not directly comparable across experiments.

4. DISCUSSION AND FUTURE WORK

The result of our 10-fold cross validation analysis can be seen in Figure 1. Both models use a minimum leaf size of 30, and are evaluated over several model complexities. In all three experiments, it is found that the CF and NCF model exhibit no reliable differences. It is also the case, however, that no significant heterogeneous effects are found by either method. Figure 2 illustrates how the methods converge with increasing sample sizes using a bootstrapping method of sampling with replacement on the largest experiment.

We compare in this work the Causal Forest method for identifying heterogeneous treatment effects to our Naïve Causal Forest method and find no reliable differences between the simpler and more complex methods. It is expected, and planned for future work, that applying these methods to experiments with larger sample sizes may show statistic reliability.

We also found that the CF model exhibited stable policy risk over increases to model complexity. This is a desirable quality of a prediction model, as it is data driven and less sensitive to changes in model structure. We found that the CF model exhibited non-converging behavior when bootstrapping, but may additionally be caused by insufficient variation or lack of heterogeneity in the dataset.

5. ACKNOWLEDGMENTS

We thank multiple current NSF grants (IIS-1636782, ACI1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428 & DRL-1031398), the US Dept. of Ed (IES R305A120125 & R305C100024 and GAANN), and the ONR.

6. REFERENCES

[1] S. Athey, G. Imbens, and Y. Kong. *causalTree: Recursive Partitioning Causal Trees*, 2016. R package version 0.0.

[2] 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.

[3] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[4] J. Roschelle, M. Feng, R. F. Murphy, and C. A. Mason. Online mathematics homework increases student achievement. *AERA Open*, 2(4), 2016.

[5] D. Selent, T. Patikorn, and N. Heffernan. Assistments dataset from multiple randomized controlled experiments. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 181–184. ACM, 2016.

[6] U. Shalit, F. Johansson, and D. Sontag. Estimating individual treatment effect: generalization bounds and algorithms. *arXiv preprint arXiv:1606.03976*, 2016.

[7] M. Szumilas. Explaining odds ratios. *Journal of the Canadian Academy of Child and Adolescent Psychiatry*, 19:227, 2010.

[8] S. Wager and S. Athey. Estimation and inference of heterogeneous treatment effects using random forests. *arXiv preprint arXiv:1510.04342*, 2015.

An Offline Evaluation Method for Individual Treatment Rules and How to Find Heterogeneous Treatment Effect

Thanaporn Patikorn, Neil T. Heffernan, Jian Zou
100 Institute Rd.
Worcester, MA 01609
{tpatikorn, nth, jzou} @wpi.edu

ABSTRACT

Heterogeneous treatment effects occur when the treatment affects different subgroups of population differently. In this work, we conducted a large scale simulation study to identify the characteristics of treatments that are more likely to have heterogeneous treatment effects, and to estimate how effective the individual treatment rules are compared to the better conditions. We found that heterogeneous treatment effects are rare. When the overall treatment effect is close to zero, we found that individual treatment rule is very likely to be effective. With large positive or negative overall treatment effect, the heterogeneous treatment effect is less likely to occur, and the individual treatment rules are more likely to be ineffective.

Keywords

Heterogeneous Treatment Effect; Individual Treatment Rule; ASSISTments; Randomized Controlled Experiment.

1. INTRODUCTION

Researchers have been using randomized controlled experiments (RCT) to test their interventions. RCTs are considered the gold standard and are widely used in many fields, from healthcare to education. Traditionally, researchers often look for treatment effects across the population. However, in many experiments, the treatment effect differs systematically from one subgroup of the population to another. For example, patients who are allergic to the treatment drugs may react negatively instead of benefiting from the drug. This type of effect is often called heterogeneous treatment effects, as there are different effects for different types of people. Many machine learning methods have been developed to detect heterogeneous treatment effects. For example, [4] introduced the Causal Forest, a decision tree-based method to determine the treatment effect on each subgroup of the population.

In many cases such as [1], it is better to tutor students with lower prior knowledge using step-by-step hints, while it is better to tutor students with high prior knowledge with full problem solutions. In this case, giving personalized tutoring to each student is better than giving the same tutoring to everyone. This type of condition assignment is often called an individual treatment rule or a personalization policy.

In order to evaluate a personalization policy, the most popular method is to deploy the policy in real time and compare the result. However, the on-line method is often costly and sometimes unavailable to the researchers (e.g. because the data have already been collected). As a result, many researchers conduct an offline policy evaluation using past data. In [3], they use the expected outcome of the policy to evaluate their personalization policy. To calculate the expected outcome using past RCT data, we must first find a subset of subjects whose random condition assignments during the RCT matches the personalized condition assignments of the policy. The expected outcome of a personalization policy is the average outcome of this subset across conditions. Comparing two policies using the expected outcome easy and intuitive; if the larger outcome values are better, the policy with larger expected outcome is better. This method is equivalent to policy risk introduced in [2].

The main goals of this work are 1) to find the characteristics of the experiments that are more likely to have heterogeneous treatment effects, and 2) to compare a personalization method, specifically Causal Forest, against assigning every subject to the best conditions to find out how effective a personalization policy can be.

2. METHODOLOGY

In order to gain a better understanding of expected outcome, we investigated how it is calculated in [3]. They first took the subset of the subjects from the RCT whose random condition assignments are the same as the condition assignments given by a personalization policy. For the rest of this paper, we will refer to this subset as the "congruent subset". Then, the expected outcome of the policy is calculated by taking the average outcome values of the congruent subset regardless of conditions. For example, in Table 1, the congruent subset consists of subject 1, 3, 4, and 5, and the expected outcome of the policy is (0.7 + 0.4 + 0.6 +0.7)/4 = 0.6.

2.1 Simulation Study

We conducted a large-scale simulation study to verify the effectiveness of using the congruent subset as an estimate of real outcome values of the policy, and to find types of experiments that are likely to have personalization. We chose simulation study because it allows us to not only calculate the real outcome values of the policy, but also investigate how different settings impact the personalization.

Table 1: an example data to show how congruent subset works

subject	RCT condition	outcome	personalized condition	Is in congruent subset?
1	C	0.7	C	yes
2	T	0.6	C	no
3	C	0.4	C	yes
4	T	0.6	T	yes
5	T	0.7	T	yes
6	C	0.5	T	no

Table 2: Different Distributions for Effect of Conditions

distribution	parameter	values	number of combinations
normal	mean	0, 1, 2, 5, 10	15
	sd	1, 2, 5	
log normal	meanlog	0, 0.5, 1, 2	16
	sdlog	0.25, 0.5, 1, 2	
gamma	shape	0.5, 1, 2, 5, 10	15
	scale	0.5, 1, 2	
total			46

For the simulation study, we focused only on experiments with two conditions. For each condition, we simulated 46 different settings, as shown in Table 2, resulting in 46 * 46 = 2116 different combinations of experiments. We also include lognormal distributions and gamma distributions because real datasets may not always follow normal distributions, for example the mastery speed in [5] resembles lognormal distribution. For each setting, we generated 1000 datasets, each of which has 1000 data points.

Every data set has 3 covariates: one with a positive, negative, and no effect on the outcome. Every covariate value is generated independently for each subject from a normal distribution with mean = 0 and sd = 1. The true effect is generated using the distribution and parameters in Table 2. The observed outcome is

$$\text{observed} = \text{effect} + \text{cov1} * \text{impact1} - \text{cov2} * \text{impact2} + \text{noise}$$

The impacts are from uniform (0,5) and remains constant within experiment. The noise is drawn from a normal (0,1) distribution.

For each personalization policy, we measured 1) if the outcome values of congruent sets are significantly different from the outcome values of actually assigning everyone using personalization policy, and 2) whether the personalization from the Causal Forest is better than the better of the two conditions.

3. RESULTS

From 2,116,000 simulated dataset, we detected the significant difference between the outcome values of the congruent sets and the real personalized outcome values less than 1% of the time, which is far lower than the threshold of 5%, regardless of parameters of the dataset. As for the effectiveness of the Causal Forest, we look at how often the personalization suggested by Causal Forest are better than assigning subjects to the better of the two conditions. We found that personalization is slightly more common when at least one of the distribution is gamma distribution.

Table 3: the Effectiveness of Personalization Suggested by Causal Forest by Overall Observed Treatment Effect

Rounded average observed treatment effect	Causal Forest suggests personalization	Causal Forest's personalization is the most effective
≤ -5	0.03%	15.26%
-4	0.04%	23.19%
-3	0.12%	22.46%
-2	0.41%	44.44%
-1	2.98%	76.43%
0	8.27%	83.56%
1	3.03%	76.26%
2	0.43%	44.79%
3	0.12%	20.76%
4	0.05%	23.35%
≥ 5	0.03%	14.67%

Table 3 shows that when the treatment effect is close to zero, the personalization suggested by the Causal Forest is very effective. Causal Forest policy is better than assigning subjects to the better of the two conditions more than 3/4 of the times when the treatment effects are between -1 and 1. The effectiveness of the personalization quickly drops as the treatment effect is far from zero. It is important to note that the Causal Forest we used in this study has never been optimized and most of parameters we used are default, except the two we specified earlier in the paper.

4. CONCLUSION

This paper has three main contributions. First, we promoted the study of heterogeneous effects and an offline personalization policy evaluation method to the Educational Data Mining. Second, we investigated several different settings of simulated experiments to find the characteristics of the experiments that are more likely to have heterogeneous treatment effects. We found that, generally heterogeneous treatment effects are not common and typically rare when the treatment effects are very large or very small. Third, we investigated the effectiveness of personalization policies given by Causal Forest. We found that the personalization policy is likely to be effective for the experiments with small treatment effects.

5. FUTURE WORK

We plan to investigate different methods for detecting heterogeneous treatment effects on real dataset from ASSISTments to see if we can detect more experiments like [1]. If we can detect such effects, we would be able to improve our system even further, which will improve student learning.

We also plan to compare different methods for detecting heterogeneous treatment effects to see what are the advantages and disadvantages of each model. We also plan to compare these pre-train models to real-time methods like bandits as well. This result will allow us to be able to choose the right tool for the right personalization task.

6. ACKNOWLEDGMENTS

We thank multiple 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.

7. REFERENCES

[1] Razzaq, L. M., & Heffernan, N. T. (2009, July). To Tutor or Not to Tutor: That is the Question. In *AIED* (pp. 457-464).

[2] Shalit, U., Johansson, F., & Sontag, D. (2016). Estimating individual treatment effect: generalization bounds and algorithms. *arXiv preprint arXiv:1606.03976*.

[3] Vickers, A. J., Kattan, M. W., & Sargent, D. J. (2007). Method for evaluating prediction models that apply the results of randomized trials to individual patients. *Trials*, *8*(1), 14.

[4] Wager, S., & Athey, S. (2015). Estimation and inference of heterogeneous treatment effects using random forests. *arXiv preprint arXiv:1510.04342*.

[5] Xiong, X., Li, S., & Beck, J. E. (2013, May). Will You Get It Right Next Week: Predict Delayed Performance in Enhanced ITS Mastery Cycle. In FLAIRS Conference

MyCOS Intelligent Teaching Assistant

Jiao Guo
MyCOS
Wanliuyichen Center A-18
Beijing China 100083
(+86)1058819001-352
amanda.guo@ mycos.com

Xinhua Huang
MyCOS
Wanliuyichen Center A-18
Beijing China 100083
(+86)1058819001-352
xinhua.huang@mycos.com

Boqing Wang
MyCOS
Wanliuyichen Center A-18
Beijing China 100083
(+86)1058819001-169
boqing.wang@ mycos.com

ABSTRACT
In this preliminary study, we introduce MyCOS Intelligent Teaching Assistant (MITA). It is an open learning platform tailored for a specific challenge of Chinese universities, i.e., undergraduates report less student-faculty interaction than those in the U.S.. Compared with existing classroom tools like Socrative, MITA leverages the app-within-an-app model of WeChat (the largest social app in China) instead of a stand-alone app. Which model is the future is debatable. MITA also uses prompt feedback to engage learners and dashboards to inform teachers and administrators. It now serves more than 3,200 teachers and near 110,000 students from 600+ Chinese universities. What the data from the platform reveal about learning deserves further study.

Keyword
Open learning platform, student engagement

1. INTRODUCTION
Researchers found that the gap in student-faculty interaction (SFI) between Chinese universities and their American peers. Based on a comparative study of 2009 National Survey of Student Engagement (NSSE) results, 27% Tsinghua (a Chinese research university) undergraduates had never received prompt feedback from faculty on academic performance while the average in the American research universities was 7% [1].

MyCOS Intelligent Teaching Assistant (MITA) is an open learning platform tailored to the context of Chinese universities. Different from existing tools such as Socrative, MITA enables teachers to interact with students through the app-within-an-app model of WeChat (the most popular social app in China). Whether this model is better than a stand-alone app to engage college students is debatable. It would be interesting to explore similar learning tools that leverage Facebook or other social apps in different countries and then compare.

Inspired by the 2011 proposal of open learning analytics [2], MITA tracks learner behaviors and provides prompt feedbacks. It has data dashboards for teachers (see Figure 1) and administrators to monitor learning process and take informed actions. Since launched in September 2016, MITA has been used by more than 3,200 teachers and near 110,000 students in 600+ Chinese universities. It is a real case of collaboration across research, industry and education sectors. The fast development and nationwide deployment of MITA can produce data useful for further study.

The rest of the poster sections is organized as follows. In section 2 we describe the data sample; in section 3 we report the learning behavior patterns the data reveal; in section 4, we discuss the need for further analysis.

2. DATA SAMPLE
The sample used in this preliminary study was selected from MITA clickstream data between 2016/09/10 and 2017/02/06. During the time period, 1,599 teachers and 45,383 students registered. Among them, 766 teachers and 32,305 students have verified their institute information and interacted through MITA at least once. They are defined as active teachers and active students in this study.

To assess student engagement, we focus on the related learning patterns the MITA data reveal. Specifically, the patterns discussed below (in section 3) are student attendance, quiz participation and questions answered.

The sample covers 278 Chinese universities, including 199 fouryear universities (71.6%) and 99 three-year vocational colleges.

3. BEHAVIOR PATTERNS
3.1 Student Attendance
Existing studies on student attendance were limited within an institution, e.g., a 2015 research on 2,141 classes of a four-year Chinese university found the average attendance rate of 89% [3]. The student attendance pattern based on the MITA sample extends to nationwide and the numbers fall within a reasonable range. The average attendance rate is higher in three-year vocational colleges (92.8%) than that of four-year universities (89.2%).

Daily attendance behaviors demonstrate a similar pattern: the attendance rate of three-year vocational colleges is higher than that of four-year universities every weekday except Friday. The lowest daily attendance rate for three-year colleges is on Friday (88.9%) while for four-year universities is on Monday (87.9%). Hourly attendance behaviors show a common challenge for both categories of universities: classes scheduled in the evening (6-9 pm) have the lowest attendance rates (85% for three-year vocational colleges and 83.9% for four-year universities).

3.2 Quiz Participation
Quiz participation is one of indicators used by researchers to monitor online learning behaviors [4]. MITA enables us to conduct the similar learning analysis in a real classroom. When students take a quiz in class by MITA, they can view the progress in realtime and get the feedback immediately after submission. With the fine-grained data, the teacher can check who participate, who get the answer wrong and which part of the course content is most challenging.

Based on the MITA sample, the quiz participation rate on average is 84.5% for 3-year vocational colleges and 81.7% for 4-year universities. Both are higher than the quiz participation rate in MOOCs. A 2014 study found that 40%~70% learners completed zero quiz in two live-MOOCs (i.e. in-session, instructor-led course with possibility of obtaining a statement of achievement) [5].

3.3 Questions Answered

Asking questions is one of teaching strategies used in college classroom. In a 2013 study, a researcher observed 30 English classes in a four-year Chinese university for two months. She also surveyed 25 teachers and 237 students to analyze the behaviors of asking and answering questions in class [6]. Data collection becomes more efficient with MITA. Based on the MITA sample data, nearly half teachers in three-year vocational colleges (51.7%) use MITA to ask questions in every class session. The proportion is lower in four-year universities (41.6%).

The proportion of answering questions, however, is quite low for students. The MITA data show that 96.7% students in three-year vocational colleges and 98% in four-year universities never answered a question in class. The result looks plausible given the large class size in the sample: 36.8% classes in three-year vocational colleges and 47.2% classes in four-year universities are larger than 50 students. It indicates that some alternative strategy (e.g., an open question in a quiz) can engage more students.

4. DISCUSSION

The focus of this preliminary study is to enhance student-faculty interaction in a real classroom. Besides, MITA has the data on learning behaviors before class (e.g. viewing the course PPT) and after class (e.g. submitting an assignment) for further exploration.

Further study is using EDM & LA (e.g. user behavior modeling) to explore the MITA data in terms of student motivation, performance and satisfaction. More clickstream data (e.g., the number of attempts students try with a quiz) can be collected and analyzed. Different learning patterns can be compared across not only institutional type (four-year universities vs. three-year vocational colleges) but also class size (small, medium and large) or course type (required courses vs. elective courses). The comparison can provide actionable information for teachers and administrators.

Based on the 2015 IMPACT report from Purdue University, nearly half faculty (48%) chose the ICT-supplemental learning model to redesign their courses, 46% chose the hybrid or flipped model and only 6% chose online-only [7]. It indicates the possibility of developing and deploying MITA or similar learning tools for a real classroom in different countries. Experiments of Facebook in classroom has been explored in the U.S. [8], Canada [9], and Singapore [10], but more third-party applications like MITA are needed to extend the capability of Facebook as a learning tool and more debate on whether we should ban or embrace using such a tool is ongoing.

Figure 1. Teacher Dashboard of MyCOS Intelligent Teaching Assistant (MITA).

5. REFERENCES

[1] Ross, H., Cen, Y. and Zhou, Z. 2011. Assessing Student Engagement in China: Responding to Local and Global Discourse on Raising Educational Quality. Current Issues in Comparative Education, Vol. 14(1): 24-37

[2] Siemens, G., D. Gasevic, C. Haythornthwaite, S. Dawson, S. B. Shum, R. Ferguson, E. Duval, K. Verbert, and R. S. J. d. Baker. 2011. *Open Learning Analytics: An Integrated & Modularized Platform*. SoLAR. DOI= http://www.elearnspace.org/blog/wp-content/uploads/2016/02/ProposalLearningAnalyticsModel_SoLAR.pdf

[3] Yao, L.M., Zhu, L.M. and Hu, J.L.2015. Survey and Analysis on College Student Attendance. Jiangsu Higher Education, Vol.15(3):67-70

[4] Wang, Y. 2014. *MOOC Learner Motivation and Learning Pattern Discovery: A Research Prospectus Paper*. In the Proceedings of the 7th International Conference of Education Data Mining, DOI= http://educationaldatamining.org/EDM2014/uploads/procs2014/YRT/452_EDM-2014-Full-Proceedings.pdf

[5] Campbell, J., Gibbs, A., Najafi, H. and Severinski, C. 2014, A Comparison of Learner Intent and Behavior in Live and Archived MOOCs, The International Review of Research in Open and Distributed Learning, Vol.15(5) DOI= http://www.irrodl.org/index.php/irrodl/article/view/1854/3097 [6] Tian, J. 2013. *A Study on the Pattern of Asking Questions in College English Classes*. Shanxi Finance & Economics University. DOI= http://cdmd.cnki.com.cn/Article/CDMD-10125-1013203176.htm

[7] Purdue University. 2015. *Instruction Matters: Purdue Academic Course Transformation (IMPACT) Annual Report.* DOI=https://www.purdue.edu/impact/assets/documents/IMPACT%20annual%20report%202015(I).pdf

[8] Walsh, K. 2011. *Facebook in Classroom, Seriously.* EmergingEdTech. DOI= http://www.emergingedtech.com/2011/03/facebook-in-theclassroom-seriously/

[9] Malhotro, N. 2013. *Experimenting with Facebook in College Classroom*. Faculty Focus. DOI= https://www.facultyfocus.com/articles/teaching-withtechnology-articles/experimenting-with-facebook-in-thecollege-classroom/

[10] Wang, Q., Woo, H. L., Quek, C. L., Yang, Y. and Liu, M. 2012. Using the Facebook group as a learning management system: An exploratory study. British Journal of Educational Technology, Vol. 43(3):428-438

Towards Automatic Classification of Learning Objects: Reducing the Number of Used Features

Pedro González[1], Eva Gibaja[1], Alfredo Zapata[2], Víctor H. Menéndez[2], Cristóbal Romero[1]
[1]University of Cordoba, Dept. of Computer Science, 14071, Córdoba, Spain
[2]Autonomous University of Yucatan, Faculty of Education, 97305, Mérida, Mexico
{pgonzalez,egibaja,cromero}@uco.es, {zgonzal, mdoming}@correo.uady.mx

ABSTRACT

The automatic classification of LOs into different categories enables us to search for, access, and reuse them in an effective and efficient way. Following this idea, in this paper, we focus specifically on how to automatically recommend the classification attribute of the IEEE LOM when a user adds a new LO to a repository. To do it, we propose the use of the multi-label classification approach, since each LO might be simultaneously associated with multiple labels. An initial problem we have found is that the number of terms or pure text features that characterize LOs tends to be very high. So, we propose to apply a dimensionality reduction process. We have carried out an experiment using 515 LOs from the AGORA repository in order to try to reduce the number of features or attributes used, improving execution time without losing prediction accuracy.

Keywords

Multi-label classification, feature selection, learning object

1. INTRODUCTION

The IEEE Learning Object Metadata standard (IEEE LOM) defines several attributes that may be assigned to each Learning Object (LO). However, manual entering all these metadata is a time-consuming process and automated techniques are required for a wider adoption of the standard [2]. In this paper, we focus on how to automatically recommend the classification attribute of the IEEE LOM when a user adds a new LO to a repository. Our idea is to recommend the user what are the possible categories that a LO belongs to from just user-provided information about the LO (such as the title, keywords and description). In order to do it, we propose to use multi-label classification for automatic categorization of LOs from the terms or pure text features that characterize these LOs. Multi-label classification (MLC) is a variant of the classification problem where multiple target labels can be assigned simultaneously to each instance [1]. In traditional classification classes are mutually exclusive, that is, a specific instance can belong to just a single class. However, there are occasions where classes present overlapping, that is, a specific instance can belong to several classes. In our case, we use MLC because a specific LO could belong to several categories.

2. PROPOSED METHODOLOGY

Our proposed approach for automatically classifying of LOs is represented in figure 1. First, we create the data file starting from the terms or pure text features that characterize LOs extracted from the LOs metadata, and categories to which the LO belongs to. Therefore, our next step consists in performing an attribute selection. The final step is the application of a MLC algorithm that will give us a model for classifying new LOs.

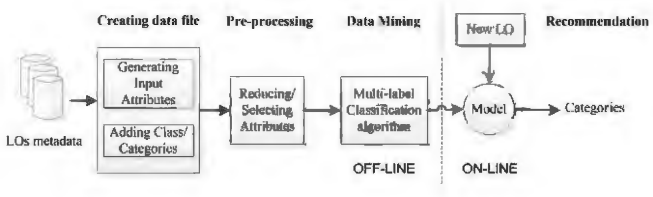


Figure 1. LO multi-label classification approach.

3. EXPERIMENTAL WORK

The data file used in this work has been extracted using 515 LOs from the AGORA repository [3] as follows. When a user adds a new LO to AGORA, he must provide information such as title, keywords, description and other related IEEE LOM metadata. Starting from these information about all the LOs we extracted 1336 terms (features) after removing stop words and stemming (to reduce the terms to their roots). Next, we compute the frequency of these roots for the LO at issue obtaining its term frequency (TF) representation. So, we obtained an example-term matrix, in which each element represents how many times a term appears in an example. We also normalized the count to term frequency to measure the importance of a term. Besides, in AGORA, a user has to specify one or several categories to which the LO belongs to from a predefined set of five academic disciplines: Engineering and Technology; Natural and Exact Science; Social and Administrative Science; Education, Humanities and Art; Health Science. So, we added the 5 labels (in binary format) to each LO as classes to predict. Then, we applied a dimensionality reduction process for reducing the number of attributes in the dataset. The motivation is to reduce training and classification times and removing noisy and irrelevant attributes, which can have a negative impact on accuracy results. Usually, there exists a wide range of possible terms that can refer to LOs of very different topics, and hence, the number of attributes describing LOs tends to be very high. Feature selection has been performed according to a specific method for MLC suggested in [5]. First, the χ^2 feature ranking method was separately applied to each label. Thus, for each label, the worth of each attribute is estimated by computing the χ^2 statistic with respect to the label to determine its independence. The core idea is that, if an attribute is independent on a class, this attribute could be removed. The result of this step is a ranking of all features for each label according to the statistic. Finally, the top-*n* features were selected based on their maximum rank over all labels. Finally, 13 different state-of-the-art MLC algorithms [1] have been applied to the different versions of the data set. They include 3 adaptation algorithms: AdaBoost.MH, Multi-Label k-Nearest Neighbor (MLkNN) and Instance-based Logistic Regression (IBLR), and 10 transformation algorithms in which the J48 implementation of C4.5 decision tree algorithm has been used as base classifier: Binary Relevance (BR), Classifier

Chais (CC), Calibrated Label Ranking (CLR), Label Powerset, Prued Sets (PS), Ensemble of Pruned Sets (EPS), Ensemble of Classifier Chains (ECC), Random-k-LabelSets (RAkEL), Hierarchy Of Mul-tilabel classifiERs (HOMER) and Stacking. The MULAN software for MLC [4] has been used for running both the feature selection method and the MLC algorithms. We have used a 10-fold cross validation with 10 seeds. Our experimentation takes into consideration two main factors: number of attributes and MLC performance. Overall, the time employed by a MLC algorithm to generate a model will be proportional to the number of training instances and the number of attributes describing each instance. So, if we reduce the number of attributes then the computational cost will be reduced as well. However, as a reduction of the number of attributes could discard relevant information, the induced model could perform poorly. This is why we have performed an attribute selection with different reduction levels in order to determine the more suitable reduction level without damaging the classification performance. Our original data set contains 515 LO instances, each one characterized by 1336 attributes. From these, we have selected 1000, 750, 500, 250, 150, 100 and 50 attributes with highest ranking to create different datasets. Next, we have applied 13 MLC algorithms to each different version of the data set, in order to know if there are differences in computational costs and performance by checking some evaluation measures. Therefore, in addition to train time the next five multi-label evaluation measures have been computed: a) Example-based metrics: Hamming loss (H-loss) and Accuracy (E-Acc) b) Label-based measures: Accuracy (L-Acc) and c) Ranking-based measures: Ranking loss (R-loss) and Average precision (A-Pre). On the one hand, we have found a significant reduction of computational costs as the number of features decrease (Figure 2), especially up to 250 features. The algorithms reducing training time at higher degrees are ECC, RAkEL and EPS.

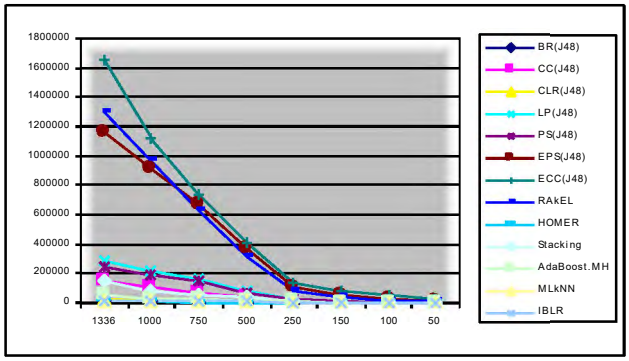


Figure 2. Training time (milliseconds).

On the other hand, in order to compare the classification performance of the algorithms, a Friedman test has been carried out for each evaluation metric by considering results for each feature reduction level. Ranking values and p-values are detailed in Table 1. These p-values (≤ 0,05) show significant differences between reduction levels with high confidence level (95%). We can also observe that for Ranking loss (R-loss) and Average Precision (A-Pre), the best ranking value is obtained for 1000 features instead of the original 1336 features. Besides, a meta-ranking (the rank of rank) of reduction levels was built performing another Friedman test. This way we can evaluate which number of features has the best overall performance in most of the metrics. The last column of Table 1 shows the resulting meta-rank. It is interesting to see that the best ranking does not correspond to the complete feature set. As the test detected significant differences between reduction levels (p-value ≤ 0,01), a Bonferroni-Dunn test was performed. This test found that algorithms performed significantly worst with less than 250 attributes at 95% confidence level. So, we established 250 as the optimum reduction level.

Table 1. Avg. rankings for all metrics and reduction levels.

Number Features	↓H-loss	↑E-Acc	↑L-Acc	↓R-loss	↑A-Pre	Meta Rank
1336	**2,92**	**3,07**	**2,92**	4,50	4,19	2,60
1000	3,76	3,23	3,76	**3,11**	**3,11**	**2,40**
750	3,11	3,57	3,11	3,88	3,42	2,80
500	2,96	3,34	2,96	4,42	3,76	2,80
250	4,19	3,96	4,19	3,96	3,88	4,40
150	5,73	5,57	5,73	4,88	5,46	6,00
100	6,50	6,50	6,50	5,96	6,23	7,40
50	6,80	6,73	6,80	5,26	5,92	7,60
p-values	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001

Finally, a comparison of 13 MLC algorithms when using the optimum reduction level (250 features) has been performed. The goal was to identify which algorithm yields the best results in this specific dataset considering the previous 5 evaluation metrics. The algorithm with the overall best results in the five evaluation measures (higher in E-Acc, L-Acc and A-Pre; and lower in H-Loss and R-Loss) was RAkEL. So, this algorithm will be used in our proposed approach for recommending the categories to which the new LOs belong. In the future we want to use more evaluation measures and also information about LO usage in order to try to improve classification performance.

4. ACKNOWLEDGMENTS

Authors gratefully acknowledge the financial subsidy provided by Spanish Ministry of Science and Technology TIN2014-55252-P.

5. REFERENCES

[1] Gibaja, E., Ventura, S. 2014. Multi-label learning: a review of the state of the art and ongoing research. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4, 6, 411-444.

[2] Kannampallil, T. G., Farrell, R. G. 2005. Automatic Learning Object Categorization For Instruction Using An Enhanced Linear Text Classifier. *Knowledge Management: Nurturing Culture, Innovation, and Technology*, 299-304.

[3] Menéndez, V., Prieto, M., Zapata, A. 2010. Sistemas de gestión integral de objetos de aprendizaje, *Revista Iberoamericana de Tecnologias del Aprendizaje*, 5, 2, 56-62.

[4] Tsoumakas, G., Spyromitros-, E., Vilcek, J., Vlahavas, I. 2011. Mulan: a java library for multi-label learning", *Journal of Machine Learning Research*, vol. 12, 2411-2414.

[5] G. Tsoumakas, I. Katakis, I. Vlahavas. 2011. Random k-labelsets for multilabel classification", *IEEE Transactions on Knowledge and Data Engineering*, 23, 7, 1079-108.

The Reading Ability of College Freshmen

Andrew M. Olney
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
aolney@memphis.edu

Breya Walker
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
bswlker2@memphis.edu

Raven N. Davis
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
rndavis2@memphis.edu

Art Graesser
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
graesser@memphis.edu

ABSTRACT
Over the past 50 years, an increasing proportion of student graduating high school attend college, but literacy levels in the United States have remained largely unchanged. We present preliminary results that suggest the literacy levels of assessed first year college freshmen are above 5th grade but below 12th grade, that only 32% of these freshmen are reading at a 12th grade level, and that this high-performing group has only a 69% chance of passing the reading portion of the GED high school equivalence test.

Keywords
adult literacy, higher education, NAEP, TABE

1. INTRODUCTION
The percentage of high school graduates immediately attending college has steadily increased from 60% in 1990 [5] to 69% in 2015 [2]. However, during this same period the average reading score of 12th grade students on the National Assessment of Educational Progress (NAEP) has declined slightly, such that in 2015, only 37% of students were deemed proficient readers [6]. If all proficient readers immediately attend college, then only 54% of college freshmen are proficient readers. Accordingly, the remaining 46% of college freshmen are either basic or below basic readers.

While it is alarming to think that approximately half of college freshmen are not proficient readers, the NAEP proficiency criteria and cut scores are not without controversy [1]. For example, in a recent mapping of NAEP standards to state standards for 8th grade reading (the highest grade available), only one state was found to have standards aligned with NAEP's proficient category. Given the controversy, it is not clear if the NAEP standards are too high or the state standards are too low.

To better understand the relationship between NAEP reading scores and college freshmen reading ability, we conducted a pilot study using questions from the Reading section of the Tests of Adult Basic Education (TABE). The TABE [3, 4] is useful for exploring the question of reading proficiency of college freshmen because i) TABE items have national norms and are aligned with grade equivalences, allowing us to categorize freshmen reading ability according to grade level and ii) TABE can be used to predict General Educational Development (GED) test performance, which is a proxy for determining whether a participant's reading ability is high school equivalent.

2. METHOD
2.1 Participants
Participants (N = 1062) were recruited through the psychology subject pool at an urban university in the southern United States in two waves of online data collection. The first wave (N = 313), which took place during the spring semester of 2015, was conducted as a regular online study, but the second wave (N = 749), which took place during the fall semester of 2015, was conducted as a screening component for the entire subject pool. Subject pool screening is used to determine eligibility for other studies later in the semester and therefore represents an even more diverse group of participants, as it largely eliminates the self-selection bias of experimental sign up. No demographics of participants were collected.

2.2 Materials
Ten items (#4-13) were selected from the nationally-normed, TABE 10 Form D Reading Survey. Form D (Difficult) is designed to assess reading ability in grade ranges 6.0 - 8.9 and therefore may seem a less obvious choice for assessing college freshmen. However, Form D items cover the widest range of grade equivalents (grades .7 - 12.9) of all TABE 10 forms and therefore has some additional utility when the underlying grade level is unknown. Because the 10 items used in the present study were selected from the 25-item TABE 10 Form D Reading Survey, the distribution of grade equivalents for items does not match the distribution of the complete survey and instead falls into three clusters: five items are at grades 4-5 (3.9, 4.4, 4.8, 5.1, and 5.2), three items are at grades 11-13 (11.4, 12, and 12.9), and two items are at grades 6-7

(6.2 and 7). All items had multiple choice format with four response options.

2.3 Procedure

Participants completed the informed consent and the 10 items using a web browser. Because the study was online and not proctored, the time guidelines of the TABE (approximately 1 minute per question) were not enforced, and due to technical problems, the time participants spent on the items could not be determined. Participants read each of three text passages in turn and answered three to four items after each passage by selecting a multiple-choice response option.

3. RESULTS

Overall, 75% of participants answered 80% or more items correctly, suggesting that the 10 items were overall too easy, as recommendations for TABE specify that participants answer 40% to 75% of the items correctly [4]. Participant performance varied across item difficulty cluster, however. While 73% of participants answered all five items correctly in the 4-5th grade cluster, only 32% answered all three items correctly in the 11-13th grade cluster. Furthermore 30% of participants answered one item or less correctly in the 11-13th grade cluster. Using the TABE guidelines above, this differential cluster performance suggests that 4-5th grade items are too easy but that 11-13th grade items are too hard for the participants assessed.

These results may also be considered in terms of scale scores and GED equivalence. According to previous work mapping TABE Reading scale scores to GED Reading test scores [3], a TABE scale score of 523 corresponds to the passing GED score of 450. Scale scores for each item cluster and items overall were calculated and compared to the GED criterion. Only participants who answered all 10 items correctly (248 participants) or all of the 11-13th grade items correctly (335 participants) surpassed the GED criterion. Using the TABE-GED mapping [3], participants who answered all of the 11-13th grade items correctly had a 69% chance of passing the GED Reading test. Thus while 32% of all participants answered the 11-13th grade items correctly, only 22% of all participants are likely to pass the GED Reading test.

4. DISCUSSION

Our preliminary results suggest that college freshmen reading ability overall is between 5th and 12th grade. This finding is plausible given NAEP results that only 37% of 12th grade students are proficient readers [6]. The lack of a more specific grade-level assessment of freshmen reading ability is attributable to the 10-item assessment used, which lacked medium difficulty items. In the present study, the duration of the complete 25 item TABE Survey was beyond what could be accommodated logistically; however, our results indicate that such logistic considerations must be overcome to assess the reading ability of college freshmen adequately.

Analysis of the 11-13th grade cluster offers suggestive results regarding freshmen reading ability, but must be treated with caution given that there were only three items in this cluster. Participants who answered all three items in this cluster correctly could reasonably be assumed to be proficient readers, and the difference between this percentage (32%) and NAEP's percentage of proficient readers (37%) could be easily explained by regional differences. Although demographic data was not collected for this study, the freshman demographics for the university where the study was conducted suggest that approximately half of students are white and half are African-American. These two groups have NAEP 12th grade Reading Proficiency rates of 46% and 17% respectively, averaging 32% as found in the present study.

However, as previously noted, only 69% of graduating seniors went straight to college in 2015 [2], suggesting that 54% of college freshmen should be proficient readers, assuming that all NAEP Proficient readers attend college. The present finding that reading proficiency is closer to the high school rate than the projected college rate could reflect a self-selection effect whereby the most proficient readers attend schools with more stringent admissions criteria on standardized tests.

The projection that only 69% of participants who answered all three items in the 11-13th grade cluster would pass the GED Reading test gives a strikingly different assessment of freshman reading proficiency (22% vs. NAEP's 37%) that cannot be easily explained by regional differences and may be a useful target for future research.

Altogether, our findings suggest that two-thirds of college freshman assessed have reading ability corresponding with below Proficient as described by NAEP. More accurate assessment and determination of regional differences are important areas of future research, as reading proficiency plays a large role in college success.

5. ACKNOWLEDGMENTS

This research was supported by the Institute of Education Sciences (IES; R305C120001). 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] Grading the nation's report card: Evaluating NAEP and transforming the assessment of educational progress, 1999. DOI: 10.17226/6296.

[2] Bureau of Labor Statistics, U.S. Department of Labor. College enrollment and work activity of high school graduates news release, 2016.

[3] CTB/McGraw-Hill. Tests of adult basic education, forms 9 and 10 norms book, complete battery and survey, all levels. Technical Report 91496, The McGraw-Hill Companies, Inc., 2004.

[4] CTB/McGraw-Hill. Tests of adult basic education, forms 9 and 10 technical report, all levels. Technical Report 91495, The McGraw-Hill Companies, Inc., 2004.

[5] National Center for Education Statistics, U.S. Department of Education. The condition of education - elementary and secondary education - transition to college - immediate college enrollment rate - indicator, 2016.

[6] The Nation's Report Card, U.S. Department of Education. NAEP - 2015 mathematics & reading at grade 12 - reading - national average scores, 2015.

Discovering Skill Prerequisite Structure through Bayesian Estimation and Nested Model Comparison

Soo-Yun Han
Dept. of Mathematics Education
Seoul National University
Seoul, South Korea
ssu1205@snu.ac.kr

Jiyoung Yoon
Dept. of Mathematics Education
Seoul National University
Seoul, South Korea
torol2@snu.ac.kr

Yun Joo Yoo
Dept. of Mathematics Education
Seoul National University
Seoul, South Korea
yyoo@snu.ac.kr

ABSTRACT
Identifying prerequisite relationships among skills is important for better student modeling in many educational systems. In this paper, we propose a new method to discover prerequisite structure from data using nested model comparisons in the context of Bayesian estimation. We evaluate our method with simulated data and real math test data.

Keywords
Prerequisite structure discovery, Bayesian Network, MCMC estimation, nested model comparison, pseudo-Bayes factor.

1. INTRODUCTION
In many educational systems, the process of learning usually proceeds sequentially according to a predetermined order that reflects cognitive theories about student learning. In this learning sequence some knowledge skills must be acquired prior to learning advanced skills. In this study, we refer to *prerequisite structure* as the relationships among skills that put strict constraints on the order in which these skills can be mastered.

Identifying skill prerequisite structure is a crucial step to construct a valid and accurate student model in adaptive tutoring system or other educational system for estimation of student's skill mastery status and provision of appropriate remediation for them. Prerequisite structure can be specified by domain experts, but such process may be time-consuming and could produce subjective models lacking validity. Using large educational data and data mining techniques, several previous studies have tried to find prerequisite relationships among knowledge skills [1,2,3,7]. To derive prerequisite structure from student performance data is somewhat challenging in that a student's mastery status of skills cannot be directly observed, but can only be estimated, i.e, is latent in nature. Previous works mostly used Expectation-Maximization (EM) estimates for latent skill variables [1,2,3].

In this paper, we present a new method for discovering prerequisite structure from student performance data using Bayesian Markov Chain Monte Carlo (MCMC) estimation and nested model comparison. For nested model comparison, we use pseudo-Bayes factor (PsBF) [4], one of the Bayesian model selection criteria.

2. METHOD
In our method, it is assumed that student performance (item response) data at a certain point in time is given and skills related to items are specified. Skills and items are considered as binary random variables and the item-skill relationships are given by Q-matrix (a binary matrix that represents the mapping of items to skills) [9]. DINA model is used for modeling the probability of correct response to an item as a function of whether all the skills required are mastered and of slip and guess parameters [5]. To represent skill prerequisite structure, (static) Bayesian Network is used as student model. Bayesian network is a probabilistic graphical model representing the relationship of a set of random variables as a directed acyclic graph (DAG) with conditional probability tables (CPTs).

We now focus on the discovery of prerequisite relationship, that is, *strict hierarchical order* between mastery of two skills. To this end, we set two types of models: a *full model*, which parameterizes all possible dependencies between skills, and a *strict model*, which assumes prerequisite relationship between a pair of skills. For example, Figure 1 illustrates DAGs and CPTs of a full model consisting of three skills (S_1, S_2, S_3) and a strict model assuming prerequisite relationship between skill S_1 and S_2 (S_1 is a prerequisite for S_2). The difference between two models is that, while the full model contains the parameter γ_{20} related to the probability $P(S_2 = 1 \mid S_1 = 0)$, the strict model put a constraint that this probability is zero (that is, the strict model is nested within the full model). If skill S_1 is a true prerequisite for S_2, the parameter γ_{20} in the full model will be estimated to be closed to zero and there will be no significant difference in the degree to which the two models explain the data. The idea of nested model comparison is to statistically test the null hypothesis that the two models present the same likelihood on the data.

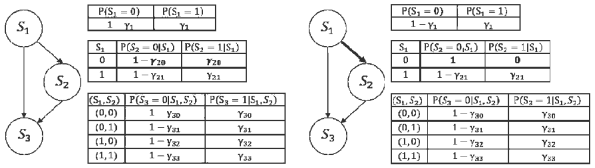


Figure 1. DAGs and CPTs of (a) a full model and (b) a strict model of skills S_1, S_2, S_3. The bolded directed edge from S_1 to S_2 in DAG of the strict model (b) means that S_1 is a prerequisite for mastery of S_2.

When two models are fitted to the data using maximum likelihood, the likelihood ratio test is used for hypothesis testing. In the context of Bayesian estimation, Bayes factor or its variants can be considered as the test method. We use *pseudo-Bayes factor*, which can be calculated by the MCMC estimation process, as the test statistic to contrast two models. The pseudo-Bayes factor for model M_1 relative to M_2 is the ratio of approximations of marginal likelihood based on predictive distributions and cross-validation strategies and defined as

$$\mathrm{PsBF}_{12} = \frac{\hat{p}(X \mid M_1)}{\hat{p}(X \mid M_2)} = \frac{\prod_{i=1}^{n} p(X_i \mid X_{-i},\ M_1)}{\prod_{i=1}^{n} p(X_i \mid X_{-i},\ M_2)} = \frac{\prod_{i=1}^{n} \int p(X_i \mid \Theta, M_1) p(\Theta \mid X_{-i}, M_1) d\Theta}{\prod_{i=1}^{n} \int p(X_i \mid \Theta, M_2) p(\Theta \mid X_{-i}, M_2) d\Theta}$$

where X_i is the response data of student i, X_{-i} is the complement of X_i in the data X, and Θ is the set of free parameters. The

calculated PsBF value in MCMC estimation is compared to a critical value to decide whether to reject the null hypothesis or not. If the null hypothesis is not rejected, then the strict model is accepted, thus concluding that the prerequisite relationship exists.

3. EVALUATIONS

To evaluate the efficiency of our method in discovering prerequisite structures, we first conducted a simulation study and then applied our method to a real dataset. In this process we faced a problem that PsBF values are dispersed from the known distribution of Bayes Factor [6]. To address this problem, we derived the critical value from the empirical distribution of PsBF values under the null hypothesis.

In our evaluation steps, all MCMC estimation algorithms were implemented using R package R2OpenBUGS [8]. For MCMC estimations, we set the priors as follows: a uniform prior $\mathrm{Unif}(0,1)$ on each structural parameters (γ_{ij}) and a beta prior $\mathrm{Beta}(6,21)$ on slip and guess parameters for each items.

3.1 Simulated Data

In this simulation part, we considered five prerequisite structures of latent skills (Figure 2). For each structure, we generated 500 datasets consisting of 1000 students' skill mastery status and their responses for test items using a balanced Q-matrix (each skills are measured with the same number and types of items) under the DINA model with low slip and guess probabilities randomly drawn from $\mathrm{Unif}(0, 0.05)$.

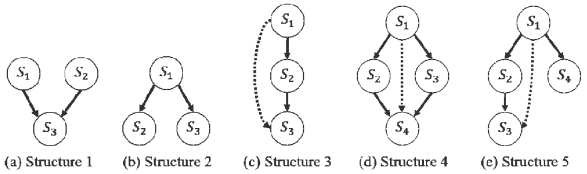


Figure 2. Five prerequisite structures of skills used in simulation study

We evaluate our method using two metrics: *true positive structure rate* (TPSR; # of correct structure recoveries in the output / # of true structures) and *true positive adjacency rate* (TPAR; # of correct adjacency recoveries in the output / # of adjacencies in true model).

The results show that our method can efficiently discover prerequisite structure (Table 1). In all cases recovery rates of true structure are over 80% (the worst rate is 81.6% in structure 4). The recovery rates of true prerequisite relationship between two skills (edges) are even higher such as over 90%.

Table 1. TPSR and TPAR results for each structure

Structure	1	2	3	4	5
TPSR	0.926	0.840	0.872	0.816	0.874
TPAR	0.937	0.942	0.943	0.942	0.962

3.2 Real Data Application

We used mathematics cognitive diagnosis assessment data from 936 eighth grade students over a set of 16 items measuring four skills related to linear equation and linear inequality (Figure 3-a). The prerequisite structure of these skills (Figure 3-b) was initially set by knowledge experts.

Figure 3-c shows the prerequisite structure discovered by applying our method to the real data. All prerequisite relationships set by experts are well discovered, and one additional prerequisite

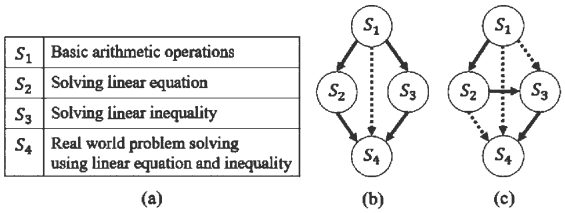


Figure 3. (a) Four skills in math test; (b) Prerequisite structure from knowledge experts; (c) Discovered prerequisite structure

relationship ($S_2 \rightarrow S_3$) is found. A possible explanation for this is that while knowledge experts judge that either linear equation or linear inequality can be learned first, students usually learn to solve linear equation first following the sequence in the curriculum.

4. CONCLUSION AND FUTURE WORK

We presented a method to discover skill prerequisite structure from data based on nested model comparison and evaluated the method using simulated data and real data. The performance of our prerequisite structure learning method was good within the settings used in our experiments. Since we used only low number of skills and certain assumptions for the evaluation, we need to further explore our method in various conditions.

In future work, we will investigate the idea of nested model comparison in the context of frequentist estimation (e.g., EM estimation) and compare with other previous methods. In this paper the focus is only on the prerequisite relationship between skills, but there may be other dependence relationships between them along with different types of response models. It would be interesting to study how to discover skill structures considering various dependency relationships in Bayesian Network modeling of skill mastery.

5. REFERENCES

[1] Brunskill, E. 2011. Estimating prerequisite structure from noisy data. In *Proceedings of the 4th International Conference on Educational Data Mining*.

[2] Chen, Y., González-Brenes, J. P., and Tian, J. 2016. Joint discovery of skill prerequisite graphs and student models. In *Proceedings of the 9th International Conference on Educational Data Mining*.

[3] Chen, Y., Wuillemin, P. H., and Labat, J. M. 2015. Discovering prerequisite structure of skills through probabilistic association rules mining. In *Proceedings of the 8th International Conference on Educational Data Mining*.

[4] Gelfand, A. E. 1996. Model determination using sampling-based methods. In W. R. Gilks, S. Richardson, and D. J. Spiegelhalter (Eds.), *Markov Chain Monte Carlo in practice* (pp. 145-161). London: Chapman & Hall.

[5] 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), 258-272.

[6] Kass, R. E., and Raftery, A. E. 1995. Bayes factors. *Journal of the American Statistical Association, 90*(430), 773-795.

[7] Scheines, R., Silver, E., and Goldin. I. 2014. Discovering prerequisite relationships among knowledge components. In *Proceedings of the 7th International Conference on Educational Data Mining*.

[8] Sturtz, S., Ligges, U., and Gelman, A. 2010. *R2OpenBUGS: a package for running OpenBUGS from R.* http://cran.r-project.org/web/packages/R2OpenBUGS/vignettes/R2OpenBUGS.pdf

[9] 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.

Text analysis with LIWC and Coh-Metrix: Portraying MOOCs Instructors

Junyi Li
Central China Normal University
University of Pennsylvania
junyili@mails.ccnu.edu.cn

Yun Tang
Central China Normal University
tangyun@mail.ccnu.edu.cn

Lijun Sun
Central China Normal University
lijunsunccnu@gmail.com

Xiangen Hu
Central China Normal University
University of Memphis
xiangenhu@gmail.com

ABSTRACT
To date, most MOOCs in major platforms (e.g. Coursera and edX) are xMOOCs, which means teacher speech is still the major part of these MOOCs. Therefore, it is necessary to evaluate the quality of lecture and to explore the relationships between lecture quality of MOOCs and learning outcomes. The present study attempted to explore the lecture styles of instructors in MOOCs by using text analysis. One hundred and twenty-nine course transcripts were collected from Coursera and edX. We also collected public data of course evaluation from the largest MOOC community in China (mooc.guokr.com) Linguistic inquiry and word count (LIWC) and Coh-Metrix were used to extract text features including self-reference, tone, affect, cognitive words, and cohesion. After combined students' comments with clustering analysis, results indicated that four different lecture styles emerged from 129 courses: "mediocre", "boring", "perfect" and "enthusiastic". Significant difference was found between four lecture styles for the notes taken, but significant differences were not found for the course satisfaction and discussion posts. Future studies should exam whether different lecture styles have impacts on students' engagement and learning outcomes in MOOCs.

Keywords
MOOCs; Lecture styles; Instructors; Text analysis

1. INTRODUCTION
Massive open online courses (MOOCs) have attracted much attention in the recent years. They provide not only free courses from high prestige universities, but also the freedom of learning for learners all over the world. Major MOOC platforms, such as Coursera, FutureLearn, edX, and Open2Study, are well received by most learners. The reason why MOOCs become a popular way to learn is that it provides each individual learner with opportunities to engage with the materials via formative assessments and the ability to personalize her learning environment (Evans, Baker & Dee, 2016).

Researchers from different discipline have conducted many studies focused on MOOCs learners, including course completion, quality of interaction, student engagement, and collaborative learning in MOOCs (Andres et al., in press; Wang & Baker, 2015). However, the complexities of teaching have been largely absent from emerging MOOC debates (Ross et al., 2014). After all, MOOC is quite different from traditional class in many aspects. For example, MOOC instructors were motivated by a sense of intrigue, the desire to gain some personal rewards, or a sense of altruism; they were challenged by difficulty in evaluating students' work, encountering a lack of student participation in online forums, being burdened by the heavy demands of time and money, and having a sense of speaking into a "vacuum" due to the absence of student immediate feedback (Hew & Cheung, 2014). Some instructors found it difficult to teach when not facing a real audience of students (Allon, 2012). To date, most MOOCs in major platforms (e.g. Coursera and edX) are xMOOCs, which is a highly structured, content-driven course and designed for large numbers of individuals working mostly alone, teacher speech is still the major part of these MOOCs. Therefore, it is necessary to evaluate the quality of lecture and to explore the relationships between lecture quality of MOOCs and learning outcomes. Some researchers have tried to build models to automatically predict if certain course content would show up by using natural language processing (Araya et al., 2012). Based on the mentioned above, the present study attempted to explore the lecture styles of instructors in MOOCs by using text analysis.

2. METHOD
2.1 Data Collection
Transcripts from 129 courses (humanities: 24.8%, social science: 38%, science: 37.2%) were collected from Coursera and edX. We also collected public data of course evaluation from the largest MOOC community in Mainland China (mooc.guokr.com). This community offered online learners a platform on which they could voluntarily evaluate MOOCs and share their opinions with fellow online learners. The data set we used included course satisfaction, the number of asynchronous discussion posts per course, notes taken per course, the number of followers per course, to name a few.

2.2 Extracting Text Features
Two text analysis tools (i.e. LIWC and Coh-Metrix) were used to extract text features from 129 course transcripts. According to previous studies, self-reference (I, me, my), affect (positive emotion and negative emotion), tone, cognitive words, and cohesion were extracted. Other features like words per sentence and big-words (words are longer than 6 letters) were also viewed as complexity measure of teacher speech.

2.3 Data Analysis

Clustering analysis and ANOVA were conducted by using RapidMiner and SPSS. We first transformed all the text features into Z score, then performed k-means algorithm with euclidean distance in RapidMiner. The k value was assigned with a value from 2 to 6, because of comprehensibility.

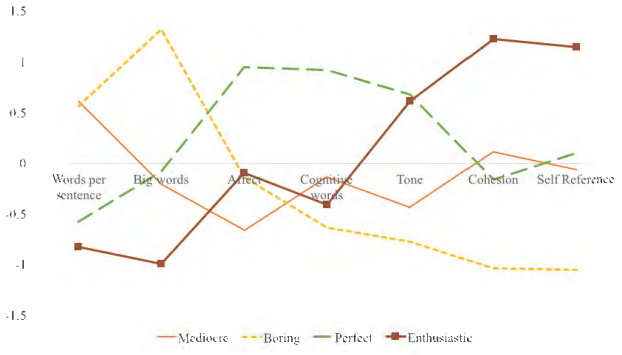


Figure 1 The four lecture styles in MOOCs

3. RESULTS AND DISCUSSION

Four clusters were found, and there were 42, 27, 36, and 24 courses in each cluster respectively. We then checked the students' comments of these courses in Guoke MOOC community, and assigned label to these clusters (Figure 1).

Concretely, instructors who used the most self-reference words (I, me, my), short sentences, and the least big-words were perceived as agreeable and enthusiastic by students (Cluster 4: Enthusiastic). Instructors who used the least self-reference words, long sentences, the most big-words, and showed a low cohesion were perceived as boring by students (Cluster 2: Boring). Instructors who used the most cognitive words to help students to understand and used medium level of self-reference words, big-words and showed medium cohesion were labeled as "perfect" (Cluster 3). Courses used the most of long sentences and showed average level in other dimensions were labeled as "mediocre" (Cluster 1). No significant differences were found between four lecture styles for the course satisfaction (F = .76, p = .52, $\eta 2$ = .02) and discussion posts (F = 1.39, p = .25, $\eta 2$ = .03). However, significant difference was found for notes taken (F = 2.80, p = .4, $\eta 2$ = .06). Concretely, the number of notes taken in "perfect" style was much more than "mediocre". Notes taken can stand for the cognitive processing of learners to some extents. These results suggested that the "perfect" lecture style may be more likely to encourage students' engagement. Since the discussion posts, notes taken and course satisfaction data in the present study were acquired from a third-party platform, further evidence are needed to verify these results. Future studies should examine whether the four lecture styles have different impacts on students' engagement and learning outcomes (e.g. academic performance and course completion) in MOOCs.

4. ACKNOWLEDGEMENTS

This research is supported by China Scholarship Council and Excellent Doctoral Dissertation Program (Central China Normal University). We appreciated these support both in finance and in spirit.

5. REFERENCES

[1] Evans, B. J., Baker, R., & Dee, T. S. 2016. Persistence Patterns in Massive Open Online Courses (MOOCs). *Journal of Higher Education,* 87, 206-242.

[2] Andres, M., Baker, R., Gasevic, D., & Spann, G. in press. Replicating 21 Findings on Student Success in Online Learning. *Technology, Instruction, Cognition, and Learning.*

[3] Wang, Y., & Baker, R. 2015. Content or Platform: Why do students complete MOOCs? *Journal of Online Learning and Teaching,* 11, 17-30.

[4] Ross, J., Sinclair, C., Knox, J., Bayne, S., & Macleod, H. (2014). Teacher experiences and academic Identity: The missing components of MOOC pedagogy. *Jounral of Online Learning and Teaching*, 10, 57-69.

[5] Hew, K. F., & Cheung, W. S. 2014. Students' and instructors' use of massive open online courses (MOOCs): Motivations and challenges. *Educational Research Review,* 12, 45-58.

[6] Allon, G. 2012. Operations Management, Udemy. *Chronicle of Higher Education,* 59, B10-11.

[7] Araya, R., Plana, F., Dartnell, P., Soto-Andrade, J., Luci, G., Salinas, E. and Araya, M. 2012. Estimation of teacher practices based on text transcripts of teacher speech using a support vector machine algorithm. *British Journal of Educational Technology*, 43, 837-846.

Identifying relationships between students' questions type and their behavior

Fatima Harrak
Sorbonne Universités
UPMC Univ Paris 06,
CNRS, LIP6 UMR 7606,
4 place Jussieu, 75005 Paris, France
fatima.harrak@lip6.fr

François Bouchet
Sorbonne Universités
UPMC Univ Paris 06,
CNRS, LIP6 UMR 7606,
4 place Jussieu, 75005 Paris, France
francois.bouchet@lip6.fr

Vanda Luengo
Sorbonne Universités
UPMC Univ Paris 06,
CNRS, LIP6 UMR 7606,
4 place Jussieu, 75005 Paris, France
vanda.luengo@lip6.fr

ABSTRACT

We present the process of categorization of students' questions, and through a clustering on students, we show the relevance of this classification to identify different profiles of students. It opens perspectives in assisting teachers during Q&A sessions.

Keywords

Clustering, question taxonomy, students' behavior.

1. INTRODUCTION

Studying learners' questions while they learn is essential [1], not only to understand their level and eventually help them learn better [2] but to help teachers in addressing these questions. Analyzing students' questions can help for instance in distinguishing deep learning vs. shallow learning [3]. In this paper, we are interested in whether the type of questions asked by students on an online platform is characteristics of their classroom behavior. We investigate this question in the context of an hybrid curriculum (like [4]), where students have to ask questions before the class to help professors prepare their Q&A session. Our goal here is threefold: (RQ1) Can we define a taxonomy of questions relevant to analyze students' questions? (RQ2) Can we automatize the identification of these questions? (RQ3) Can annotated questions asked by a student inform us about their performance, attendance and questioning behavior?

2. RESEARCH METHODOLOGY

We addressed these research questions in 3 successive steps: (1) we conducted a manual process of categorization of students' questions, which allowed us to propose a taxonomy of questions, (2) we used this taxonomy for an automatic annotation of a corpus of students' questions, (3) to identify students' characteristics from the typology of questions they asked, we used clustering technique over two courses and then characterized the obtained clusters using a different set of features, as in [5].

The dataset used for this work is made of questions asked in 2012 by 1st year medicine/pharmacy students from a major public French university (Univ. Joseph Fourier). Each course is made of 4 to 6 4-week sequences on the PACES[1] platform. After a 1st week dedicated to learning from online material, during week 2 students must ask questions and vote for questions asked by other students on an online forum to help professors prepare their Q&A session in week 3. Therefore, for each of the 13 courses, we have 4 to 6 sets of questions asked by students (6457 questions overall) during the 2nd week of each sequence.

[1] paces.medatice-grenoble.fr

3. RESULTS

3.1 Categorization of questions

To answer to RQ1, we took a sample of 600 questions (around 10% of the corpus size) from two courses (biochemistry [BCH], histology & developmental biology [HBDD]), which are considered to be among the most difficult courses and had the highest number of questions asked. This sample was randomly divided in 3 sub-samples of 200 questions to apply 3 different categorization steps: a discovery step, a consolidation step and a validation step. Step 1 consisted in grouping sentences with similarities to extract significant concepts. Then we segmented the combined questions to standardize the previous annotation and we grouped the extracted categories into independent dimensions, where each dimension grouped similar concepts in sub-categories. Step 2 consisted in annotating the second sub-sample to validate the dimensions previously identified and to make sure they were indeed independent from each other. In step 3, we performed a double annotation to validate the generality of our categories on the remaining sub-sample of 200 sentences. Two human annotators used as a unique reference the taxonomy previously created. They annotated independently each dimension (average kappa = 0.70) – discussions to fix discrepancies led to a final refinement of the categories' description. Finally, a re-annotation was performed on the entire sample (600 sentences) to consider the changes and to provide a grounded truth for the automatic annotation. The final taxonomy is provided in Table 1.

Table 1. Final question taxonomy from manual annotation

Dim1	**Type questions**	**Description**
1	Re-explain / redefine	Ask for an explanation already done in the course material.
2	Deepen a concept	Broaden a knowledge, clarify an ambiguity or request for a better understanding
3	Validation / verification	Verify/validate a formulated hypothesis
Dim2	**Modality explanation**	**Description**
0	N/A	None – attributed when neither of the other values below applies
1	Example	Example application (course/exercise)
2	Schema	Schema application or an explanation about it
3	Correction	Correction of an exercise in course/exam
Dim3	**Type of explanation**	**Description**
0	N/A	None – attributed when neither of the other values below applies
1	Define	Define a concept or term
2	Manner (how?)	The manner how to proceed
3	Reason (why?)	Ask for the reason
4	Roles (utility?)	What's the use / function
5	Link between concepts	Verify a link between two concepts

Dim4	Optional: if question is a verification	Description
1	Mistake / contradiction	Detect mistake/contradiction in course or in teacher's explanation.
2	Knowledge in course	Verify knowledge
3	Exam	Check exam-related information

3.2 Automatic annotation

To answer to RQ2 and to annotate the whole corpus (and on the long term, to use it online to analyze the questions collected), we identified keywords representative of each value in each dimension (e.g. the word "detail" is representative of a "deepen a concept" question). Then we developed an automatic tagger which identifies for each question the main value associated to each dimension and tags the question as such. We validated the automatic annotator by comparing its results on the manually annotated subsample of 600 questions and obtained a kappa value of 0.74, enough to consider applying it to the full corpus.

3.3 Links between questions and behavior

To identify whether the type of questions asked can inform us on students' characteristics, first we performed two clustering analyses using K-Means algorithm (with k varying between 2 and 10) over two datasets: students who asked questions in the BCH course (1227 questions asked by N_1=244 students) and in the HBDD course (979 questions asked by N_2=201 students). We performed the clustering using as features for each student the proportion of each question asked in each dimension (*e.g.* the proportion of questions with value 1 in dimension 1) asked (a) overall, (b) during the first half of the course, and (c) during the second half of the course (44 features overall). Distinguishing (b) and (c) in addition to (a) allowed us to take into account whether it was a change in questions asked that could be meaningful, more than the overall distribution. We obtained 4 clusters in both cases.

The second step consisted in characterizing the clusters by considering attributes not used for the clustering: students' grade on the final exam on this course (out of 20), attendance ratio (from 0 [never there] to 1 [always there]), the number of questions asked in this course, and the number of votes from other students on their questions in this course. Students for whom this data was not available were excluded from the datasets, leading to two smaller sample sizes (N'_1=173 and N'_2=161). We performed two one-way ANOVA for grades on these two clusterings and found statistically significant differences (p<0.001 and p<0.001). For the other variables, the distribution did not follow a normal law and we therefore performed a Kruskal-Wallis H test on ranks associated to each variable. The test showed that there was a statistically significant difference for attendance (p=0.04 and p=0.02), number of questions asked (p<0.001 and p<0.001) and number of votes received (p=0.04 and p<0.001) for BCH and HBDD respectively. Results are summarized in Table 2.

Table 2. Differences between the 4 BCH and HBDD clusters

Course	Cluster	*N*	Grade (/20)	Attendance	# quest.	# votes
BCH	A	53	7.97	0.86	2.83	3.06
	B	63	8.54	0.90	2.92	2.69
	C	86	9.38	0.93	6.23	2.61
	D	42	11.2	0.93	11.74	1.22
HBDD	A	59	7.43	0.89	3.53	5.57
	B	34	9.78	0.92	2.44	2.47
	C	72	10.11	0.92	6.54	3.69
	D	36	11.78	0.95	7.00	1.71

4. DISCUSSION AND CONCLUSION

Overall, when considering the results presented in Table 2, we see two similar clusters in both cases: A and D. Cluster A is made of around 28-41% of the students with grades lower than average, attending less to classes, asking less questions than average but which are particularly popular (probably because of votes from similar students, but that information was unfortunately not available). In terms of questions asked, they had a higher number of "how to" questions (*cf.* dim3-2 in Table 1) than any other cluster. On the other end of the spectrum, cluster D is made of around 21% of the students with grades above average, high attendance, who ask more questions than average that are fairly unpopular – we can assume these must be very precise questions that already require a good understanding of the content of the course, and are thus not deemed as important by other students. Interestingly, when comparing the proportion of questions asked in the first vs. second half of the class, cluster D students are the only ones who asked more questions in the 2nd half of the 4-6 sequences than in the 1st half, presumably because the concepts presented at the beginning were simpler and easier for them to understand. In between, clusters B and C represent more average students who differ mostly in terms of number of questions asked.

Therefore, to answer to RQ3 we have shown that although the clustering was performed exclusively on semantic features (*cf.* taxonomy in Table 1), it correlates with information relative to students' performance, attendance and questioning/voting behavior. Our work has some limits: we have applied it only to 2 courses (because a minimum number of questions is required) and we have not considered if it would be possible to classify students in clusters online or even if the same clusters could be found in the same courses on different years. Furthermore, not all questions could be automatically annotated, which reduced the dataset size and is particularly problematic for students who asked few questions. However, this work demonstrates the validity and the usefulness of our taxonomy, and shows the relevance of this classification to identify different students' profiles. It also suggests the taxonomy could be useful for our long-term goal which is to assist teachers in choosing questions to be explained in Q&A sessions. We also intend to apply this taxonomy to different datasets (*e.g.* questions asked in a MOOC) to see if it can also be useful in these contexts and if similar patterns appear.

5. REFERENCES

[1] A. C. Graesser and N. K. Person, "Question Asking During Tutoring," *Am. Educ. Res. J.*, vol. 31, no. 1, pp. 104–137, 1994.

[2] J. Sullins *et al.*, "Are You Asking the Right Questions: The Use of Animated Agents to Teach Learners to Become Better Question Askers.," in *FLAIR*, 2015, pp. 479–482.

[3] C. Chin and J. Osborne, "Students' questions: a potential resource for teaching and learning science," *Stud. Sci. Educ.*, vol. 44, no. 1, pp. 1–39, Mar. 2008.

[4] Q. Liu, W. Peng, F. Zhang, R. Hu, Y. Li, and W. Yan, "The Effectiveness of Blended Learning in Health Professions: Systematic Review and Meta-Analysis," *J. Med. Internet Res.*, vol. 18, no. 1, Jan. 2016.

[5] F. Bouchet, J. M. Harley, G. J. Trevors, and R. Azevedo, "Clustering and Profiling Students According to their Interactions with an Intelligent Tutoring System Fostering Self-Regulated Learning," *J. Educ. Data Min.*, vol. 5, no. 1, pp. 104–146, May 2013.

Metacognitive Prompt Overdose: Positive and Negative Effects of Prompts in iSTART

Kathryn S. McCarthy, Amy M. Johnson, Aaron D. Likens, Zachary Martin, Danielle S. McNamara
Arizona State University
Tempe, AZ, USA
{ksmccar1, amjohn43, alikens, zsmartin, dsmcnama}@asu.edu

ABSTRACT
Interactive Strategy Training for Active Reading and Thinking (iSTART) is an intelligent tutoring system that supports reading comprehension through self-explanation (SE) training. This study tested how two metacognitive features, presented in a 2 x 2 design, affected students' SE scores during training. The *performance notification* feature notified students when their average SE score dropped below an experimenter-set threshold. The *self-rating* feature asked participants to rate their own SE scores. Analyses of SE scores during training indicated that neither feature increased SE scores and, on the contrary, seemed to decrease SE performance after the first instance. These findings suggest that too many metacognitive prompts can be detrimental, particularly in a system that provides metacognitive strategy training.

Keywords
intelligent tutoring systems; metacognition; educational games; system interaction logs

1. INTRODUCTION
Intelligent tutoring systems (ITSs) provide an opportunity for extended training and individualized feedback to support the development of skills and strategies. One such ITS, Interactive Strategy Training for Active Reading and Thinking (iSTART) uses self-explanation (SE) training as a means of increasing students' comprehension of complex texts [4]. iSTART provides instruction on SE strategies through lesson videos, guided demonstration, and practice. Research indicates that prompting metacognition, or reflection on one's own knowledge, can enhance the benefits of training within computer-based learning [1]. In this study, we expand upon previous research to investigate how two metacognitive features affect the SE scores during iSTART practice.

In iSTART's generative practice, students write their own SEs and a natural language processing (NLP) algorithm immediately provides a score of poor (0), fair (1), good (2), or great (3). The two metacognitive features were implemented within this generative practice. The first feature is a *performance notification* that alerts students that their SE score is below 2.0 and sends them to Coached Practice for remediation. The second feature is a *self-rating* that prompts students to rate the quality of their SE before receiving the computer-generated score. The performance notification encourages metacognition indirectly, whereas the self-rating is a direct metacognitive prompt [6]. The current study expands on data reported in [3], which further demonstrated the positive effects of iSTART on deep comprehension, but also indicated that neither metacognitive feature affected post-training learning outcomes. In this study, we explore the log-data to investigate how these two metacognitive features, both individually and in combination, affect SE scores during iSTART generative practice.

Based on previous work [6], we predicted that the performance notification would increase SE scores immediately after the first instance of the notification. In [6], however, the instruction was brief, and did not allow examining further instances of the notification. In this study, we examine the effects of the notification after the initial instance during a longer duration study. Consistent with previous research [5], we had predicted that self-ratings would improve performance. Of particular interest was the interaction of the two features. One hypothesis is that there would be an additive effect such that having both features would yield the greatest SE score improvement [2]. An alternative hypothesis is that the redundancy of the two features would result in an interactive, and possibly negative effect [4].

2. METHODS
2.1 Participants
As part of the larger study reported in [3], 116 high school students (M_{age}=17.67, SD=1.30) received monetary compensation for their participation.

2.2 Design and procedure
The study employed a 2(performance notification: off, on) x 2(self-rating: off, on) between-subjects design. Participants completed iSTART training in three 2-hour sessions. Participants first watched iSTART video lessons that provide instruction on the purpose of SE training and five comprehension strategies (comprehension monitoring, paraphrasing, prediction, elaboration, and bridging). Next, participants completed one round of Coached Practice, in which a pedagogical agent provides individualized feedback on students' self-explanations. Participants were then allowed to move freely throughout the system to interact with videos, Coached Practice, identification games, and generative games for the remainder of the training sessions. The metacognitive features were implemented only during generative games. Performance notifications were triggered each time the average SE score was less than 2.0 and self-rating prompts were triggered on randomly-determined self-explanations approximately 1/3 of the time.

3. RESULTS

We calculated a *gain score* to compare the average SE score in the game before and immediately following an average generative game score of 2.0 indicative of when the performance notification was triggered (or *would have triggered* in the notification off conditions). We used log-data to identify participants who completed at least one game in which their average SE score was less than 2.0 (n=78). Though the performance notification could be triggered as many times as necessary, most participants had no more than two instances of less than 2.0 average SE scores (Fig. 1). As participants were able to move freely through the system, only 48 participants (across all conditions) followed the *generative game, notification, generative game* sequence needed to calculate a gain score. These participants were relatively evenly distributed across the conditions. We analyzed the first two instances of average SE scores less than 2.0 for these 48 participants.

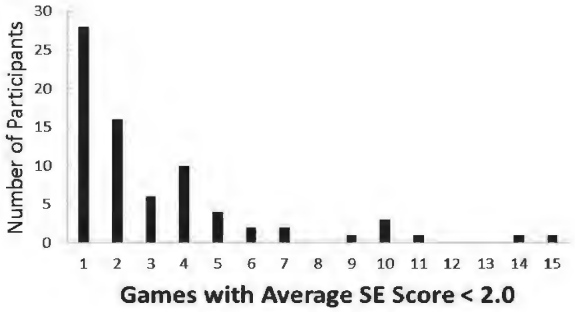


Figure. 1 *Frequency of Games with Average SE Scores < 2.0*

For the first instance of notification, the average gain scores in all conditions were positive. Though the pattern of gain scores for the performance notification is consistent with previous findings [3], an ANOVA indicated no effect of notification, of self-rating, and no interaction, all $F(1, 47) < 2.00$ (Fig. 2, *left*).

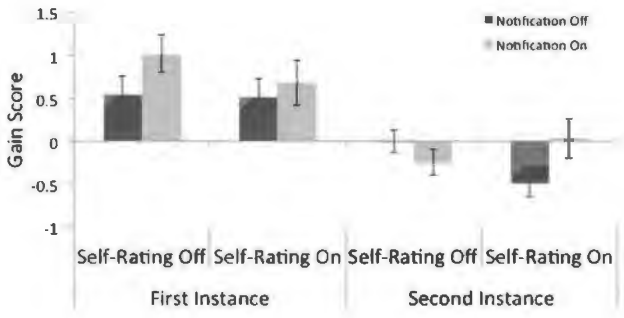


Figure 2. *Gain score in 1st and 2nd instance of avg. SE score < 2.0 as a function of performance notification and self-rating*

Fewer participants (n=27) had a second instance of notification. Contrary to the scores following the first instance, in this second instance, average gain scores were either near zero or negative, indicating that the scores after notification were the same or lower than before the notification. An ANOVA revealed no main effect of performance notification or self-rating, Fs < 1.00, *ns.* There was a significant notification by self-rating interaction indicating that having neither feature or both features did not affect SE score, but that the presence of only one metacognitive feature was detrimental to SE score, $F(1, 26)=5.46$, $p < .05$, $\eta^2_p=.17$ (Fig. 2, *right*).

4. CONCLUSIONS

These findings indicate that neither metacognitive feature had a consistent effect on SE quality during iSTART training. Though there was an overall increase in SE score in the first instance (as indicated by positive gain scores), there was no significant effect of either performance notification or self-rating compared to control. In the second instance, the interaction should be interpreted with caution given the small sample size. Nonetheless, the features did not improve SE score, and were potentially detrimental to performance. One explanation for these findings is that iSTART intrinsically instructs on metacognitive strategies. Hence, the inclusion of additional metacognitive prompts may be redundant, if not overwhelming, at least after the first instance.

These results were not consistent with extant research, and may be particular to iSTART. Certainly further analyses and studies are merited and will be explored. Nonetheless, given that neither prompt showed post-training learning outcomes [3] or sustained training benefits, we do not intend to include these features in future implementations of iSTART, and we would caution other researchers to consider the possibility of potential metacognitive prompt over-dosages.

5. ACKNOWLEDGMENTS

This research was supported in part by IES Grant R305A130124. Opinions, conclusions, or recommendations do not necessarily reflect the views of the IES.

6. REFERENCES

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

[2] Flavell, J. H.: Metacognition and cognitive monitoring: A new area of cognitive– developmental inquiry. American Psychologist. 34(10), 906 (1979)

[3] McCarthy, K.S., Jacovina, M. E., Snow, E.L. Guerrero, T. A., & McNamara, D.S.. *iSTART therefore I understand: But metacognitive supports did not enhance comprehension gains.* In R. Baker, E. André, X. Hu, M.T. Rodrigo, B. du Boulay (eds.) *Proceedings of the 18th International Conference on AIED.* Wuhan, China (2017).

[4] McNamara, D.S., Levinstein, I.B., & Boonthum, C. (2004). iSTART: Interactive strategy trainer for active reading and thinking. *Behavioral Research Methods, Instruments, & Computers, 36*, 222-233.

[5] Schraw, G. & Dennison, R.S.: Assessing metacognitive awareness. *Contemporary Educational Psychology, 19*: 460-475 (1994)

[6] Snow**,** E.L., McNamara, D. S., Jacovina, M. E., Allen, L. K., Johnson, A. M., Perret, C. A., Dai, J., Jackson, G. T., Likens, A. D., Russell, D. G., & Weston, J. L. Promoting metacognitive awareness within a game-based intelligent tutoring system. In Mitrovic, A., Verdejo, F., Conati, C., Heffernan, N. (eds), *Proceedings of the 17th International Conference on AIED 2015.* Madrid, Spain: Springer, pp 786-789 (2015)

Tracking Online Reading of College Students

Andrew M. Olney
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
aolney@memphis.edu

Eric Hosman
Department of Counseling, Educational Psychology and Research
University of Memphis
Memphis, TN 38152
ehosman@memphis.edu

Art Graesser
Institute for Intelligent Systems
University of Memphis
Memphis, TN 38152
graesser@memphis.edu

Sidney K. D'Mello
Departments of Psychology & Computer Science
University of Notre Dame
Notre Dame, IN 46556
sdmello@nd.edu

ABSTRACT

We conducted a pilot study that used kernel-level packet capture to record the web pages visited by college students and the reading difficulty of those pages. Our results indicate that i) no students were fully compliant in their participation, ii) the number of texts encountered by participants was highly skewed, iii) the reading difficulty of texts was about 7th grade, $M = 7.24$, $CI_{95}[7.04, 7.43]$, though difficulty varied by participant, and iv) the increasing use of encryption is likely a limiting factor for using kernel-level packet capture to measure online reading in the future.

Keywords

reading, Internet, measurement, text difficulty

1. INTRODUCTION

A recent survey revealed that approximately 90% of undergraduate respondents used laptops for their electronic course readings even though 68% did not prefer electronic textbooks to print [3]. The increase in online reading behavior has created new opportunities for researchers to track ecologically valid reading behavior. Online reading reflects true interests and goals (unlike artificial experimental paradigms) and further allows measures of the time spent reading and of the text itself over extended periods of time.

To better understand the online reading behavior of college freshmen, we conducted a pilot study using custom-designed online reading tracking software based on kernel-level packet capture. Tracking naturalistic online reading behavior appears to be novel to the literature, as most studies of online reading behavior either use lab-based methods like eye-tracking or self-report methods like surveys. Our main research objectives were to determine whether i) participants would comply with the tracking, ii) the reading behavior of participants was measured consistently, and iii) the text difficulty of measured texts was in a reasonable range.

2. METHOD

2.1 Participants

Participants (N = 7) were recruited through the psychology subject pool at an urban university in the southern United States. Self-reported ACT scores ($M = 21.29, SD = 3.64$) ranged from 18 to 29. Participants were required to own and bring a laptop to the study when they enrolled.

2.2 Materials

Kernel-level packet capture software for tracking online reading behavior was developed in C♯ using the WinPcap and PcapDotNet packet capture libraries. The resulting software, called Snarf, runs as a Microsoft Windows service in the background whenever the computer is turned on. Snarf monitored all http packet traffic on all network devices and sent anonymized timestamped records of web page URLs to an online Google Fusion Tables service for collection. Records were anonymized by using the media access control (MAC) address of the participant's network card as an identifier. To minimize data traffic, Snarf sent only URLs that did not match a blacklist of known non-reading-related URLs, such as Windows Update and image/audio/video filetypes. Also excluded from collection was any service using the encrypted https protocol. Encrypted traffic was excluded for two reasons. First, it is highly likely that encrypted traffic is of a personal nature that the participants would prefer not to share, e.g. email, banking, or health information. Secondly, breaking encryption could potentially introduce security vulnerabilities and put participants at significant risk.

2.3 Procedure

Approval for the research protocol was obtained from our institutional review board. Participants were enrolled in the study in the fall of 2015. After consent was obtained,

Table 1: Participant reading behavior

			Flesch-Kincaid Grade Level				Word Count			
					95% CI				95% CI	
Id	Texts	Days	M	(SD)	LL	UL	M	(SD)	LL	UL
1	1	0^-	-							
2	23	4^-	9.30	(8.05)	6.01	12.59	1137.10	(1985.10)	325.83	1948.30
3	170	100^+	6.98	(5.74)	6.12	7.85	509.72	(1578.30)	272.46	746.97
4	210	101^+	9.20	(6.67)	8.30	10.11	1152.50	(2086.00)	870.37	1434.60
5	829	94^+	7.15	(5.57)	6.77	7.53	963.39	(1778.20)	842.34	1084.40
6	4	50^+	7.28	(7.13)	0.29	14.26	14.00	(8.98)	5.20	22.80
7	3116	119^+	7.10	(6.76)	6.86	7.34	417.77	(1236.40)	374.36	461.18

Note: CI = confidence interval; LL = lower limit; UL = upper limit; -/+ indicates under/over study length.

an experimenter installed the SNARF online reading behavior tracker onto the participant's laptop and confirmed that SNARF was logging data to the Google Fusion Table service. At the end of the study, each recorded URL was queried and, if it was accessible, downloaded. Text from downloaded files was extracted using the Apache Tika library, tokenized into sentences using the Stanford CoreNLP tools [2], and then measured for word count and text difficulty using the Flesch-Kincaid Grade Level metric [1].

3. RESULTS & DISCUSSION

Of the 327,179 timestamped URLs collected, only 87,029 were unique, and of those unique URLs, only 26,762 (31%) were downloadable at the end of the study. Inspection of the timestamped URLs revealed that, despite efforts to blacklist non-reading-related web traffic, many URLs were not reading-related, e.g. antivirus updates, ads, and video websites.

Texts from downloadable URLs had extreme Flesch-Kincaid Grade Level (FKGL) values ranging from -3.40 to 7431, and extreme word count values ranging from 0 to approximately 10 million. Inspection of the data revealed that the FKGL frequency distribution dropped precipitously at grade level 20 and that the word count frequency distribution likewise dropped at 10,000 words. These values would be possible if a participant read a document with an average sentence length of 22 and average syllables per word of 2.3 (FKGL) or a 20-page single spaced paper (word count); thus these values are plausible but may be overly generous. Descriptive statistics for the texts and downloadable URLs after applying these filtering criteria are shown in Table 1.

Table 1 presents evidence addressing our research objectives. First, participants did not comply with tracking: two participants uninstalled the software within a week (one within the same day) and the remaining five participants failed to uninstall the software or meet the experimenter to uninstall the software after being reminded by email. Secondly, participant's online reading behavior was not measured evenly: the number of texts (as measured by downloadable URLs) read by participants was highly skewed, ranging from 1 to over 3,000. This skewed distribution could be caused by some participants mostly using encrypted sites like Wikipedia or the New York Times which, by virtue of being encrypted, SNARF would not record. Finally, the reading difficulty of texts was in a reasonable range, generally 7th grade, $M = 7.24$, $CI_{95}[7.04, 7.43]$, and word count on average was comparable to a page of single spaced text, $M = 564$, $CI_{95}[521, 507]$, though both varied somewhat by participant as shown in Table 1. These results are slightly lower than might be expected when reading for academic purposes, but for general reading seem reasonable.

4. CONCLUSIONS

Our results indicate that kernel-level packet capture is a viable means for measuring online reading behavior save for the increasingly prevalent use of encryption on all web sites. While it would be possible to modify a browser to record the text displayed to the user, this alternative could inadvertently collect email, banking, or health information that should remain private. Thus it may be that the balance between privacy concerns and reading research is best struck by avoiding general purpose reading applications like web browsers and instead focusing on reading-specific applications that are not otherwise used to access personal information.

5. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation (NSF; 1235958 and 1352207) and Institute of Education Sciences (IES; R305C120001). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the author and do not represent the views of the NSF or IES.

6. REFERENCES

[1] J. P. Kincaid, R. P. Fishburne, R. L. Rogers, and B. S. Chissom. Derivation of new readability formulas (Automated Readability Index, Fog Count, and Flesch Reading Ease Formula) for Navy enlisted personnel. Research branch report 8-75., Naval Air Station, Memphis, 1975.

[2] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[3] D. Mizrachi. Undergraduates' academic reading format preferences and behaviors. *The Journal of Academic Librarianship*, 41(3):301 – 311, 2015.

Dropout Prediction in MOOCs using Learners' Study Habits Features

Han Wan Jun Ding Xiaopeng Gao
School of Computer Science and Engineering
Beihang University
Beijing, China
+86-10-82338059
{wanhan, dingjun, gxp}@buaa.edu.cn

David Pritchard
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, MA, 02139
617-253-6812
dpritch@mit.edu

ABSTRACT
Many educators have been alarmed by the high dropout rates in MOOC. There are various factors, such as lack of satisfaction or attribution, may lead learners to drop out. Educational interventions targeting such risk may help reduce dropout rates. The primary task of intervention design requires the ability to predict dropouts accurately and early enough to deliver timely intervention. In this paper, we present a dropout predictor that uses student activity features and then we add learners' study habits features to improve the accuracy. Our models achieved an average AUC (receiver operating characteristic area-under-the-curve) as high as 0.838 (if lacking study habits is 0.795) when predicting one week in advance. The model with learners' study habits features attained average increase in AUC of 0.03, 0.06, 0.08 and 0.05 in different cohorts (passive collaborator, wiki contributor, forum contributor, and fully collaborative).

Keywords
MOOC, dropout prediction, study habits

1. INTRODUCTION
One way to solve the high dropout rates in MOOC is to deliver timely intervention by predicting the dropout probability. Some researchers focused on extracting features of learners' study activities (such as resource accessing) from MOOCs' log, and then building machine learning models. Balakrishnan [1] used the discrete single stream HMMs model to predict whether a student would dropout or not. [2] tried to establish an extensible real-time predicting model, which is fit for any different courses. Loya [3] demonstrated that who executed their learning process on schedule has greater probability to finish the course in MOOCs. Liang J [4] predicted a student's dropout state 10 days later with 3 months' data into four typical machine learning models (LR/SVM/GBDT/RF).

Taylor C. [5] used the dataset of 6.002x: Circuits and Electronics taught in Fall of 2012 on edX, includes course information and students' activity data. In addition to the common simple features, they produced some complex, multi-layered interpretive features, and then used them as the input of predicting models. They divided the students into four groups according to their participation: *passive collaborator* are those learners never actively participated in either the forum or the Wiki, they just view the resources, but did not have contributions; *wiki contributor* are those learners generated Wiki content, but never posted in the forum; *forum contributor* are those learners posted in the forum, but never actively participated in the Wiki; *fully collaborative* are those learners actively participated by generating Wiki content and posting in the forum. Their results shown that if the sample size of the students group is small (especial for wiki contributor, forum contributor and fully collaborative), the predicting accuracy is relative low.

In our work, we focus on extracting more important features of learners' study habits features to improve the accuracy of predicting models, particularly for the small sample size group.

2. PREDICTION PROBLEM DEFINITION
Our data obtained from the 2014 instance of the introductory physics MOOC 8.MReV through the edX platform. We considered defining the dropout point as the time slice (week) a learner fails to submit any further assignments or problems / exam.

The instructor could use the data from week 1 to the current week i to make predictions. The model will predict existing learner dropout during week (i + 1) to week 16. For example, current week is week 7, and we use the logging data from week 1 to week 7 to predict the learners' performance at week 12 with *lead* equals to 4 and *lag* equals to 7.

3. FEATURES ENGINEERIN
Table 1. Self-proposed covariates

	NAME	Definition
x1	stopout	Whether the student continue submit problem
x2	total_duration	Total time spent on all resources
x3	number_forum_posts	Number of forum posts
x4	number_wiki_posts	Number of wiki posts
x5	average_length_forum_post	Average length of forum posts
x6	number_distinct_problems_submitted	Number of distinct problems attempted
x7	number_submissions	Number of submissions
x8	number_distinct_problems_submitted_correct	Number of distinct correct problems
x9	average_number_submissions	Average number of submissions per problem (x7 / x6)
x10	observed_event_duration_per_correct_problem	Total time spent /number of distinct correct problems (x2 / x8)
x11	submissions_per_correct_problem	Number of problems attempted / number of correct problems (x6 / x8)
x12	average_time_to_solve_problem	Average time between first and last problem submissions for each problem (average(max(submission.timestamp) -min(submission.timestamp) for each problem in a week))
x13	observed_event_variance	Variance of a student's observed event timestamps
x14	number_collaborations	Total number of collaborations (x3 + x4)
x15	max_observed_event_duration	Duration ofLongest observed event
x16	total_lecture_duration	Total time spent on lecture resources
x17	total_book_duration	Total time spent on book resources
x18	total_wiki_duration	Total time spent on wiki resources

We extracted 18 self-proposed features, 7 crowd-proposed features (according to Taylor's work [5]) and 6 study habits related behavioral features on a per-learner basis, these features are list in table 1, table 2 and table 3. And then these features are

assembled from different weeks as separate variables to build predictive models.

Table 2. Crow-proposed covariates

	NAME	Definition
x201	number_forum_responses	Number of forum responses
x202	average_number_of_submissions_percentile	A student's average number of submissions / the average of all the students' submission
x203	average_number_of_submissions_percent	A student's average number of submissions / maximum average number of submissions
x204	pset_grade	Number of the week's homework problems answered correctly / number of that week's homework problems
x205	pset_grade_overtime	Difference in grade between current pset grade and average of student's past pset grade
x206	correct_submissions_percent	Percentage of the total submissions that were correct (x 8 / x 7)
x207	average_predeadline_submission_time	Average time between a problem submission and problem due date over each submission

Table 3. Study habits related behavioral features

	NAME	Definition
x301	problem_finish_percent_pre_start24h	The number of problem learner finished correctly in the first 24h after the problem issued
x302	problem_finish_percent_pre_deadline24h	The number of problem learner finished correctly in the last 24h before the problem due
x303	time_first_visit	Min(time_first_problem_get, time_first_html etext_access) - project_issue_time
x304	time_till_first_check	Average of all problem the time between problem_first_check and problem_first_get
x305	study_before_submit	Total book duration before problem submit + total video duration before problem submit
x306	discussion_duration_after_incorrect_submit	Total discussion duration after incorrect submission

4. RESULTS

As shown in figure 1, for all learners, our models achieved an average AUC as high as 0.838 (and lacking study habits features is 0.795) when predicting one week in advance.

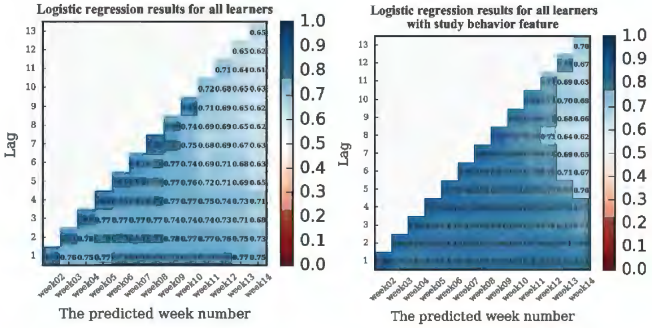


Figure 1. Heatmap for the logistic regression dropout prediction problem

From feature importance analysis as shown in figure 2, the study habits related behavioral features (x301-306) had played more important roles in the dropout prediction. Top features that had the most predictive power including *problem_finish_percent_pre_deadline24h*, *study_before_submit*, and *time_first_visit*.

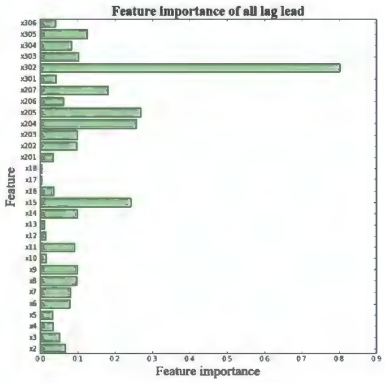


Figure 2. Feature importance

With new features related to study habits, the AUC of our predicting improved (figure 3), especially for the small sample size group (wiki / forum contributor and fully collaborative).

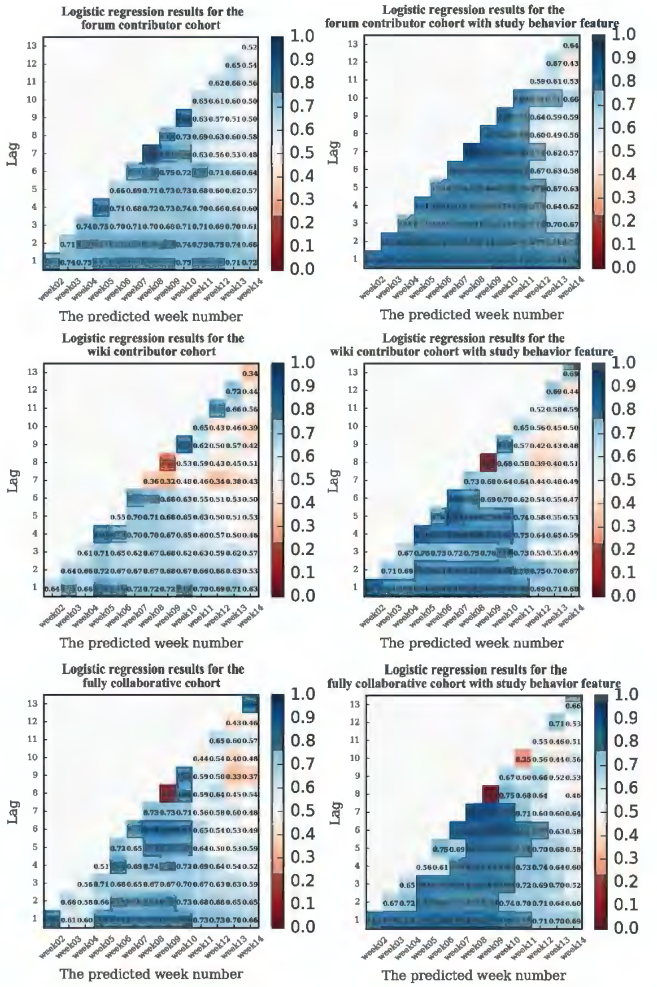


Figure 2. Heatmap for the logistic regression dropout prediction problem for three groups

In the future, we will try to using improved predictor each week within the course progress to deliver the intervention into small private online course.

5. REFERENCES

[1] Balakrishnan, G., & Coetzee, D. (2013). Predicting student retention in massive open online courses using hidden markov models. *Electrical Engineering and Computer Sciences University of California at Berkeley*.

[2] Whitehill, J., Williams, J. J., Lopez, G., Coleman, C. A., & Reich, J. (2015). Beyond prediction: First steps toward automatic intervention in MOOC student stopout.

[3] Loya, A., Gopal, A., Shukla, I., Jermann, P., & Tormey, R. (2015). Conscientious behaviour, flexibility and learning in massive open on-line courses. *Procedia-Social and Behavioral Sciences*, *191*, 519-525.

[4] Liang, J., Yang, J., Wu, Y., Li, C., & Zheng, L. (2016, April). Big Data Application in Education: Dropout Prediction in Edx MOOCs. In *Multimedia Big Data (BigMM), 2016 IEEE Second International Conference on* (pp. 440-443). IEEE.

[5] Taylor, C., Veeramachaneni, K., & O'Reilly, U. M. (2014). Likely to stop? Predicting stopout in massive open online courses. *arXiv preprint arXiv:1408.3382*.

Exploring the Relationship Between Student Pre-knowledge and Engagement in MOOCs Using Polytomous IRT

Jingxuan Liu and Hongli Li
Georgia State University, USA
jliu56@student.gsu.edu, hli24@gsu.edu

ABSTRACT

One of the issues that MOOCs face since its emergence is the low engagement rate and accomplish rate. As an open and free education source, MOOCs are available for people around the world with different motivations and previous knowledge to join. It is a challenge to keep students engaged in a MOOC environment. In the present study, we implement a polytomous item response model (IRT) to explore the relationship between students' self-evaluation of their previous knowledge and students' engagement behaviors in a Geography MOOC. Specifically, we estimate students' latent trait, pre-knowledge, through 15 likert-scale items. Engagement behaviors include assignment, peer review, forum, comment, quiz, and lecture. Each of them is quantified by the aggregated frequency. Then we examine the correlation between pre-knowledge and each type of engagement behavior. We find self-evaluation on previous knowledge cannot predict students' engagement behaviors for any type of engagement. This application indicates that the self-evaluation of pre-knowledge does not predict student engagement in MOOC environment. However, it shows that traditional psychometric models used for standardized tests may be useful and promising in the MOOC context.

Keywords

MOOC, engagement, pre-knowledge, Polytomous IRT

1. INTRODUCTION

A massive open online course (MOOC) is a model for delivering learning content online to anyone who wants to take a course, with no limit on attendance. MOOC engagement is a concept to describe students' involvement of a MOOC. Usually it includes behaviors like posting questions and comments in the MOOC system, submitting assignment and quiz, and other behaviors, which can directly predict students' achievement. Although during the past decade, the number of MOOC students increased tremendously across the world, the low accomplishment and low level of active engagement is always a problem for MOOC development [1]. MOOC engagement is important to predict students' achievement and to show whether students really learned something from the course or not. Students' prior knowledge, which was defined by first two assignments' performance, in computer science and problem solving had impact on their MOOC performance [3]. In the current research, we used pre-course survey data to define pre-knowledge of Geography and to explore if it can predict students' MOOC engagement. Also we use a polytomous IRT model to exam each item and their performance.

2. POLYTOMOUS IRT

Polytomous IRT model is an important model in the IRT family, which is designed for items with more than 2 possible options. Within polytomous IRT models, there are mainly four types: the partial credit model, the rating scale model, the generalized partial credit model, and the graded response model. One example of the application of the graded response model is attitude survey data. Usually the format of item in an attitude survey is likert-scale. For example, for question, "how much do you think you like this opera?", the options can be 5 likert scale from "I like it very much" to "I don't like it at all". The mathematic equation for polytomous IRT model is the following:

$$P^*_{x_{ij}}(\theta_i) = P(X_{ij} \geq x_{ij}|\theta_i) = \frac{e^{Da_j(\theta_i - b_{ij})}}{1 + e^{Da_j(\theta_i - b_{ij})}}$$

In the above equation, D equals to 1.7. For each item j, a_j is a discrimination parameter, and b_{ij} is the difficulty parameter for each option i in each item j ($b_1 < b_2 < ... < b_n$) [2]. Figure 1 indicates a graded response function of a polytomous item. Take the blue line as an example, people with higher theta level seldom choose this option, since the slope is roughly negative.

3. METHOD

3.1 Data

Data comes from a MOOC in Geography. It has enrolled over 100,000 students from 200 countries to date. Data from its 2014 class was used in the present study. In total, after excluding students with little data, there were 3058 students in the current analysis.

3.2 Measure

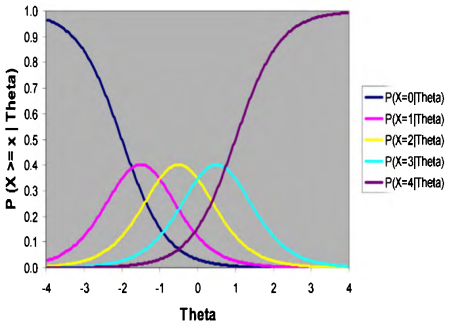


Figure 1: Graded response function

Table 1: Factor loading for each item

Item	1	2	3	4	5
Factor Loading	0.630	0.427	0.608	0.782	0.522
Item	6	7	8	9	10
Factor Loading	0.726	0.705	0.769	0.798	0.697
Item	11	12	13	14	15
Factor Loading	0.668	0.657	0.656	0.698	0.800

There are 15 seven-point likert-scale items, from "strongly agree" to "strongly disagree" designed for students to evaluate their pre-knowledge of Geography. One example is "I enjoy reading maps." In terms of the students' engagement behavior, there are six criteria including assignment, peer review, forum, comment, quiz, and lecture. The method for quantify them is to aggregate the number of times they participate in each type of behavior.

3.3 Procedure

The graded response model was applied using package mirt in R to estimate students' pre-knowledge of Geography. Then the Pearson correlation coefficients between pre-knowledge and each type of engagement behaviors were calculated respectively to examine if students' pre-knowledge influence their engagement behaviors in the MOOC environment.

4. RESULTS

The model fit indices verify a good model fit (RMSEA=0.047, RMSEA_5=0.041, RMSEA_95=0.053, CFI=0.959). The factor loading estimation shows that these 15 items can be used to measure the latent trait, pre-knowledge of Geography (table 1). The parameter estimates are presented in table 2, and the graded response function for each items is shown in the following figure 2. Additionally, table 3 presents the correlation coefficients between pre-knowledge of Geography and each type of engagement behavior.

5. CONCLUSIONS

Table 2: Parameter estimation for each item.

item	a	b1	b2	b3	b4
1	1.38	-1.741	-0.454	1.074	N/A
2	0.804	-2.746	-0.255	2.151	N/A
3	1.302	-6.566	-1.68	0.12	1.842
4	2.133	-5.112	-1.695	-0.553	0.581
5	1.041	-7.801	-2.436	-0.504	0.969
6	1.795	-5.465	-1.72	-0.199	1.096
7	1.693	-5.613	-2.494	-1.345	-0.301
8	2.049	-5.19	-1.768	-0.606	0.639
9	2.257	-5.027	-1.878	-0.757	0.23
10	1.654	-5.692	-1.828	-0.3	0.986
11	1.529	-5.932	-1.553	-0.026	1.268
12	1.482	-6.049	-2.67	-1.278	0.001
13	1.478	-6.048	-2.213	-0.933	0.253
14	1.66	-1.771	-0.307	0.898	N/A
15	2.268	-5.02	-1.621	-0.545	0.52

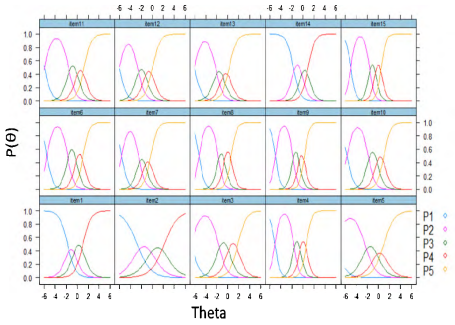


Figure 2: Graded response function for each item.

Table 3: The Pearson correlation coefficient between pre-knowledge and Engagement Behavior Type (EBT)

EBT	Pre-knowledge of Geography
	The Pearson correlation coefficient
assignment	-0.018
peer review	-0.022
forum	-0.016
comment	-0.022
quiz	-0.019
lecture	-0.025

All of the 15 items have relatively good loading on one factor, so it is reasonable to use one-dimensional IRT model. Also, the fit indices show that this graded response model fit well with the data. In terms of the discrimination index, item 8, item 9, item 15 have very good discrimination level. It indicates that these three items can provide more information in terms of students' pre-knowledge of Geography than other items. In terms of the difficulty parameter, b4 cannot be estimated for item 1, item 2, and item 14. This indicates that these items might be problematic.

All of the correlation coefficients are negative and nonsignificant (p-value>.05). This results indicates that although the general trend is students with less pre-knowledge of Geography will have less frequency of engagement behavior, none of them are statistically significant. In other words, whether students report a relative rich or poor pre-knowledge of Geography cannot predict their engagement behaviors. One of the explanation may be the pre-knowledge here is measured by self-evaluation, which relates to the meta-cognitional ability of students. This subjective report is different from objective questions, such as "have you taken any university level courses related to this MOOC course?" In further research, more direct measure of pre-knowledge is needed.

6. REFERENCES

[1] J. Daniel. Making sense of moocs: Musings in a maze of myth, paradox and possibility. *Journal of interactive Media in education*, 2012(3), 2012.

[2] R. J. De Ayala. *The theory and practice of item response theory*. Guilford Publications, 2013.

[3] G. Kennedy, C. Coffrin, P. De Barba, and L. 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.

An Analysis of Students' Questions in MOOCs Forums

Meng Cao
School of Psychology, Central China Normal University, Wuhan, 430079
caomeng@mails.ccnu.edu.cn

Yun Tang
School of Psychology, Central China Normal University, Wuhan, 430079
tangyun@mail.ccnu.edu.cn

Xiangen Hu
School of Psychology, Central China Normal University, Wuhan, 430079
xiangenhu@mail.ccnu.edu.cn

ABSTRACT

When learners become frustrated or confused, they can ask for help by posing questions in MOOCs forums. Students' questions reveal their needs and learning problems. If not answered timely and effectively, they may drop out. In the present study, students' questions from one Chinese MOOCs forum were collected and classified. Results showed that most of the posts in the forum were questions and the quantity of questions decreased over time although in some weeks the number of questions increased. Different types of questions have their own variation characteristics which means that the instructors need to focus on certain types of questions in the corresponding period.

Keywords

Student questions, MOOCs forum, classification, time-variation.

1. INTRODUCTION

Educators think highly of students' question asking. Questions posed by students can reflect active learning, knowledge construction, curiosity and the depth of the learning process [1]. Through analysis of these questions, instructors can better understand a student's thinking, so as to make more targeted teaching decisions [2]. Besides, students' questioning asking has association with their achievement. Learners with good performance behave better in the frequency or quality of questioning [3][4]. Thus, Teachers can also assess students learning based on their questions.

Researchers have investigated students' questioning behavior in a variety of educational settings, such as classroom, tutoring, online learning environments[1]. MOOCs allow students to pose their questions in a forum format and then wait for their questions to be answered by instructors and peer students. This online learning mode and asynchronous discussion pattern influences students' questioning behavior. Students may pose different kinds of questions at any time and at any place anonymously. The present study investigated students' questioning behaviors in the MOOCs forums including the quantity, classification and variations over time. According to previous research and forum data, we first establish standards to screen question posts, then classify and count the quantity of them, and finally observe the variation in the entire course.

2. DATA AND ANALYSIS

2.1 Platform and Data

We analyzed a forum of the course *The Introduction to Psychology* on the Chinese MOOCs platform XuetangX, which was launched in October 2013. This course has been opened for several sessions and has a large enrollment with tens of thousands learners. We chose the data for the 2015 Spring Session as it had the largest number of posts in the forum, starting from March 4th to September 15th. The whole course had 12-week lectures and two exams. The mid-term test took place between the 10th week and the 12th week. The final exam period ran from the 15th to 16th week. All the data came from www.kddcup2015.com and www.xuetangx.com.

2.2 Question Selection and Classification

First, we selected question posts from all the data. We regarded the question mark in the sentence as a marker feature. Some modal words and question words were also taken into consideration, such as" 是不是 (whether or not)", " 什么 (what)","怎么(how)","为什么(why)". And there are some fixed expression of questions, such as "我不懂(I do not know)","我很困惑/疑惑(I am confused)"[4]. Two researchers labeled the posts separately, then compared and made an agreement on the differences. The inter-rater agreement was 86% (representing agreement on 880 items out of 1029 opportunities for agreement multiplied by 100).

After filtering posts, a taxonomy of the questions was created based on Brinton's[5] classification on MOOCs discussion threads and question posts in the forum, including five categories: (1) Course management questions, relating to course design, time arrangement, learning resources, etc.; (2) Course content questions, involving learner's understanding of the learning materials or exercises; (3) Interaction questions, where learners ask and exchange experiences, learning methods and emotions; (4) Platform operation questions, students encounter when operating the platform; (5) Other, including vague expression and irrelevant questions. Two researchers classified the question posts separately and then reached an agreement. The inter-rater agreement was 82% (representing agreement on 613 items out of 751 opportunities for agreement multiplied by 100).

We calculated the total amount of students' question posts, the distribution of different classifications and different types of question variation over the weeks of the course.

3. RESULTS

3.1 The Quantity of Students' Question Posing

In the forum, 1002 people participated in the discussion, accounting for only 3 per cent of the total registers. Among them, 569 students posed 1029 posts, getting 3165 replies, which means that the average reply per post is 3.1. Two researchers screened 751 question posts, accounted for about 73% of the total posts,

indicating that learners' main activity in the MOOCs forum was question asking and answering. Figure 1 shows the quantity of students' questions over the course weeks. The number of posts decreased in general with a few fluctuations.

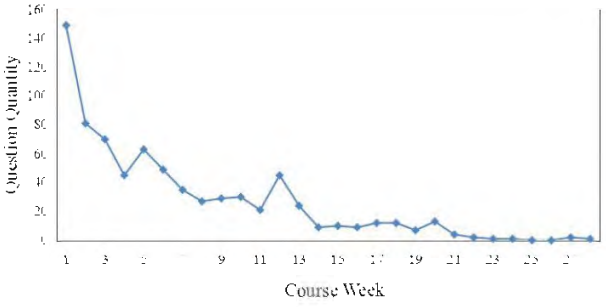


Figure 1: The quantity of students' questions over course weeks

3.2 The Distribution of Five Categories

Table 1 shows the amounts and proportions of five categories, as well as number of replies and average reply per question on each category. The quantity of course management questions are the most while course content questions are only the second. This may due to instructors' low participation in the forum. In the whole course, only some community assistants and administrators posed a limited number of posts and answers. However, course management questions and platform operation questions mainly rely on instructors' answers. As for the course content questions and interaction questions, they can be answered by both instructors and peer learners. Without prompt and proper replies, the first and forth kinds of questions will be repeatedly asked. So the average reply of them are lower than course content questions and interaction questions.

Table 1. The quantity of questions and their replies

Question type	Quantity	Proportion of the total questions	Replies	Average reply per question
Course management questions	334	44.5%	827	2.5
Course content questions	248	33.0%	875	3.5
Interaction questions	49	6.5%	218	4.4
Platform operation questions	111	14.8%	274	2.5
Others	9	1.2%	20	2.2

3.3 The Time-variation of Three Categories

As only a very small number of questions belong to the third and fifth category, we removed them from further analysis and calculated the quantity of the other three categories by course week. Figure 2 shows the relationship between course weeks and question quantity, suggesting a decreasing trend for all the types of questions. However, each type also has its specific characteristics. Course management questions existed throughout the course, because learners will generate a series of questions on textbook, exam, and certificate from start to end. At some time, these questions increased significantly. In contrast, course content questions disappear after the lectures are over. Questions mainly emerge in certain chapters. As to the platform operation questions, the proportion is lower while students may encounter more problems in some weeks on the practice submission.

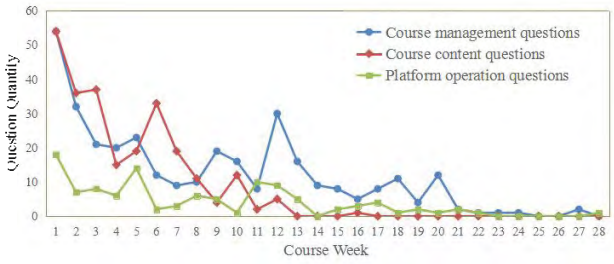


Figure 2: Question quantity of three categories in every course week

To summarize, through the analysis of students' questions in the forum, we can learn the patterns of their questioning behavior and in turn improve instructions in MOOCs. Instructors need to focus on certain kind of questions during different periods and provide appropriate guidance and answers. Course management questions and platform operation questions will influence learners' learning progress, so instructors should clearly describe details of course arrangement to avoid misunderstanding and confusion. When platform errors occur, they need to solve the problem as quickly as possible or give suggestions to learners. As to the course content questions, even without instructors' replies, learners and peers will try to discuss and find answers by themselves. So the main task of instructors are guiding their discussion and giving answers at the proper time.

The current study is part of a larger project studying the long-term impact of question asking/answering in MOOCs. We expect a significant relation between student's completion rate and the way students questioning/answering behaviors. Further study will be reported in the future.

4. REFERENCES

[1] Li, H., Duan, Y., Clewley, D. N., Morgan, B., Graesser, A. C., & Shaffer, D. W., et al. 2014. *Question Asking During Collaborative Problem Solving in an Online Game Environment. Intelligent Tutoring Systems*.

[2] Colbert, J. T., Olson, J. K., & Clough, M. P. 2007. Using the web to encourage student-generated questions in largeformat introductory biology classes. *Cbe Life Sciences Education, 6*(1), 42-48.

[3] Harper, K. A., Etkina, E., & Lin, Y. 2003. Encouraging and analyzing student questions in a large physics course: meaningful patterns for instructors. *Journal of Research in Science Teaching, 40*(8), 776-791.

[4] Graesser, A. C., & Person, N. K. 1994. Question asking during tutoring. *American Educational Research Journal, 31*(31), 104-137.

[5] Brinton, C. G., Chiang, M., Jain, S., Lam, H., Liu, Z., & Wong, F. M. F. 2014. Learning about social learning in moocs: from statistical analysis to generative model. *IEEE Transactions on Learning Technologies, 7*(4), 346-359.

Tutorials

Real-time programming exercise feedback in MOOCs

Zhenghao Chen, Andy Nguyen, Amory Schlender, Jiquan Ngiam
Coursera
381 East Evelyn Ave
Mountain View, CA, USA
{zhenghao, anguyen, aschlender, jngiam}@coursera.org

ABSTRACT

We present an active learning system for coding exercises in Massively Open Online Courses (MOOCs) based on real-time feedback. Our system enables efficient collection of personalized feedback via an instructor tool for automated discovery and classification of bugs.

1. INTRODUCTION

Active learning is a learning approach that "requires students to do meaningful learning activities" in contrast to traditional lecture-based approaches where "students passively receive information from the instructor" [2]. In active learning, timely feedback is important as it helps learning and reduces the risk of learner disengagement due to repeated failure to complete learning activities.

MOOCs have leveraged in-videos quizzes as an active learning strategy, but these quizzes have traditionally been limited to multiple choice questions. One reason that introducing higher order tasks, such as coding exercises, has been challenging is that it is difficult to provide good feedback. Most automated code grading systems allow for efficient grading through unit testing, but these methods are often limited in the forms of feedback they can provide.

Feedback that helps learners understand their errors can improve learning outcomes. Stamper et al. [5] demonstrated significant problem completion rate improvements in a logic course when feedback was available to learners. This has motivated related developments in data-driven methods to generate such feedback [3, 4, 1].

In this demo, we will show a system that enables instructors to efficiently generate and provide real-time feedback for programming exercises in MOOCs through extensions to Executable Code Blocks (ECBs) [6] and the Codewebs engine [1]; these exercises can be embedded throughout the learning experience to enable rich active learning.

2. EXECUTABLE CODE BLOCKS

Executable code blocks (ECBs) [6] enable learners to write and execute code directly in their web browser. The primary advantage of ECBs is that they can be tightly integrated into the course experience. For example, immediately after a concept is explained in a video, a learner can be asked to implement the specific concept in an ECB.

ECBs usually employ unit testing strategies to evaluate if a learner's implementation is correct. We extend ECBs such that when a learner makes an incorrect submission, they can request additional feedback that highlights potential errors in their submission and provides hints that guide the learner towards correcting these errors (see figure 1). These hints are provided efficiently by an instructor through an extension of the Codewebs engine.

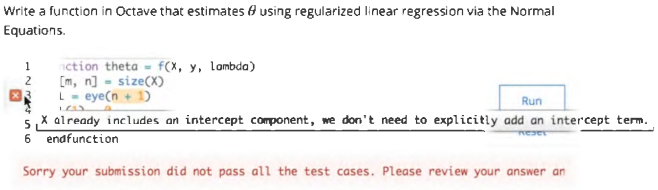


Figure 1: Hints provided in an ECB for an incorrect submission.

3. CODEWEBS ENGINE

We use the Codewebs engine [1] to localize errors in learner code submissions and identify common classes of errors. We describe here the relevant process of doing so automatically at a high level, and refer the reader to [1] for details.

The Codewebs engine operates on the abstract syntax tree (AST) representation of code submissions. Let n be a node in the AST, T_n be the subtree rooted at n, and P_n be the subtree rooted at the *parent of* n. The local context of T_n, denoted by T_n^c, is P_n with T_n removed (see figure 2).

We say that T_n^c is a buggy context if submissions containing T_n^c are more likely to be incorrect than by random chance. The Codewebs engine declares that P_n is a bug if T_n^c is a buggy context but no subtree of T_n has a buggy context. Given a bug P_n, the Codewebs engine then searches for a correction C such that replacing P_n with C results in a correct program.

We extend Codewebs in two ways. First, we modify the localization process to consider local contexts that are semantically equivalent[1]. This allows us to discover more bugs across submissions that might have *syntactically* distinct but *semantically* equivalent contexts. We also use this to improve correction discovery in a similar way (see figure 3) and improve correction searching to handle instances where multiple bugs occur within a submission.

Second, we introduce the concept of bug groups or error modes. Two bugs B and B' belong to the same group *iff* B

[1]We follow the definition of semantic equivalence used in [1].

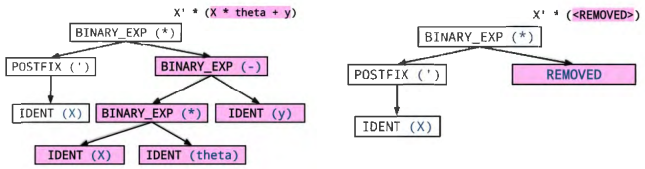


Figure 2: Left: Subtree P_n containing subtree T_n in pink. Right: T_n^c, the local context of subtree T_n.

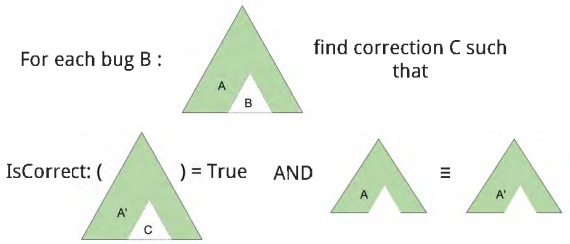


Figure 3: Visual illustration of finding corrections for bug B, C is a correction for B if we can find a correct submission where C is surrounded by A' and A' is semantically equivalent to A.

is semantically equivalent to B' and the correction for B is semantically equivalent to the correction for B'.

4. INSTRUCTOR ANNOTATIONS

By grouping bugs together, instructors can provide a hint for each error mode (instead of for individual submissions). These hints power the feedback features mentioned in section 2 (see figure 1).[2]

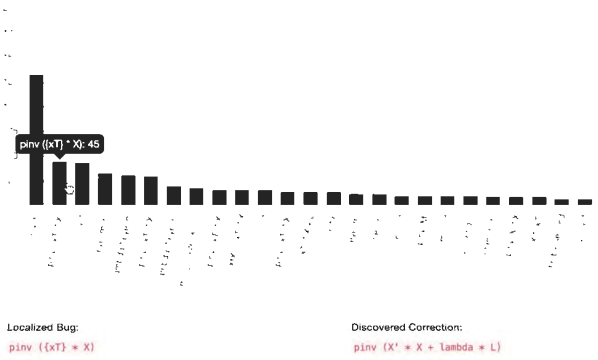


Figure 4: Instructor tool for exploring common errors based on bug equivalences classes.

Furthermore, we can provide instructors with a tool (see figure 4) to explore these common error modes. This tool orders bug groups by the frequency at which they appear in learner submissions. This enables instructors to quickly understand the most common errors made by learners. This breakdown is useful for course material improvement as they can expose common learner misconceptions.

5. RESULTS

We introduced 3 ECBs into the Machine Learning MOOC on Coursera involving tasks of varying levels of complexity (e.g., implementing the cost function for regularized linear regression). Each ECB required between 10 and 20 lines of code each to solve.

For each ECB we collected between 3,118 and 5,550 submissions, consisting of between around 1,000 and 3,000 distinct ASTs (see table 1). These submissions were used to train the Codewebs model. We find that a relatively small number of error groups (40) is required to achieve good coverage

[2]It is also possible to show learners automatically generated corrections when instructor input is not available.

	Submissions	% Correct	Unique ASTs	% Coverage (40 bug groups)
RLR Normal Eqn	3,118	52.8%	1,338	61.0%
Matrix Inv Cost Fn	3,892	19.5%	1,440	49.5%
Matrix Inv Grad	5,550	11.5%	3,050	36.7%

Table 1: 3 ECBs added to the Machine Learning MOOC on Coursera

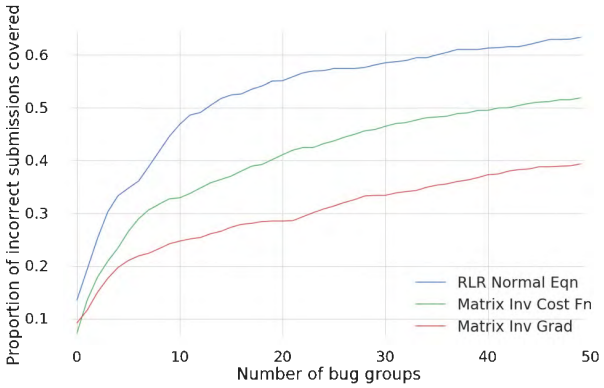


Figure 5: Percentage of incorrect submissions by number of error modes.

of a large fraction of incorrect submissions (see figure 5). Between 28.6% and 55.0% of incorrect submissions contain at least 1 of the 20 most common error modes, and between 36.7% and 61.0% contain at least 1 of the 40 most common error modes (see figure 5).

A teaching assistant was recruited to label the top 40 discovered error groups, and we are now running tests to understand the effects of this intervention on learning outcomes.

6. REFERENCES

[1] A. Nguyen, C. Piech, J. Huang, and L. Guibas. Codewebs: Scalable homework search for massive open online programming courses. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 491–502, New York, NY, USA, 2014. ACM.

[2] M. Prince. Does active learning work? a review of the research. *J. Engr. Education*, pages 223–231, 2004.

[3] 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*, 27(1):37–64, 2017.

[4] R. Singh, S. Gulwani, and A. Solar-Lezama. Automated feedback generation for introductory programming assignments. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '13, pages 15–26, New York, NY, USA, 2013. ACM.

[5] J. C. Stamper, M. Eagle, T. Barnes, and M. Croy. *Experimental Evaluation of Automatic Hint Generation for a Logic Tutor*, pages 345–352. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[6] C. Wong. Active learning experiences with code executable blocks. `https://building.coursera.org/blog/2016/09/30/active-learning-experiences-with-code-executable-blocks/`.

Tutorial: Why data standards are critical for EDM and AIED

Robby Robson
Eduworks Corporation, Inc.
IEEE Learning Technology Standards
robby.robson@eduworks.com

Avron Barr
Aldo Ventures, Inc.
IEEE Learning Technology Standards
avron@aldo.com

Xiangen Hu
The University of Memphis
Central China Normal University
xhu@memphis.edu

SUMMARY

As EDM and AIED innovations proliferate, the ability for diverse products to consistently interpret each other's data will emerge as a critical issue. Formal data interoperability standards that enable diverse datasets to be curated, accessed, merged/compared and fruitfully analyzed will play a crucial role in research and in the successful mass adoption of products based on that research, as will standards that enable systems to produce data that can be mined by existing and yet-to-be-invented algorithms. Yet this important topic is often neglected by researchers and system developers, who naturally focus on the specific problems they set out to solve and do not consider how they can either contribute or consume data produced by other systems or how their innovations will fit into larger ecosystems. This tutorial is intended to:

- Raise awareness of the role of standards and their criticality for EDM and AIED;
- Provide participants with an understanding of the nature, status, and current activity of multiple international standards development effort relevant to educational data;
- Provide participants with insight into how they can beneficially apply standards and, in some cases, contribute to their development.

TOPICS

This tutorial will cover following topics:

- **Why schools, corporations, and government agencies require standards conformance in procurement**: How standards interact with regulations and requirements to facilitate the free exchange of information and data, to prevent "lock-in" and thereby lower costs, to ensure quality and minimal levels of functionality, and to protect the integrity and privacy of data.
- **How standards shape product categories and markets**: How standards can define functionality, product capabilities, and market segmentation. In many instances, standards determine which of a number of competing approaches will dominate. They can shape markets and lead to winners and losers and long-term consequences for producers, consumers, and researchers alike. There are obvious examples in areas such as telecommunications and manufacturing, but there are also examples in educational technology relevant to EDM and AIED.
- **How standards can support research and lower market entry barriers for innovative products**: How standards make it possible for innovative component technologies to be independently developed without requiring a vertical monopoly, and how they support research by making it possible for data produced by one system to be understood by another.
- **Types of standards (governance, process, and data interoperability)**: People often think of standards as relevant only to technical interoperability, e.g. to determining data formats, sizes, shapes, tolerances, and the like. But there are other types of standards as well, including process standards such as ISO 9001 and Software Engineering Standards and governance standards that address issues such as data preservation, curation, ethics, and privacy. All of these will play a critical role for EDM and AIED.
- **International standards organizations**: A survey of standards development organizations (SDOs). This segment will briefly explain the structure of international standardization, the principles by which ISO, IEC, IEEE, W3C, and similar SDOs abide (openness, consensus, balance, due process, right of appeal), the differences (and similarities) between these and industry consortia, and the SDOs that are most relevant to EDM and AIED.
- **How standards are made**: The standards development process has been refined over many years to ensure that each SDO can be productive within its principles and goals. This segment will describe how standards development works so that participants have an idea of what it entails and how to participate.
- **A brief history of standards related to educational and training technology**: Starting circa 1996, various organizations and consortia began developing standards, some better known and more widely adopted than others. We will briefly survey this history with a view towards extracting some key "lessons learned" that apply generally to standards development: The perfect is the enemy of the good; standards are a poor way to define systems but a great way to define how they interoperate; simplicity and modularity leads to adoption; industry participation is vital; and how to avoid standards wars.
- **Current international standards activity relevant to EDM and AIED**: This is a major segment that will touch on a large number of relevant standards, including:
    - Metadata standards
    - Format standards (e.g. data shop)
    - Competency and learner information standards
    - Data reporting and curation standards

- Platform standards
- Big data and AI ethics
- Student data governance
- Possibly needed additional standards

Each standard will be summarized and described in terms of what problem(s) it solves, how it works, who developed it, who uses it, how it fits in with other standards, and what the presenters see as its future.

- **Tools for applying standards to EDM and AIED**: This segment will focus in on a few high-value standards and applications of standards to EDM and AIED. This segment is the punchline of the tutorial and will cover the standards that the presenters feel are most important. It will focus on existing or emerging technologies that participants can apply now or in the near future and will provide concrete examples of how standards are applied in software.
  - Using standards to report and collect data
  - Data set efforts (Datashop, Dataport)
  - The US DoD's Total Learning Architecture and related unification efforts
- **How to get involved in the standards development process**: This last, short segment will provide participants with information on how to get involved if they are interested, to be followed up offline.
- **Questions and Answers**: Adequate time will be set aside to address participants' questions and issues.

Presenter Relevant Bios:

- http://transformingedu.com/speakers/avron-barr/
- http://eduworks.net/robby/
- http://www.xiangenhu.info/

Tutorial: Principal Stratification for EDM Experiments

Adam C Sales
University of Texas College of Education
1912 Speedway Stop D5700
Austin, Texas, USA
asales@utexas.edu

ABSTRACT

Principal stratification (PS), which measures variation in a causal effect as a function of post-treatment variables, can have wide applicability in educational data mining. Under the PS framework, researchers can model the effect of an intelligent tutor as a function of log data, can account for attrition, and study causal mechanisms. Participants in this tutorial will learn how and when PS works and doesn't work, and will learn three methods of estimating principal effects.

1. PRINCIPAL STRATIFICATION IN EDM RESEARCH

Educational data miners are increasingly interested in causal questions—what interventions work, for whom, and how. Accompanying this interest is the widespread realization that there is no such thing as "the effect": actually, effects can vary widely between individuals. Estimating the differences in effects between types of learners is (in principal) straightforward for types defined prior to the onset of an experiment. But what about learners who use the software in different ways—or, even given the opportunity, don't use it at all? Traditionally, "post-treatment" variables, observed subsequent to treatment assignment, are treated as mediators whose analysis requires the kind of untestable assumptions randomization is supposed to avoid.

Principal stratification (PS) [2] offers a different approach: categorizing learners based on how they *would* (or would not) use the software if given the opportunity. Under the PS approach, an analyst begins by defining types, or "principal strata" of learners based on post-treatment measurements, then estimates the probability each learner is a member of each stratum (conditional on baseline covariates), and finally the average effect of the treatment within each stratum. In a randomized experiment, the final step of the process proceeds from the randomization (and, possibly, testable modeling assumptions). That is, researchers need not assume unconfoundedness, or that all relevant variables have been measured. The result is a principal effect, or separate estimate of an average treatment effect for each usage mode of interest; these may be used to explore causal mechanisms, study the conditions under which software might work better (or worse), learn dosage effects (i.e. does more usage translate to larger effects), and many other applications.

1.1 EDM Questions PS may Help Answer

PS could help address a wide range of research questions in EDM. Some examples are:

- Does the effect of an intervention depend on learners' (measured) emotional state?
- Are some sections of a software more effective than others?
- Do some learner strategies—such as hint usage or mastery learning—correspond to larger effects than others?
- Are there intermediate outcomes, such as mastery speed or error rate, that can serve as good surrogates for a final outcome, such as a post-test?
- Estimating treatment effects after attrition

Each of these questions estimates an average treatment effect for a group of learners which is defined based on variables measured only after the intervention began. This is the type of question principal stratification was designed to answer.

1.2 Estimating Principal Effects

The catch is that principal effects can be difficult to estimate. Estimating effects within principal strata depends on knowing who is in which stratum—for instance, which students in the control condition *would have* been frustrated, had they been assigned to treatment, or which students would have attritted, had they been assigned to the opposite condition—which is unobserved and must be inferred. The most popular and powerful approach begins by assuming a model (typically the normal distribution) for the outcome within each stratum and a model for who is in which stratum (typically logistic regression). Next, it fits a mixture model for those subjects with unobserved stratum membership. For instance, in an experiment comparing students assigned to use an intelligent tutor with students assigned to

use traditional curricula, a researcher looking to estimate average effects for high-hint users might model post-test scores for subjects in the control condition as a mixture of two distributions: one for students who would use many hints, and one for students who would not. The success of this approach depends on the fit of the model—misspecified models may yield misleading results—so extensive model checking is necessary. Further, even when the model is correctly specified, its success can depend on factors beyond the researcher's control [1].

Two other approached depend less on modeling assumptions, but may yield less precise estimates. One approach [3] estimates bounds for principal effects, rather than estimating the effects themselves. Another [4], applicable in some PS studies but not others, uses non-parametric techniques to identify plausible candidates for unobserved principal strata, and estimates effects based on those. These approaches are more "automatic" than the model-based approach, in that they do not require careful model fitting and checking, but still require researchers to specify the problem carefully.

1.3 My Expertise

For the past three years, I have been working on an NSF-funded project to use the PS framework to study data from the Cognitive Tutor Algebra I effectiveness study. With Dr. John Pane of the RAND Corporation, I have estimated various associations between Cognitive Tutor treatment effects and student usage. This has produced two EDM proceedings papers, [5] and [6]. As part of the project, I have developed a new method for estimating principal effects which expands on [4] and set of new diagnostic and model checking techniques. I have also worked extensively with Neil Heffernan's lab using PS to model data from ASSISTments experiments.

2. TUTORIAL PLAN

2.1 Introduction to Principal Stratification

The beginning of the tutorial will introduce the PS framework. First, we will discuss why principal stratification is necessary: participants will learn to distinguish post-treatment from pre-treatment variables and understand the conceptual and methodological issues with conditioning causal inference on post-treatment variables. Next, we will describe PS framework, so participants understand how it solves the problems with post-treatment conditioning. Finally, we will discuss methods for estimating effects within principal strata: what assumptions they depend on and the source for their identification. We will give a brief overview of the various PS methods that we will explore hands on, in more depth, during the remainder of the tutorial.

2.2 Hands on PS Estimation

The second half of the tutorial will focus on three classes of methods to estimate principal effects: nonparametric bounds, nonparametric randomization inference, and model based PS.

I will provide two real EDM datasets that participants can use for exercises. The first will be a subset of the data from the Cognitive Tutor effectiveness study, comparing subjects assigned to use the Cognitive Tutor to those assigned to traditional curricula. The study produced rich log-data—PS can be used to compare treatment effects between sets of learners who used, or would have used, the tutor differently. The second dataset will come from an experiment run on the ASSISTments platform [7]. I will also give participants the opportunity to bring their own datasets to the tutorial.

The methods will be taught in `R`, a free, open-source language for statistical computing. We will begin with a brief introduction to the software: how to read in data, and how to write and execute simple code.

The bounding portion will be based on [3], which describes a set of bounds on principal effects, depending on available covariates and certain identification assumptions. We will set out a number of real or realistic data scenarios and discuss which bounds may be appropriate when. Next, we will use `R` to calculate the appropriate bounds for principal effects.

The randomization inference portion will be based on [4] and extensions I have developed. They depend on the assumption of monotonicity—that principal stratum membership is directly observable for all members of either the treatment or the control group. I will provide code in `R` to estimate confidence intervals for principal effects with and without covariates the predict stratum membership.

The model based portion will use Bayesian methods, with the JAGS language, via `R` and the `R2Jags` package. We will practice estimating principal effects with pre-written JAGS code (which I will explain) as well as discuss diagnostic tools: model checking, convergence diagnostics, and small simulation studies.

References

[1] A. Feller, E. Greif, L. Miratrix, and N. Pillai. Principal stratification in the twilight zone: Weakly separated components in finite mixture models. *arXiv preprint arXiv:1602.06595*, 2016.

[2] C. E. Frangakis and D. B. Rubin. Principal stratification in causal inference. *Biometrics*, 58(1):21–29, 2002.

[3] L. Miratrix, J. Furey, A. Feller, T. Grindal, and L. C. Page. Bounding, an accessible method for estimating principal causal effects, examined and explained. *arXiv preprint arXiv:1701.03139*, 2017.

[4] T. L. Nolen and M. G. Hudgens. Randomization-based inference within principal strata. *Journal of the American Statistical Association*, 106(494):581–593, 2011.

[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.

[6] A. C. Sales, A. Wilks, and J. F. Pane. Student usage predicts treatment effect heterogeneity in the cognitive tutor algebra i program. In *Proceedings of the 9th International Conference on Educational Data Mining*, 2016.

[7] D. Selent, T. Patikorn, and N. Heffernan. Assistments dataset from multiple randomized controlled experiments. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 181–184. ACM, 2016.

Whitebox: A Device To Assist Group Work Evaluation

Daisuke Yukita
Keio University Graduate School of Media Design
4-1-1 Hiyoshi Kohoku
Yokohama, Japan
daisuke@kmd.keio.ac.jp

ABSTRACT

With the growing trend of Active Learning, group work is becoming increasingly common among education of all ages. Among the many advantages of group works, we have also witnessed how difficult it is for teachers to keep an eye on the activities within each group, thereby turning the group work process itself into a black box from the teachers' perspective. In order to propose a solution for this problem, this study introduces Whitebox, a device that discreetly gathers several types of data within group work, which are then visualized for the teacher to reference after the group work. The user study with high shcool students showed that group work analysis by Whitebox led to deeper understanding of how each student performed within their group.

1. INTRODUCTION

Considering the fact that there can be more than 30 students in a typical high school class in Japan, it is highly difficult for teachers to look over the activities within each group during group work. In other words, the students' processes of their group work remain a blackbox for teachers. In addition, how we evaluate group work is still an often debated issue, especially in formal education where a standard evaluation method is required. Whitebox was developed in order to suggest a solution towards such obstacles for schools in adopting group work. By placing the Whitebox in the middle of a group work table, it tracks the activities within the group. Later the recorded data will be visualized for the teacher to check, enabling teachers to get a rough idea of what kind of process each group went through without being physically present all the time. Furthermore, Whitebox quantifies the group work process by measuring talking ratios, volumes, etc., suggesting novel evaluation measurement units for group work, which can be used as the future standard.

2. LITERATURE REVIEW

While many of EDM / LA related researches have been limited to online or digital learning environments, recent studies have stepped in to face-to-face classroom activities with the help of advanced sensors and devices. Martinez-Maldonado et al. [1] created a realtime feedback system for teachers to provide feedback just at the right time using the data obtained from MTClassroom, a multi-touch tabletop that analyzes the strategies of student groups. Evans et al. [2] also proposed to identify touch patterns of students on an interactive tabletop to analyze the quality of collaboration. Whitebox aims to provide similar feedback to the teachers without relying heavily on each hardware. In terms of providing measurement units for conversation and collaboration, Lederman et al. [3] proposed Open Badges, an open source toolkit to measure face to face interaction and human engagement in real-time with custom hardware. Olguin et al. [4] states that such sociometric badges can make group collaborations more efficient by providing context, but such badges are mainly used for business and work environments, and they must be designed alongside students and teachers if it were to be used in a classroom setting.

3. SYSTEM DESCRIPTION

Initially, Whitebox used Kinect's mic arrays to determine which direction the audio is coming from, thereby distinguishing who is currently speaking. Following the feedbacks from a pilot test, however, audio recording was also done with separate pin microphones attached to the students' clothing. The attained audio is processed to obtain the volume as well. Using Kinect's depth camera, Whitebox also obtains the participants' body skeletons, allowing it to track their hand coordinates and their posture angles. Due to the way the current system is designed, Whitebox can only track the participants' data when they are sitting down and are not moving around or switching positions. The entire group work is also recorded, and when the group work is finished the audio data is converted into text using Google Cloud Speech API.

4. USER STUDY

A user study was conducted during a 4 day Design Thinking workshop at Tokyo Metropolitan College of Industrial Technology high school. In this user study, we especially focused on one group of four students, student A, B, C and D, and recorded only those 4 students' activities. After the workshop, the 4 visualizations and speech-to-texts were shown to both teachers and students separately, followed by an hour long discussion each on what those data meant to them.

Figure1 shows the study setup.

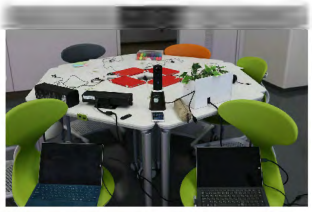


Figure 1: User study setup

The data acquired from the 4 workshops was processed, then visualized in to A4 infographic posters as shown in Figure 2.

Figure 2: visualizations from user study

To provide a more fine grained analysis of each session, we also provided additional visualization that plotted the students' audio data, posture data and hand position data along the timeline of the workshop. Figure 3 is an examples of the additional visualization.

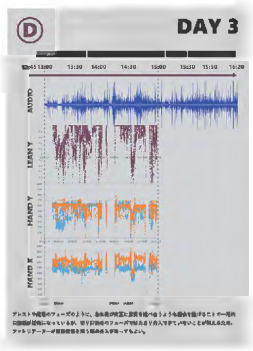


Figure 3: additional visualization of student D from day 3

By visualizing the data from all four sessions, it was possible to get a grasp of how each student behaved in the workshops. It is important to note here that what the visualizations suggested matched with the thoughts of facilitators who were in charge of this group (e.g. that student D would speak the least and student B would take charge of the overall discussion), meaning that Whitebox would be able to assist teachers to evaluate group work without them having to be present at each group's table all the time. As for the speech-to-text, it helped the teachers to see what words were mentioned most frequently. With improved conversion accuracy, it would become possible to process the text to search the most frequenty mentioned conjunctional phrases per student in order to see the characteristics of their contributions.

By post processing the audio data recorded, we were also able to provide visualizations on the order of conversational turn taking during the discussion. The data was plotted for each 30 seconds of conversation. This enables the teacher to examine specific points in a discussion and analyse how it transitioned between the group members. An example is shown in Figure 4.

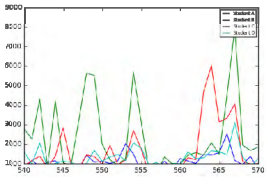


Figure 4: conversation transition

5. CONCLUSION

In this study we proposed Whitebox, a device that tracks the activities within a group work. Through the discussions with the teachers, we were able to see that Whitebox analysis certainly functioned as a guideline for a deeper understanding of the group and its students, and it also functioned as signs for what was and was not working in the group work, ultimately leading to improvements in the design of the class. Although not all the data we recorded seemed useful to the teachers, the measurements that Whitebox proposed, especially talking ratios, volumes and posture were valuable information for the teachers, uncovering the activities within the group that they otherwise would have missed. By using these measurements continuously, they can become a standard measurement unit in assessing group work.

6. REFERENCES

[1] R. Martinez-Maldonado, A.Clayphan, K.Yacef and J. Kay. "MTFeedback: providing notifications to enhance teacher awareness of small group work in the classroom." IEEE Transactions on Learning Technologies, 8(2): 187-200, 2015.

[2] A.C. Evans, J.O. Wobbrock, K.Davis. "Modeling Collaboration Patterns on an Interactive Tabletop in a Classroom Setting." Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work Social Computing, 860-871, 2016.

[3] O. Lederman, D. Calacci, D.C. Fehder, F.E. Murray, A. (Sandy) Pentland. "Open Badges: A Low-Cost Toolkit for Measuring Team Communication and Dynamics ." 2017 International Conference on Social Computing, Behavioral-Cultural Modeling, Prediction and Behavior Representation in Modeling and Simulation, 2017.

[4] D.O. Olguin, A. (Sandy) Pentland . "Sociometric Badges: State of the Art and Future Applications." IEEE 11th International Symposium on Wearable Computers, 2007.

Understanding Student's Reviewing and Reflection Behaviors Using Web-based Programming Grading Assistant

Yancy Vance Paredes
Arizona State University
699 S Mill Ave
Tempe, AZ 85281
yvmparedes@asu.edu

Po-Kai Huang
Arizona State University
699 S Mill Ave
Tempe, AZ 85281
phuang24@asu.edu

I-Han Hsiao
Arizona State University
699 S Mill Ave
Tempe, AZ 85281
sharon.hsiao@asu.edu

ABSTRACT
Paper-based assessment is still one of the most preferred methods in assessing students in a blended learning environment. However, it has several drawbacks such as having a high turnaround time before feedback is provided to the students. Furthermore, understanding how students attend to their graded papers is difficult to investigate because of the absence of empirical evidence. We describe in this paper a web-based system we developed that addresses some key issues when trying to understand the reviewing and reflection behaviors of the students. This system also aims to help instructors to efficiently and effectively grade paper-based assessments.

Keywords
Reviewing Behavior, Paper-Based Assessment, Educational Technology

1. INTRODUCTION
Paper-based assessment is still one of the most preferred methods in assessing students in a blended learning environment. Aside from being convenient to prepare, the possibility of students committing academic dishonesty is lower. However, it also has its drawbacks. Evaluating large amounts of test paper gives rise to the possibility of inconsistency among or even within graders [2]. Additionally, the feedback is limited [5]. Moreover, there is a high turnaround time before students receive their graded papers [1]. In terms of understanding the reviewing and reflecting behaviors of the students, it is difficult to systematically estimate how students review their paper-based assessments because of the absence of empirical evidence. It is not possible to determine whether students really do review their graded test papers. Thus, it is challenging to estimate the impacts of reviewing on learning.

2. WEB-BASED PROGRAMMING GRADING ASSISTANT (WPGA)
A web-based system was developed to address the above-mentioned issues. More specifically, it is designed to help students to review effectively. In addition, it aims to help instructors to efficiently and effectively grade paper-based assessments. The name of the system is Web-based Programming Grading Assistant (WPGA). The system is capable of capturing all activities performed by the users, which is mostly comprised of students' clickstream.

2.1 Documentation of Paper-Based Assessments
WPGA uses quick response (QR) codes to label the paper exam of a student. These generated codes are manually placed on the students' papers prior to scanning. Using an automatic document feeder, all the papers are scanned and uploaded to the system. The system automatically associates the scanned image to the corresponding student and the corresponding assessment. There are instances where the system may not accurately associate an image to a student. One possible reason would be due to the QR code being not readable. It could also be because the student is not registered in the system. When this happens, the instructor can just manually label the images.

2.2 Interface for Grading Assessments
After the exams are digitized, instructors can distribute the questions to be evaluated by different graders. The system allows multiple graders to work on the same assessment simultaneously. In effect, the turnaround time in the distribution of grades is reduced. The grading coherence will improve since graders will only be working on the question assigned to him or her.

Figure 1. The grading interface of WPGA

The grading interface is shown in Figure 1. Buttons on the upper right portion represent a learning concept or a rubric that is used to evaluate a question. Every rubric default to a perfect score, which

translates to a full understanding of the concept. Whenever the button is clicked, the grade for the rubric is decremented and the overall score is recalculated. Also, the color of the button changes depending on the grade for the rubric. It could be blue (full understanding), red (partial understanding), or grey (missed the concept). The overall score can also be overridden, if necessary. The graders can also add markings on top of the student's paper. This will enable them to highlight the mistakes. Lastly, using the comment section, the graders can provide free form feedback. In previous studies [2,3], we found out that graders prefer to type their feedback rather than physically writing them on paper. One advantage of this over the traditional way of checking is the ability to copy and paste feedbacks of common and similar mistakes.

2.3 Interface to Encourage Student Reflection

After the instructor publishes the results of an assessment, the students can log in to the system and review it. There are two levels how the students can view the results: assessment level and question level. In the assessment level (shown in Figure 2), the general result is displayed. This includes the overall score obtained by the student along with the individual scores for each question. In the question level (shown in Figure 3), a detailed feedback for the particular question is provided. This includes the scores for all the rubrics, the markings on the student's paper, and the free form text provided by the grader.

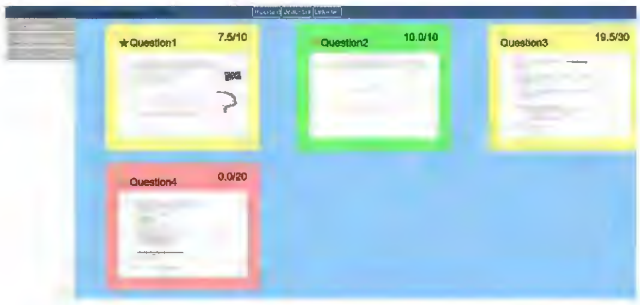


Figure 2. The assessment level view of the student interface

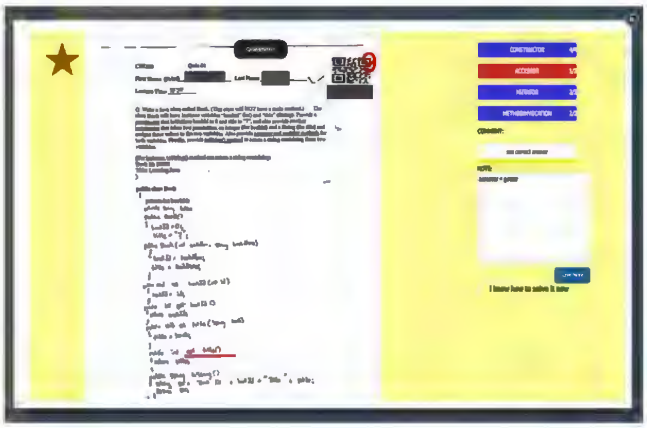


Figure 3. The question level view of the student interface

In addition to letting the students access a digital copy of their paper assessments, the system also allows them to reflect on the feedback given to them by the graders. We incorporated some features that help students track and monitor their learning. For example, in the question level, there is a checkbox that the students can tick to indicate whether they already know how to solve the problem after reviewing it. This is particularly useful for questions where they committed mistakes. Another feature is the bookmark which enables students to highlight the importance of a question. This could be used in future targeted reviews along with the use of filters. We also provided a free form text area to allow the students to type in his or her personal notes. The collection of these bookmarks, checkbox ticks, and notes are externalization of what the student knows. Through these features, it is hoped that students will be encouraged to reflect on their answers.

3. CASE STUDY

Using the system, we designed a classroom study and analyzed the logs collected from an *Object-Oriented Programming and Data Structures* class. We tracked and modeled students' reviewing and reflecting behaviors. Results show that students demonstrated an effort and desire to review assessments regardless whether they are graded or not [4].

4. FUTURE WORK

We intend to improve the system by using the feedback obtained from the users. For the next iteration, we are integrating the analytics module that will enable the instructors to quickly see a snapshot of the class performance and will enable them to gain insight on the assessments they gave to the students. Furthermore, we intend to do more research in understanding the reviewing behaviors of the students. This would allow us to create personalized review sessions that will help students do effective reviews.

5. REFERENCES

[1] Susan A. Ambrose, Michael W. Bridges, Michele DiPietro, Marsha C. Lovett, and Marie K. Norman, *How Learning Works: Seven Research-Based Principles for Smart Teaching*.: John Wiley & Sons, 16 April 2010.

[2] I.-Han Hsiao, "Mobile Grading Paper-Based Programming Exams: Automatic Semantic Partial Credit Assignment Approach," in *Lecture Notes in Computer Science*., 2016, pp. 110-123.

[3] I.-Han Hsiao, Sesha Kumar Pandhalkudi Govindarajan, and Yi-Ling Lin, "Semantic visual analytics for today's programming courses," in *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge (LAK'16)*, 2016.

[4] I.-Han Hsiao, Po-Kai Huang, and Hannah Murphy, "Uncovering reviewing and reflecting behaviors from paper-based formal assessment," in *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*, 2017, pp. 319-328.

[5] Hannah E. Murphy, "Digitalizing Paper-Based Exams: An Assessment of Programming Grading Assistant," in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, New York, NY, USA, 2017, pp. 775-776. [Online]. http://doi.acm.org/10.1145/3017680.3022448

Doctoral Consortium

A Framework for the Estimation of Students' Programming Abilities

Ella Albrecht
Institute of Computer Science
University of Goettingen
Göttingen, Germany
ella.albrecht@cs.uni-goettingen.de

ABSTRACT

In times of increasing numbers of students and high usage of e-learning systems, student models are a good way to get an overview of what is currently occurring in the classroom, analyze students' behavior and estimate their learning progress. In our work, we develop a framework which estimates a student's programming knowledge by looking at his responses to open-ended programming assignments. The model we construct incorporates multiple applications of multiple skills in one exercise, multiple submissions and varying knowledge components involved in the same exercise.

1. INTRODUCTION

During the last years, the number of students has increased rapidly. Especially in introductory courses, hundreds of students are attending. This makes it infeasible for educators to take care of each student individually. On the other hand, to deal with large amounts of students, many institutes use e-learning and e-assessment systems to support their teaching. These systems allow large data collection on which data mining and learning analytics techniques can be applied to build student models. Student models are used to estimate a student's cognitive state, e.g., his/her motivation, knowledge, misconceptions or learning style and preferences [4]. A student model can be used to provide students personalized course material fitting to their current knowledge and learning habits. Furthermore student models can be used to predict student's performance and identify students which are at risk to intervene in a timely manner. Besides, we can use a student model to identify problematic course contents. This knowledge can be used as a basis for restructuring and redesigning the course.

In our research, we want to develop a framework for the estimation of student's knowledge regarding programming. Therefore, we look at students' solutions to open-ended programming exercises. For each exercise, it is defined which knowledge components (KC) are required to solve the exercise correctly. KCs describe the individual components of knowledge which are required to solve a particular task or problem. The task in an introductory programming course is to learn to write simple programs which meet the specifications given in text form, i.e., the exercise description. Therefore KCs can be, e.g., the programming language's constructs, i.e., syntax and semantics, correct usage of a compiler or IDE, error understanding and debugging ability, or the translation of specifications to program code. Then, it is checked whether the student has applied the KCs in his/her solution correctly. From theses observations a student model can be constructed which is able to estimate a student's knowledge state.

2. PROBLEM STATEMENT

Knowledge cannot be assessed directly, because there may be several reasons why a student made a mistake. For example, a missing `break` in a `switch-case`-block may be just due to sloppiness, because the student does not know the `break`-statement, or because the student does not understand how the commands in a `switch-case`-block are executed. Because of these uncertainties often probabilistic models are used for student modeling.

Bayesian Knowledge Tracing (BKT) [5] is one of the most widely spread student modeling approaches. It uses Hidden Markov Models to model students' learning. It was at first applied to programming exercises for LISP in the ACT Programming Tutor. The domain knowledge was represented by production rules of the form "to achieve goal X do Y" where Y may be a subgoal. The knowledge of a student was described as the probability that the student knows a rule. Since there was a deterministic order of which rules need to be applied to solve an exercise correctly, the student's knowledge could be estimated by looking at the student's solutions rules order. But in imperative or object-oriented languages like C, C++, or Java one can only extremely rarely define a deterministic order of statements.

Kasurinen and Nikula [7] have applied BKT on students' results to Python exercises. As domain knowledge they have defined guidelines for preferred solutions, e.g., each open file should be closed. Moreover, they have checked whether the student has used the guideline in his/her solution. However, the set of KCs was very limited.

Berges and Hubwieser [2] as well as Yudelson et al. [10] used the Rasch model from Item Response Theory (IRT) to estimate student's knowledge of object-oriented concepts in Java instead. In IRT, the relationship between responses to items, i.e., exercises, and a latent trait, i.e., an ability or KC, is described as a logistic function. Different from BKT,

it also takes the difficulty of an item into account.
BKT as well as IRT have the main drawback that they are single skill models, i.e., for each KC a separate model is constructed, and it is assumed that each exercise only requires one KC. For programming assignments, this assumption is of course not sustainable. Performance Factor Analysis (PFA) [8] is able to deal with multiple skills per exercise but as BKT and IRT also does not consider dependencies between KCs. However, in the programming domain there are dependencies between KCs, e.g., one needs to know how assignments or incrementing works when using a `for`-loop, or that the knowledge of a `while`-loop can influence the knowledge of a `for`-loop. It was also shown that integrating dependencies of knowledge into a student model can improve the model [3, 6]. Another special property of programming assignments is that KCs can be required multiple times in one exercise, e.g., if multiple loops are needed to solve the exercise. We also want to investigate the influence of substeps during the solution process to a model's accuracy. To the best of our knowledge, there does not exist a modeling approach so far which fulfills all of the requirements for programming assignments we have stated above.

3. RESEARCH METHODOLOGY AND APPROACH

Before we can make use of a student model in a course, several steps have to be taken. First, we need to identify what we expect the students to learn in our course, i.e., which KCs shall be acquired. In the first iteration of our research, the KCs we want to use for our model are the concepts of the programming language, e.g., `if`, `for`, variables, arrays etc., rules for good programming practice, e.g., each declared variable shall be used, allocated memory has to be freed, etc., as well as the fulfillment of the specifications by checking whether the program produces the correct output.
In a second step, we need to know which KCs are required to solve a particular exercise as we want to build our student model from the data we gain from their solutions to programming assignments. For example, summing up the numbers from 1 to 100 requires among others the knowledge of loops or recursion. This example also shows us, that it is actually not that easy to define which concrete concepts are really mandatory to solve the exercise as we could write a correct solution without knowing loops if we know recursion and vice versa. In our work, we develop a knowledge requirements model (KRM) which models required KCs related to language concepts for a particular exercise. The general mapping of language constructs, e.g., elements of an abstract syntax tree (AST), to concrete KCs has to be done beforehand by a domain expert. The KRM for a particular exercise is learned automatically from different correct solutions to that exercise based on their ASTs and structural analysis. We divide correct solutions into blocks and determine the set of KCs used in the block. From these sets we construct a tree where each path describes an alternative solution. By comparing a student's solution to the KRM, one can get the KCs which were applied correctly, incorrectly or are missing in the student's solution.
Despite the comparison with the KRM, we also use compiler and static analysis tool messages to assess the incorrect application of a KC, e.g., static analysis tools can deliver hints on, e.g., misunderstanding of control flow. Dynamic tests like unit tests, help us to evaluate a student's general program writing ability, i.e. whether a student is able to write a program which meets the specifications, i.e., does what it is intended to do.

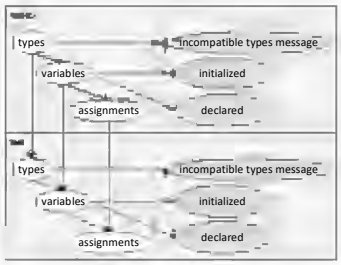


Figure 1: Example structure for a part of a DBN student model

The third step deals with the construction of the student model. We use Dynamic Bayesian Networks (DBN) for student modeling as they seem most appropriate to us. A DBN is a two-time-sliced Bayesian network where the state of a hidden variable depends on the states of the variables it depends on and the variable's state in the previous time step. Making observations in each time step updates the probability distribution of a hidden variable being in a particular state.
In our case, the hidden variables are the KCs, e.g., in Figure 1 the hidden variables (blank circles) are the concepts *types*, *variables*, and *assignments*. Observations in our student model are the results from the comparison of the student's solution with the KRM, compiler and static analysis tool messages as well as results from dynamic tests, e.g., in Figure 1 the observations (filled circles) are whether the student has declared and initialized a variable as well as whether an error message regarding incompatible types in an assignment appears. These variables can have the states *true* or *false*. With DBNs, we are able to deal with multiple KCs per exercise, their interdependencies, the uncertainty of which KC is affected by a certain observation and the uncertainty of which KCs are required to solve a particular exercise.
In our work, the structure of the DBN is defined manually by a domain expert. Though, one could also learn dependencies between KCs from data. The parameters of the DBN are learned from data using an expectation maximization algorithm with reasonable parameter constraints defined by an expert, e.g., limits for guess and slip probabilities. One problem that may occur, is that the parameter space is too large and we get computational problems when estimating the parameters of the model, if we use a very fine-grained KC definition. Therefore, we need to evaluate which granularity to choose to be able to estimate the parameters and still have an accurate model. Furthermore, we have to reason how to integrate multiple occurrences of the same KC in one exercise. Possible treatments are, e.g., majority vote or using uncertain evidences with a probability according to the ratio of correct/incorrect applications. We also want to analyze, whether multiple submissions, i.e., substeps preceding the final solution, improve the model.
In the second iteration of our research, we want to add further KCs which concentrate on more cognitive skills. The

first one is the debugging ability, which we want to assess by comparing two subsequent submissions when the first one indicates an error (or a failure) and check whether the problem was fixed.
As a further KC, we want to include variable roles [9]. Variable roles describe patterns of variable usage. They are defined by the successive values the variables obtain. An example for a role would be the most-wanted holder which is a variable that holds the best value encountered so far when going through a succession of values, e.g., when searching the smallest value in an array. The proper collocation of variable roles is essential for solving a task or achieving a goal in a program. Usually, students intuitively use variable roles in their programs. The lack of knowledge of a particular role could explain why a student may have problems to solve an exercise.
We want to evaluate our model by comparing it to common student modeling approaches like BKT, IRT and PFA.
In a last step, we want to analyze the model constructed from the data of our introductory C course to find out what students which are at risk have in common, which KCs seem most difficult to the students and how many exercises are required at least (on average, to reach a particular percentage of students) to gain sufficient knowledge in a certain KC.

4. CURRENT STATUS & NEXT STEPS

We have implemented a framework for the collection of metrics regarding students' solutions [1] which was successfully introduced in our introductory C programming course. It is mainly an e-assessment system where students can upload their solution and get some basic feedback. It collects compiler messages, results from static analysis tools, and results from dynamic tests to capture the correctness of the solution. In the first year, we got about 10,000 submissions of on average 250 students. We expect similar numbers this year.
Furthermore, we have identified the different KCs that we have in our course by going through the course material and previous programming errors of students. Based on that, we defined a hierarchical structure of KCs where the sinks are basic observations in form of rules like, e.g., the function returns a value if the return type is not void. We have also mapped compiler/static analysis tool messages to different concepts and implemented an AST parser. In a next step, we want to use the AST to filter the KCs from source code and construct our KRM.
Next, we plan to conduct a small case study with only a few KCs to evaluate the feasibility of our DBN student model.

5. EXPECTED CONTRIBUTIONS

In our work, we develop a framework for the estimation of students' knowledge regarding programming. One of our main contributions is the definition of a student model which has the following properties which are needed to construct the model based on solutions to programming assignments: multiple KCs per exercise are possible and their interdependencies are considered, uncertainty of affected KCs can be handled, individual KC requirements and usages can be treated, multiple submissions can be integrated, and a KC can be used multiple times in the same exercise.
Another contribution will be a KRM which is automatically generated from model solutions for each exercise and can be used to evaluate which KCs were applied correctly or incorrectly by the student.
Furthermore, we plan to not just look at language related KCs, but also more cognitive skills like, e.g., debugging ability. We hope that our model helps to get better insights into the learning process of students.
From the doctoral consortium we expect to get some feedback on our student model, especially hints for the evaluation w.r.t. metrics and data sets. We are also looking forward for further ideas for additional or alternative KCs which we can integrate in our model.

6. REFERENCES

[1] E. Albrecht and J. Grabowski. Towards a framework for mining students' programming assignments. In *2016 IEEE Global Engineering Education Conference (EDUCON)*, pages 1096–1100, 2016.

[2] 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*, pages 51–56, New York, NY, USA, 2015. ACM.

[3] A. Botelho, H. Wan, and N. Heffernan. The prediction of student first response using prerequisite skills. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*, pages 39–45, New York, NY, USA, 2015. ACM.

[4] K. Chrysafiadi and M. Virvou. Review: Student modeling approaches: A literature review for the last decade. *Expert Syst. Appl.*, 40(11):4715–4729.

[5] A. T. Corbett and A. Bhatnagar. *Student Modeling in the ACT Programming Tutor: Adjusting a Procedural Learning Model With Declarative Knowledge*, pages 243–254. Springer, Vienna, 1997.

[6] Y. Huang, J. Guerra, and P. Brusilovsky. A data-driven framework of modeling skill combinations for deeper knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining EDM*, pages 593–594, 2016.

[7] J. Kasurinen and U. Nikula. Estimating programming knowledge with bayesian knowledge tracing. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education*, pages 313–317, New York, NY, USA, 2009. ACM.

[8] 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, Amsterdam, The Netherlands, 2009. IOS Press.

[9] J. Sajaniemi. An empirical analysis of roles of variables in novice-level procedural programs. In *Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02)*. IEEE Computer Society, 2002.

[10] M. Yudelson, R. Hosseini, A. Vihavainen, and P. Brusilovsky. Investigating automated student modeling in a java MOOC. In *Proceedings of the 7th International Conference on Educational Data Mining EDM*, pages 261–264, 2014.

Student Use of Inquiry Simulations in Middle School Science

Elizabeth McBride
University of California Berkeley
Tolman Hall
Berkeley, CA, USA
bethmcbride@berkeley.edu

ABSTRACT

My research focuses on the integration of science and design through the use of interactive simulations and other scaffolding tools. I specifically look at patterns of use in interactive simulations. To conduct this research, I have developed a curriculum about solar ovens used by middle school students, during which students are guided by an online curriculum to design, build, and test physical solar ovens. This curriculum utilizes interactive simulations as a tool to help students plan the design for their solar ovens. I have evaluated scaffolding for the simulation steps, and plan to evaluate other patterns of student use, based on action log data.

Keywords

Interactive Simulations, Science Education, Inquiry, Log Data

1. RESEARCH TOPIC

My research focuses on the integration of science and design through the use of interactive simulations and other scaffolding tools. I specifically look at patterns of use in interactive simulations. I conduct this research in secondary schools, and work in collaboration with teachers. Through my dissertation work, I aim to answer the following questions:

- What types of use patterns in interactive simulations are beneficial for integrating science and design learning?
- How can we use tools to support integrated understanding in writing activities (e.g.,automated guidance)?

My work is situated in the learning sciences, using techniques from educational data mining and artificial intelligence to understand how students' activities impact their learning and how to improve the learning experience. Recently, I have used natural language processing to develop automated classifiers for multiple short response questions [6]. Using these classifiers, I plan to develop automated guidance for student writing during the curriculum, which will deploy during spring 2017. I have also studied student use of interactive simulations, using log data, feature engineering, and clustering to make sense of patterns (submitted to EDM 2017).

To conduct this research, I have developed a curriculum that is run using an online platform and offers students the opportunity to use interactive simulations while they design a physical artifact. In previous work, I have found that the simulation is beneficial, especially when students use it during the design phase of the curriculum [8]. My work has also been published in a variety of other conference venues [7, 10, 11, 9].

1.1 Curriculum

My research utilizes a curriculum about solar ovens that is run using the Web-based Inquiry Science Environment (WISE). During this curriculum, students design, build, and test a solar oven. They go through the design, build, test process two times to get an idea of how engineers iterate on their designs based on results from testing (Figure 1). This curriculum was designed using the knowledge integration framework [5]. The knowledge integration framework has proven useful for design of instruction featuring dynamic visualizations [14] and engineering design [1, 12]. The framework emphasizes linking of ideas by eliciting all the ideas students think are important and engaging them in testing and refining their ideas [5].

Students are allowed to use only a certain set of materials (e.g., tin foil, black construction paper, plastic wrap, Plexiglas, tape), in addition to a cardboard box they bring from home. Students use an interactive computer simulation to test the different materials in their oven. This simulation helps to elicit student ideas before they get to the building process, consistent with the knowledge integration framework. The testing portion of the project allows students to distinguish their ideas.

Throughout the project, students respond to short response questions about the choices they are making in their design and how their ovens work. This curriculum is unique, since it is guided by an online platform, but students also design, build, and test their solar ovens in a hands on portion of the project.

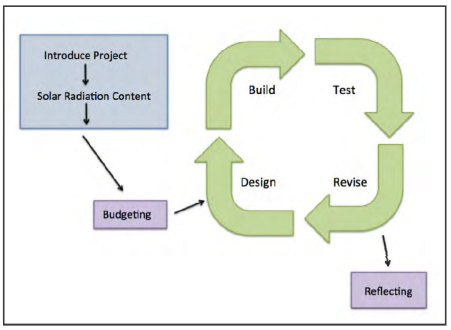


Figure 1: Outline of the solar ovens curriculum

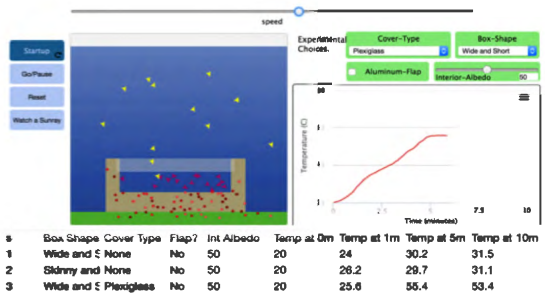


Figure 2: The interactive simulation used by students to test solar ovens and visualize energy transformation; below the table simulation is output from the automatically generated table

The curriculum takes between 10-15 class periods (45 minutes per class period). Students complete this project in groups of 2 or 3 students. Students also complete a pretest the day before the project begins and a posttest the day after completing the solar ovens project. Students do the pretest and posttest individually. The pre-/posttests measure student understanding of science concepts and practices.

1.2 Interactive Computer Simulation

The interactive simulation (figure 2) was built using NetLogo [15]. Students can manipulate the simulation in a number of ways. They can change the cover on top of the oven, whether or not there is a reflective flap on top of the box, the shape of the box (wide and short or skinny and tall), and the albedo (reflectivity) of the inside of the box. Students may also manipulate the speed at which the simulation runs. Once a simulation runs to the end of the graph (10 simulated minutes), a new row is added to the table below the visualization with the settings and results from the trial. If the students do not allow the simulation to run until the simulated 10 minutes finish, nothing is added to the table.

The scaffolds we developed for the interactive simulation are twofold; short response questions direct students to investigate capabilities and limitations of the simulation and an automatically generated table helps students to keep track of trials they have run. The table includes information about all of the settings used in that trial, as well as the results of the trial at certain time points (e.g. 5 minutes, 10 minutes).

2. PROPOSED CONTRIBUTIONS

Making sure students use interactive simulations to aid in learning is a difficult task. To try to encourage students to take advantage of these simulations during learning, various scaffolding methods have been used. Often, these scaffolds are implicit, or built into the system with the simulation [13]. For example, guiding questions are used with inquiry simulations to direct students' attention toward certain features of simulations [4]. Students are also often encouraged in science classes to run multiple trials and control variables between trials (only change one variable between trials). A control of variables strategy can help students to determine the effect of a single variable on a more complex system, although in some cases students may benefit from more exploratory strategies [12].

Using log files from student interactions with the curriculum and output from the automatically generated tables (simulation scaffolding), we use feature engineering to identify how students use the model and whether these uses have an impact on learning. I developed features that have to do with the control of variables strategy, such as the number of trials (rows) a student runs and the percent of those trials that are systematic. These types of techniques have also been used with more complex simulations and microworlds (e.g., [3, 2]). We use results from pre- and posttests to assess student learning in tandem with the log data from the curriculum.

The data in this work comes from 635 students across three schools and five teachers. During this study, students participated in a pretest and posttest (each lasting one class period), as well as the 2-3 week long curriculum. During the curriculum, students worked in teams of 2-3. These 635 students formed 255 teams.

3. RESULTS

I used pretest and posttest scores to understand the effect of actions with the simulation on learning. I then examined the role the number of rows of data a student generated using the table scaffolding on learning. I found that the number of rows generated in iteration 1 of the simulation is a significant predictor of individual posttest scores, when controlling for pretest scores and curriculum group (b = 0.10, t(546) = 2.68, $p < 0.01$). Next, I examined the impact of controlling variables on learning. I found that the number of *Control Of Variables (COV) Trials* run, however, is not quite a significant predictor of posttest score, when controlling for group and pretest score (b = 0.06, t(546) = 1.63, $p = 0.10$). In addition, using a dummy variable for conducting any *COV Trials* does not significantly predict posttest scores when controlling for pretest scores and group (b = 0.005, t(546) = 0.13, $p = 0.90$). Together, these results indicate that the control of variables strategy, while a good practice in science, is not as helpful for developing an understanding of the scientific principles at play in a simulation. More experimentation using the model is beneficial for developing a better understanding of the scientific concepts.

I then split the students up based on their actions during the

simulation step (did not generate any rows in table, generated one row, generated 2 or more rows). I found that generating 2 or more rows in the table significantly predicts posttest scores, when controlling for pretest score and working group (b = 0.12, t(546) = 3.11, $p < 0.01$), though generating no rows or 1 row were not significant predictors. I also developed a variable, *Percent Systematic*, that is the percentage of the total rows a group generated that used the control of variables strategy. This variable has the ability to show more nuance in how students were employing the control of variables strategy, but was also not predictive in determining posttest scores, when controlling for pretest and group id (b = 0.05, t(508) = 1.32, $p = 0.188$).

There were also two short response scaffolding questions on the same step as the interactive simulation. I generated a variable based on the number of questions students answered (0, 1, or 2). This was predictive of posttest score, when controlling for pretest score and group id (b = 0.10, t(546) = 2.56, $p = 0.011$).

Overall, evidence suggests that students should be encouraged to experiment with the model and guided to produce at least two rows of data in the table to improve learning outcomes and use the short response questions. Perhaps changing more than one variable at a time in this type of environment indicates that students are spending more time thinking about possible outcomes. I have further examined this data using k-means clustering algorithms.

4. FURTHER QUESTIONS

I have finished the majority of data collection for my dissertation. I will conduct one more study during the spring of 2017, and there will be the potential for a follow-up study later. This is an important time for me to get feedback on my work, especially on the analysis of the action log data I have collected from over a thousand students. I will begin the writing phase of my dissertation work during the summer, and expect to complete my dissertation within the next 12 months.

During the doctoral consortium, I would like to discuss the following:

- How to assess patterns in student actions in interactive simulations (Tools and packages for doing this and assessment of what it means to be a meaningful pattern)
- Designing studies that integrate education theory and data mining
- Assessment of inquiry skills in online environments
- Use of event logs in online curriculum to assess student use of curriculum and how this can be used to assess learning in tandem with other methods

5. REFERENCES

[1] J. Chiu, P. Malcolm, D. Hecht, C. DeJaegher, E. Pan, M. Bradley, and M. Burghardt. Wisengineering: Supporting precollege engineering design and mathematical understanding. *Computers & Education*, 67:142–155, 2013.

[2] C. Conati, L. Fratamico, S. Kardan, and I. Roll. Comparing representations for learner models in interactive simulations. In *Artificial Intelligence in Education*, pages 74–83. Springer, 2015.

[3] J. D. Gobert, Y. J. Kim, M. A. Sao Pedro, M. Kennedy, and C. G. Betts. Using educational data mining to assess studentsâĂŹ skills at designing and conducting experiments within a complex systems microworld. *Thinking Skills and Creativity*, 18:81–90, 2015.

[4] C. Hmelo and R. Day. Contextualized questioning to scaffold learning from simulations. *Computers & Education*, 32(2):151–164, 1999.

[5] M. Linn and B. Eylon. *Science learning and instruction: Taking advantage of technology to promote knowledge integration.* Routledge, 2011.

[6] E. McBride, A. Dixit, and M. Linn. Submitted – using machine learning to automatically score text responses in middle school science projects. In *Proceedings of the 18th International Conference on Artificial Intelligence in Education*, 2017.

[7] E. McBride, M. Martinez-Garza, J. Vitale, and M. Linn. Middle school student use of interactive climate change simulations: Periods of observation and activity. In *Paper presented at the Annual Meeting of the American Educational Research Association, San Antonio, TX*, 2017.

[8] E. McBride, J. Vitale, L. Applebaum, and M. Linn. Use of interactive computer models to promote integration of science concepts through the engineering design process. In *Proceedings of the 12th International Conference of the Learning Sciences*, Singapore, Singapore, June 2016 2016.

[9] E. McBride, J. Vitale, L. Applebaum, and M. Linn. Examining the flow of ideas during critique activities in a design project. In *Proceedings of the 12th International Conference of Computer Supported Collaborative Learning*, June 2017 2017.

[10] E. McBride, J. Vitale, L. Applebaum, J. Madhok, and M. Linn. Using virtual models to improve science understanding in a hands-on solar ovens unit. In *Paper presented at the Annual Meeting of the American Educational Research Association, San Antonio, TX*, 2017.

[11] E. McBride, J. Vitale, H. Gogel, M. Martinez, Z. Pardos, and M. Linn. Predicting student learning using log data from interactive simulations on climate change. In *Learning @ Scale*, Edinburgh, UK, April 2016 2016.

[12] K. McElhaney and M. Linn. Investigations of a complex, realistic task: Intentional, unsystematic, and exhaustive experimenters. *Journal of Research in Science Teaching*, 48(7):745–770, 2011.

[13] N. Podolefsky, E. Moore, and K. Perkins. Implicit scaffolding in interactive simulations: Design strategies to support multiple educational goals. *Chemistry Education Research and Practice*, 14(3):257–268, 2013.

[14] K. Ryoo and M. Linn. Can dynamic visualizations improve middle school students' understanding of energy in photosynthesis? *Journal of Research in Science Teaching*, 49(2):218–243, 2012.

[15] U. Wilensky. Netlogo. 1999.

Developing Chinese Automated Essay Scoring Model to Assess College Students' Essay Quality

Ju-Lu, Yu
Graduate Institute of Educational Information and Measurement, National Taichung University of Education, Taiwan
No.140, Minsheng Rd., West Dist., Taichung City 40306, Taiwan (R.O.C.)
ddog5633@yahoo.com.tw

Bor-Chen Kuo
Graduate Institute of Educational Information and Measurement, National Taichung University of Education, Taiwan
No.140, Minsheng Rd., West Dist., Taichung City 40306, Taiwan (R.O.C.)
kbc@mail.ntcu.edu.tw

Kai-Chih Pai
Graduate Institute of Educational Information and Measurement, National Taichung University of Education, Taiwan
No.140, Minsheng Rd., West Dist., Taichung City 40306, Taiwan (R.O.C.)
minbai0926@gmail.com

ABSTRACT

The present study aimed at proposing a Chinese automated essay scoring model to assess college students writing quality. Thirty-one related Chinese linguistic indicators were developed based on Coh-Metrix indices and characteristics of Chinese texts. Essay collected from 277 college students were analyzed using automated Chinese text analyze tool. A stepwise regression was used to explain the variance in human scores. The number of words, number of low strokes, content words frequency, minimal edit distance (all words) and minimum frequency for content words predicted 55.8% variance in human scores. On the other hand, seven indicators: number of words, content words frequency, concreteness, Measure of Textual Lexical Diversity, minimal edit distance (part of speech), minimal edit distance (all words) and words per sentence were predictive of human essay ratings by using discriminant analysis. The present study further explored the effectiveness of the Chinese automated essay scoring model by using three different methods: stepwise linear regression, discriminant analysis, and Nonparametric Weighted Feature Extraction classification (NWFE). The preliminary results showed that NWFE classification method produced higher exact matches (51.3%) between the predicted essay scores and the human scores than stepwise regression (47.3%) and discriminant analysis (47.3%).

Keywords

Chinese automated essay scoring, writing quality, NWFE classification, Chinese linguistic indicators

1. INTRODUCTION

Essay scoring has traditionally relied on expert raters. These scoring methods need to spend more time and a large amount of human scoring. Based on these limitations, automated essay scoring becomes the important research for essay assessment. According to the results of past studies, automated essay scoring reported perfect agreement (i.e., the exact match of human and computer scores) from 30-60% and adjacent agreement (i.e., within 1 point of the human score) from 85-99% [1]. Moreover, recently the study of analyzing the scored essays using Coh-Metrix has increased noticeably [2, 4, 5, 6, 7, 8, 13, 14, 15]. Coh-Metrix is an automated text analysis tool that provides lots of different linguistic indices [10]. The tool can provide these indices by combining lexicons, a syntactic parser, and several other components that are widely used in computational linguistics.

Chinese language features in the characteristics of different from the English, cannot be directly applied to the Chinese essay writing. Most of the experts will consider the following sections: Number of words, structure organization, vocabulary diversification, typos, and punctuation. Based on the development of Coh-Metrix, automated text analyze tool were developed in Chinese. Totally 66 Chinese related linguistic indicators were used to analyze the characteristics of Chinese texts [12].

Writing the literacy assessment is an important standardized testing to assess college students' writing skill in Taiwan. The assessment is to detect whether students can express personal comments on specific issues. Students need to read an article, respectively, and express personal comments by writing the essay in two hundred words. These essays were scored by two experts and score from 0-5. However, we need to a lot of experts and spend more time to score. To propose a suitable automated scoring model is important and needed.

2. PROPOSED CONTRIBUTIONS

The purpose of the study is to explore the characteristics of Chinese writing and propose a suitable Chinese automated essay scoring model to assess college students writing quality. Past studies explored the variety of human scoring were predicted by different text features using regression analysis. Moreover, they proposed automated essay scoring model and examined the essay matches by linear regression and discriminant analysis. A Nonparametric Weighted Feature Extraction (NWFE) classification method was also used to examine the essay matches in the present study.

Nonparametric Weighted Feature Extraction (NWFE) is based on a nonparametric extension of scattering matrices. It could reduce parametric dimensional and increase classification accuracy [11]. The present study used linear regression analysis and discriminant analysis of the gradual selection of variables for the NWFE classification method and examine the accuracy of essay matches.

3. Method

3.1 Text Indices Selection Procedure

The present study collected Chinese essay from college students in Taiwan. All essay was analyzed by Chinese automated text

analyze tool. The tool provides 62 Chinese linguistic indices, includes basic text measures (e.g., text, sentence length), words information (e.g., word frequency, concreteness), cohesion (semantic and lexical overlap, lexical diversity, along with the incidence of connectives), part of speech and phrase tags (e.g., nouns, verbs, adjectives), and syntactic complexity (e.g., Sentence syntax similarity, Minimal Edit Distance).

The first step, correlation analyses was conducted to examine the strength of relations between the selected indices and the human scores of essay quality. Text indices retained based on a significant correlation with human scores. Multicollinearity was then assessed between the indices (r >.900). The index retained based the strongly with human scores when two or more indices demonstrated multicollinearity. Finally, totally thirty-one indices were used in the study.

3.2 Essay Scoring

277 essays were collected from college students in Taiwan. Each essay in the study was scored independently by two expert raters using a 5-point rating. The rating scale was used to assess the quality of the essays and had a minimum score of 0 and a maximum score of 5. The experts evaluated the essays based on a standardized rubric used in the Chinese writing literacy assessment in Taiwan. The results of correlation between two experts are 0.788. It indicated that consistency of expert scoring.

3.3 Essay Evaluation

Three different methods were used to examine the accuracy of automated essay scoring: linear regression analysis, discriminant analysis, and NWFE classification. Text features were selected by linear regression and discriminant analysis. The leave-one-out method was used to experiment with training essay set and testing the essay set. The present compared the exact matches of the essay by using the three methods.

4. Preliminary Results

4.1 Linear Regression Analysis: Text Features

A stepwise regression analysis was conducted to examine which text indicators were predictive of human essay ratings. 40 Chinese text features were used in the study. The results presented in Table 1. Five indicators were a significant predictor in the regression model: Number of words, the number of low strokes, content word's frequency, minimal edit distance (all words) and the minimum frequency of content words, F = 12.074, p <.001, r =.747, r^2 =.558. The results from the linear regression demonstrate that the five variables account for 55.8% of the variance in the human scoring of writing quality.

Table 1. Stepwise regression results for text features

Indicators	*B*	*SE*	B
number of words	.011	.001	.529
number of low strokes	.000	.000	-.131
content words frequency	.824	.402	.086
minimal edit distance (all words)	2.334	.618	.238
minimum frequency for content words	-.148	.042	-.154

4.2 Discriminant Analysis: Text Features

The purpose of the discriminant analysis was to examine whether features are predictive of human scoring. The results of the discriminant analysis showed that seven text features could predict human scorning, includes the number of words, content word frequency, concreteness, Measure of Textual Lexical Diversity, minimal edit distance (part of speech), minimal edit distance (all words) and words per sentence.

4.3 Exact and Adjacent Matches

Table 2 and Table 3 presented the results of exact and adjacent matches. The linear regression analysis (stepwise) selected features: The number of words, number of low strokes, content words frequency, minimal edit distance (all words) and minimum frequency for content words. The exact matches (leave-one-out) between the predicted essay scores (rounded to 0-5) and the human scores is 47.3% exact accuracy and 95.3% adjacent accuracy.

The discriminant analysis (stepwise) selected features had the number of words, word frequency of content words, minimal edit distance (local), MTLD, the number of terms, concreteness, and minimal edit distance (part of speech). The exact matches (leave-one-out) between the predicted essay scores and the human scores is 47.3% exact accuracy and 93.9% adjacent accuracy.

The present study conducted NWFE classification method to examine the effectiveness of automated essay scoring. The results showed that 48.7% exact matches between predicted scores and human scoring, which text features selected by linear regression. Moreover, 51.3% exact matches between predicted scores and human scoring, which text features selected by discriminant analysis.

Table 2. Comparison of Exact

Classification method	Text features selected by linear regression	Text features selected by Discriminant
Linear regression	47.3%	46.6%
Discriminant	45.5%	47.3%
NWFE	48.7%	51.3%

Table 3. Comparison of Adjacent

Classification method	Text features selected by linear regression	Text features selected by Discriminant
Linear regression	95.3%	93.9%
Discriminant	94.2%	93.9%
NWFE	89.9%	90.3%

5. Conclusion

Past studies have found that the number of words was an important indicator of human score [4, 15]. The results of the study also presented that the number of words has a high significant correlation with human scores. The number of words,

the minimal edit distance (local), and the number of low strokes three indicators belong to Descriptive and Syntactic Complexity categories in Coh-Metrix. MTLD belongs to Lexical Diversity. These indicators are related the scoring guide of writing for college students in Taiwan.

Comparing exact matches between linear regression analysis (stepwise) and discriminant analysis (stepwise). The results of leave-one-out of exact matches linear regression and discriminant analysis showed consistency. Moreover, regardless of method linear regression analysis (stepwise) or discriminant analysis (step-wise) selection indicators, the accuracy of exactly matched of NWFE method is higher than the other two classification methods.

6. Future Works

Past studies have investigated the potential for component scores that are calculated using the linguistic features by Coh-Metrix in assessing text readability [9, 12]. Moreover, one study has explored correlations between human ratings of essay quality and component scores based on similar natural language processing indices and weighted through a principal component analysis [2]. However, this approach has not been extended to computational assessments of essay quality In Chinese. The present study will adapt a similar approach to passing studies [9, 12]. We will conduct a principle component analysis (PCA) or factor analysis to reduce the number of indices selected from Chinese automated text analyze tool into a smaller number of components comprised of related features. The present study will further explore the correlation between component scores and human scoring. A Chinese automated essay scoring model based on text component scores will be developed and explored.

7. ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of this study by the Ministry of Education.

8. REFERENCES

[1] Attali, Y., & Burstein, J. 2006. Automated Essay Scoringwith E-rater V.2. *Journal of Technology, Learning and Assessmen*t, 43.

[2] Crossley, S. A., & McNamara, D. S. 2014. Does writing development equal writing quality? A computational investigation of syntactic complexity in L2 learners. *Journal of Second Language Writing*, 26, 66-79.

[3] Crossley, S. A., & McNamara, D. S. 2014. Developing component scores from natural language processing tools to assess human ratings of essay quality. In W. Eberle & C. Boonthum-Denecke (Eds.), *Proceedings of the 27th International Florida Artificial Intelligence Research Society (FLAIRS) Conference* (pp. 381-386). Palo Alto, CA: AAAI Press.

[4] Crossley, S. A., Dempsey, K., & McNamara, D. S. 2011. Classifying paragraph types using linguistic features: Is paragraph positioning important? *Journal of Writing Research*, 3, 119-143.

[5] Crossley, S. A., Roscoe, R. D., & McNamara, D. S. 2013. Using automatic scoring models to detect changes in student writing in an intelligent tutoring system. In C. Boonthum-Denecke & G. M. Youngblood (Eds.), *Proceedings of the 26th International Flordia Artificial Intelligence Research Society (FLAIRS) Conference*, 208-213. Menlo Park, CA: The AAAI Press.

[6] Crossley, S.A. & McNamara, D.S. 2010. Cohesion, coherence, and expert evaluations of writing proficiency. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*, 984-989. Austin, TX: Cognitive Science Society.

[7] Crossley, S.A., & McNamara, D.S. 2011. Text coherence and judgments of essay quality: Models of quality and coherence. In L. Carlson, C. Hoelscher, & T.F. Shipley (Eds.), *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, 1236-1231. Austin, TX: Cognitive Science Society.

[8] Guo, L., Crossley, S. A., & McNamara, D. S. 2013. Predicting human judgments of essay quality in both integrated and independent second language writing samples: A comparison study. *Assessing Writing*, 18, 218-238.

[9] Graesser, A.C., McNamara, D.S., and Kulikowich, J. 2012. Coh-Metrix: Providing multilevel analyses of text characteristics. *Educational Researcher*, 40(5), 223-234.

[10] Graesser, A. C., McNamara, D. S., Louwerse, M. M., & Cai, Z. 2004. Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, and Computers*, 36, 193-202.

[11] Kuo B.-C., and Landgrebe, D. A. 2004. Nonparametric weighted feature extraction for classification, *IEEE Transactions on Geoscience and Remote Sensing*, 42(5), 1096-1105.

[12] Kuo B.-C., and Liao C.-H. 2014. The Automated text analysis for Chinese text. *2014 Workshop on the Analysis of Linguistic Features (WoALF 2014)*, Taipei, Taiwan.

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

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

[15] Roscoe, R.D., Crossley, S.A., Weston, J.L., & McNamara, D.S. 2011. Automated assessment of paragraph quality: Introductions, body, and conclusion paragraphs. In R. C. Murray & P. M. McCarthy (Eds.), *Proceedings of the 24th International Florida Artificial Intelligence Research Society (FLAIRS) Conference*, 281-286. Menlo Park, CA: AAAI Press.

Teaching Informal Logical Fallacy Identification with a Cognitive Tutor

Nicholas Diana
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
ndiana@cmu.edu

John Stamper
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
john@stamper.org

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
koedinger@cmu.edu

ABSTRACT

In this age of fake news and alternative facts, the need for a citizenry capable of critical thinking has never been greater. While teaching critical thinking skills in the classroom remains an enduring challenge, research on an ill-defined domain like critical thinking in the educational technology space is even more scarce. We propose a difficulty factors assessment (DFA) to explore two factors that may make learning to identify fallacies more difficult: type of instruction and belief bias. This study will allow us to make two key contributions. First, we will better understand the relationship between sense-making and induction when learning to identify informal fallacies. Second, we will contribute to the limited work examining the impact of belief bias on informal (rather than formal) reasoning. We discuss how the results of this DFA will also be used to improve the next iteration of our fallacy tutor, how this tutor may ultimately contribute to a computational model of informal fallacies, and some potential applications of such a model.

Keywords

Cognitive Tutors, Informal Logical Fallacies, Informal Reasoning, Cognitive Task Analysis, Difficulty Factors Assessment

1. INTRODUCTION

Despite the recognized importance of critical thinking in traditional education, critical thinking is largely absent from the educational technology space (e.g., online courses/MOOCs, cognitive tutoring systems, etc.). Some of the recent work on critical thinking in educational technology has focused on comparing critical thinking in face-to-face and computer-mediated interactions. Researchers often use content-analysis to identify instances of critical thinking in online and face-to-face discussions [3, 10]. In this work, critical thinking is not the primary focus of the course, but rather an epiphenomenon.

Other work, particularly in the domains of philosophy, writing and law, has addressed critical thinking more directly. For example, some recent work has demonstrated that argument diagramming using a graphical interface improved argumentative writing skills [6] as well as critical thinking skills more generally [5]. However, similar gains are seen using paper-and-pencil argument diagramming as well, suggesting the software may be more of a convenience than a necessary factor [4].

Despite the challenges of working in an ill-defined domain [8], another intersection of critical thinking and e-learning has been in intelligent tutoring systems (ITS). For example, Ashley and Aleven [1] built an ITS to teach law students to argue with cases more effectively. The study we propose extends this work on critical thinking in the ITS space to a more general population. We will build a cognitive tutor that teaches users to identify several common informal logical fallacies. We chose informal fallacies because they offer a degree of structure to the otherwise ill-defined domain of informal reasoning, making the content more amenable for use in a cognitive tutor. Using this tutor, we will conduct a difficulty factors assessment (a type of a cognitive task analysis) [7] to evaluate the impact of two factors on the user's ability to identify logical fallacies.

The first factor explored will be *type of instruction*. The Knowledge-Learning-Instruction (KLI) framework lists three types of learning processes, and suggests that the best instruction for teaching a specific skill depends on the type of process used to learn that skill. The purpose of the *type of instruction* manipulation is to better understand the learning processes that underpin the identification of logical fallacies. Specifically, we are interested in whether this skill is more efficiently learned using induction (e.g., showing many examples of the fallacy) or sense-making (e.g., providing detailed descriptions of the fallacy's mechanics). Textbooks used to teach logical fallacies often take both approaches, giving readers an explanation of a fallacy followed by some small number of examples. As this skill may consist of multiple, more fundamental skills (or knowledge components), the mixed approach used by textbooks may prove to be the most efficient. Nevertheless, the proportion of time to devote to each learning process remains an open question that this experiment may help answer.

The second factor that may negatively impact a student's ability to identify logical fallacies is *belief bias*, the tendency

Table 1: Breakdown of the problems used in the tutor. Note that *(F)*, *(A)*, *(C)*, and *(L)* correspond to *for*, *against*, *conservative* and *liberal*, respectively. For example, in the first cell of the table, we see an *apolitical* prompt, which *fallacy 1* is used to argue *for*.

	Apolitical	Political	Apolitical	Political	Apolitical	Political
Fallacy 1	(F)	(C)	(A)	(L)	(F)	(C)
Fallacy 2	(A)	(L)	(F)	(C)	(A)	(L)
Fallacy 3	(F)	(C)	(A)	(L)	(F)	(C)
Fallacy 4	(A)	(L)	(F)	(C)	(A)	(L)
Fallacy 5	(F)	(C)	(A)	(L)	(F)	(C)
Fallacy 6	(A)	(L)	(F)	(C)	(A)	(L)

to judge arguments more favorably if we agree with the conclusion. Early work on belief bias explored its effect on formal reasoning using syllogisms [9, 2], but there is some evidence that suggests that belief bias may operate differently in informal reasoning [11]. The proposed study builds on and contributes to this research by empirically testing the effect of belief bias on learning to identify informal fallacies.

2. FUTURE RESEARCH PLANS

2.1 Difficulty Factors Assessment

We will use a Difficulty Factors Assessment (DFA) to identify the factors (if any) that make it more or less difficult for students to learn how to identify logical fallacies. The proposed experiment will explore the impact of two primary factors as well as several secondary factors.

2.1.1 Type of Instruction

The proposed experiment will explore the impact of *type of instruction* by randomly assigning each participant to one of three conditions. In each condition, when the participant is given a problem and asked to identify the logical fallacy, they will be given a set of possible answers and the option to view more information about each of the answers. In the first condition, when participants ask for more information they will be shown a brief, but detailed description of the mechanics of each fallacy (sense-making). In the second condition, participants will be shown two examples of each fallacy (induction). In the the third condition, participants will be shown a description and one example for each fallacy (mixed).

In addition to comparing the effect of increased examples between groups, we will be able to compare this effect within groups by treating completed problems as viewed examples. This analysis will help us pinpoint the average number of examples needed to be able to identify the fallacies used in the experiment, and compare that number to the average numbers seen in common textbooks.

2.1.2 Belief Bias

The proposed experiment will explore the impact of *belief bias* on a student's ability to identify logical fallacies by altering the political orientation of problem content and comparing performance on those problems with the participant's personal political orientation. Of the 36 problems presented, half will be apolitical (i.e., politically neutral) and half will be political. Of the political problems, half will have a conservative orientation, half a liberal orientation. The apolitical problems are also split into two categories (for an issue or against an issue) for balance. Problems can be broken down into three subcomponents: the prompt (either political or apolitical), the fallacy, and the conclusion (either for/against or conservative/liberal). Table 1 shows the breakdown of each problem.

2.1.3 Secondary Factors Explored

In addition to the main effects of *type of instruction* and *belief bias*, our design also allows us to explore several secondary factors. We can test whether *type of instruction* has a differential effect on specific fallacies. For example, sense-making may be more important for learning to identify a circular argument, while examples may be sufficient for learning to identify a Post Hoc fallacy. We can also test whether participants are more likely to identify a fallacy given the nature of the prompt (political vs. apolitical) or the valence of the conclusion (for/against or conservative/liberal).

2.2 Towards a Computational Model of Logical Fallacies

The ultimate goal of this work is to develop a computational model of logical fallacies. Achieving this goal requires overcoming several large challenges.

2.2.1 Lack of Labeled Examples

First, to train a model to detect such a nuanced use of language will most likely require a large number of labeled examples. Furthermore, these examples will most likely have to be varied and authentic (perhaps unlike many of the purposefully illustrative examples used in textbooks). To solve this shortage of labeled examples, we propose using our cognitive tutor to train crowd workers to identify fallacies in real-world media sources. The quality of those labels can be evaluated using traditional crowdsourcing methods (e.g., consensus of the crowd). High quality labels can then be automatically integrated into the tutor training system, increasing the number of potential examples crowd workers can use to achieve mastery. This increase in the number of examples may be especially important if our DFA reveals that learning to identify informal fallacies is a primarily inductive skill. Figure 1 shows the feedback loop relationship between crowd workers and the cognitive tutor.

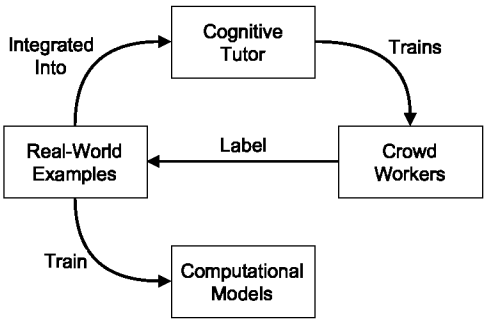


Figure 1: Feedback loop relationship between the cognitive tutor and crowd workers. The real-world examples labeled by crowd workers can be used to both improve the cognitive tutor and train computational models.

2.2.2 Modeling the Semantic Nature of Fallacies

Informal Logical Fallacies is an umbrella term that encompasses a diverse array of fallacies. Some of these fallacies may be easier for a machine learning model to detect. For example, the *Slippery Slope* fallacy often has the generic structure: "First X, pretty soon there'll be Y too!" These kinds of syntactic features will likely be easier to detect than the semantic features necessary to identify a fallacy like *Circular Reasoning*. Finding the right method for approaching these more difficult cases will be one of the key challenges of this work.

2.2.3 Potential Applications

If we meet these challenges and are able to detect logical fallacies in real-world text, there are potential applications in media (both traditional and social), politics, and education. One could imagine a plugin for your favorite word processor that underlines an *Appeal to Ignorance* just as it would a misspelled word. Similarly, one could imagine how broadcasts of presidential debates in the future might be accompanied by a subtle notification anytime a candidate uses *Moral Equivalence*.

In conclusion, we propose a plan to develop a computational model of informal logical fallacies. The first, and most concrete, step of this process is developing a better understanding of the factors that promote and hinder how we learn to identify informal fallacies. We propose a difficulty factors assessment to explore the impact of sense-making versus induction support, as well the impact of belief bias. Discovering how these factors regulate learning will not only allow us to build a better tutor, but will improve our understanding of how we learn informal logical fallacies in general.

3. REFERENCES

[1] K. D. Ashley and V. Aleven. Toward an intelligent tutoring system for teaching law students to argue with cases. In *Proceedings of the 3rd international conference on Artificial intelligence and law*, pages 42–52. ACM, 1991.

[2] J. S. Evans, J. L. Barston, and P. Pollard. On the conflict between logic and belief in syllogistic reasoning. *Memory & cognition*, 11(3):295–306, 1983.

[3] J. Guiller, A. Durndell, and A. Ross. Peer interaction and critical thinking: Face-to-face or online discussion? *Learning and Instruction*, 18:187–200, 2008.

[4] M. Harrell. No computer program required: Even pencil-and-paper argument mapping improves critical-thinking skills. *Teaching Philosophy*, 31(4):351–374, 2008.

[5] M. Harrell. Assessing the efficacy of argument diagramming to teach critical thinking skills in introduction to philosophy. *Inquiry: Critical Thinking Across the Disciplines*, 27(2):31–39, 2012.

[6] M. Harrell and D. Wetzel. Improving first-year writing using argument diagramming. *Proc. of the 35th Annual Conf. of the Cognitive Science Society*, (1987):2488–2493, 2013.

[7] K. R. Koedinger and A. Terao. A cognitive task analysis of using pictures to support pre-algebraic reasoning. In *Proceedings of the Twenty-Fourth Annual Conference of the Cognitive Science Society*, pages 542–547. Citeseer, 2002.

[8] C. Lynch, K. Ashley, V. Aleven, and N. Pinkwart. Defining ill-defined domains; a literature survey. In *Proceedings of the workshop on intelligent tutoring systems for ill-defined domains at the 8th international conference on intelligent tutoring systems*, pages 1–10, 2006.

[9] J. J. B. Morgan and J. T. Morton. The distortion of syllogistic reasoning produced by personal convictions. *Journal of Social Psychology*, 20(1):39–59, 1944.

[10] D. R. Newman, B. Webb, and C. Cochrane. A content analysis method to measure critical thinking in face-to-face and computer supported group learning. *Interpersonal Computing and Technology*, 3(September 1993):56–77, 1995.

[11] V. Thompson and J. S. B. T. Evans. Belief bias in informal reasoning. *Thinking & Reasoning*, 18(3):278–310, 2012.

Automated Extraction of Results from Full Text Journal Articles

R. Wes Crues
University of Illinois
Dept. of Educational Psychology
1310 South Sixth Street
Champaign, Illinois
crues2@illinois.edu

ABSTRACT

Recent mandates by federal funding agencies and universities to create open access repositories of published research allow researchers a wealth of texts to analyze. Furthermore, some publishers of academic texts have begun creating policies to permit non-commercial text mining of journal articles. This project follows the approach of [7], which automatically extracts result sentences from full-text biomedical journal articles by using support vector machines and naive Bäyes classifiers. I also experiment with using the least absolute shrinkage and selection operator (LASSO) [6, 18] as a method to select features for the classifiers. I compare this new approach with other feature selection strategies used in previous studies.

Keywords

Information extraction, text classification, feature selection

1. INTRODUCTION

Information overload is hardly a new concept, with even the Ancient Roman scholar Seneca the Elder claiming in 1 AD, "the abundance of books is distraction" [8]. Similarly, the automatic summarization of text has been researched since at least the 1950's, with Luhn's work on creating abstracts automatically [11]. In concert, United States (US) federal funding agencies, such as the National Institutes of Health (NIH) [13], the National Science Foundation (NSF) [14], and the Institute for Educational Sciences (IES) [9], and university systems such as the University of California (UC) [1] have adopted open access policies for funded and published research. Publishers of academic journals, such as Elsevier [4] and Springer [15], have adopted policies for non-commercial research of texts. Finally, some national governments (e.g., the United Kingdom (UK) [10]) have adopted changes to copyright law allowing for non-commercial research of copyright protected works.

Given these open-access and legal policy changes, a wide swath of researchers now have access to a wealth of texts to automatically analyze. Specifically, the shifts in policies and laws allows for text mining to extract result sentences from full-text journal articles. Further, publishers have created APIs which allow for access to texts. It is unlikely that future researchers will be able to carefully read and analyze all of the texts in order to extract pertinent results. However, open-access policies in the US by the NIH have enabled automated extraction since the late 2000s in some fields.

My research seeks to first expand the work done in the biomedical sciences, particularly in [7] to the educational sciences, but also to explore an additional feature selection technique. This experiment is to complement the work in [20] by using the LASSO as a feature selection technique.

2. BACKGROUND

Text mining has been recognized as a tool to reduce the time required to complete a systematic literature review [17]. There are several tasks text mining can simplify when creating a systematic review. Current text mining approaches allow relevant studies to be identified, by identifying relevant search terms, and describing the characteristics of prior investigations can be accomplished by automatic summarization [17]. This proposal is inspired by the systematic search of literature using targeted queries by the information scientist, Don Swanson, who revealed a link between magnesium and migraines in the late 1980s [16]. This finding is novel because it linked medical literature with chemistry literature. Thus, I want to uncover previously unrealized links, contradictions, and confirmations in the current literature on on how students utilize computers to enhance or hinder their educational experience.

Supervised learning using text has been heavily researched in the biomedical sciences. For example, [12] proposed to use a modified naïve Bayes classifier which can determine whether an abstract is relevant for a given topic, based on the words in previously seen abstracts. They also propose a unique weighting scheme which allows for high recall and reasonable precision. In their work, they show their proposed process can significantly reduce the time required to conduct a systematic literature review. Given the amount of publications available following from the aforementioned changes, these results could help educational researchers significantly reduce time to determine which previously published work

is most relevant.

More broadly, this work addresses the need to have a "living systematic literature review" where the most up-to-date published findings can be included for practitioners and researchers to implement and be informed of these findings [3]. One study found the average time between a published finding and inclusion in a systematic literature review to average between 2.5 and 6.5 years [3]. This relates directly to an initiative by the US's Institute of Educational Sciences to use evidence based practices [19]; that is, connecting the knowledge from research to practicing the knowledge.

3. APPROACH

This project will extract sentences containing results from full-text journal articles in peer-reviewed journals. Given that journals have dozens of volumes and issues, it is likely not feasible to read and find all relevant articles needed to understand prior research. This process will create a systematic review of literature from educational journals in a targeted area: student interaction and behavior in computing environments. The systematic review will inform researchers on previous findings and update practitioners on the most current research.

3.1 Extracting Results

To extract result sentences, I will parse full-text journal articles into sentences, using a tokenizer, for example, Python's NLTK [2]. Next, I label the sentences as either containing a result or not, as well as indicate the section of the article where the sentence lies, and whether the sentence is the first or last in the respective paragraph, following from [7]. In [7], result sentences were distributed throughout the journal articles and were most common in the first or last sentence of the paragraph. Then, I will experiment with various classifiers, such as support vector machines, naïve Bayes classifiers, decision trees, and various ensemble models. The output of the classifiers will be the sentences containing results, which can then be used to form a thorough systematic review.

To train these models, I will select features using traditional metrics, such as information gain, mutual information, and the χ^2 statistic [20], which are the ones used by [7]. Interestingly, using these three feature selection strategies, not one term was selected by all three methods; however, there was overlap with terms for the χ^2 statistic and information gain, and information gain and mutual information. Because of this finding, I propose to use a different feature selection technique to select words or surface level knowledge (e.g., sentence position, section of paper) to train these classifiers.

3.2 Feature Selection

Another experiment I plan to conduct to extract words from the corpus of sentences from the journal articles is to utilize the LASSO to select words to use to train classifiers to discern sentences containing results from those that do not. Given that the LASSO is used for high dimensional data sets as a variable selection technique, in fields such as gene-expression analysis [5], this approach seems reasonable given the high dimensionality and sparseness of text data. I will experiment with various parameters of the LASSO to ensure reasonable feature selection; that is, a feature set which is not prohibitively small to provide high recall and reasonable precision, but one which is not too big to prohibit generalizablity.

The specific binomial logistic LASSO model I will use to select terms is

$$\log \frac{P(result = 1|\mathbf{x})}{P(result = 0|\mathbf{x})} = \beta_0 + \mathbf{x}^T\beta, \quad (1)$$

where $result$ equals one if the sentence x_i contains a result, and zero otherwise. Note that $\mathbf{x}$ is a matrix, where each row is a sentence, one column is $result$, and the other columns are words and surface-level features about the sentence. In the estimation phase, the model's likelihood function is penalized by a shrinkage parameter λ. This shrinkage parameter shrinks unimportant βs towards zero, thus leaving only the most important terms with nonzero βs. These terms will then be used to train the classifiers to extract result sentences to be used in systematic literature reviews. Further, the magnitude of each β can be beneficial in determining relative importance of a term.

For this portion of the project, I will experiment with various λs to determine which give the best performance when training the models to extract result sentences. A comparison of the feature selection strategies in [7, 20] will be conducted to determine any relationship between these feature selection strategies and the LASSO.

4. CURRENT STATUS

My current tasks are to complete a literature review of text classification. In this literature review, I address traditional classifiers from multivariate statistics and machine learning, but also accompany background on generating systematic literature reviews. The literature review also includes a discussion of evidence based practices and speculates on how a living systematic literature review might impact education research.

A concurrent stage is procuring and processing texts for analysis. In [7], seventeen full-text articles were analyzed, with around 2550 total sentences being considered. Thus, once all texts have been selected, I will begin labeling the sentences as containing a result or not containing a result. Efforts are underway to procure a small research fund to pay a research assistant to also label sentences as a measure of inter-rater reliability.

5. PROPOSED CONTRIBUTIONS

This work provides contributions to the fields of information science and educational data mining. One contribution is an alternative feature selection strategy which could improve performance of supervised learning methods. Because feature selection is arguably the most important analysis phase in text classification, using the LASSO in addition to strategies already used might help better performance in text classification.

Another contribution of the work is introducing the concept of a living systematic literature review to educational research. Due to the explosion of the amount of published research in education, and the interest in evidence based

practice to be utilized in education, this work can address those desires.

6. ADVICE SOUGHT

I would like advice on any or all of these concerns:

1. Are there other approaches, besides classifiers such as support vector machines, naïve Bayes, discriminant analysis, neural networks, and decision tree classifiers that would be useful for this approach?
2. What suggestions do you have for analyzing the result sentences once they have been discovered by the classification algorithms?
3. Do you have any suggestions for experiments with the shrinkage parameter, λ, for selecting terms when using the LASSO?
4. Are there any specific metrics you would suggest to use for analyzing the results of either result extraction or selecting terms?

7. REFERENCES

[1] Academic Senate of the University of California. UC systemwide academic senate open access policy, 2013.

[2] S. Bird. NLTK: The natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[3] J. H. Elliott, T. Turner, O. Clavisi, J. Thomas, J. P. Higgins, C. Mavergames, and R. L. Gruen. Living systematic reviews: an emerging opportunity to narrow the evidence-practice gap. *PLoS Med*, 11(2):e1001603, 2014.

[4] Elsevier, Inc. Text and data mining policy, 2014.

[5] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer, 2001.

[6] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[7] H. A. Gabb, A. Lucic, and C. Blake. A method to automatically identify the results from journal articles. *iConference 2015 Proceedings*, 2015.

[8] Hewlett Packard. Dizzying volumes of data is nothing new.

[9] Institute of Educational Sciences. IES policy regarding public access to research, 2016.

[10] Intellectual Property Office. Exceptions to copyright: Research, 2014.

[11] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

[12] S. Matwin, A. Kouznetsov, D. Inkpen, O. Frunza, and P. O'Blenis. A new algorithm for reducing the workload of experts in performing systematic reviews. *Journal of the American Medical Informatics Association*, 17(4):446–453, 2010.

[13] National Institutes of Health. Revised policy on enhancing public access to archived publications resulting from NIH-funded research, 2008.

[14] National Science Foundation. NSF's public access plan: Today's data, tomorrow's discoveries (NSF 15-22), 2015.

[15] Springer. Springer's text- and data-mining policy, 2016.

[16] D. R. Swanson. Migraine and magnesium: Eleven neglected connections. *Perspectives in Biology and Medicine*, 31(4):526–557, 1988.

[17] J. Thomas, J. McNaught, and S. Ananiadou. Applications of text mining within systematic reviews. *Research Synthesis Methods*, 2(1):1–14, 2011.

[18] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[19] US Department of Education: Institute of Educational Sciences. Identifying and implementing educational practices supported by rigrous evidence: A user friendly guide, 2003.

[20] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, volume 97, pages 412–420, 1997.

Intelligent Argument Grading System for Student-produced Argument Diagrams

Linting Xue
North Carolina State University
Raleigh, North Carolina, USA
lxue3@ncsu.edu

ABSTRACT

Current automated essay grading systems are typically focused on the semantic and syntax analysis of written arguments via Natural Language Processing techniques. Few systems focus on the automatic assessment of argument *structure*. In this work, we propose to build an Intelligent Argument Grading System to automatically assess and provide feedback on the structure of arguments of student-produced argument diagrams, which are graphical representations for real-word argumentation. The proposed system contains two stages. In the first, it automatically induces empirically-valid graph rules for expert-graded argument diagrams. An assessment model is trained from the dataset of manually-graded argument diagrams with the feature of induced graph rules. In the second stage, the assessment model automatically grades and provides feedback by identifying both good features and structural flaws in students' work. The significance of this work will be that the proposed system can save high cost of labor by automatically inducing empirically-valid rules, grading, and providing feedback on the structure of arguments for students. We anticipate that the automatic feedback can help students revise their structural plans accordingly before they start to write essays, which will in turn lead them to produce more high-quality arguments.

Keywords

Argument Diagrams, Structure of Arguments, Automated Grading System, Automatic Feedback

1. INTRODUCTION

Argumentation is an essential skill in scientific domains including physics, engineering, and computer science, where students must articulate and justify testable hypotheses through argumentative reasoning. As a consequence, automated essay grading systems have become particularly useful tools for argument assessment (e.g. [1, 3, 9]). Prior research has shown that automated assessment systems can be used to assess student-produced arguments correctly and cost-effectively. Current automated grading systems rely on either surface-level analysis of linguistic features within a bock of text (as in [3]) or deeper Natural Language Processing (NLP) that utilizes machine learning techniques (as in [9, 1]). These systems are typically designed to evaluate on the basis of readability (e.g. the number of prepositions and relative pronouns or the complexity of the sentence structure), shallow semantic analysis (e.g. lexical semantics or the relationships analysis among named entities), and syntax analysis (e.g. grammatical analysis). Ultimately, these systems return the scores or feedback on the content and the qualities of the students' writing based on a predictive model that is trained by the dataset stored in the system.

However, very few active systems are focused on automatic analysis of the rhetorical structure of arguments to address structural flaws. Argument structure refers to the organization of the key components of argumentation (e.g. hypotheses, citations, or claims), which can reveal how the students justify their research hypotheses by using relevant evidence to support or oppose conclusory statements. In real-life teaching, the students are encouraged to structure their argumentative essays before they start writing by formulating a research hypothesis based on the research question, listing relevant evidence and factual information, and identifying the logical relationships between them. Evaluating the draft structure of these arguments and identifying flaws can help students to revise their plans and to produce high-quality arguments in the future. It is possible for human experts to grade draft arguments. However that process is costly and time-consuming.

In this work, we propose to build an Intelligent Argument Grading System that can automatically grade and provide feedback on the structure of students' arguments. The system will be based upon LASAD [4], an online tool for argument diagramming and collaboration. The input to the system will be a valid argument diagram, the output is the grade and feedback pointing out the outstanding substructures and structural flaws in the student's work.

2. BACKGROUND

2.1 Argument Diagrams

Argument diagrams are visual representations of real-world argumentation that reify the essential components of arguments such as *hypotheses* statements, *claims*, and *citations* as nodes and the *supporting*, *opposing*, and *clarification* relationships as arcs [6]. These complex nodes and arcs can

include text fields describing the node and arc types or free-text assertions, links to external resources and other data. Argument diagrams have been used in a variety of domains, including science [10], law[8] and philosophy [2] to help students learn written argumentation. Prior researchers have shown that argument diagrams can be used to scaffold students' understanding of existing arguments [2] and can help to support scientific reasoning [10].

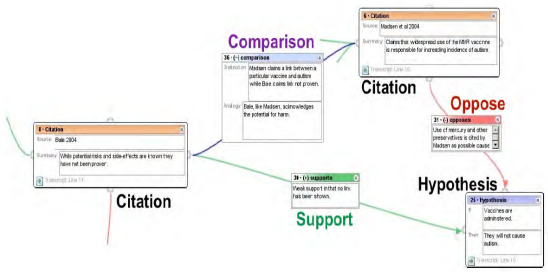


Figure 1: A student-produced Argument Diagram.

A sample student-produced diagram is shown in Figure 1. The diagram includes a *hypothesis* node at the bottom right, which contains two text fields, one for a conditional or *if* field, and the other for a consequent or *then* field. Two *citations* are connected to the *hypothesis* node via *supporting* and *opposing* arcs colored green and red, respectively. They are also connected via a *comparing* arc. 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.

3. PRELIMINARY RESULTS

In Lynch's study of diagnosticity of argument diagrams [5], a set of 104 paired diagrams and essays were collected at the University of Pittsburgh in a course on Psychological Research Methods. The diagrams and essays were independently graded by an experienced TA according to a parallel grading rubric. They showed that hand-authored graph rules were *empirically-valid* and were correlated with the diagram and essay grades; and thus that they could be used as the basis of predictive models for automatic grading.

Our prior work has also shown that Evolutionary Computation (EC) can be used to automatically induce empirically-valid graph rules for student-produced argument diagrams, and that the induced graph rules can be used as features for automatic grading [11, 12]. It is possible to harvest a set of diverse rules that were filtered via post-hoc Chi-Squared analysis [7]. This includes both good rules that are positively correlated with the diagram and essay grades and bad rules which are negatively correlated with the former representing positive structural features and the latter indicating flaws in the argument.

Figure 2 shows an example of a positive graph rule (P-G) and a negative graph rule (N-G) induced in our prior work. P-G shows a graph structure where the students identified at least two related citations ($c0$ & $c1$) that can be synthesized to support a single claim ($k0$) and where they included both a separate hypothesis (h) and an additional claim ($k1$).

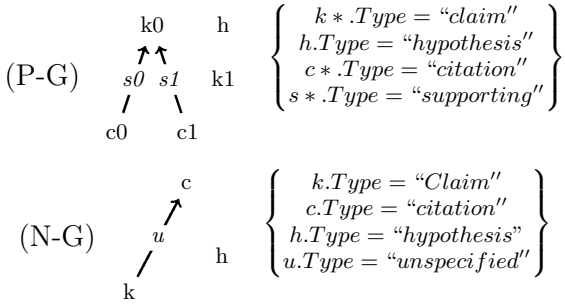


Figure 2: Examples of positive and negative graph rule.

It shows one of the structures that students have been encouraged to incorporate into their arguments as it shows an ability to synthesize citations to form a complex claim.

N-G is a negative rule that contains a single claim node (k) which is connected to a citation node (c) via an undefined arc (u), and a separate hypothesis node (h) which may or may not be connected to the rest structure. This rule is a clear violation of the semantic guidance that students were given. In our experiment, the students were instructed to use unspecified arcs for definitions or clarifications. Some students instead used them only when they were unsure about the strength of their evidence or did not understand the citation.

4. PROPOSED SYSTEM

In this work, we propose to build an Intelligent Argument Grading System (iARG) for student-produced argument diagrams. Our goal is to automatically grade the structure of arguments for students and provide feedback that reflects the good features and structural flaws in students' work. The proposed system includes two stages, which are shown in Figure 3.

The top part of Figure 3 illustrates the first stage, **Automatic Rule Induction**, in which the system automatically induces empirically-valid graph rules for expert-graded argument diagrams. The system will contain a database of argument diagrams and expert-assigned grades, along with a database of graph rules induced by the EC algorithm with a χ-Squared filter as described in [11, 7]. After the system produces a set of individual rules, the induced rules are evaluated by domain experts to determine whether or not they are semantically valid. Only valid rules will be incorporated into the database. Note that the induced rules contain both positive and negative examples. At the end of the process, we will use supervised learning methods to train an assessment model based upon the feature of induced rules and other graph feature (e.g. the degree of diagram nodes, the complexity of diagrams, and the attribute of the hub nodes in diagrams).

In the second stage of **Automatic Grading and Feedback**, the trained model will automatically grade and provide feedback on students' submissions by identifying both good features and structural flaws of the arguments. After

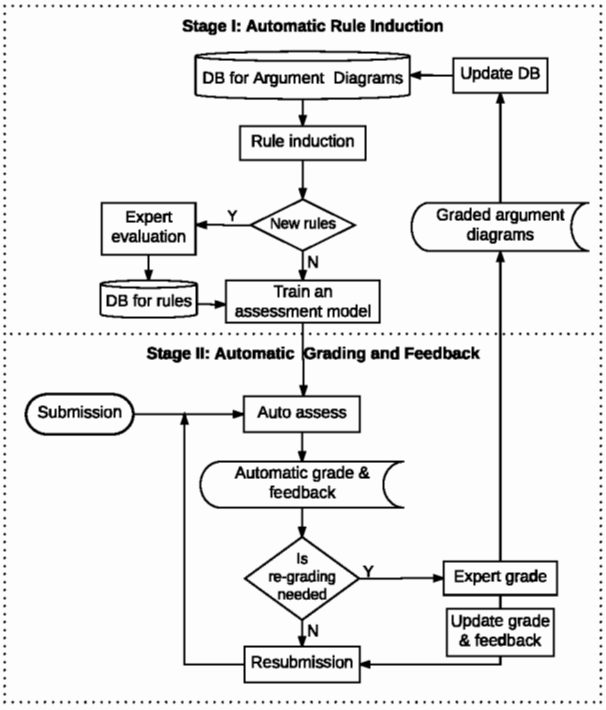


Figure 3: Flowchart for the proposed iARG

this, we will have experts re-evaluate the automatic grades and give feedback periodically, and if necessary, to re-grade the submission. We include this step because the students' submissions may include novel structures that are not included in the current rule database. In this case, the assessment model may treat these novel structures as outliers and provide uncorrected feedback. If the submissions are re-graded by experts, they will be updated to the database for argument diagrams. The rule database and assessment model will also be updated for future use.

5. FUTURE WORK & OPEN QUESTIONS

In the future work, we plan to achieve the following:

1. In Fall 2017, we plan to work with domain experts to determine whether the induced graph rules are semantically valid; whether they can be used for automatic grading; and whether they include all of the good features and structural flaws in students' work. This gives rise to our first research question: *how can we improve the performance of the graph rule induction algorithm by inducing more empirically-valid graph rules?*

2. In Spring 2018, we will leverage different supervised learning methods to train an assessment model from our current dataset of expert-graded argument diagrams with the feature of valid graph rules and other graph features. We will evaluate the assessment model on a new set of student-produced argument diagrams. Our second research question is that *what other graph features can we use to build the assessment model?*

3. In Fall 2018, we plan to implement the proposed system based upon LASAD by building databases for the argument diagrams and for the graph rules, and integrating the assessment model into the system.

4. In 2019, we will test the performance of our system in an augmentative writing class at NCSU. We will focus on accessing the automatic grades and feedback from the student's perspective and determine whether they find the automatic feedback to be useful. Thus we will not have experts to examine the automatic feedback in the second stage. Based upon the students' feedback, we will consider whether to have experts to regrade the new submission and to update the database and assessment model.

6. REFERENCES

[1] J. Burstein, C. Leacock, and R. Swartz. Automated evaluation of essays and short answers. 2001.

[2] M. Harrell and D. Wetzel. Improving first-year writing using argument diagramming. In *The 35th CogSci*, pages 2488–2493, 2013.

[3] M. A. Hearst. The debate on automated essay grading. *IEEE Intelligent Systems and their Applications*, 15(5):22–37, 2000.

[4] F. Loll and N. Pinkwart. Lasad: Flexible representations for computer-based collaborative argumentation. *International Journal of Human-Computer Studies*, 71:91–109, Januart 2013.

[5] C. F. Lynch and K. D. Ashley. Empirically valid rules for ill-defined domains. In J. Stamper and Z. Pardos, editors, *Proceedings of The* 7^{th} *International Conference on EDM*. IEDMS, 2014.

[6] 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, *ITS*, Lecture Notes, pages 260–265. Springer, 2014.

[7] C. F. Lynch, L. Xue, and M. Chi. Evolving augmented graph grammars for argument analysis. GECCO, 2016.

[8] N. Pinkwart, K. D. Ashley, C. F. Lynch, and V. Aleven. Evaluating an intelligent tutoring system for making legal arguments with hypotheticals. *IJAIED*, 19(4):401 – 424, 2009.

[9] L. M. Rudner and T. Liang. Automated essay scoring using bayes' theorem. *The Journal of Technology, Learning and Assessment*, 1(2), 2002.

[10] 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.

[11] L. Xue, C. Lynch, and M. Chi. Unnatural feature engineering: Evolving augmented graph grammars for argument diagrams. In *Internatinal Educational Data Mining*, pages 255–262. IEDMS, 2016.

[12] L. Xue, C. F. Lynch, and M. Chi. Mining innovative augmented graph grammars for argument diagrams through novelty selection. EDM, 2017.

Industry Track

Dropout Prediction in Home Care Training

Wenjun Zeng*
University of Minnesota
Minneapolis, Minnesota
zengx244@umn.edu

Si-Chi Chin
SEIU 775 Benefits Group
Seattle, Washington
sichichin@gmail.com

Brenda Zeimet
SEIU 775 Benefits Group
Seattle, Washington
brenda.zeimet
@myseiubenefits.org

Rui Kuang
University of Minnesota
Minneapolis, Minnesota
kuang@cs.umn.edu

Chih-Lin Chi
University of Minnesota
Minneapolis, Minnesota
cchi@umn.edu

ABSTRACT

In Washington state (WA), SEIU 775 Benefits Group provides basic home care training to new students who will deliver care and support to older adults and people with disabilities, helping them with self-care and everyday tasks. Should a student fail to complete their required training, it leads to a break in service, which can result in costly negative health outcomes (e.g. emergency rooms and hospitalization) for their clients [1].

In this paper we describe the results of utilizing machine learning predictive models to accurately identify students who exhibit a higher risk of drop out in two areas: (1) dropping out before attending first class[first class attendance]; and (2) dropping out before completing the training[training completion]. Our experimental results show that AdaBoost algorithm gives a useful result with ROC_{AUC} = 0.627±0.013 and Precision at 10 = 0.73±0.12 for first class attendance and ROC_{AUC} = 0.680±0.024 and Precision at 10 = 0.67±0.20 for training completion without relying on additional assessment data about students. In addition, we demonstrate the use case for constructing larger decision trees to help front-line training operations staff identify intervention strategies that create the most impact in preventing dropout.

1. INTRODUCTION

By 2050, the number of Americans needing long-term home care services and supports will double[2], implying increased demand for workers providing home care services (called "personal care aides" nationally and "home care aides (HCA)" in WA). This will also increase the demand of training for HCAs to provide quality care to their clients. In WA, should an individual wish to work as a home care aide, they are required to complete a 75 hour, 2 week, Basic Training (BT) course within 120 days of their hire date. In WA, an HCA can begin providing care before completing their training as long as their deadline has not passed. In the event that an HCA fails to complete BT, she or he will fall out of compliance, leading to the HCAs termination and a break in service for the clients served by the HCA [1].

Educators have frequently used assessment tools that measure cognitive skills, engagement, self-management and social support to accurately predict student successes. However, conducting assessments at scale is time consuming for both students and instructors. In the absence of a validated assessment specific to HCA profession, there is great interest in utilizing existing learning data to isolate the strongest predictors of dropout through the predictive power of machine learning algorithms. Our research questions are two-folds: 1) Can machine learning algorithms successfully predict student dropouts? 2) What are the risk factors related to early dropout from basic home care training?

Many studies[3] have been conducted to explain academic performance and to predict the success or failure across a variety of students in a wide-range of educational settings. Machine learning algorithms have been successful in predicting graduation[4], course participation[5], and other academic outcomes[6].

However current research has not fully investigated the area of using machine learning algorithms for on-the-job training, healthcare training programs, or adult education in general. In this paper, we focus on the dropout problems in home care training using machine learning methods. We were granted the latitude to be creative with our feature engineering, utilizing readily available data to meet business requirements.

2. EXPERIMENTAL SETUP

Figure 1 illustrates the four sequential time-based milestones in home care training: 1) Complete Orientation & Safety (O&S); 2) Register for a 70-hours BT course; 3) Attend the first class in this course; 4) Complete the 70-hour training. At the moment that a prospective home care aide enters the system, a 'Tracking Date' is assigned to their O&S training

*This work has been done during the author's internship at SEIU 775 Benefits Group

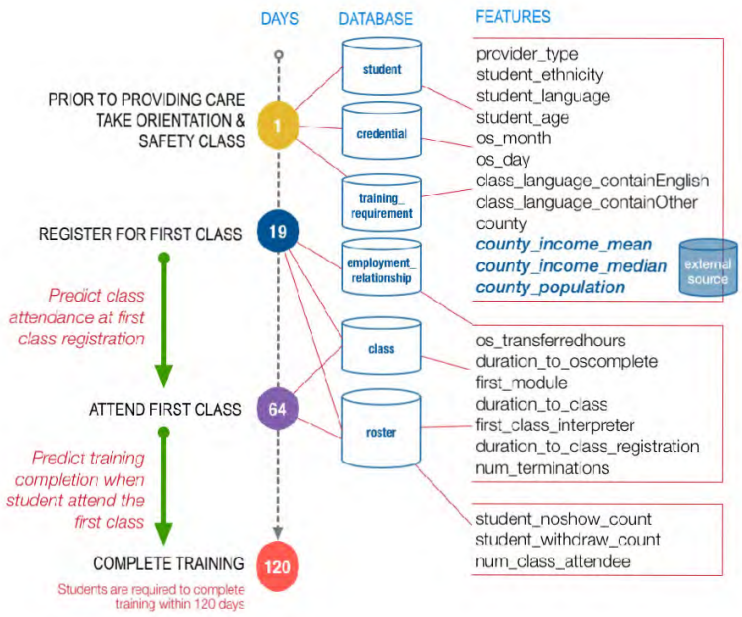


Figure 1: Predicting Targets and Features

requirement, signifying the start of their training journey. On average a student will register for his or her first class approximately 19 days after completing O&S and will actually attend his or her class about 64 days after entering our system.

Predicting dropouts at different stages has the potential to allow for timely interventions that may improve a students' learning experience. This paper focuses on two stages: First, **Class Attendance**: Will the newly hired students show up for their first scheduled class? We attempt to predict this at the point of registration. Second, **Training Completion**: Will a student complete all 70 hours of their required training? We attempt to predict this at the point that a student attends his or her first class. As shown in Figure 1, some basic but sometimes incomplete student demographic data are captured at the time a student is assigned to take O&S training. As a student progresses in his or her training journey, we are able to extract more features about learning behavior, such as the amount of time a student needed to complete O&S or the number of days it took a student to register for class. In addition, we leveraged external government census data to augment the existing feature set by adding income and population data of the student's county of residence.

We built four models – Logistic Regression, SVM, Random Forests, and AdaBoost – for the two predicting targets described above. Our final data set contained 5,303 records for predicting first class attendance and 5,182 records for predicting training completion. For both predicting targets, we reserved 2,000 records for testing data set and the remaining were utilized as the training data set. We collected 22 features to predict class completion and used the first 19 features to predict first class attendance(the last three features are not available at our prediction point of registration). Table 1 summarizes the features we used for the model.

3. EXPERIMENT RESULTS

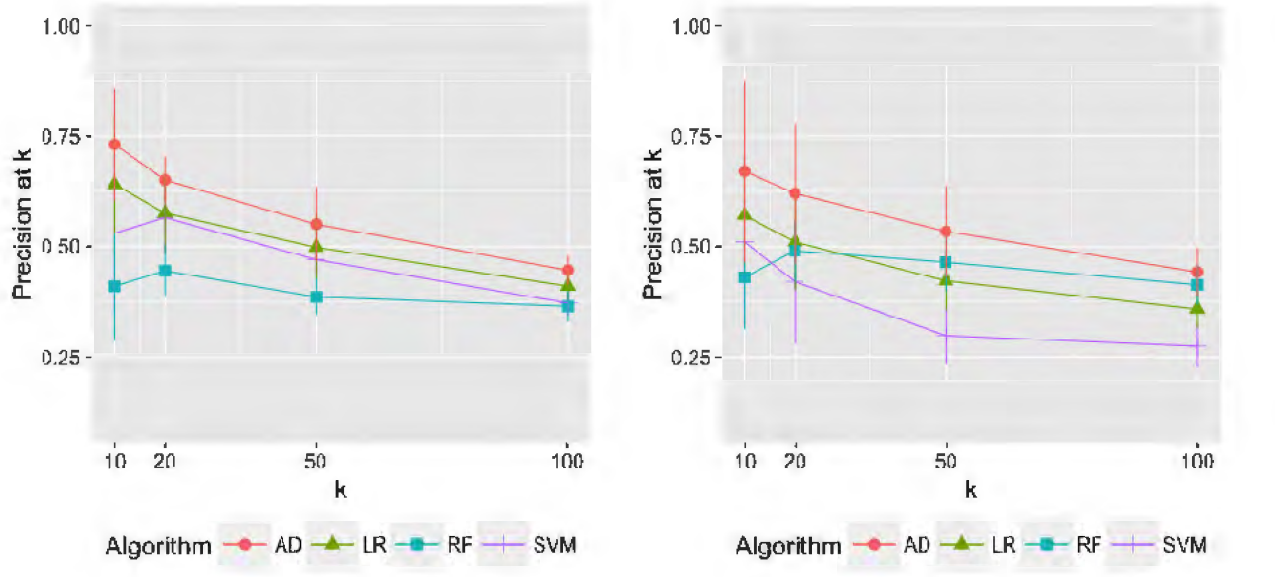
3.1 Prediction Performance: ROC-AUC and Precision at k

We use area under curve of the receiver operating characteristic (ROC_{AUC}) and precision at k (Prec@k) to evaluate prediction quality of each machine learning technique. ROC_{AUC} was used as a standard evaluation metric to measure the quality of overall ranking results. Prec@k was used to determine the quality of predicting the top k outcomes, in our case, the top k students of highest drop out risk at each stage. It is assuming that, with limited resources, front-line staff could only outreach to k number of students per week to provide support and assistance to HCAs struggling to meet their individual learning needs. Therefore, it is essential to accurately predict the first k students exhibiting the highest dropout risk.

Figures 2a and 2b depict the prediction results of our 4 models articulated by precision at k. The AdaBoost model gives the best prediction result for both targets. For predicting first class attendance, AdaBoost with tree number = 2000 has the highest precision at 10 which equals to 0.73 and AdaBoost with tree number = 1000 gives the best precision at 20, 50, 100 which equals to 0.67, 0.56 and 0.46 respectively. For predicting BT completion, AdaBoost with tree number = 100 gives the best precision at 10, 20, 50, 100, which equals to 0.67, 0.62, 0.53, 0.44 respectively. As there are more students who did not attend the first class (385/2000

Table 1: Features used for class attendance and training completion prediction

Feature	Type	Remarks
provider_type	Nominal	Individual provider (paid by the Department of Social and Health Services) or agency provider (paid by private home care agencies). {IP, AP}
student_ethnicity	Nominal	student ethnicity. {Asian Indian, White etc}
student_language	Nominal	student language. {English, Russian, etc}
student_age	Numerical	student age. {Mean = 39, Median = 37}
os_month	Numerical	Month of O&S tracking date. {1,2,$\cdots$,12}
os_day	Numerical	Day of O&S tracking date {1,2,$\cdots$, 31}
class_language_containEnglish	Boolean	Whether the student's profile includes an English language selection. {Yes, No}
class_language_containOther	Boolean	Whether the student's profile includes a language other than English.{Yes, No}
county	Nominal	student's county of residence {King County, Pierce County,etc}
county_income_mean	Numerical	The mean income(in USD) for the county.{mean = 67011, median = 65498}
county_income_median	Numerical	The medium income(in USD) for the county. {mean = 55468, median = 54727}
county_population	Numerical	The population for the county. {mean = 28672, median = 29582}
os_transferredhours	Numerical	Transferred hours for O&S. {mean = 0.9965, median = 0}
duration_to_oscomplete	numerical	Duration(in number of days) beween O&S completion date and O&S tracking date.{mean = 0.842, median = 1.500}
first_module	Nominal	The module of first registered class {Module 1, Module 2,..., Module 20, etc}
duration_to_class	Numerical	Duration(in number of days) between class date and O&S tracking date.{mean=72.05, median = 67.42}
first_class_interpreter	Boolean	Whether the student articulated a need for interpreter services.{Yes,No}
duration_to_class_registration	Numerical	duration(in number of days) between class registration date and O&S tracking date.{mean = 32.647, median = 19.784}
num_terminations	Numerical	Number of terminating employment relationships before attending first class.{0,$\cdots$,7}
student_noshow_count	Numerical	Number of class absences before attending the first class. {0,$\cdots$,58}
student_withdraw_count	Numerical	Number of class withdrawals before attending the first class. {0,$\cdots$,60}
num_class_attendee	Numerical	Number of attendees in the first class. {3,$\cdots$,33}



(a) Precision at k for first class attendance

(b) Precision at k for training completion

Figure 2: Precision at k results

ROC_{AUC}		
Model	1st Class Attendance	Training Completion
SVM(radial)	0.578±0.012	0.600±0.011
LR	0.612±0.020	0.634±0.018
AD(1000)	0.627±0.013	0.673±0.025
AD(2000)	0.626±0.015	0.680±0.024
RF(2000)	0.608±0.012	0.672±0.023

Table 2: ROC_{AUC} results

= 19.25%) than the number of students who did not complete the training (229/2000 = 11.45%), it was slightly easier to predict top k students who were likely to not show up for their first class and explains the higher Prec@k for predicting class attendance.

Table 2 shows ROC_{AUC} results. For predicting first class attendance, AdaBoost with tree number = 1000 gives the best ROC_{AUC} at 0.627. For predicting BT completion, AdaBoost with tree number = 2000 gives the best ROC_{AUC} at 0.68. Low ROC_{AUC} indicates the need for stronger inputs and feature attributes to the models. Although 19 out of 22 attributes were shared in both predicting problems, attributes such as duration to class registration, duration to class and first module were more useful in predicting BT completion than in predicting class attendance. This explains the increased ROC_{AUC} results for BT completion predictions. It provides an opportunity to understand why students choose to not attend their registered training classes and to collect more data at this early stage of the training journey.

3.2 Risk Profile Analysis

In this section, we illustrate how we use insights derived from decision tree modeling to profile students with different dropout rates, providing a tool to isolate target segments of high risk students so the business can take measures that can decrease dropout rate. Decision tree modeling enable us to acquire foundational knowledge necessary to develop educated hypotheses for customized interventions to support students with different risk profiles. Variable importance analysis using Random Forest also enhances our understanding of what factors influence training dropout and assists in our predictions.

At the root note of Figure 3a, the average first class attendance rate is almost 81% among 5,303 students. That is, the overall dropout rate is 19%. For students who didn't enroll in either module 1 or 2 as their first class[1], they demonstrated a significantly higher risk of not attending the training – 54% will not show up for their first registered class. Using the same decision tree, we are also able to infer that both county and age are important factors. For example, students who do not reside in certain counties [2] above and are younger than 49 are less likely to attend the first class compared to those who are older than 49. Younger students, English speaking students and students who take longer to complete O&S exhibit higher risk of not attending their first class. The variable importance from random forest shows that duration to class registration, duration to class are other most important indicators. The larger the time gaps, the higher the dropout rates are.

Figure 3b gives a decision tree for training completion. From the display, we can see if students have two or more class absence records before actually attending the first class, their completion rate decreases to 60%, which is much lower than the average completion rate of 89%. Among these students, if their first class is not Module 1, then the likelihood that the student will complete training drops to 27%. It shows duration to class registration and class location (i.e county) play important role for training completion. Duration to class and student age are also shown as important indicators using random forest variable importance analysis. In addtion, knowing the count of class absence record and first class module gives a much better understanding about the BT completion. Figure 3b shows that even for students who had one or zero class absences. If they register for the class too late (in our case this amounts to more than 52 days after being hired), then the probability of completing the training is even lower.

4. RELATED WORK

Prior studies([3],[7],[8]) have been conducted to explain academic performance and to predict the success or failure across a variety of students in a wide-range of educational settings. These studies focused heavily on the explanatory factors associated with a student's learning behavior and training journey and which of those may cause separation between student types. Machine learning algorithms have been successful in high school and college education settings, most helpful in predicting graduation[4], course participation[5], and other academic outcomes[6]. These algorithms also provide great value to the student success[9].

Lakkaraju et al.[6] used several classification models to identity students at risk of adverse academic outcomes and used precision_at_top_K and recall_at_top_K to predict risk early. The authors compared ROC curves for two cohorts for algorithms Random Forest, AdaBoost, Linear Regression, SVM and Decision Tree. The authors demonstrated that Random Forests outperformed all other methods. Aguiar et al.[10] selected and prioritized students who are at risk of not graduating high school on time by prediction the risk for each grade level and reported precision at top 10%, accuracy, and MAE for ordinal prediction of time to off-track.

[1]Currently, students are allowed to attend classes out of sequence in order to complete their training before the mandatory deadline.

[2]Counties include: Benton, Clark, Cowlitz, Douglas, Grays Hoarbor, Lewis, Mason, Skagit, Stevens, Walla Walla and Whatcom

Johnson et al.[11] used d-year-ahead predictive model to predict on-time graduation for different grade level. Vihavainen et al.[5] found a higher likelihood of failing their mathematics course could be detected in an early stage using Bayesian network. Radcliffe et al.[4] used logit probability model and parametric survival models to found that demographic info, academic preparation and first-term academic performance have a strong impact to graduation. Dekker et al.[12] gave experimental results which showed decision trees gave a high accuracy for predicting student success and improved prediction accuracy using cost-sensitive learning.

Other prior studies have highlighted some important indicators that influence students' performance like a student's age and absence rates[6]. Based on these features, Early Warning Indicator (EWI) systems are rapidly being built and deployed using machine learning algorithms[6]. Similar to other research in Educational Data Mining (EDM), we use precision at k to measure the prediction result([6], [10], [13]) and, like in traditional education systems, our motive is to most effectively and efficiently target our limited resources to assist and suppor students. Typically, ensemble models outperformed individual models[7] and this held true in our case as well. While random forest has proven to be an extremely useful and powerful machine learning technique in educational research[11], our results indicated that AdaBoost outperformed random forest.

5. CONCLUSION AND FUTURE WORK

In this study, we demonstrated preliminary results for predicting home care student training dropout from a large, heterogeneous dataset containing student demographics and engineered features extracted from training patterns. Predicting dropout at varying stages of an adult learner's training journey yielded promising results from a skewed dataset of over 5,303 students with AdaBoost (2,000 trees) providing the strongest predictions (prec@10 = 0.73 and ROC_{AUC} = 0.625. Prior history of class absence and time effects (duration to registration, duration to first class) were among the strongest individual predictors of dropout, as were class module sequence, county, and student age. The results demonstrate that applying machine learning techniques to demographic data and learning behavior data (e.g. duration to registration, duration to first class) can achieve adequate prediction quality in predicting the top k highest risk students out of a pool of newly hired HCAs. This enables efficient use of limited capacity and resources to support students of greatest need. Insights revealed in this study inspired training operation staff to explore alternatives, including encouraging newly hired HCAs to register for training early and strongly recommend proper class sequence to support students success in their training.

Future work will investigate collecting more information about students, such as their motivations, propensity for self-efficacy, and life circumstances to determine if there are other factors at play on a personal level that my uncover additional features that can contribute to our target predictions around training dropout.

6. REFERENCES

[1] Charissa Raynor. Innovations in training and promoting the direct care workforce. *Public Policy & Aging Report*, 24(2):70–72, 2014.

[2] Colombo Francesca, Llena-Nozal Ana, Mercier Jérôme, and Tjadens Frits. *OECD Health Policy Studies Help Wanted? Providing and Paying for Long-Term Care: Providing and Paying for Long-Term Care*, volume 2011. OECD Publishing, 2011.

[3] S Kotsiantis, Christos Pierrakeas, and P Pintelas. Predicting students'performance in distance learning using machine learning techniques. *Applied Artificial Intelligence*, 18(5):411–426, 2004.

[4] P Radcliffe, R Huesman, and John Kellogg. Modeling the incidence and timing of student attrition: A survival analysis approach to retention analysis. In *annual meeting of the Association for Institutional Research in the Upper Midwest (AIRUM)*, 2006.

[5] Arto Vihavainen, Matti Luukkainen, and Jaakko Kurhila. Using students' programming behavior to predict success in an introductory mathematics course. In *Educational Data Mining 2013*, 2013.

[6] Himabindu Lakkaraju, Everaldo Aguiar, Carl Shan, David Miller, Nasir Bhanpuri, Rayid Ghani, and Kecia L Addison. A machine learning framework to identify students at risk of adverse academic outcomes. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1909–1918. ACM, 2015.

[7] Dursun Delen. A comparative analysis of machine learning techniques for student retention management. *Decision Support Systems*, 49(4):498–506, 2010.

[8] S Kotsiantis, Kiriakos Patriarcheas, and M Xenos. A combinational incremental ensemble of classifiers as a technique for predicting students' performance in distance education. *Knowledge-Based Systems*, 23(6):529–535, 2010.

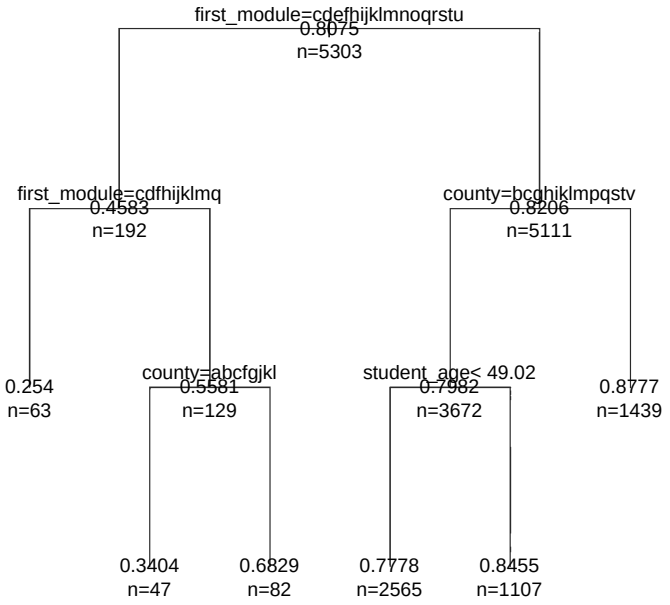
[9] Cristóbal Romero and Sebastián Ventura. Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(6):601–618, 2010.

[10] Everaldo Aguiar, Himabindu Lakkaraju, Nasir Bhanpuri, David Miller, Ben Yuhas, and Kecia L Addison. Who, when, and why: A machine learning approach to prioritizing students at risk of not graduating high school on time. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 93–102. ACM, 2015.

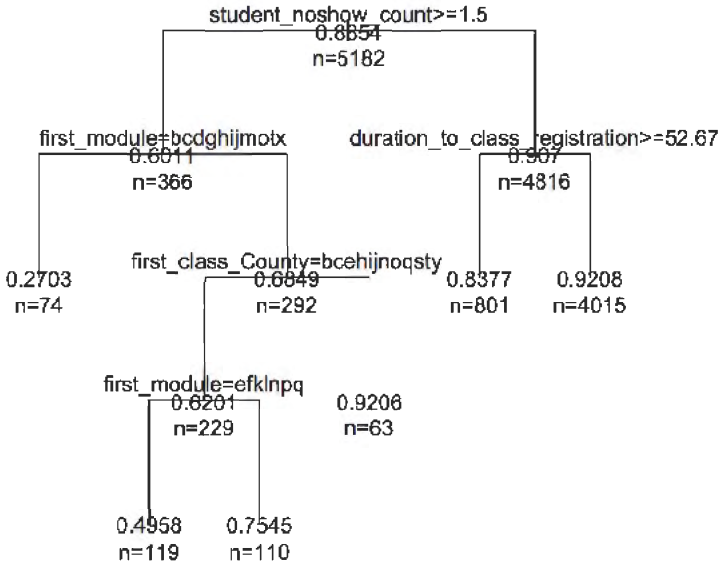
[11] Reid A Johnson, Ruobin Gong, Siobhan Greatorex-Voith, Anushka Anand, and Alan Fritzler. A data-driven framework for identifying high school students at risk of not graduating on time.

[12] Gerben Dekker, Mykola Pechenizkiy, and Jan Vleeshouwers. Predicting students drop out: A case study. In *Educational Data Mining 2009*, 2009.

[13] Everaldo Aguiar, G Alex Ambrose, Nitesh V Chawla, Victoria Goodrich, and Jay Brockman. Engagement vs performance: Using electronic portfolios to predict first semester engineering student persistence. *Journal of Learning Analytics*, 1(3):7–33, 2014.



(a) First Attend



(b) Training Completion

Figure 3: Decision Trees

Few hundred parameters outperform few hundred thousand?

Amar Lalwani
funtoot
2nd floor, Sancia House,
14th Cross,1st Stage Domlur,
Bengaluru 560071, India
amar.lalwani@funtoot.com

Sweety Agrawal
funtoot
2nd floor, Sancia House,
14th Cross, 1st Stage Domlur,
Bengaluru 560071, India
sweety.agrawal@funtoot.com

ABSTRACT

Knowledge Tracing plays a key role to personalize learning in an Intelligent Tutoring System including funtoot. Bayesian Knowledge Tracing, apart from other models, is the simplest well-studied model which is known to work well. Recently, Deep Knowledge Tracing based on Deep Neural Networks, was proposed with huge promises. But, soon after, it was discovered that the gains achieved by DKT were not of significant magnitude as compared to Performance Factor Analysis [13] and BKT and its variants proposed in [6]. In the quest of examining and studying these models, we experiment with them on our dataset. We also introduce a logical extension of DKT, Multi-Skill DKT, to incorporate items requiring knowledge of multiple skills. We show that PFA clearly outperforms all the above mentioned models when the AUC results were averaged on skills while PFA and DKT, both were equally good, when they were averaged on all data points.

Keywords

Deep Knowledge Tracing, Adaptive Learning, funtoot, Bayesian Knowledge Tracing, Intelligent Tutoring System, Performance Factor Analysis

1. INTRODUCTION

An Intelligent Tutoring System's main aspect is to deliver the instruction and provide feedback as and when required. To do that, the system requires to measure the knowledge state of a student with respect to the content available. The system continuously monitors the student's performance, updates the knowledge state and based on that takes further decisions. The techniques capable of performing these functions are called Knowledge Tracing models.

Bayesian Knowledge Tracing [2] has been one of the most predominantly researched models in the educational data mining domain. BKT is a 2-state skill specific model, where the student's knowledge state can take either of the two values: learned or unlearned. Moreover, a skill once learned cannot be unlearned. These assumptions make it a very simple and constrained model and has led lots of researchers to extend the model by enhancing it with new features to improve its performance; making it less constrained so to say. For instance [10] extend BKT in the scenario where the students do not necessarily use the system in the same day.

Authors of [14] proposed an individualized BKT model which fits not only the skill specific parameters, but also student specific parameters and have reported significant gains over standard BKT.

Educational data mining techniques can now very accurately predict how much a student has learned a Knowledge Component (KC). But it doesn't give information about the exact moment when the KC was learnt. [3] discusses a technique about finding a moment of learning.

Another model Performance Factor Analysis (PFA) is a logistic regression model proposed in [7] which showed better performance than standard BKT. Unlike BKT, PFA can incorporate items with multiple skills. PFA makes predictions based on the item difficulty and historical performances of a student. [4] has compared BKT and PFA by using various model fitting parameter models like Expectation Maximization (EM) and Brute Force (BF). Knowledge tracing models with EM have shown performance comparable to PFA[4].

The most recently published model - DKT [9] is the newest technique in this area of research. DKT is an LSTM [5] network, a variant of recurrent neural network [11] which takes as input a series of exercises attempted by the student and correspondingly a binary digit suggesting if the exercise was answered correctly or not. DKT has shown significant gains over BKT which is a very tempting gain for any researcher in this community to look into and study further. Papers like [6], [13] and [12] did just that.

Authors in [13] have pointed out few irregularities in the dataset used by authors in [9] which, when accounted for, reduce the gain reported by using DKT. They also reported that DKT doesn't quite hold an edge when the results are compared with PFA.

Another standard framework for modelling student responses, Temporal extension of Item Response Theory (IRT) is compared with DKT in [12]. Authors have reported that the variants of IRT consistently matched or outperformed DKT.

Recent paper [6] studies DKT even further and explains why DKT might be better. It has been pointed out that DKT inherently exploits the characteristics of the data which standard models like BKT cannot. So, in order to make a fair comparison between the two, authors have presented three

different variants of BKT with forgetting, skill discovery and latent abilities which might help BKT make use of information from the data the way DKT does.

Having introduced these variants, the authors also make a point that Knowledge Tracing might not require the "depth" that deep learning models offer.

Being an Intelligent Tutoring System, funtoot's tutor module requires sophisticated knowledge tracing technique which models the process of knowledge acquisition and helps students achieve mastery. One such model operates at the level of LGs (discussed in section 2) which models the committance and avoidance of them with time and practice. In the context of this paper, these LG models are of prime importance to us and henceforth we will refer LGs as skills. Also, considering user experience, we need a model which can be used for predictions in real time without compromising on user latency.

In this paper, we test standard BKT, the variants of BKT, DKT and PFA on the funtoot dataset and examine the results. We also introduce a logical and trivial extension of DKT to accommodate the items which involve multiple skills. Out of all the models considered in this article, PFA is one such model which allows items with multiple skills. But in our dataset, each of the skills in the item has its own response and hence it is modelled separately in PFA.

The rest of the paper is organized as follows: section 2 gives a brief introduction to our product funtoot and its knowledge graph. Section 3 discusses the experiments on funtoot dataset and results. Section 4 discusses the future work and conclusion.

2. FUNTOOT

Funtoot[1] is a personalized digital tutor which is currently being used actively in around 125 schools all over India with the total of 99,842 students registered. The curriculum of math and science for grades 2 to 9 is covered by funtoot.

Schools in India are typically affiliated with one of the boards of education[2]. Curriculum for math and science from the following boards of education are included in funtoot:

- CBSE[3] board for grades 2 to 9,
- Karnataka State Board[4] for grades 2 to 8,
- ICSE[5] board for grades 2 to 8 and
- IGCSE[6] board for grades 2 to 3.

[1] http://www.funtoot.com/
[2] https://en.wikipedia.org/wiki/Boards_of_Education_in_India
[3] https://en.wikipedia.org/wiki/Central_Board_of_Secondary_Education
[4] https://en.wikipedia.org/wiki/Karnataka_Secondary_Education_Examination_Board
[5] https://en.wikipedia.org/wiki/Indian_Certificate_of_Secondary_Education
[6] https://en.wikipedia.org/wiki/International_General_Certificate_of_Secondary_Education

2.1 Funtoot Knowledge Graph

Pedagogy team at funtoot has created a funtoot ontology around the subjects Math and Science. This ontology represents the various learning units of any subject and their relationships, which is created based on human expertise in the subject matter. All the above mentioned curricula are later derived from this funtoot ontology based on the age group and grade.

An ontology for a subject is created as follows:

1. a subject is broken down into the smallest teachable sub-sub-concepts
2. it is then mapped to determine inter-dependencies/connections between concepts, sub-concepts (*sc*) and sub-sub-concepts (*ssc*) as shown in the figure 1,
   Consider the example shown in figure 1. Subject *Math* contains a concept *Triangle*, and *Triangle* contains a sub-concept *Congruency*. Sub-concept contains two sub-sub-concepts: *Rules of Congruency* and *Applications of Congruency*. Sub-sub-concepts are connected by "depends-on" relationship. Here, *Applications of Congruency* is dependent on *Rules of Congruency*, which suggests that the latter is a prerequisite for the former.
3. learning gaps (definition 1) are determined in the sub-sub-concepts

   DEFINITION 1. *Learning Gap (LG): "A learning gap is a relative performance of a student in a specific skill, i.e. difference of what a student was supposed to learn, and what he actually learned in a skill.* [7] *"*

   "A misunderstanding of a concept or a lack of knowledge about a concept that is required for a student to solve or answer a particular question is also a learning gap"

   For instance, a question "Solve 12 + 18" is given to student *Alice*. If *Alice* makes a mistake while adding carry and answers 20, we say that a LG (*carry-over error*) has been *committed*. Had she answered 30, this LG would have been said to be *avoided*. This question might also have other LGs which could have been committed simultaneously with the LG mentioned above. If the response is correct, all the LGs of a question are said to have been avoided.

   In figure 1, *Applications of Congruency* is an ssc containing LG_1, LG_2 and LG_3. Learning gaps can have "induce" relationships. In our example, LG_1 induces LG_2.
4. inter-dependencies get refined based on the data-points received by funtoot through the user's interaction
5. an SSC is further divided into six Bloom's Taxonomy Learning Objectives (*btlos*) using Bloom's Taxanomy [1]. Each learning objective has five difficulty

[7] http://edglossary.org/learning-gap/

Difficulty Level \ BTLO	Remember	Understand	Apply	Analyze	Evaluate	Create
1						
2						
3						
4						
5						

Table 1: Btlos, Difficulty levels ⇒ Complexities

levels as shown in table 1. Each cell (for instance, Remember1, Apply2 and so on) in table 1 is called a *complexity* in funtoot.

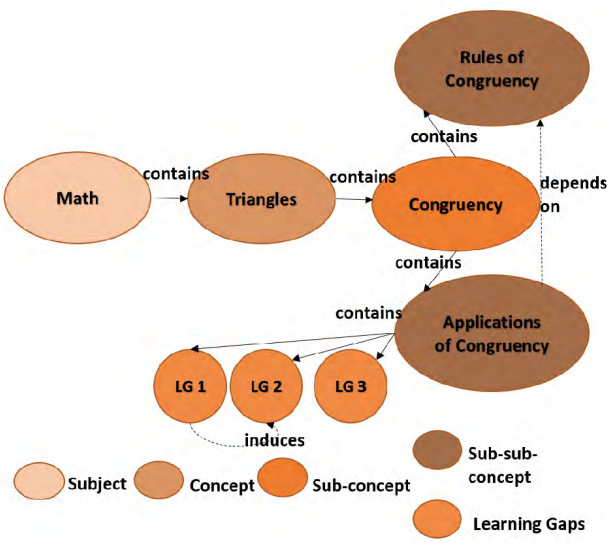


Figure 1: Funtoot Knowledge Graph

2.2 Dataset

During a student's interaction with funtoot, information like: session, the scope of the question (which includes $grade - subject - topic - subtopic - subsubtopic - complexity - question$), question identifier, start time, total attempts allowed based on the student's performance, time taken, attempts taken, information about hints, LGs committed in each attempt, assistance provided and so on is logged.

In the study presented in this paper, we model LG as a skill. We aim to predict a student's proficiency in a particular LG. When a student is presented with an item, several attempts are provided to solve it. In an unsuccessful attempt a student might commit more than one LG as explained in section 2 and the same LG can also be committed in several attempts. We know apriori the set of LGs that are exposed by a question. With this information at hand, we need an impression of each of these LGs for the student in the context of this item.

Consider a hypothetical example. *Alice* attempts an item q from a subtopic *Rules of Congruency* having skills s_1, s_2, s_3. The series of attempts is shown in table 2.

Attempt no.	s_1	s_2	s_3
1	0	1	1
2	0	1	1
3	0	0	1
4	1	1	1
Overall Outcome	0	0	1

Table 2: Attempts made by *Alice* while solving q

In the above table, 1 represents avoidance and 0 represents committance. As shown in the table, *Alice* committed s_1 in attempts 1, 2 and 3. *Alice* committed s_2 in attempt 3. *Alice* avoided s_3 in all attempts. The overall outcome of *Alice* in LGs s_1, s_2 and s_3 is $(0, 0, 1)$ which is a logical AND over all attempts. This means that s_1 and s_2 are committed and s_3 is avoided. From now on, we will refer these outcomes as committances and avoidances and they will be used for modelling. So this problem attempt of *Alice* gives rise to three data points.

For this experiment we have used data of 6^{th} grade CBSE math from date $2015-07-25$ to $2017-01-30$. Syllabus descendant hierarchy for this dataset is as follows: 22 topics, 69 subtopics, 119 sub-sub-topics, 541 complexities and $1,524$ problems. This dataset has $26,06,022$ entries of problem attempts involving 442 skills. This data is about 176 schools with $11,820$ students and $1,524$ problems. From this dataset, the data of students having less than 100 problem attempts were excluded. This gives us $24,47,027$ problem attempts involving 442 skills with 7780 students and $1,523$ problems. Finally, we have $56,04,227$ data points where $42,68,503$ are avoidances (class 1) and $13,35,724$ are committance (class 0).

In the context of the example shown in table 2, the length of $Alice's$ attempt to solve a question q can be said as three, as there are three skills involved. Given this definition, of length of the problem attempt, figure 2 shows the distribution of the length of the problem attempts in the dataset. 38.18% of the total problem attempts have 1 skill, i.e., length is 1 and 29.47% of the problem attempts have length 2.

3. EXPERIMENTS

In this section, we discuss the experiments done on our dataset and report the results. Consider a hypothetical dataset of student *Alice* attempting questions q_1 and q_2 in the same order. Question q_1 has three skills A, B and C, question q_2 has two skills B and C. *Alice* gets only one attempt for both the questions wherein she commits skill B and C and skill B in questions q_1 and q_2 respectively. This example is used in this section to explain the training datasets for each of the techniques.

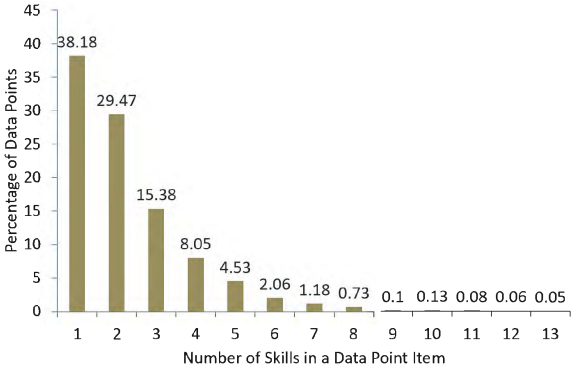


Figure 2: Data Distribution

3.1 Bayesian Knowledge Tracing

After DKT [9], authors in [6] have explored and hypothesized the properties of the data which DKT exploits while the standard BKT cannot. To equip BKT with those capabilities, the authors have proposed three variants of BKT: BKT with forgetting (BKT+F), BKT with skill discovery (BKT+S) and BKT with latent-abilities (BKT+A).

We have used the author's implementation of BKT and its three variants published on `https://github.com/robert-lindsey/WCRP/tree/forgetting` to train on our dataset. The data format required by these BKT variants is as shown in table 3. As discussed in the earlier section 1,

skill ID	response series
A	1
B	0, 0
C	0, 1

Table 3: BKT data format

BKT is a skill specific model and thus, three models need to be built one each for skills A, B and C. Each model needs the time series of responses as shown in the table 3.

All variants of BKT except the ones where skill discovery is involved, namely BKT, BKT+F, BKT+A and BKT+FA operate on the skills provided by the data. The remaining variants: BKT+S and BKT+FSA completely ignore the expert tagged skills available in the data. This is achieved by setting the non-parametric prior, β on the expert tagged skills as 0.

3.2 Performance Factor Analysis

Like BKT, PFA being a skill specific model requires a different model to be built for each skill. Logistic Regression model of [8] is used in the implementation of PFA. For each skill, the response is a function of the skill difficulty, number of prior student success (avoidances) responses and number of prior student failure (committances) responses for the skill. From the implementation point of view, the decision function has two variables - the number of prior success instances and the number of prior failure instances for the skill. Also, a bias is added in the decision function (achieved by the intercept) which serves as the skill difficulty. The data format needed by PFA is as shown in figure 4.

skill ID	no. of failures	no. of successes	response
A	0	0	1
B	0	0	0
C	0	0	0
B	1	0	0
C	1	0	1

Table 4: PFA data format

3.3 Deep Knowledge Tracing

The implementation of LSTM based DKT published on `https://github.com/mmkhajah/dkt` is used to train our dataset. The neural network of DKT requires the input as one hot encoding of skills as well as responses for each of them, while output is the probability of correctness of each of the skills. Hence the size of the input is twice the number of skills and that of the output is the number of skills. The serial number in the table 5 shows the order in which the inputs are fed into the network. The input in the table signifies the previous output while the response shows the expected output out of the network. The odd bits in the input represent one hot encoding of the skills while the even bits represent their responses. X in the output shows that the bit can take either 0 or 1.

serial no.	input	response
1	0, 0, 0, 0, 0, 0	1, X, X
2	1, 1, 0, 0, 0, 0	X, 0, X
3	0, 0, 1, 0, 0, 0	X, X, 0
4	0, 0, 0, 0, 1, 0	X, 0, X
5	0, 0, 1, 0, 0, 0	X, X, 1

Table 5: DKT data format

As discussed in subsection 2.2 that to figure out the final outcomes for the LGs in an item attempt, there is no clear or fixed ordering. But the time series to be fed into the network of DKT requires us to establish the ordering between them. We sample the orderings randomly and average the results on them. The sample dataset in the table 5 is one such ordering. Another random ordering can be seen in the table 6. The skills of the item q_1 are in the order A, B, C in table 5 while their order is B, A, C in table 6. The other way to get an ordering is to get rid of the ordering itself by merging the data points of the skills in an item which is explained in the following subsection.

serial no.	input	response
1	0, 0, 0, 0, 0, 0	X, 0, X
2	0, 0, 1, 0, 0, 0	1, X, X
3	1, 1, 0, 0, 0, 0	X, X, 0
4	0, 0, 0, 0, 1, 0	X, X, 1
5	0, 0, 0, 0, 1, 1	X, 0, X

Table 6: Shuffled skills DKT data format

3.4 Multi-skill DKT

As explained in the context of DKT, the orderings among the skills in the item are sampled randomly. In order to get rid of such orderings, we introduce an extension of DKT: Multi-skill DKT which can incorporate the items having multiple skills efficiently. It can be seen from the table 7 that the three data points of q_1 and two data points of q_2 are consolidated and we are left with two data points in total. The size and structure of the inputs and outputs still remain the

serial no.	input	response
1	0, 0, 0, 0, 0, 0	1, 0, 0
2	1, 1, 1, 0, 1, 0	X, 0, 1

Table 7: Multi-Skill DKT data

same. The only difference is that the input and output can have the information about multiple skills simultaneously.

3.5 Results

For all the algorithms, we use three replications of 2-fold cross validation, which gives us 6 folds in total on which the results are averaged. We use Area under the curve of Receiver Operating Characteristics (ROC), which we will refer as the AUC. Paper [6] discusses the inconsistent procedures used to compute and compare performance of BKT and DKT. We therefore compute AUC both by averaging on all data points and by averaging on skills. The results of our experiments on funtoot dataset are shown in figure 3.

When AUC is averaged on all the data points, the relative difference in performance between algorithms is very low, 0.83 being the lowest and 0.88 being the highest. PFA and DKT share the highest performance of 0.88 AUC. Multi-skill DKT lags a bit behind DKT by 0.03 AUC units (0.85 AUC). All the variants of BKT also lag behind DKT and PFA by not a very big margin, the highest being 0.05 AUC units. BKT has the lowest AUC of 0.83, BKT+FSA has the highest AUC of 0.85 and the rest of them have an AUC of 0.84, which depicts that they all show equivalent performance.

The relative difference in performance between algorithms is higher when AUC is averaged on skills, the lowest being 0.64 AUC of BKT+F and highest being 0.88 AUC of PFA which is 37.5% gain. PFA with an AUC of 0.88 outperforms all the methods by having a minimum gain of 17% (0.75 AUC of DKT and BKT+FSA) and maximum gain of 37.5% (0.64 AUC of BKT+F). Here also, the magnitude of difference between DKT and Multi-skill DKT is very less, 0.04 AUC units to be precise with Multi-skill DKT lagging behind.

With BKT, BTK+F, BKT+A and BKT+FA having AUCs of 0.65, 0.64, 0.68 and 0.67 respectively, it is clear that *Forgetting* adds no value. The number of skills discovered by both BKT+S and BKT+FSA are in the range of $145 - 175$ compared to 442 original skills. The *Skill Discovery* extension provides reasonable gains which are evident from the AUCs of BKT and BKT+S (9% gain) and BKT+FA and BKT+FSA (12% gain). The magnitude of the gains achieved by *Abilities* extension is very less, 0.003 AUC units in the case of BKT, BKT+A and BKT+F, BKT+FA. Finally, the different variants of BKT achieve a gain of maximum 15% over standard BKT. Notably, the best version of BKT, that is, BKT+FSA and DKT, perform equally.

4. DISCUSSION AND FUTURE WORK

Our aim of this study was to explore the performance of standard BKT, all of its variants proposed in [6], PFA and DKT on funtoot dataset. The results we have got are in sync with the results in [6]. When the AUC results were computed by averaging over skills, DKT and BKT+FSA perform equally well while DKT outperforms standard BKT with the gain of 15%. Also, BKT+S gave a performance

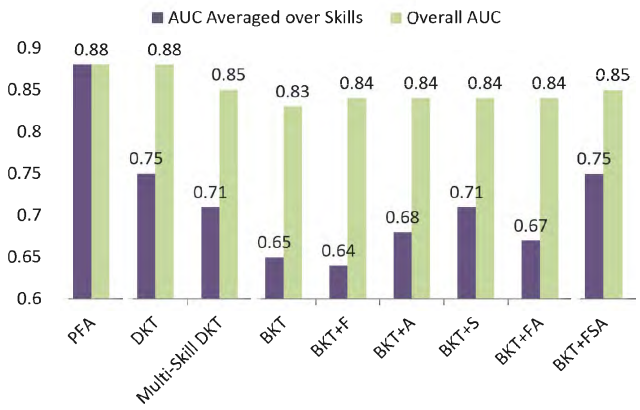


Figure 3: A comparison of PFA, DKT, Multi-skill DKT, BKT and its variants

which was very close to DKT. Though DKT does perform better when the AUC results are averaged over all data points, the magnitude of the gain is significantly low.

Similar kind of results hold true for PFA. PFA achieves a high gain compared to all the models when AUC results are averaged over skills. When AUC results are averaged over all data points, PFA equals DKT's performance and outperforms the rest of the models, though not with a very high margin. This is not consistent with the results in [13] where DKT outperforms PFA though, not overwhelmingly.

The above results reinforce the hypothesis proposed in [6] that the domain of knowledge tracing seems to be shallow and may not require the depth that the deep neural networks offer. The predictive or the explanatory power of a model can also be characterized in terms of the number of parameters the model fits. One of the reasons why DKT is expected to be more successful than other models, at the cost of interpretability, is that it has weights in the order of hundreds of thousands. Moreover, being made up of a layer of LSTM cells, DKT has the capability of looking back arbitrary number of timesteps. On the contrary, variants of BKT and PFA are very simple and interpretable models. Their simplicity can easily be attributed to the small number of parameters they fit.

Standard BKT needs four parameters: *pInit* (the probability that the student is in learned state before the first practice), *pLearn* (the probability that the student transitions from not learned state to the learned state at each practice), *pGuess* (the probability that the student guesses the answer being in the unlearned state) and *pSlip* (the probability that the student accidentally makes a mistake being in the learned state). In PFA, it is even better, only three parameters are learned per skill - item difficulty and one coefficient each for prior failures and successes. With this, the total parameters for a few hundred skills (which is true in our case) would be a few hundred parameters: $three \times number\ of\ skills$. Hence, in our context, it seems appropriate to say that few hundred parameters are better than few hundred thousand parameters.

Both BKT and DKT, in an abstract sense, are the models

which maintain the knowledge state of the student. With each response of the student, the knowledge states are updated and those states are used to generate future predictions. They both require the time series data of the student's responses. This is significantly different than the type of data required by PFA. PFA operates on abstract features of student's interactions like total number of prior successes and failures. It occurs to us that the abstract features are smoother than the time series data of responses. It seems the domain of knowledge tracing can be deciphered better if the abstract features are used instead of detailed trail of responses which might be noisy. More studies and experiments are required to validate this point.

The skills used in our experiment are the LGs from the funtoot Knowledge Graph which are tagged at the level of subsubtopic which acts as a context of LG. Also, an LG can occur in multiple subsubtopics. The discovered skills in our experiments of BKT+S and BKT+FSA were in the range of $145 - 175$ which is close to the number of subsubtopics (119) in our dataset. We suspect that there is some relation between the subsubtopics in our dataset and the skills discovered. We would like to investigate this further in future. DKT also supports skill discovery as proposed in [9] which we would look into in future to compare the skills discovered by several algorithms.

Funtoot dataset has items with multiple skills which forced us to extend DKT and come up with Multi-skill DKT. This variant of DKT underperformed marginally as compared to DKT. We do not have a clear understanding about why this is so and hence this also requires further study. Since we have used a very crude dataset, that is, does not contain features about attempts, time durations, hints, item context, etc., it would be interesting to use them with DKT and see if the depth of DKT can exploit them.

5. REFERENCES

[1] L. W. Anderson, D. R. Krathwohl, and B. S. Bloom. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Allyn & Bacon, 2001.

[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, 1994.

[3] R. S. d Baker, A. B. Goldstein, and N. T. Heffernan. Detecting the moment of learning. In *International Conference on Intelligent Tutoring Systems*, pages 25–34. Springer, 2010.

[4] Y. Gong, J. E. Beck, and N. T. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *International conference on intelligent tutoring systems*, pages 35–44. Springer, 2010.

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[6] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? In *Proceedings of the 9th International Conference on Educational Data Mining (EDM 2016)*, pages 94–101, 2016.

[7] P. I. Pavlik Jr, H. Cen, and K. R. Koedinger. Performance factors analysis–a new alternative to knowledge tracing. *Online Submission*, 2009.

[8] 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.

[9] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.

[10] Y. Qiu, Y. Qi, H. Lu, Z. Pardos, and N. Heffernan. Does time matter? modeling the effect of time with bayesian knowledge tracing. In *Educational Data Mining 2011*, 2010.

[11] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[12] K. H. Wilson, Y. Karklin, B. Han, and C. Ekanadham. Back to the basics: Bayesian extensions of irt outperform neural networks for proficiency estimation. In *Proceedings of the 9th International Conference on Educational Data Mining (EDM 2016)*, pages 539–544, 2016.

[13] X. Xiong, S. Zhao, E. G. Van Inwegen, and J. E. Beck. Going deeper with deep knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining (EDM 2016)*, pages 545–550, 2016.

[14] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180. Springer, 2013.

Tell Me More: Digital Eyes to the Physical World for Early Childhood Learning

Vijay Ekambaram*, Ruhi Sharma Mittal*, Prasenjit Dey,
Ravi Kokku, Aditya K Sinha, Satya V Nitta
IBM Research
vijaye12@in.ibm.com, ruhisharma@in.ibm.com, prasenjit.dey@in.ibm.com
rkokku@us.ibm.com, adksinha@in.ibm.com, svn@us.ibm.com

ABSTRACT

Children are inherently curious and rapidly learn a number of things from the physical environments they live in, including rich vocabulary. An effective way of building vocabulary is for the child to actually interact with physical objects in their surroundings and learn in their context [17]. Enabling effective learning from the physical world with digital technologies is, however, challenging. Specifically, a critical technology component for physical-digital interaction is visual recognition. The recognition accuracy provided by state-of-the-art computer vision services is not sufficient for use in Early Childhood Learning (ECL); without high (near 100%) recognition accuracy of objects in context, learners may be presented with wrongly contextualized content and concepts, thereby making the learning solutions ineffective and un-adoptable. In this paper, we present a holistic visual recognition system for ECL physical-digital interaction that improves recognition accuracy levels using (a) domain restriction, (b) multi-modal fusion of contextual information, and (c) semi-automated feedback with different gaming scenarios for right object-tag identification & classifier re-training. We evaluate the system with a group of 12 children in the age group of 3-5 years and show how these new systems can combine existing APIs and techniques in interesting ways to greatly improve accuracies, and hence make such new learning experiences possible.

1. INTRODUCTION

Children learn a lot from the physical environment they live in. One of the important aspects of early childhood learning is vocabulary building, which happens to a substantial extent in the physical environment they grow up in [17]. Studies have shown that failure to develop sufficient vocabulary at an early age affects a child's reading comprehension and hence their ability to understand other important concepts that may define their academic success in the future. It is also evident from a study that failure to expose a child to sufficient number of words by the age of three years leads to a 30 million word gap between kids who have been exposed to a lot of quality conversations, versus the ones that have not been exposed as much [13].

Vocabulary building has been a theme for early childhood learning and is closely associated with its context in the physical world. The exploration of physical surroundings of the child triggers new vocabulary and vice-versa. Relating physical world objects and concepts, to digital world content requires seamless flow of information. Increasingly, availability of cheap sensors such as camera, microphone etc. on connected devices enable capture of physical world information and context, and translate them to personalized digital learning.

An envisioned system uses mobile devices to take pictures of the child's physical surroundings and make the best sense out of the picture. This is then translated to a learning session where the child is taught about the object in focus, its relation to other objects, its pronunciation, it's multiple representations, etc. Recognition of pictures for teaching a child requires high recognition accuracy. In-the-wild image recognition accuracies are in general low, especially for images taken with mobile devices. Moreover, pictures taken by a child is even more challenging given the shake, blur, lighting issues, pose etc. that come with it.

To this end, in this paper, we take a holistic approach of recognition-in-context using a combination of (a) domain restriction, (b) multi-modal fusion of contextual information, and (c) gamified disambiguation and classifier re-training using child-in-the-loop. Specifically, we use object recognition results from a custom-trained (with images from restricted domains) vision classifier, and combine them with information from the domain knowledge that is available whenever a new domain of words is taught to a child in the classroom or at home. We use a new voting based multimodal classifier fusion algorithm to disambiguate the results of vision classifier, with results from multiple NLP classifiers, for better accuracy. We show that using such a framework, we can attain levels of accuracy that can make a large majority of the physical-digital interaction experiences fruitful to the child, and also get useful feedback from the child at a low cognitive load to enable the system to retrain the classifier and improve accuracy. We tested our system with a group of 12 children in the age group of 3-5 years and show that children can play an image disambiguation game (that allows the child to verify what class label has actually been identified by the system) very easily with graceful degradation of performance on difficult images. In most cases, multi-modal context disambiguation improves object recognition accuracy significantly, and hence the human disambiguation step remains limited to one or two rounds, which ensures the child's continuing interest in the games and learning activities. The system learns from the child feedback, and the child in turn feels engaged to enable the system to learn over time. The nuggets of information made available about the object in focus at the end of playing a game were also found to be very engaging by the child.

In summary, this paper makes the following contributions:

- We take a holistic approach to address the challenges with

*these authors contributed equally to this work

automatic visual recognition for physical digital interaction to enable early childhood learning in context. Our three-stage approach includes (a) domain restriction, (b) contextual disambiguation and (c) gamified human disambiguation, which enables a platform for building a variety of early childhood learning applications with physical-digital interaction.

- We propose a novel re-ranking algorithm that uses the notion of strong vouching to re-order the output labels of a vision classifier based on strong supporting evidence provided by the additional context from semantic representation models in NLP, namely GloVe [6], Word2vec [11] and ConceptNet [5] (which can be textual cues in the form of classroom and curriculum context, domain focus, conversational input and clues, etc.). Note that, we use the terms "re-order" and "re-rank" interchangeably throughout this paper.

- We evaluate a simple disambiguation game for children to choose the right label from the Top-K labels given out by the system. Through an usability study with 12 children, we make the case that engaging user experiences can indeed be developed to bridge the gap between automatic visual recognition accuracies and the requirement of high accuracy for meaningful learning activities.

2. MOTIVATION AND RELATED WORK

Early childhood learning applications with physical-digital interaction fall into two categories: (i) *Application-initiated activities:* In this category, the child is given a context by the application and is required to find relevant physical object and take a picture [2]. For example, the application may prompt the child to take a picture of "something that we sit on", "a fruit", "something that can be used to cut paper", etc. (ii) *Child-initiated activities:* In this category, the child takes a picture of an object and intends to know what it is, where it comes from, other examples of the same type of objects, etc. For example, the child may take a picture of a new gadget or machine found in school, a plant or a leaf or a flower, etc. and wants to know more about them.

In each of these categories, the application is required to identify what the object is with Top-1 accuracy (i.e. a vision recognition solution should emit the right label at the top with high confidence). While a lot of advancement has been made in the improvement of accuracy of vision classifiers, Top-1 accuracy levels are still relatively low, although Top-5 accuracy levels (i.e. the right label is one of the top 5 labels emitted) are more reasonable. Nevertheless, the goal is to be able to work with the Top-5 list, and using the techniques described earlier, push the Top-1 accuracy to acceptable levels for a better interaction.

2.1 Vision Recognition Accuracy

To understand the efficacy of state-of-the-art solutions quantitatively, we experimented with two deep convolution neural networks (Baseline Model 1: VGGNet [18] and Baseline Model 2: Inception V3 [19]). Inception V3 has been found to have 21.2% top-1 error rate for ILSVRC 2012 classification challenge validation set [8]. Even in experiments where baseline models were custom trained with 300 training images per class and tested with images taken from iPad, we observed low Top-1 accuracy (of 72.6% in Baseline Model 1 and 79.1% in Baseline Model 2); i.e. one in about four images will be wrongly labeled. Even the Top-5 accuracy is 88.05% in Baseline Model 1 and 89.3% in Baseline Model 2. We also trained the Baseline models with the complete Imagenet[8] images for the considered classes and we observed <1% improvement. Further, when multiple objects are present in the image frame, the Top-1 accuracy degrades further (38.2% in Baseline Model 1 and 44.5% in Baseline Model 2 for 2 objects in a frame), and so does Top-5 accuracy (of 77.9% in Baseline Model 1 and 85.6% in Baseline Model 2). Note that this could be a common scenario with children taking pictures, in which multiple objects get captured in a single image frame. Observe that recent Augment Reality (AR) Applications such as Blippar [4], Layer [9], Aurasma [3] rely on similar vision recognition task, and hence run into similar inaccuracies in uncontrolled settings. While adult users of such applications may be tolerant to inaccuracies of the application, children may get disengaged when the system detects something wrongly or is unable to detect at all.

2.2 Multi-modal Information Fusion

Using additional information to identify the objects holds promise in imporving the accuracy of vision recognition. For instance, several past works ([22], [14], [15]) improve the image classification output based on the text features derived from the image. Specifically, authors in [20] propose techniques that train the model specifically with images that contain text, for efficient extraction of text and image features from the image. They also propose fusion techniques to merge these features for improving image recognition accuracies. While this may be possible in some scenarios, the application's accuracy will remain a challenge when such textual information embedded in the image is not present. Several works in literature propose indexing of images based on text annotations for efficient image search. [12] surveys and consolidates various approaches related to efficient image retrieval system based on text annotations. Likewise, [21] proposes techniques to label images based on image similarity concepts. These works are complementary, and do not address the problem of correctly determining the labels right when a picture is taken based on a context.

In summary, the early childhood learning scanarios require a holistic solution that leverages the state-of-the-art vision recognition solutions, but goes beyond in improving the detection accuracy of the image captured to make engaging applications for children. We describe one such holistic solution next.

3. PROPOSED APPROACH

Our goal is to enable a holistic solution for applications to provide as input an image taken by a child, and emit as output the final label that should be used as an *index* into the relevant learning content. A high level overview of our solution is depicted in Figure. 1. In one of the envisioned applications built for physical-digital interaction, a child takes a picture that is sent as input to the proposed ECL Image Recognition (ECL-IR) Module that emits the correct label of the image by applying the following three stages: (i) Stage 1: Domain Specific Customized Training (which improves Top-K accuracy), (ii) Stage 2: Domain Knowledge (DK) based disambiguation and reordering (which improves Top-1 accuracy) and (iii) Stage 3: Human Disambiguation game (confirmation step). We now discuss each of these stages in detail.

3.1 Stage 1: Domain Specific Customized Training of Baseline Models

The first stage of our solution strives to improve the Top-K accuracy of the vision classifiers by constraining the domain of child learning in which they are applied. In order to achieve this, we perform custom training of the baseline models with domain-specific data sets. This step is very commonly applied in most of the vision recognition use-cases for improving the Top-K accuracy and several reported statistics indicate good Top-K accuracy improvements through custom training. For example current state-of-art vision classifier [19] reports 94.6% Top-5 accuracy on ILSVRC 2012 classification challenge validation set. However, even this state-of-art vision classifier reports 21.2% Top-1 error rate on the same validation set. In the next section, we discuss how ECL-IR module improves Top-1 accuracy through contextualized reordering (Stage 2).

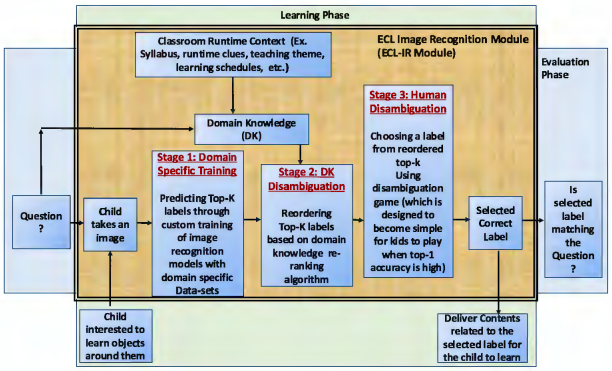


Figure 1: High Level Solution Overview

Algorithm 1: Algorithm to calculate DK Correlation Score

Input: Label, Domain Knowledge text (DK)
Output: DK Correlation Score (i.e. Semantic correlation between DK and Label)

1 For every word in DK, fetch its corresponding N-dimensional semantic vector from the semantic representation model.
2 Representation(DK) <– Compose N-dimensional vector for the complete DK by combining word level vectors to a phrase level vector using linear average technique
3 Representation(Label) <– Fetch N-dimensional vector for the label from the semantic representation model
4 DK Correlation Score = Cosine Distance between Representation(DK) and Representation(Label)
5 **return** *DK Correlation Score*

3.2 Stage 2: Domain Knowledge based Disambiguation and Reordering

In this section, we propose to improve the Top-1 accuracy through intelligent reordering of the Top-5 labels from the vision classifier. In-order to achieve this, we leverage the domain knowledge associated with the teaching activity as a second source of information to re-order the Top-5 output labels. Domain Knowledge refers to the classroom learning context (derived from teacher's current syllabus, teaching themes, object related clues, collaborative clues) based on which the learning activity is conducted. Note that the Domain Knowledge could be a *word* or a *phrase* too. We now discuss various important aspects of this stage in detail.

Enabling Semantic Capability. Domain Knowledge is a text representation of the intent or activity derived from the classroom context. However, same intent or information could be conveyed through different keywords, and hence traditional bag-of-word approaches [23] will not solve the problem in our use-cases. We leverage the support of semantic representations (i.e. distributed word representation [16]) of words for enabling keyword independent re-ranking algorithm. In distributed word representation, words are represented as N-dimensional vectors such that distance between them capture semantic information. There are various pre-trained semantic representation models (also called word embedding models such as Word2Vec [11], GloVe [6]) available which enable semantic comparison of words. Likewise, there is also ConceptNet [5] which is a multilingual knowledge base, representing words and phrases that people use and the common-sense relationships between them. This paper leverages these existing works to achieve an effective re-ranking of the output label-set with semantic capability.

Existing Approach Results. One naive way to approach the problem of re-ranking is to find the DK Correlation Score (DK-CS) using Algorithm. 1 and re-rank the Top-5 labels in descending order of their DK-CS. However, this approach has strong bias towards the semantic representation output and completely ignores the ranking that is produced by the vision classifier.

Other fusion approaches that have been tried are combining one or more of the classifier outputs (i) Word2Vec (S1), (ii) GloVe (S2), (iii) ConceptNet (S3), (iv) Vision (S4) in different ways. The most common are the product rule and the weighted average rule where the confidence scores are combined by computing either a product of them or a weighted sum of them. The improvement in Top-1 accuracy of such combinations varies from -11% to 6%. We observe that the Top-1 accuracy of the system did not increase significantly and in many cases Top-1 accuracy of the system dropped after re-ranking as compared to the original list. The reason being the need for proper and more efficient resolution of conflicts between DK-CS wins vs. vision confidence score wins. In the next section, we explain the proposed novel re-ranking algorithm which highly improves the Top-1 accuracy of the system by effectively resolving the conflicts between DK-CS and vision rankings.

Proposed Re-Ranking Approach. In our proposed approach, we fuse the inferences from various semantic models and vision model using *Majority-Win Strong Vouching algorithm* for re-ordering the Top-5 output list. There are two important aspects of this approach: (i) Strong Vouching of Semantic Models, (ii) Majority Voting across Semantic Models.

Strong Vouching of Semantic Models: As discussed earlier, the reason for failure of the traditional fusion approaches is the need for efficient resolution of conflicts between the semantic model ranks and the vision model ranks. Let us understand this problem through 2 example scenarios. (i) Scenario 1: Top-1 prediction is "orange", Top-2 prediction is "apple", domain Knowledge is "fruits"; (ii) Scenario 2: Top-1 prediction is "orange", Top-2 prediction is "apple", domain knowledge is "red fruits". In the first scenario, since the domain knowledge is semantically correlated towards both Top-1 and Top-2 predicted labels, system should maintain the same order as predicted by the vision model. However, in the second scenario, since the domain knowledge (i.e. "red fruits") is highly correlated towards Top-2 (i.e. "apple")as compared to Top-1(i.e. "orange"), system should swap the order of Top-1 and Top-2 labels. It turns out that just having a higher DK-CS to swap the labels is not enough. We show that DK-CS of one label (label-1) should override the other label (label-2) by a specific threshold value to indicate that label-1 is semantically more correlated with as compared to label-2 and hence effect a swap against the vision rank. Through empirical analysis in Section. 4.2, we show that, in the context of reordering Top-K labels, if normalized DK-CS of a label is greater than the other label by a value equal to $1/k$ (threshold value), then the former label is more semantically correlated with domain knowledge as compared to the latter.

Majority Voting across Semantic Models: As mentioned before, many semantic models exist in the literature and each of them are trained on various data-sets. Therefore, it is not necessary that the strong vouching behavior of all these semantic models to be same. In order to resolve this, our approach considers multiple semantic models together (such as GloVe, Word2Vec and ConceptNet) and enables swapping of i-th label with j-th label (i<j) in the Top-K output list only when majority of semantic models are strongly vouching that j-th label is more correlated with DK as compared to the i-th label. This makes the system more intelligent in resolving across

semantic models as well as resolving conflicts across DK correlation score wins vs. vision confidence score wins. Algorithm. 2 explains the overall flow of the proposed re-ranking algorithm.

Algorithm 2: Fusion based on Majority Win Strong Vouching Concept

Input: Top-K output label from image recognition model, Domain Knowledge(DK)
Output: Reordered Top-K output label list

```
Sort Top-K labels based on vision confidence score
Re-rank the Top-K label by sorting using the following compare logic
Compare logic (i-th label, j-th label, DK): begin
    [Note] i-th label precedes j-th label in the ranked Top-K list.
    X1 = Total number of semantic models strongly-vouching for j-th label as compared to i-th label
    X2 = Total number of semantic models strongly-vouching for i-th label as compared to j-th label
    if X1>X2 then
        swap i-th label and j-th label in the Ranked Top-K list
    else
        Maintain the same order of i-th label and j-th label
return Re-Ranked Top-K List
```

3.3 Stage 3: Human Disambiguation Game

It is important to note that, due to limitation of existing state-of-art vision models, though we achieve effective improvements, we never reach an accuracy of 100%. Even after effective custom training and DK based Top-K re-ranking, accuracy of the system is not 100% (though high improvements are observed). So, there has to be a confirmation step involving human-in-loop to confirm whether the predicted label is the right label to prevent teaching wrong objectives. Since we are dealing with Kids, this step has to be extremely light, simple, and also engaging for the Kids so that, they do not feel any extra cognitive load. In this section, we propose a simple disambiguation game which is designed in a way that, (i) Kids easily play with it correctly, (ii) Kids interaction with the game highly reduces when Top-1 accuracy of the system is high. Through enhancements as explained in previous sections, we make vision model to reach high Top-1 accuracy which in-turn reduces the Kids interactions in the disambiguation game, thereby reducing the overall cognitive overload to the Kids.

Our system leverages image matching for the disambiguation game. Re-ranked Top-K list (which is the output from Stage 2) is fed as input to the disambiguation game. This game is depicted in Figure. 2 renders reference images of the label (with possible variants of a same object) one by one in the order of the re-ranked list and asks the Kid to select the image, if it looks similar to the object clicked (through camera). If not, system show the next reference image and continues till all K labels are rendered. Since the input to the game is a re-ranked Top-K list (which has high Top-1 accuracy), Kid has high chances of encountering the right image in the first or second step itself, thus reducing the cognitive load of the kid to traverse till the end. Usability Guidelines [10] [1] for Child based Apps suggest large on-screen elements which are well spatially separated for Kids to easily interact with them. So, based on the display size of the form-factor, system could configure the no of images to be rendered in one step/cycle. Through usability study with 15 Kids, we show that Kids are able to easily play image similarity based disambiguation games. In scenarios when the right label is not in the predicted Top-K labels, system executes the exit scenarios as configured. Few possible exit scenarios could be: (i) Continue the game with other labels in the learning vocabulary set in the sorted order of DK, (ii) Request for teacher intervention, etc.

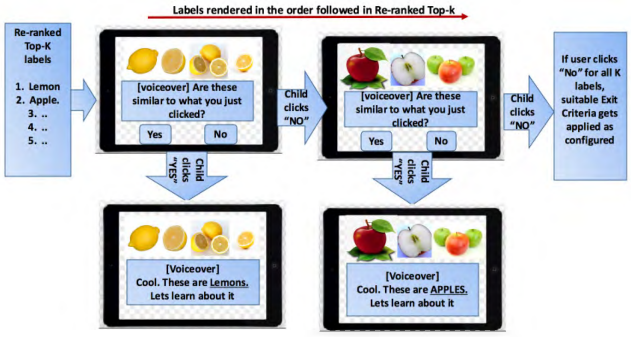


Figure 2: Basic Disambiguation Game

4. EVALUATION

We present here the experimental setup and results of improvement in the vision classifier results achieved by the re-ordering approach. We then explain and present the results of the empirical analysis to determine the value of threshold for strong vouching of the semantic models. To show that our approach is independent of domain knowledge, test set, training class set, and baseline image classification models (generality of approach), we performed various experiments as explained in following subsections. Later in this section, we present the usability study and inferences from the study conducted with a group of 12 children in the age group of 3-5 years.

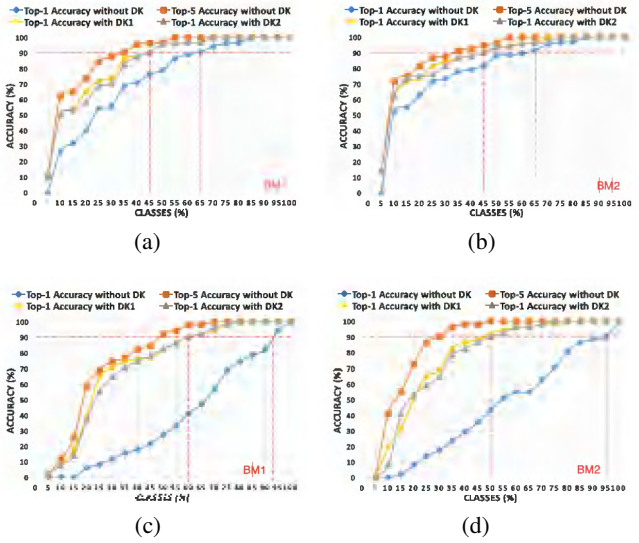
Datasets: The training dataset includes images from Imagenet [8]. We used 52 classes and approximately 400 images per class for training. These 52 selected classes are objects commonly used in early childhood learning, for example, apple, car, book, and violin, etc. The test datasets include real images taken from mobile phones and tablets. The test dataset I includes 1K images where single object (from training set) is present in an image frame. The test dataset II includes 2.6K images where two objects (from training set) are present in an image frame. All the experiments were performed using two baseline image classification models: (i) Baseline Model 1 (BM1): Model based on VGGNet architecture [18], (ii) Baseline Model 2 (BM2): Model based on Inception-V3 architecture [19].

Domain Knowledge: During all the experiments, we used two different domain knowledge (DK): Domain Knowledge 1 (DK1), which is the google dictionary definition [7] of each object class; Domain Knowledge 2 (DK2), which is the merged description of each object class collected from three different annotators (crowdsourced approach). By this way, we make sure that the domain knowledge is not keyword dependent and re-ordering happens at semantic level rather than at any specific keyword matching level.

Evaluation Metrics: In order to illustrate the performance of the proposed approach, evaluation parameters such as Top-1 accuracy, Top-5 accuracy, and improvements in Top-1 accuracy are used. The Top-1 accuracy is computed as the proportion of images such that the ground-truth label is the Top-1 predicted label. Similarly, the Top-5 accuracy is computed as the proportion of images such that the ground truth label is one of the Top-5 predicted labels.

4.1 Experimental Results

The cumulative accuracy distribution of Baseline Model 1 (BM1) and Baseline Model 2 (BM2) on test dataset I and II is shown in Figure. 3. Figures 3(a), 3(b) shows the improvement in the Top-1 accuracy after re-ordering on dataset I which has one object in an image frame. As shown in Figure. 3, for BM1, without re-ordering only 35% of object classes have Top-1 accuracy more than 90%, whereas with re-ordering using DK1 or DK2 around 55% of classes



(a) (b)

(c) (d)

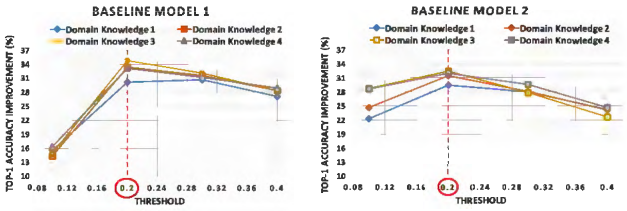
Figure 3: Cumulative accuracy distribution of Baseline Model 1 & Baseline Model 2 on the data set. I(a-b), II(c-d)

.

have more than 90% Top-1 accuracy. Similarly, for BM2 our approach shows 20% improvement in number of classes for 90% or above Top-1 accuracy on dataset I as shown in Figure 3(b).
When a child takes an image, it is common that multiple objects get captured in that image. If more than one object is present in an image, then the confusion of the classifier highly increases which leads to low Top-1 accuracy. Figure. 3(c), 3(d) show the improvement in Top-1 accuracy on data set II, where two objects (from training set) are present in an image frame. As shown in Figure. 3(c), for BM1, without re-ordering only 7% of object classes have Top-1 accuracy more than 90% whereas with re-ordering using DK1 or DK2 around 40% of classes have more than 90% Top-1 accuracy. Similarly, for BM2, our approach shows improvement of 45% in number of classes for 90% or more Top-1 accuracy on dataset II as shown in Figure. 3(d).

4.2 Empirical analysis to determine threshold for strong vouching of semantic models

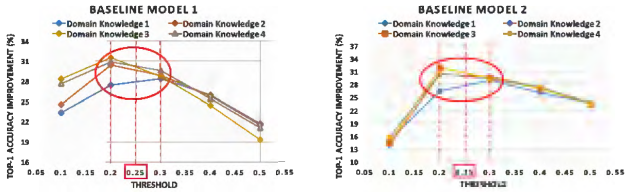
In this section, we explain the empirical analysis which determines the threshold value required by semantic models for strong vouching as discussed in Section. 3.2. In comparing two elements with respect to their semantic correlation with domain knowledge (i.e. DK-CS), the threshold stands for the minimum value by which DK-CS of one element should be higher than the other to confidently say that the element is semantically more correlated with the domain knowledge as compared to the other element. Choice of correct threshold value is very cruicial for the proposed approach. *The threshold value should be as high so as to avoid wrong swapping of labels, and as low to allow correct swapping of labels for better Top-1 accuracy improvements.*
For the empirical analysis of threshold value, we conducted experiments on dataset II with the following combinations (i) four different domain knowledges collected through crowd sourcing, (ii) four different threshold values, and (iii) for both baseline models (BM1&BM2) to make it independent of any local data-behavior. The results are shown in Figure. 4. From the results, we noticed that the correct threshold value is 0.2 for reordering Top-5 predicted labels. As observed in Figure. 4, Top-1 accuracy reaches the peak value when the threshold value is 0.2. We now discuss the reason behind this magical number.



(a) (b)

Figure 4: Improvement in Top-1 accuracy while reordering predicted Top-5 labels for different domain knowledge, threshold values and baseline models

In our approach, we use normalized DK-CS, which means if we consider equal distribution of labels while reordering Top-5 predicted labels, then the DK-CS for each label is 0.2 (i.e. 1/5). We propose that, if DK-CS of one label overrides the semantic score of another label by a value near or equal to the $1/k$ (i.e. individual DK-CS of the labels considering equal distribution of each label), then it is considered as **strong vouching** by semantic model for the former label.
In order to confirm the above proposed claim, we performed ex-



(a) (b)

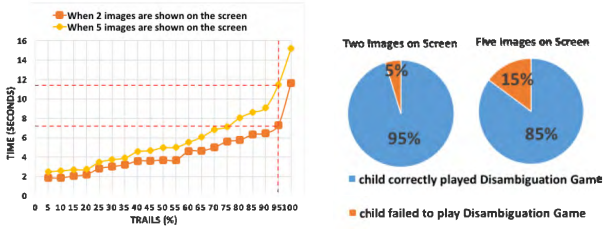
Figure 5: Improvement in Top-1 accuracy while reordering predicted Top-4 labels for different Domain Knowledge, Threshold Values and Baseline Models

periments to reorder Top-4 predicted labels (results are shown in Figure. 5). From the results, we can see that the performance is at peak for threshold between values 0.2 and 0.3, which is near to 0.25 (1/k where k is number 4). There is very noticeable degradation in performance when threshold is below 0.2 or above 0.3. Similar trends were also observed when experimenting with Top-3 re-ordering.
Therefore, the correct choice of threshold while re-ordering Top-k predicted labels is $1/k$. When system is tuned to vouch strongly using this threshold value, we observe high improvements in Top-1 accuracy.

4.3 Usability Study

The main purpose of this usability study is to observe the following key points in children of ages between 3-5 years: (i) whether they can take images using the camera of a phone or tablet, (ii) whether they can perform visual comparison between the physical object for which picture was taken, and its reference image provided by the classifier in the disambiguation game, (iii) comparison of cognitive load on children when they see less vs. more number of images on a device screen during the game. To conduct this study, we asked the child to play with our app installed on iPads, which logged the complete click stream data of the app for tracking various quantitative parameters. We also noted down the feedback from parents/observer during the activity play.

We conducted this usability study on 12 children with a total of 29 trials. In each trail, a child was allowed to play with the app as long



(a) (b)

Figure 6: (a) Cumulative distribution of time taken by children to play disambiguation game when two and five objects are shown on the device screen (b) % of times children correctly played disambiguation game when two and five objects are shown on the device screen

s(he) wanted. We observed that some children played only one time in a trial and some played upto 8 times in a trail. There were no limitations on the number of trails per child. We did not observe even a single instance where a child was asked to capture an image of a relevant object using the camera and s(he) failed to do it. This shows that children of that age group can easily take pictures. The average time taken by a child to search for an object in the environment and take picture was 20 seconds. From the collected data, we observed that around 90% of times, children are able play the disambiguation game correctly. The common feedback which we got from parents/observers is that children liked this app and wanted to play it again and again.

The comparison of cognitive load on a child when (s)he got 2 object images (from Top-2) vs. 5 object images (from Top-5) on a screen during disambiguation game is shown in Figure.6 (a). Around 95% of children took upto 7 and 11 seconds for disambiguation game when they got 2 and 5 options on a screen, respectively (as shown in figure 6a). Similarly, Figure. 6 (b) shows that on an average a child failed to make a visual comparison only 5% of time (when there were 2 images on a screen) and 15% of time (when there were 5 images on a screen). These results indicate that, child is able to easily play the disambiguation game but the cognitive load reduces when less number of images were rendered in each turn of the game. Since the proposed re-ranking algorithm increases the Top-1 accuracy of the system, the child could reach the right object in initial rounds of the disambiguation game with high chance, thereby providing a good user experience.

5. CONCLUSION

We present a holistic visual recognition system for Early Childhood Learning through physical-digital interaction that improves recognition accuracy levels using (a) domain restriction and custom training of vision classifiers, (b) a novel re-ranking algorithm for multi-modal fusion of contextual information, and (c) semi-automated feedback with different gaming scenarios for right object-tag identification & classifier re-training. Through a usability study with 12 children, we make the case that engaging user experiences can indeed be developed to bridge the gap between automatic visual recognition accuracies and the requirement of high accuracy for meaningful learning activities.

Extensive evaluations on large datasets brought forth the deficiency of existing multimodal fusion techniques in combining the domain knowledge context with the vision classification results. Using a data driven approach we show the efficacy of our proposed re-ranking algorithm based on strong vouching, and also show that the swapping threshold (derived from data) is also anchored in a physical meaning.

For future work we would like to conduct extensive pilot study with children to demonstrate evidence-of-learning for vocabulary acquisition using physical-digital interaction. We would also like to use other implicit contexts such as location, speech cues, wearable sensors etc. to derive domain knowledge for better multimodal disambiguation.

6. REFERENCES

[1] Usability guidelines for kids. *http://rosenfeldmedia.com/wp-content/uploads/2014/11/DesignforKids-excerpt.pdf*.

[2] Alien assignment app. *http://my.kindertown.com/apps/alien-assignment*, 2017.

[3] Aurasma. *https://www.aurasma.com/*, 2017.

[4] Blippar. *https://blippar.com/en/*, 2017.

[5] Conceptnet. *https://github.com/commonsense/conceptnet5/wiki*, 2017.

[6] Glove. *http://nlp.stanford.edu/projects/glove/*, 2017.

[7] Google dictionary. *http://www.dictionary.com/browse/google*, 2017.

[8] Imagenet. *http://www.image-net.org/*, 2017.

[9] Layar - augmented reality. *http://appcrawlr.com/ios/layar-reality-browser-augmented*, 2017.

[10] Usability guidelines for kids. *http://hci.usask.ca/publications/2005/HCI_TR_2005_02_-Design.pdf*, 2017.

[11] Word2vec. *https://github.com/dav/word2vec*, 2017.

[12] A. N. Bhute and B. Meshram. Text based approach for indexing and retrieval of image and video: A review. *arXiv preprint arXiv:1404.1514*, 2014.

[13] B. Hart et al. The early catastrophe: The 30 million word gap by age 3. *American educator*, 27(1):4–9, 2003.

[14] Y. Lin et al. Text-aided image classification: Using labeled text from web to help image classification. In *Web Conference (APWEB), 2010 12th International Asia-Pacific*, pages 267–273. IEEE, 2010.

[15] H. Ma et al. Bridging the semantic gap between image contents and tags. *IEEE Transactions on Multimedia*, 12(5):462–473, 2010.

[16] T. Mikolov et al. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[17] B. C. Roy et al. Predicting the birth of a spoken word. *Proceedings of the National Academy of Sciences*, 112(41):12663–12668, 2015.

[18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[19] C. Szegedy et al. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[20] L. Tian et al. Image classification based on the combination of text features and visual features. *International Journal of Intelligent Systems*, 28(3):242–256, 2013.

[21] G. Wang et al. Building text features for object image classification. In *Computer Vision and Pattern Recognition*, pages 1367–1374. IEEE, 2009.

[22] C. Xu et al. Fusion of text and image features: A new approach to image spam filtering. In *Practical Applications of Intelligent Systems*, pages 129–140. Springer, 2011.

[23] Y. Zhang et al. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1-4):43–52, 2010.

Student Learning Strategies and Behaviors to Predict Success in an Online Adaptive Mathematics Tutoring System

Jun Xie
University of Memphis
3720 Alumni Ave,
Memphis, TN 38152
(901)678-2000
Jxie2@memphis.edu

Shirin Mojarad
McGraw Hill Education
281 Summer Street,
Boston, MA 02210
(800)338-3987
shirin.mojarad@mheducation.com

Keith Shubeck
University of Memphis
3720 Alumni Ave,
Memphis, TN 38152
(901)678-2000
kshubeck@memphis.edu

Alfred Essa
McGraw Hill Education
281 Summer Street,
Boston, MA 02210
(800)338-3987
alfred.essa@mheducation.com

Ryan S. Baker
University of Pennsylvania
3451 Walnut Street,
Philadelphia, PA 19104-6291
(215)573-2990
rybaker@upenn.edu

Xiangen Hu
University of Memphis
3720 Alumni Ave,
Memphis, TN 38152
(901)678-5736
xhu@memphis.edu

ABSTRACT

Student learning strategies play a critical role in their overall success. The central goal of this study is to investigate how learning strategies are related to student success in an online adaptive mathematics tutoring system. To accomplish this goal, we developed a model to predict student performance based on their strategies in ALEKS, an online learning environment. We have identified student learning strategies and behaviors in seven main categories: help-seeking, multiple consecutive errors, learning from errors, switching to a new topic, topic mastery, reviewing previous mastered topics, and changes in behavior over time. The model, developed by using stepwise logistic regression, indicated that requesting two consecutive explanations, making consecutive errors and requesting an explanation, and changes in learning behaviors over time, were associated with lower success rates in the semester-end assessment. By contrast, the reviewing previous mastered topics strategy was a positive predictor of success in the last assessment. The results showed that the predictive model was able to predict students' success with reasonably high accuracy.

Keywords

Help-seeking, errors, learning strategy, math, student success, adaptive tutoring system

1. INTRODUCTION

Computer-based learning environments, particularly intelligent tutoring systems (ITS), are becoming more commonly used to assist students in their acquisition of knowledge. Computer-based tutors provide tailored instruction and one-to-one tutoring, which can improve students' learning experiences and their motivation. These learning systems also provide unique and critical insight to learning science researchers by creating exhaustive archives of student learning behaviors. A central goal of investigating student learning processes is to unveil the associations between learning behaviors and performance, ultimately allowing learning system developers and researchers to predict and understand student performance. This knowledge allows for evidence-based and individually tailored feedback to be provided to students who are struggling to learn.

2. RELATED WORK

Many studies have investigated the relationships between learning behaviors and success in learning [1, 2]. The most frequent learning behaviors used in the current literature involve help-seeking, making errors, persistence, and changes in learning behaviors over time [3, 4, 5]. For example, worked examples, an effective and commonly used type of help, can be overused by students, negatively affecting learning [6]. However, asking for help after making an error has been found to be an effective help-seeking strategy, particularly for high prior knowledge students [7]. Additionally, reading a worked example after solving a problem can foster better learning than practice alone and reading a worked example before solving a problem can improve learning when compared to reading a worked example after solving a problem [8, 9].

Clearly, there is a delicate interplay between help-seeking strategies students use, their prior knowledge, and learning success. Whether students benefit from making errors often depends on how errors are approached pedagogically. Errors, when treated as stemming from student inadequacies, can trigger math anxiety, which negatively affects students' learning [10, 11]. An extreme example of making errors during learning is seen in wheel-spinning behaviors, in which students attempt ten problems or more without mastering the topic. While too many consecutive errors (i.e. wheel-spinning) undermine learning performance [12], repeated failure in the low-skill phase has been found to improve the likelihood of success in the next step [5] and to lead to more robust learning [13]. Furthermore, the errors that naturally occur from desirable difficulty are considered to be an essential element in learning [14] and facilitate long-term knowledge retention and transfer [15, 16, 17].

Many of the current computer-based tutoring systems are designed to provide students more autonomy by allowing them to learn at their own pace. In self-paced or self-regulated tutoring systems, students' learning behaviors tend to change over time during learning. These changes in learning behaviors over time represent an important aspect of learning for researchers to understand. Relatively more well-structured behavior over time is positively related to reading performance, whereas more chaotic, less-structured learning behaviors are related to poor reading performance [4].

Persistence is another increasingly studied behavior in learning research. For example, persistence is measured as time spent on unsolved problems during solving anagrams and riddles [18]. Persistence on challenging tasks is associated with mastery goals, which benefit learning [19]. Given these definitions of persistence, a contrasting learning behavior could be considered frequently switching topics within a learning system to find easier topics, an example of gaming the system [20]. Based on students' self-reports, persistence was also found to positively related to student satisfaction with the computer-based tutoring system [21]. However, unproductive persistence (i.e. wheel-spinning) impedes learning, but various formats of problems and spaced practice can reduce unproductive persistence and improve learning [22].

Reviewing previous learned materials is an efficient way to improve learning. Per Ebbinghuas' forgetting curve [23], memory retention declines over time. Repeated exposure to previously learned materials can enhance memory retention and improve learning [24]. An example of reviewing previously learned materials is seen in the retrieval practice, which was found to improve students' memory retention of reading materials [25] as well accuracy in solving "student-and-professor" algebra word problems [26].

This study aims to investigate which learning behaviors predict student success in ALEKS (Assessment and Learning Knowledge Spaces), a math tutoring system that adapts to students' knowledge [27]. Given the above literature, help-seeking behaviors, multiple consecutive errors, learning from errors, temporal behavioral changes, persistence (i.e. switch to a new topic without mastering the current topic), and reviewing previous mastered topics were selected as potential predictors of success in ALEKS. In addition, the percentage of topics that have been mastered, an indicator of learning progress, is included in the model to predict success.

3. Description of ALEKS

ALEKS is a web-based artificially intelligent learning and assessment system [27]. Its artificial intelligence is based on a theoretical framework called Knowledge Space Theory (KST) [28]. KST allows domains to be represented as a knowledge map consisting of many knowledge states, which represent the prerequisite relationships between different knowledge states (KS). Therefore, KST allows for a precise description of a student's current knowledge state, and what a student is ready to learn next. ALEKS can estimate a student's initial KS by conducting a diagnostic assessment (based on a test) when the student first begins to interact with the system. ALEKS conducts assessments during students' progress through the course to update their knowledge states and to decide what the student is ready to learn next.

For each topic within ALEKS, a problem is randomly generated, with adjustments made to several parameters for each problem type. This results in an enormous set of unique problems. Students are required to provide solutions in the form of free-response answers, rather than by selecting an answer from multiple choices. Explanations in the form of worked examples can be requested by students at any time. When an explanation is requested, a worked example for the current problem is provided and a new problem is provided to the student. The interface of ALEKS is displayed in Figure 1.

ALEKS is self-paced; students can choose topics to learn and can choose when they want to request help. All the topics that the student is most ready to learn (per the KST model) are displayed in his or her knowledge pie (Figure 2). The knowledge pie presents the student's learning progress in each math subdomain as well.

Research has shown ALEKS produces learning outcomes comparable with other effective tutoring systems for teaching Algebra [29]. Using ALEKS as an after-school program has also been observed to be as effective as interacting with expert teachers [32]. Students need less assistance during learning when using ALEKS than in traditional curricula [31]. Additionally, ALEKS has been found to reduce the math performance discrepancies between ethnicities in an after-school program [32].

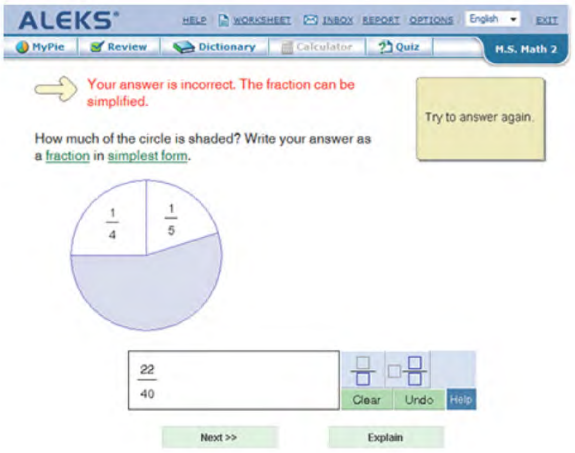


Figure 1. The ALEKS interface

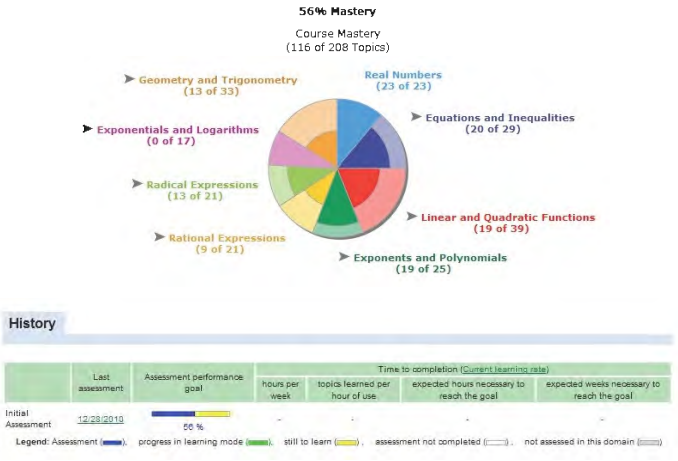


Figure 2. The ALEKS knowledge pie

4. Data

The data used in this study was collected from 179 students within 11 college classes that used ALEKS for developmental mathematics in Fall 2016. The data is comprised of information about students' learning actions and assessment scores. These actions include "correct" (C), "wrong" (W), mastering a topic (S; three C's in a row within a single topic), failing a topic (F; five W's in a row within a single topic) and explanations (E; requesting an explanation). The data also contains students' last assessment scores in ALEKS which account for students' performance in ALEKS.

5. METHODS

We employed stepwise logistic regression with backward elimination to predict students' success in ALEKS, using a training-test split. More details of this process are described below.

5.1 Student success

Success in ALEKS is defined as students knowing 60% or more of the topics in their last assessment. Therefore, we adopted 60% in the semester-end assessment as a cut-off value for success. Students whose last assessment score was 60% or greater were grouped as "successful students", whereas those with last assessment scores under 60% was grouped as "unsuccessful students". The dataset was randomly split into two parts: 60% of students' data were used to train the model (N=107), and 40% were used to test the model's generalizability (N= 72). Success was labeled as 1 and failure was labeled as 0 in the prediction model.

5.2 The features to predict success

The following behavior patterns were used to predict student success: (1) help-seeking i.e., requesting an explanation after making an error (WE), and requesting two sequential explanations (EE); (2) multiple consecutive errors i.e., making two sequential errors (WW), making an error again after an error and requesting an explanation (WEW), making an error again after an error and requesting two explanations (WEEW), and the overall percentage of failure labeled by ALEKS (PF); (3) learning from errors i.e., providing a correct answer after making an error (WC), providing a correct answer after making an error and requesting an explanation (WEC), and providing a correct answer after making an error and requesting two explanations (WEEC); (4) switching to a new topic i.e., switching to a new topic after making an error or requesting an explanation (PNew), and switching to a new topic because of failure on a topic (FNew); (5) topic mastery (PS), i.e. providing three correct responses in a row; (6) reviewing previous mastered topics (PReview); and finally, (7) changes in learning behaviors over time (measured using the entropy metric).

The features of the first four aspects mentioned above were generated by using D'Mello's likelihood metric [33] (Equation 1).

The likelihood metric is used to compute the transition probability of an event to another event. In the case of multiple events, we calculate a proportion of each sequence out of the number of sequences of that length. For example, the probability of WEEW means the transition probability of WEE to W. In this case, WEE is represented as M_t and W is represented as M_{t+1} in the formula. When the value produced by the likelihood metric is higher than 0, it signifies that M_{t+1} occurs after M_t more frequently than the base rate of M_{t+1}. Otherwise, M_{t+1} occurs after M_t at a rate lower or equal than the base rate of M_{t+1}.

$$L(M_t \rightarrow M_{t-1}) = \frac{\Pr(M_{t+1}|M_t) - \Pr(M_{t+1})}{1 - \Pr(M_{t+1})} \quad (1)$$

Shannon entropy is used to compute the degree of regularity in the changes in students' learning behaviors over time (specifically focusing on the shifts between making an error, give a correct answer, and requesting an explanation) [34] (Equation 2). High entropy values represent disordered leaning behavior patterns. On the contrary, low entropy implies ordered pattern of learning behaviors:

$$H(x) = -\sum_{i=0}^{N} P(x_i)(\log_e P(x_i)) \quad (2)$$

The details on how the features were computed are listed below in table 1.

Table 1. Descriptions of features used to predict success

Features	Description
WE	The transition probability from making an error to requesting an explanation
EE	The transition probability from requesting an explanation to requesting another explanation
WW	The transition probability from making an error to making an error again
WEW	The transition probability from making an error and requesting an explanation to making an error again
WEEW	The transition probability from making an error and requesting two sequential explanations to making an error again
PF	The proportion of times a student made five consecutive errors
WC	The transition probability from making an error to giving a correct answer
WEC	The transition probability from making an error and requesting an explanation to giving a correct answer
WEEC	The transition probability from making an error and requesting two sequential explanations to giving a correct answer
PNew	The probability of starting a new topic after making an error or requesting an explanation on the current topic
FNew	The probability of starting a new topic after failing a topic
PS	The proportion of the mastered topics out of the number of the attempted topics during learning
PReview	The percentage of mastered topics that the student reviews after mastering them
Entropy	The entropy value produced based on students' learning behaviors

6. RESULTS

6.1 Description of features

Before building the prediction model, we calculated basic descriptive statistics. The mean and standard deviations are listed in Table 2.

Table 2. Feature means and standard deviations

Features	M	S.D.
WE	.40	.11
EE	-.07	.07
WW	-.02	.08
WEW	.15	.09
WEEW	.06	.25
PF	.07	.07
WC	-.69	.37
WEC	-.07	.18
WEEC	-.22	.46
PNew	.001	.01
FNew	.76	.33
PS	.87	.10
PReview	.14	.11
Entropy	.51	.11

6.2 Model development

Stepwise logistic regression with backward elimination was used to generate the predictive model of students' success. The final model included requesting an explanation after making an error (WE), requesting two sequential explanations (EE), making an error again after making an error and requesting an explanation (WEW), changes in learning behaviors over time (entropy) and review on the topic (PReview). Each of these metrics were statistically significant predictors of students' success (i.e. the score in the last assessment is greater or less than 60%) in ALEKS. The details on the prediction model are displayed in Table 3.

Table 3. The results of multi-feature logistic regression on students' success

	B	S.E.	Z value	p
Intercept	3.32	1.63	2.04	.04*
WE	4.25	2.31	1.84	.07
EE	-8.31	4.05	-2.06	.04*
WEW	-11.33	3.40	-3.33	.00***
Entropy	-10.34	2.91	-3.55	.00***
PReview	9.44	2.57	3.67	.00***

Note. p<.000 ***, p<.05 *

The results of multicollinearity indicated that there were low correlations between features. The VIF value (i.e. variance inflation factor) for each feature is illustrated in Table 4.

Furthermore, logistic regressions that only include one single feature were conducted to examine suppression effect. The results were listed in Table 5. The results showed that compared to the results of multi-feature logistic regression, the direction of relationship between each feature and success did not change in the single-feature logistic regression. Therefore, the relationship between features and success was not impacted by suppression effect.

Then, based on the results of logistic regressions, students were less likely to be successful in the last assessment if they tend to read two consecutive explanations, or made an error after making an error and requesting an explanation, or demonstrated irregularity in their learning behaviors. By contrast, the more frequently students reviewed topics they have already mastered, the more likely they were to pass the last assessment in ALEKS.

Table 4. Multicollinearity between features in the prediction model

	WE	EE	WEW	entropy	PReview
VIF	1.02	1.32	1.17	1.66	1.50

Table 5. The summary of single-feature logistic regressions on students' success

	B	Z value
WE	4.40	2.32
EE	-0.61	-.23
WEW	-9.12	-3.44
Entropy	-5.01	-2.62
PReview	5.24	2.60

6.3 Model goodness

The fitness index of the prediction model (i.e. AIC) of training data was 115.67. McFadden pseudo r^2 of training data was .30, indicating that this model predicts a substantial amount of the variance in student success.

The model's accuracy of prediction on test data was 0.71. The AUC of test data (area under the ROC curve) was 0.77. The plot of the ROC curve is illustrated in Figure 3.

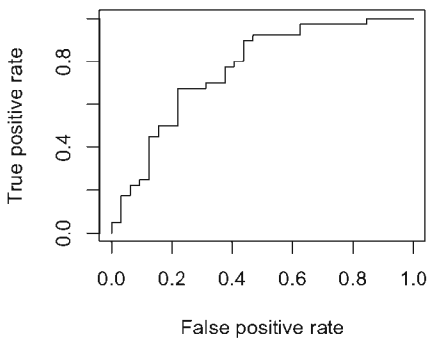


Figure 3. The ROC plot of the prediction model

7. DISCUSSIONS

The current study developed a logistic model to predict student overall success in ALEKS, as well as the relationship between various learning behaviors and success. Our findings contribute to the current understanding of the relationship between student learning behaviors and their delayed performance in adaptive tutoring systems, as well as provide evidence-based suggestions for improving the feedback and interventions in ALEKS.

Requesting two sequential explanations (EE) had a negative relationship with success in the last assessment, a finding in line with previous research on the negative effect of overusing help on learning [8]. However, the EE behaviors may suggest that students did not understand the first explanation rather than indicating that the students were "gaming the system". This can be concluded for the following reason. After requesting a worked-examples explanation, the student typically receives a new problem. Making an error again after making an error and requesting an explanation (WEW) was negatively related to students' success. The relationship between WEW and success suggests that students frequently make multiple consecutive errors, even after receiving the provided worked examples. These students may have trouble understanding the example. Therefore, if students frequently demonstrate those two behaviors on a specific problem, more individually-tailored and deeper-level instructions may be needed to provide the necessary help to overcome the impasse, such as concept-specific conversations with tutor agents that are integrated in ALEKS.

Another finding conforming to the previous research was that regular behaviors during learning is positively related to students' performance [cf. 5]. In this study, the measurement of changes of behaviors over time (via Shannon entropy) is relatively coarse-grained. Moving forward, deeper and finer-grained investigations of changes in behavior over time may shed further light on why regularity is associated with better outcomes.

Another finding worth noting was that the percentage of topics mastered (PS) during learning was not found to be a significant predictor of success on the last assessment. An explanation of this finding may lie in the adaptive design of ALEKS. During learning, ALEKS continually matches students' existing knowledge with topic difficulty and provides the topics that students are most ready to learn, so students focus their time on topics that have an appropriate level of difficulty [22]. Thus, the percentage of topics being mastered may not differ much between students who were successful in the last assessment and those who failed the last assessment. Finally, reviewing previously mastered topics (PReview) was found to be positively linked to students' success in the last assessment, which confirmed the findings of literature [24].

Our model was able to accurately predict student success. However, some improvements can be made in the future. The current model only includes percentages or probabilities of behaviors without considering the time spent on these behaviors. In the future, adding the time duration of behaviors may increase the prediction accuracy of the model. Additionally, refining the measurements of behaviors may increase the prediction accuracy of the model. For example, changes in learning behaviors over time could be measured during different learning phases or in specific temporal sequences.

By better understanding the factors associated with success in ALEKS, we can design interventions that will improve student success – the ultimate goal of any intelligent tutoring system.

8. ACKNOWLEDGMENTS

This paper is based on work supported by McGraw-Hill Education. We would like to extend our appreciation for all the informational support provided by the ALEKS Team. 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.

9. REFERENCES

[1] 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, 185-224. DOI= http://www.learntechlib.org/p/24328

[2] Aleven, V., Stahl, E., Schworm, S., Fischer, F., and Wallace, R. 2003. Help seeking and help design in interactive learning environments. *Review of Educational Research* 73, 3, 277-320. DOI= 10.3102/00346543073003277

[3] Baker, R. S., Corbett, A. T., and Koedinger, K. R. 2004. Detecting student misuse of intelligent tutoring systems. In *Proceedings of International Conference on Intelligent Tutoring Systems (*Maceió, Alagoas, Brazil, August 30 - September 3). 531-540. Springer Berlin Heidelberg. DOI= 10.1007/978-3-540-30139-4_50

[4] Snow, E. L., Jackson, G. T., and McNamara, D. S. 2014. Emergent behaviors in computer-based learning environments: Computational signals of catching up. *Computers in Human Behavior*, 41, 62-70. DOI= http://dx.doi.org/10.1016/j.chb.2014.09.011

[5] Roll, I., Baker, R. S. D., Aleven, V., and Koedinger, K. R. 2014. On the benefits of seeking (and avoiding) help in online problem-solving environments. *Journal of the Learning Sciences* 23, 4, 537-560. DOI= http://dx.doi.org/10.1080/10508406.2014.883977

[6] Kalyuga, S., Chandler, P., Tuovinen, J., and Sweller, J. 2001. When problem solving is superior to studying worked examples. *Journal of Educational Psychology* 93, 3, 579-588. DOI= http://dx.doi.org/10.1037/0022-0663.93.3.579

[7] Wood, H., and Wood, D. 1999. Help seeking, learning and contingent tutoring. *Computers & Education*, *33*(2), 153-169. DOI= http://dx.doi.org/10.1016/S0360-1315(99)00030-5

[8] Van Gog, T., and Kester, L. 2012. A test of the testing effect: acquiring problem-solving skills from worked examples. *Cognitive Science* 36, 8, 1532-1541.DOI = 10.1111/cogs.12002

[9] Van Gog, T., Kester, L., and Paas, F. 2011. Effects of worked examples, example-problem, and problem-example pairs on novices' learning. *Contemporary Educational Psychology* 36, 3, 212-218. DOI= http://dx.doi.org/10.1016/j.cedpsych.2010.10.004

[10] Ashcraft, M. H., and Kirk, E. P. 2001. The relationships among working memory, math anxiety, and performance. *Journal of Experimental Psychology: General*, 130, 2, 224-237. DOI=http://dx.doi.org/10.1037/0096-3445.130.2.224

[11] Moser, J. S., Moran, T. P., Schroder, H. S., Donnellan, M. B., and Yeung, N. 2013. On the relationship between anxiety and error monitoring: a meta-analysis and conceptual framework. *Frontiers in Human Neuroscience*,7,1-19. DOI= 10.3389/fnhum.2013.00466

[12] Beck J., and Rodrigo M.M.T. 2014. Understanding Wheel Spinning in the Context of Affective Factors. In *Proceedings of 12th Intelligent Tutoring System* (Honolulu, Hawaii, USA, June 5- 9, 2014). Lecture Notes in Computer Science, 162-167. Springer, Cham. DOI= 10.1007/978-3-319-07221-0_20

[13] Baker, R.S.J.d., Gowda, S., and Corbett, A.T. 2011. Towards predicting future transfer of learning. In *Proceedings of 15th International Conference on Artificial Intelligence in Education* (Canterbury, New Zealand, June 27- July 1, 2011). 23-30. DOI= 10.1007/978-3-642-21869-9_6

[14] Bjork, R. A., Dunlosky, J., and Kornell, N. 2013. Self-regulated learning: Beliefs, techniques, and illusions. *Annual Review of Psychology*, *64*, 417-444. DOI= 10.1146/annurev-psych-113011-143823

[15] Lee, T. D. 2012. Contextual Interference: Generalizability and limitations. *In Skill Acquisition in Sport: Research, Theory, and Practice II*. 79-93. Routledge, London.

[16] Simon, D. A., and Bjork, R. A. 2001. Metacognition in motor learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 27, 4, 907-912. DOI= http://dx.doi.org/10.1037/0278-7393.27.4.907

[17] Taylor, K., and Rohrer, D. 2010. The effects of interleaved practice. *Applied Cognitive Psychology* 24, 6, 837-848. DOI= 10.1002/acp.1598

[18] Ventura, M., Shute, V., and Zhao, W. 2013. The relationship between video game use and a performance-based measure of persistence. *Computers & Education* 60, 1, 52-58. DOI= http://dx.doi.org.ezproxy.memphis.edu/10.1016/j.compedu.2012.07.003

[19] American Psychological Association, Coalition for Psychology in Schools and Education. 2015. *Top 20 principles from psychology for preK–12 teaching and learning*.

[20] Kai, S., Almeda, M. V., Baker, R. S., Shechtman, N., Heffernan, C., and Heffernan, N. 2017. Modeling wheel-spinning and productive persistence in skill builders. *Journal of Educational Data Mining* (in press).

[21] Levy, Y. 2007. Comparing dropouts and persistence in e-learning courses. *Computers & education* 48, 2, 185-204. DOI= http://dx.doi.org/10.1016/j.compedu.2004.12.004

[22] Baker, R.S.J.d., Mitrovic, A., and Mathews, M. 2010. Detecting Gaming the System in Constraint-Based Tutors. In *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization* (Big Island, HI, USA, June 20-24, 2010), 267-278. DOI= 10.1007/978-3-642-13470-8_25

[23] Averell, L., and Heathcote, A. 2011. The form of the forgetting curve and the fate of memories. *Journal of Mathematical Psychology* 55, 1, 25-35. DOI=http://dx.doi.org/10.1016/j.jmp.2010.08.009

[24] Rohrer, D. 2015. Student instruction should be distributed over long time periods. *Educational Psychology Review* 27, 4, 635-643. DOI=10.1007/s10648-015-9332-4

[25] Roediger III, H. L., and Karpicke, J. D. 2006. Test-enhanced learning: Taking memory tests improves long-term retention. *Psychological Science 17,* 3, 249-255. DOI= 10.1111/j.1467-9280.2006.01693.x

[26] Christianson, K., Mestre, J. P., and Luke, S. G. 2012. Practice makes (nearly) perfect: Solving 'students-and-professors'-type algebra word problems. *Applied Cognitive Psychology* 26, 5, 810-822. DOI= 10.1002/acp.2863

[27] https://www.aleks.com/about_aleks

[28] Falmagne JCJ-C, Thiéry N, and Cosyn E, et al 2006. The Assessment of Knowledge, in Theory and in Practice. *Form Concept Anal* 3874, 61–79. DOI= 10.1109/KIMAS.2003.1245109

[29] 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. DOI= http://dx.doi.org/10.1016/j.chb.2013.03.001

[30] Craig SD, Hu X, and Graesser AC, et al 2013. The impact of a technology-based mathematics after-school program using ALEKS on student's knowledge and behaviors. *Comput Educ* 68, 495–504. DOI= 10.1016/j.compedu.2013.06.010

[31] Hu, X., et. 2012. The effects of a traditional and technology-based after-school program on 6th grade student's mathematics skills. *Journal of Computers in Mathematics and Science Teaching* 31, 1, 17-38. DOI= http://www.editlib.org/p/38628/

[32] 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. DOI= http://dx.doi.org/10.1016/j.lindif.2016.01.012

[33] D'Mello, S. and Graesser, A. 2012. Dynamics of affective states during complex learning. *Learning and Instruction* 22, 2, 145-157. DOI= http://dx.doi.org/10.1016/j.learninstruc.2011.10.001.

[34] Shannon, C. E. 1951. Prediction and entropy of printed English. *Bell System Technical Journal* 30, 1, 50–64. DOI=10.1002/j.1538-7305.1951.tb01366.x.

Adaptive Assessment Experiment in a HarvardX MOOC

Ilia Rushkin
Harvard University
Cambridge, USA
ilia_rushkin@harvard.edu

Yigal Rosen
Harvard University
Cambridge, USA
yigal_rosen@harvard.edu

Andrew Ang
Harvard University
Cambridge, USA
andrew_ang@harvard.edu

Colin Fredericks
Harvard University
Cambridge, USA
colin_fredericks@harvard.edu

Dustin Tingley
Harvard University
Cambridge, USA
dtingley@gov.harvard.edu

Mary Jean Blink
TutorGen, Inc
Fort Thomas, USA
mjblink@tutorgen.com

Glenn Lopez
Harvard University
Cambridge, USA
glenn_lopez@harvard.edu

ABSTRACT

We report an experimental implementation of adaptive learning functionality in a self-paced HarvardX MOOC (massive open online course). In MOOCs there is need for evidence-based instructional designs that create the optimal conditions for learners, who come to the course with widely differing prior knowledge, skills and motivations. But users in such a course are free to explore the course materials in any order they deem fit and may drop out any time, and this makes it hard to predict the practical challenges of implementing adaptivity, as well as its effect, without experimentation. This study explored the technological feasibility and implications of adaptive functionality to course (re)design in the edX platform. Additionally, it aimed to establish the foundation for future study of adaptive functionality in MOOCs on learning outcomes, engagement and drop-out rates. Our preliminary findings suggest that the adaptivity of the kind we used leads to a higher efficiency of learning (without an adverse effect on learning outcomes, learners go through the course faster and attempt fewer problems, since the problems are served to them in a targeted way). Further research is needed to confirm these findings and explore additional possible effects.

Keywords

MOOCs; assessment; adaptive assessment; adaptive learning.

1. INTRODUCTION

Digital learning systems are considered adaptive when they can dynamically change the presentation of content to any user based on the user's individual record of interactions, as opposed to simply sending users into different versions of the course based on preexisting information such as user's demographic information, education level, or a test score. Conceptually, an adaptive learning system is a combination of two parts: an algorithm to dynamically assess each user's current profile (the current state of knowledge, but potentially also affective factors, such as frustration level), and, based on this, a recommendation engine to decide what the user should see next. In this way, the system seeks to optimize individual user experience, based on each user's prior actions, but also based on the actions of other users (e.g. to identify the course items that many others have found most useful in similar circumstances). Adaptive technologies build on decades of research in intelligent tutoring systems, psychometrics, cognitive learning theory and data science [1, 3, 4].

Harvard University partnered with TutorGen to explore the feasibility of adaptive learning and assessment technology implications of adaptive functionality to course (re)design in HarvardX, and examine the effects on learning outcomes, engagement and course drop-out rates. As the collaboration evolved, the following two strategic decisions were made: (1) Adaptivity would be limited to assessments in four out of 16 graded sub-sections of the course. Extra problems would be developed to allow adaptive paths; (2) Development efforts would be focused on Harvard-developed Learning Tools Interoperability (LTI) tool to support assessment adaptivity on edX platform. Therefore, in the current prototype phase of this project, adaptive functionality is limited to altering the sequence of problems, based on continuously updated statistical inferences on knowledge components a user mastered. As a supplement to these assessment items, a number of additional learning materials are served adaptively as well, based on the rule that a user should see those before being served more advanced problems.

While the prototype enabled us to explore the feasibility of adaptive assessment technology and implications of adaptive functionality to course (re)design in HarvardX, it is still challenging to judge its effects on learning outcomes, engagement and course drop-out rates due to the prototype limitations. However, we believe that the study will help to establish a solid foundation for future research on the effects of adaptive learning and assessment on outcomes such as learning gains and engagement. [5]

2. SETUP AND USER EXPERIENCE

The HarvardX course in this experiment was "Super-Earths and Life". It deals with searching for planets orbiting around stars other than the Sun, in particular the planets capable of supporting life. The subject matter is physics, astronomy and biology. Roughly speaking, the course aims at users with college-level knowledge of physics and biology. Some of the assessment material in the course requires calculations, and some requires extensive factual knowledge (e.g. questions about DNA structure).

Two versions of the course have already run in the edX platform, our adaptivity was implemented as part of the course re-design for the third run.

A number of subsections in the course contained assessment modules (homeworks). The experiment consisted of making four of these homeworks adaptive for some of the users. At the moment of their registration, the course users were randomly split 50%-50% into an experimental group and into a control group. When arriving to a homework, users in the control group see a predetermined, non-adaptive set of problems on a page. The same is true for the experimental group in all homeworks except the four where we deployed the adaptive tool. In these homeworks, a user from the experimental group is served problems sequentially, one by one, in the order that is individually determined on-the-fly based on the user's prior performance. In addition to problems, some instructional text pages were also included in the serving sequence.

To enable adaptivity, we manually compiled a list of knowledge components (KCs, for our purposes synonymous with "learning objectives", "learning outcomes", or "skills") and tagged problems in the course with one or several knowledge components. This tagging was done for *all* assessment items in the course (as well as for some learning materials), enabling the adaptive engine to gather information from any user's interaction with any problem in the course, not only with those problems that are served adaptively. Additionally, the problems in the 4 adaptive homeworks were tagged with one of three difficulty levels: advanced, regular and easy (other problems in the course were tagged by default as regular). No pre-requisite relationships or other connections among the knowledge components were used.

The adaptive engine (a variety of Bayesian Knowledge Tracing algorithm) decides which problem to serve next based on the list of KCs covered by the homework and course material. Additional rules could be incorporated into the serving strategy. Thus, we had a rule that before any problem of difficulty level "Advanced", the user should see a special page with advanced learning material.

The parity between experimental and control groups was set up as follows. In the pool from which problems are adaptively served to the experimental group, all the regular-difficulty problems were the ones that the control group saw in these homework. The control group had access to the easy and advanced problems as well: students in this group saw a special "extra materials" page after each of the 4 experimental homeworks. This page contained the links to all the advanced instructional materials and advanced and easy problems for this homework, for no extra credit. Thus, all the materials that an experimental user can see, were also available to the control students. There were two main reasons for this: obvious usefulness for comparative studies, and enabling all students, experimental and control, to discuss all problems in the course forum.

When an experimental group user is going through an adaptive homework, the LTI tool loads edX problem pages in an iFrame. Submitting ("checking") an answer to the problem triggers an update of user's mastery, but does not trigger serving the next problem. For that to happen, the user has to click the button "Next Question" outside the iFrame. The user always can revisit any of the previously served problems.

In edX, users usually get several attempts at a problem. Thus, it may be possible for a user to submit a problem after the next problem has already been served. Fig. 1, for instance, shows a situation, where so far 4 problems have been served (note the numbered tabs in the upper left), but the user is currently viewing problem 2 in this sequence, not the latest one. The user is free to re-submit this problem, which will update the user's mastery (although in this case there is no need to do so, since it appears that problem 2 has been answered correctly). It will not alter the existing sequence (problems 3 and 4 will not be replaced by others), but it may have effect on what will be served as 5 and so on.

The user interface keeps track of the total number of points earned in a homework (upper right corner in Fig. 1). The user knows how many points in total are required and may choose to stop once this is achieved (earning more points will no longer affect the grade). Otherwise, the serving sequence ends when the pool of questions is exhausted. Potentially, it could also end when the user's probability of mastery on all relevant KCs passes a certain mastery threshold (a high probability, at which we consider the mastery to be, in practical terms, certain; it was set to 0.9). However, in this particular implementation, due to having only a modest number of problems, this was not done.

In order to explore possible effects of adaptive experiences on learners' mastery of content knowledge competence-based pre- and post-assessment were added to the course and administered to study participants in both experimental and control groups. Typical HarvardX course clickstream time-stamped data and pre-post course surveys data was collected.

2.1 Course Design Considerations

Adaptive learning techniques require the development of additional course materials, so that different students can be provided with different content. For our prototype, tripling the existing content in the four adaptive subsections was considered a minimum to provide a genuine adaptive experience. This was achieved by work from the project lead and by hiring an outside content expert. This did not provide each knowledge component with a large number of problems, reducing the significance of knowledge tracing, but it was sufficient for the purpose of our experiment. The total time outlay was ~200 hours. Keeping the problems housed within the edX platform avoided substantial amounts of software development.

The tagging of content with knowledge components was done by means of a shared Google spreadsheet, which contained a list of content items in one sheet (both assessment and learning materials), a list of knowledge components in another, and a correspondence table (the tagging itself), including the difficulty levels, in the third.

Most of the time was spent on creating new problems based on the existing ones. For these the tagging process was "reversed": rather than tag existing content with knowledge components, the experts created content targeting knowledge components and difficulty levels. Commonly, an existing problem was considered to be of "regular" difficulty, and the expert's task was to create an "easy" and/or an "advanced" version of it.

103 distinct knowledge components were used in tagging. The experts used their judgement in defining them. 66 of these were used in tagging problems, and in particular the 39 adaptively served problems were tagged with 25 KCs. The granularity of KCs was such that a typical assessment problem was tagged with one learning objective (which is desirable for knowledge tracing). Namely, among the adaptively served problems, 31 were tagged with a single KC, 7 problems – with 2 KCs, and 1 problem – with 3.

2.2 LTI Tool Development

To enable the use of an adaptive engine in an edX course, Harvard developed the Bridge for Adaptivity (BFA) tool (open-source, GitHub link available upon request). BFA is a web application that uses the LTI specification to integrate with learning management systems such as edX. BFA acts as the interface between the edX course platform and the TutorGen SCALE (Student Centered Adaptive Learning Engine) system, and handles the display of problems recommended by the adaptive engine. Problems are accessed by edX XBlock URLs.

This LTI functionality allows BFA to be embedded in one or more locations in the course (4 locations in our case). The user interface seen by a learner when they encounter an installed tool instance is that shown in Fig. 1.

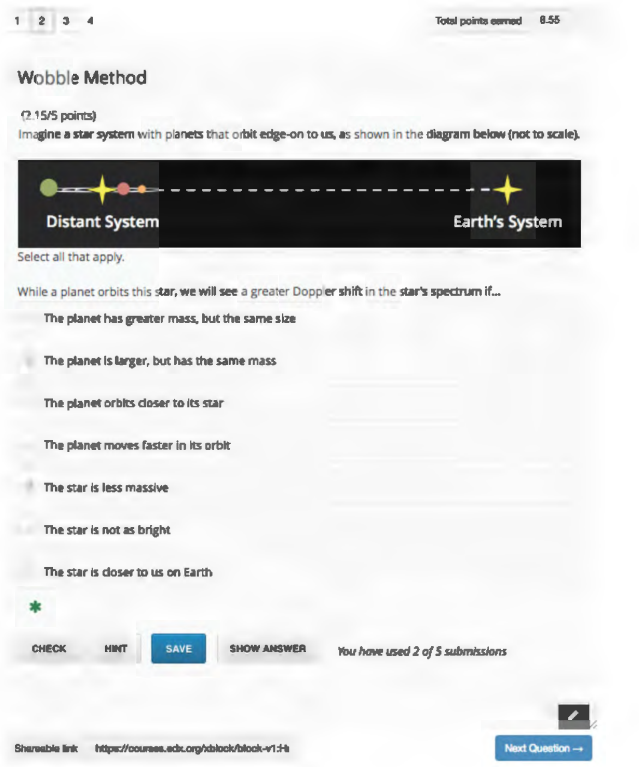


Figure 1. Adaptive assessment user interface

Problems from the edX course are displayed one at a time in a center activity window, with a surrounding toolbar that provides features such as navigation, a score display, and a shareable link for the current problem (that the learner can use to post to a forum for help). The diagram in Fig. 2 describes the data passing in the system. The user-ids used by edX are considered sensitive information and are not shared with SCALE: we created a different user-id system for SCALE, and the mapping back and forth between the two id-systems happens in the back end of the app.

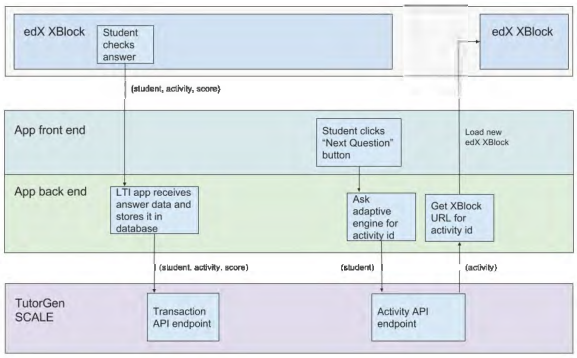


Figure 2. Diagram of data passing in the system

Every problem-checking event by the user (both inside and outside the adaptive homeworks) sends the data to SCALE, to update the mastery information real-time. Every "Next Question" event in an adaptive homework sends to SCALE a request for the next content item to be served to the user (this could be instructional material or a problem). SCALE sends back the recommendation, which is accessed as an edX XBlock and loaded.

The edX support for LTI is highly stable. The challenge is that edX exports data on a weekly cycle, but we needed to receive the information about submits in real time. We achieved this by creating a reporting JavaScript and inserting it into every problem.

2.3 TutorGen Adaptive Engine

TutorGen SCALE is focused on improving learning outcomes using data collected from existing and emerging educational technology systems combined with the core technology to automatically generate adaptive capabilities. Key features that SCALE provides include knowledge tracing, skill modeling, student modeling, adaptive problem selection, and automated hint generation for multi-step problems. SCALE engine improves over time with additional data and/or with the help of human input by providing machine learning using a human-centered approach. The algorithms have been tested on various data sets in a wide range of domains. For successful implementation and optimized adaptive operations, it is important that the knowledge components be tagged at the right level of granularity.

SCALE has been used in the intelligent tutoring system environment, providing adaptive capabilities during the formative learning stages. SCALE with HarvardX for this course is being used more as in the assessment stage of the student experience. In order to accomplish the goals of the prototype for this pilot study, we extended our algorithms to consider not only the knowledge components (KCs), but also problem difficulty. This will accommodate the needs for this course by providing an adaptive experience for students while still supporting the logical flow of the course. Further, the flexible nature of the course, having all content available and open to students for the duration of the course, presents some additional requirements to ensure that students are presented with problems based on their current state and not necessarily where the system believes they should navigate.

A variety of serving strategies are available in SCALE and can be swapped in and out. In this particular implementation, while the algorithm did trace the students' knowledge, the results were used minimally in the serving strategy: it did not make sense to do otherwise given the small size of the adaptive problem pool. SCALE was configured to consider after each submit: the probability of the learner has mastered the KCs from the problem most recently worked, the difficulty of that problem, and the correctness of the submitted answer. A general and simplified explanation of the process is as follows. Each of the four adaptive modules was treated as a separate instance, with its own pool of problems. Each problem can be served to each learner no more than once. Given the last problem submitted by a learner in the module, the candidate to be served next is the (previously unseen) problem, whose KC tagging overlaps with the KCs of the last submitted problem and includes at least one KC, on which the user has not yet reached the mastery threshold. If multiple candidates are available, SCALE will serve the one with a KC closest to mastery. If no candidates are available, other problems of the same difficulty within the same module will be served (i.e. SCALE switches to another KCs). The difficulty level of the next served problem is determined by the last submit correctness. As long as problems of the same difficulty level as the last one are available, the learner will remain at that difficulty level. Once such problems are exhausted, SCALE will serve a more or less difficult problem, depending on whether the last submit in the module was correct or incorrect.

2.4 Quantitative Details and Findings

The course was launched on Oct 19, 2016. The data for the analysis presented in this paper were accessed on Mar 08, 2017 (plus or minus a few days, since different parts of the data were extracted at different times), after the official end date of the course.

Table 1. Number of students attempting assessment items of different difficulty level

	Experimental group	Control group
Regular level only	58	73
Easy level only	0	0
Advanced level only	1	0
(Regular ∪ Easy) levels only	1	35
(Regular ∪ Advanced) levels only	105	0
(Easy ∪ Advanced) levels only	0	1
(Regular ∪ Easy ∪ Advanced) levels	99	145
Total students attempting new problems	264	254

We will refer to the list of problems from which problems were served adaptively to the experimental group as "new problems". The control group may have interacted with these as well, although not adaptively (as additional problems that do not count towards the grade). There were 39 new problems, out of which 13 were regular difficulty (these formed the assessments for the control group of students), 14 were advanced and 12 were easy. For the control group, the advanced and easy problems were offered as extra material after assessment, with no credit toward the course grade. The numbers of students attempting assessment problems of different difficulty levels are given in Table 1.

To get a sense of how the two groups of students performed in the course, we compared the group averages of the differences in scores in the pre-test and post-test. For reasons unrelated to this study, both tests were randomized: in each test each user received 9 questions, randomly selected from a bank of 17. All questions were graded on the 0-1 scale. The users knew that the pre- and post- tests do not contribute to the grade, and so only about ~40% of users took both. Moreover, not all of these questions were *relevant* for (i.e. tagged with) those 25 knowledge components, with which the adaptively served problems were tagged. So the number of offered *relevant* questions varied randomly from user to user. For these reasons the pre- and post-test are not the most reliable measure of knowledge gain, but it was still important for us to make sure that adaptivity did not have any adverse effect. Each question was graded on the scale 0-1, and in Fig. 3 we subset the student population to those individuals who attempted a "new problem" *and* a relevant pre-test question *and* a relevant post-test question, and used the average score from relevant questions as the student's relevant score. For instance, if one user attempted two relevant questions in a pre-test, and another user attempted three, and the questions were answered correctly, both users have the relevant score 1: (1+1)/2=(1+1+1)/3.

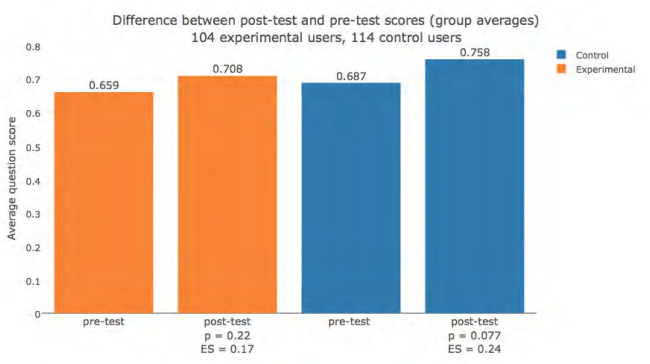


Figure 3. Comparison of relevant post-test and pre-test scores. Here and everywhere below, the p-values are two-tailed from the Welch two-sample t-test, and the effect size is the Cohen's d (Cohen suggested to consider d=0.2 as "small", d=0.5 as "medium" and d=0.8 as "large" effect size).

There is no significant between-group difference, neither in the pre-test scores (p-value 0.49, effect size 0.093) nor in the post-test scores (p-value 0.21, effect size 0.17). The two populations of pre-test takers remain comparable after subsetting to those who attempted new problems and the post-test and we see no statistically significant difference in the knowledge gaining between the experimental and control groups.

We did not see a difference in the final grade of the course: the mean grade was 83.7% in the experimental group vs. 82.9% in the control group, which is not a significant difference (p-value 0.76, effect size 0.06). Likewise, there is no significant between-group difference in the completion and certification rates (about 20%), or in demographics of students who did not drop out.

Students in the experimental group tended to make more attempts at a problem (Fig. 4), and they tried fewer problems (Fig. 5), most strikingly among the easy new problems: for these we have 1,162 recorded scores in the control group and only 423 in the experimental group.

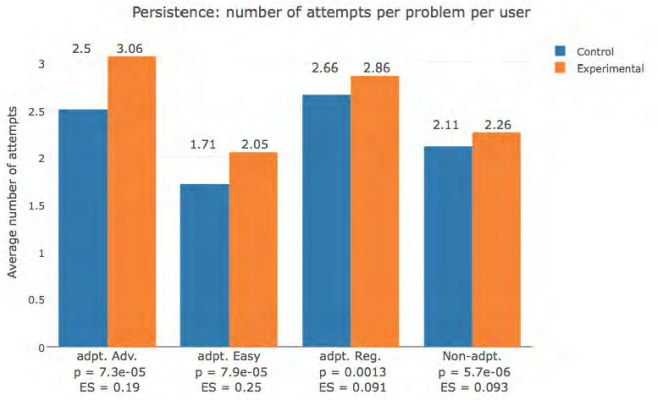


Figure 4. Comparison of attempt numbers between the experimental and control groups in the chapters where adaptivity was implemented. The attempt numbers are averaged both over the problems and over the users. Non-adaptive problems are problems not from the 4 experimental homeworks but from the same two chapters of the course as the experimental homeworks.

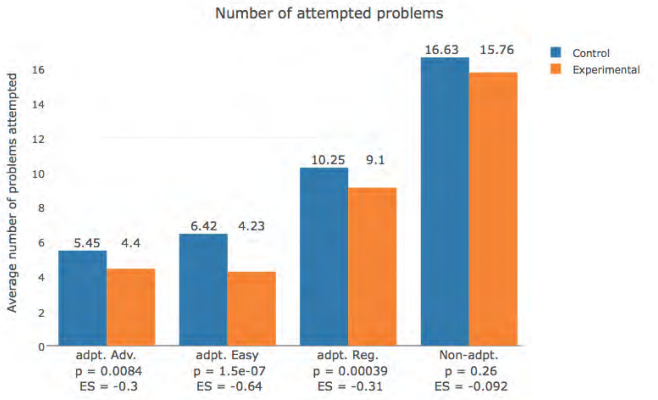


Figure 5. Comparison of attempt numbers between the experimental and control groups in the chapters where adaptivity was implemented. Non-adaptive problems are problems not from the 4 experimental homeworks but from the same two chapters of the course as the experimental homeworks.

The interpretation emerges that the students who experienced adaptivity showed more persistence by giving more attempts per problem (presumably, because adaptively served problems are more likely to be on the appropriate current mastery level for a student), while taking a faster track through the course materials. We also observed that the experimental group students tended to have a lower net time on task in the course: an average of 5.47 hours vs. 5.85 in the control group (although in this comparison the p-value is high, 0.21, and the effect size is –0.11).

Thus, we conjecture that the adaptivity of this kind leads to a higher efficiency of learning. Students go through the course faster and attempt fewer problems, since the problems are served to them in a targeted way. And yet there is no evidence of an adverse effect on the students' overall performance or knowledge gain. Given the limited implementation of adaptivity in this course, it is not surprising that we cannot find a statistically significant effect on student overall performance in the course. We expect to refine these conclusions in the future courses with a greater scope of adaptivity.

3. FUTURE WORK

Our implementation of adaptivity provided some insights for future work. For instance, assessment questions in MOOCs can vary greatly in nature, difficulty and format (multiple choice, check-all-that-applies, numeric response, etc.), and may often be tagged with more than one knowledge component. To be suitable for a MOOC, an adaptive engine should be able to handle these features.

There appear to be extensive opportunities to expand adaptive learning and assessment in MOOCs. The low total number of problems was the most severe restriction on the variability of learner experience in this study. In the future applications, larger sets of tagged items could provide a more adaptive learning experience for students, while also providing a higher degree of certainty of assessment results. Interestingly, in some MOOCs (for example, those teaching programming languages) it may be possible to create very large numbers of questions algorithmically, essentially by filling question templates with different data.

In this study, adaptivity was implemented mostly on assessment problems. Given the structure of many MOOCs, more integration between learning content and assessment could provide an adaptive experience that would guide students to content that could improve their understanding based on how they perform on integrated assessments.

Affective factors could be included to provide a more personalized learning experience. We can conceive an adaptive engine which decides what item to serve next based not just on the mastery but also on the behavioral patterns interpreted as boredom or frustration.

Finally, this work could lead to improved MOOC platform features that would contribute to improved student experiences, such as optimized group selection [2]. In addition, we anticipate expanding this adaptive assessment system to work with other LTI-compliant course platforms. Enabling use in a platform such as Canvas, the learning management system used university-wide at Harvard (and many other schools), would enable adaptivity for residential courses on a large scale. An adjustment to the current system architecture would be the use of OpenEdX as the platform for creating and hosting problems.

4. ACKNOWLEDGMENTS

We are grateful for the support from the Office of the Vice Provost for Advances in Learning at Harvard University for thoughtful leadership and support to HarvardX and the VPAL-Research group. Special thanks to Professor Dimitar Sasselov from the Harvard Department of Astronomy whose "Super-Earths and Life" MOOC made this project possible. TutorGen gratefully acknowledges support of SCALE® from the National Science Foundation award numbers 1346448 and 1534780 and from the Commonwealth of Kentucky Cabinet for Economic Development, Kentucky Science and Engineering Foundation, and The Kentucky Science and Technology Corporation, award numbers KSTC-184-512-14-182 and KSTC-184-512-16-241.

5. REFERENCES

[1] Koedinger, K., and Stamper, J. 2010. A Data Driven Approach to the Discovery of Better Cognitive Models. In Baker, R.S.J.d., Merceron, A., Pavlik, P.I. Jr. (Eds.) *Proceedings of the 3rd International Conference on*

Educational Data Mining. (EDM 2010), 325-326. Pittsburgh, PA.

[2] Rosen, Y. 2017. Assessing students in human-to-agent settings to inform collaborative problem-solving learning. *Journal of Educational Measurement* 54, 1: 36-53.

[3] Rosen, Y. 2015. Computer-based assessment of collaborative problem solving: Exploring the feasibility of human-to-agent approach. *International Journal of Artificial Intelligence in Education* 25, 3: 98-129.

[4] Stamper, J., Barnes, T., and Croy, M. 2011. Experimental Evaluation of Automatic Hint Generation for a Logic Tutor. In Kay, J., Bull, S. and Biswas, G. eds. *Proceeding of the 15th International Conference on Artificial Intelligence in Education (AIED2011)*. 345-352. Berlin Germany: Springer.

[5] A preliminary report of our study (based on the data obtained prior to the course end) is to appear in the *Proceedings of the Fourth Annual ACM Conference on Learning at Scale* (L@S 2017) as: Rosen, Y., Rushkin, I., Ang A., Fredericks C., Tingley D., Blink M.J. 2017. Designing Adaptive Assessments in MOOCs.

Workshops

Graph-based Educational Data Mining (G-EDM 2017)

Collin F. Lynch, Tiffany Barnes, Linting Xue, & Niki Gitinabard
North Carolina State University, Raleigh, North Carolina, USA
cflynch, tmbarnes, lxue3, & ngitina@ncsu.edu

ABSTRACT

With the growing popularity of MOOCs and computer-aided learning systems, as well as the growth of social networks in education, we have begun to collect increasingly large amounts of educational graph data. This graph data includes complex user-system interaction logs, student-produced graphical representations, and conceptual hierarchies that large amounts of graph data have. There is abundant pedagogical information beneath these graph datasets. As a result, graph data mining techniques such as graph grammar induction, path analysis, and prerequisite relationship prediction has become increasingly important. Also, graphical model techniques (e.g. Hidden Markov Models or probabilistic graphical models) has become more and more important to analyze educational data.

While educational graph data and data analysis based on graphical models has grown increasingly common, it's necessary to build a strong community for educational graph researchers. This workshop will provide such a forum for interested researchers to discuss ongoing work, share common graph mining problems, and identify technique challenges. Researchers are encouraged to discuss prior analyses of graph data and educational data analyses based on graphical models. We also welcome discussions of in-progress work from researchers seeking to identify suitable sources of data or appropriate analytical tools.

1. PRIOR WORKSHOPS

So far, we have successfully held two international workshops on Graph-based Educational Data-Mining. The first one was held in London, co-located with EDM 2014. It featured 12 publications of which 6 were full-papers, the remainder short papers. Having roughly 25 full-day attendees and additional drop-ins, it led to a number of individual connections between researchers and the formation of an e-mail list for group discussion. The second one was co-located with EDM 2015 in Spain. 10 authors presented their published work including 4 full papers and 6 short papers there.

2. OVERVIEW AND RELEVANCE

Graph-based data mining and educational data analysis based on graphical models have become emerging disciplines in EDM. Large-scale graph data, such as social network data, complex user-system interaction logs, student-produced graphical representations, and conceptual hierarchies, carries multiple levels of pedagogical information. Exploring such data can help to answer a range of critical questions such as:

- For social network data from MOOCs, online forums, and user-system interaction logs:
  - What social networks can foster or hinder learning?
  - Do users of online learning tools behave as we expect them to?
  - How does the interaction graph evolve over time?
  - What data we can use to define relationship graphs?
  - What path(s) do high-performing students take through online materials?
  - What is the impact of teacher-interaction on students' observed behavior?
  - Can we identify students who are particularly helpful in a course?
- For computer-aided learning (writing, programming, etc.):
  - What substructures are commonly found in student-produced diagrams?
  - Can we use prior student data to identify students' solution plan, if any?
  - Can we automatically induce empirically-valid graph rules from prior student data and use induced graph rules to support automated grading systems?

Graphical model techniques, such as Bayesian Network, Markov Random Field, and Conditional Random Field, have been widely used in EDM for student modeling, decision making, and knowledge tracing. Utilizing these approaches can help to:

- Learn students' behavioral patterns.
- Predict students' behaviors and learning outcomes.

- Induce pedagogical strategies for computer-aided learning systems.
- Identify the difficult level of the knowledge components in the intelligent tutoring systems.

Researches related to these questions can help us to better understand students' learning status, and improve the teaching effectiveness and student learning. Our goal in this workshop is to bring together researchers with special interest in graph-based data analysis to 1) discuss state of the art tools and technologies, 2) identify common problems and challenges, and 3) foster a community of researchers for further collaboration. We will consider the submission of full and short papers as well as posters and demonstrations covering a range of graphics topics that include, but are not limited to:

- Social network data
- Graphical solution representations
- Graphical behavior models
- Graph-based log analysis
- Large network datasets
- Novel graph-based machine learning methods
- Novel graph analysis techniques
- Relevant analytical tools and standard problems
- Issues with graph models
- Tools and technologies for graph grammar (pattern) recognition
- Tools and technologies for automatic concept hierarchy extraction
- Computer-aided learning system development involved with graphical representations
- Use of graphical models in educational data

We particularly welcome submissions of in-progress work both from students and researchers with problems who are seeking appropriate analytical tools, and developers of graph analysis tools who are seeking new challenges.

3. WORKSHOP ORGANIZERS

Dr.Collin F. Lynch is an Assistant Professor in the Department of Computer Science at North Carolina State University. His primary research is focused on graph-based educational data mining, the development of robust intelligent tutoring systems, and adaptive educational systems for ill-defined domains such as scientific writing, law, and engineering. In his more recent work he has also been involved in the development of Intelligent Tutoring Systems for Logic and Probability and social networking analysis for research communities.

Dr.Tiffany Barnes is an Associate Professor of Computer Science at NC State University. She received an NSF-CAREER Award for her novel work in using data to add intelligence to STEM learning environments. That grant supported the development of InVis a novel tool that use graph-based representations of student-tutor interaction data to evaluate the impact of intelligent tutoring systems on student problem-solvers and to automatically extract hints and student advice from log data using graph-analysis. More recently she has received grants for the analysis of large-scale online courses and the development of procedural guidance from intelligent tutoring system data.

Linting Xue is a third year Ph.D. student in the Department of Computer Science at North Carolina State University. She is interested in the graph data mining methods for educational graph data. Her current research is focused on automatically graph grammars induction for student-produced argument diagrams. The induced graph grammars can be used as features for automatic grading and provide the hints for argumentative writing.

Niki Gitinabard is a second year Ph.D. student in the Department of Computer Science at North Carolina State University. She is interested in social network analysis in learning environments. She is currently working on social graph generation and analysis based on students' explicit and implicit interactions.

4. WORKSHOP ORGANIZATION

We will organize this workshop as a full or half-day mini-conference with time set aside for paper presentations, large-group discussion, and individual networking. We will open the workshop with a summary of prior meetings. We will spend the morning on presentations with a short discussion session before lunch. The afternoon session will be divided between presentations and working groups which will focus on identifying shared problems, small-group networking, and planning for follow up work. We will invite submissions of full papers which describe mature work. We will also accept short papers describing in-progress work or student projects, and poster/demo submissions for those presenting available data, tools, and methods. This last category is particularly targeted at researchers who have data or methods available and are seeking to identify potential collaborators.

Workshop on deep learning with educational data

Ryan Baker
University of Pennsylvania
Philadelphia, PA 19104
ryanshaunbaker@gmail.com

Joseph E. Beck
Worcester Polytechnic Institute
Worcester, MA, 01609
josephbeck@wpi.edu

Min Chi
North Carolina State University
Raleigh, NC 27695
mchi@ncsu.edu

Neil T. Heffernan
Worcester Polytechnic Institute
Worcester, MA, 01609
nth@wpi.edu

Mike Mozer
University of Colorado Boulder
Boulder, CO 80309
mozer@colorado.edu

1. WORKSHOP TOPIC

This workshop focuses on applications of deep learning for educational data. Deep learning is a machine learning approach using neural networks with multiple levels of representational transformation (i.e., hidden layers). Deep learning has been used in a variety of domains over the past five years with impressive results. Recently, it has been used for educational data sets with mixed results when compared to traditional modeling methodologies.

We are interested in work on a variety of topics with deep learning: new prediction and modeling problems, best practices for featurizing data, network architectures, approaches to pre-training and whether it is necessary, interpreting the learned models, end-to-end deep learning approaches with low-level non-symbolic data, toolkits people have developed, empirical results on known problems to help the field develop best practices. The workshop is also interested in negative results such as analyses of data sets and domains where deep learning fails to achieve state of the art performance.

2. GOALS OF WORKSHOP

The primary goal of this workshop is to provide a venue for researchers to present emerging work. There is not much prior art on applying deep learning to educational data, and it is unclear even what the scope of possible applications are: although most work has focused on student modeling, some work has focused on using deep learning to assist in scoring essays. Having a discussion about possible application areas will be productive.

In addition, this workshop will focus on recent big topics in deep learning for educational data. A paper published in 2016 "How deep is knowledge tracing" questions the need for deep models, and will be discussed at the workshop.

Finally, this workshop will provide researchers on deep learning for EDM a chance to get focused feedback on their work. Ensuring that the research is critiqued by a roomful of people interested in the topic is more useful to the presenters (and the community) than counting on haphazard interactions at the conference.

Sharing and Reusing Data and Analytic Methods with LearnSphere

Ran Liu
ranliu@cmu.edu

Kenneth Koedinger
koedinger@cmu.edu

John Stamper
jstamper@cs.cmu.edu

Philip Pavlik
ppavlik@memphis.edu

ABSTRACT

This workshop will explore LearnSphere, an NSF-funded, community-based repository that facilitates sharing of educational data and analytic methods. The workshop organizers will discuss the unique research benefits that LearnSphere affords. In particular, we will focus on Tigris, a workflow tool within LearnSphere that helps researchers share analytic methods and computational models. Authors of accepted workshop papers will integrate their analytic methods or models into LearnSphere's Tigris in advance of the workshop, and these methods will be made accessible to all workshop attendees. We will learn about these different analytic methods during the workshop and spend hands-on time applying them to a variety of educational datasets available in LearnSphere's DataShop. Finally, we will discuss the bottlenecks that remain, and brainstorm potential solutions, in openly sharing analytic methods through a central infrastructure like LearnSphere. Our ultimate goal is to create the building blocks to allow groups of researchers to integrate their data with other researchers in order to advance the learning sciences as harnessing and sharing big data has done for other fields.

Keywords

Learning metrics; data storage and sharing; data-informed learning theories; modeling; data-informed efforts; scalability.

1. INTRODUCTION

Due to a confluence of a boom of interest both in educational technology and in the use of data to improve student learning, student learning activities and progress are increasingly being tracked and stored. There is a large variety in the kinds, density, and volume of such data and to the analytic and adaptive learning methods that take advantage of it. Data can range from simple (e.g., clicks on menu items or structured symbolic expressions) to complex and harder-to-interpret (e.g., free-form essays, discussion board dialogues, or affect sensor information). Another dimension of variation is the time scale in which observations of student behavior occur: click actions are observed within seconds in fluency-oriented math games or in vocabulary practice, problem-solving steps are observed every 20 seconds or so in modeling tool interfaces (e.g., spreadsheets, graphers, computer algebra) in intelligent tutoring systems for math and science, answers to comprehension-monitoring questions are given and learning resource choices are made every 15 minutes or so in massive open online courses (MOOCs), lesson completion is observed across days in learning management systems, chapter/unit test results are collected after weeks, end-of-course completion and exam scores are collected after many months, degree completion occurs across years, and long-term human goals like landing a job and achieving a good income occur across lifetimes. Different paradigms of data-driven education research differ both in the types of data they tend to use and in the time scale in which that data is collected. In fact, relative isolation within disciplinary silos is arguably fostered and fed by differences in the types and time scale of data used [4, 5].

Thus, there is a broad need for an overarching data infrastructure to not only support sharing and use within the student data (e.g., clickstream, MOOC, discourse, affect) but to also support investigations that bridge across them. This will enable the research community to understand how and when long-term learning outcomes emerge as a causal consequence of real-time student interactions within the complex set of instructional options available [2]. Such an infrastructure will support novel, transformative, and multidisciplinary approaches to the use of data to create actionable knowledge to improve learning environments for STEM and other areas in the medium term and will revolutionize learning in the longer term.

LearnSphere transforms scientific discovery and innovation in education through a scalable data infrastructure designed to enable educators, learning scientists, and researchers to easily collaborate over shared data using the latest tools and technologies. LearnSphere.org provides a hub that integrates across existing data silos implemented at different universities, including educational technology "click stream" data in CMU's DataShop, massive online course data in Stanford's DataStage and analytics in MIT's MOOCdb, and educational language and discourse data in CMU's new DiscourseDB. LearnSphere integrates these DIBBs in two key ways: 1) with a web-based portal that points to these and other learning analytic resources and 2) with a web-based workflow authoring and sharing tool called Tigris. A major goal is to make it easier for researchers, course developers, and instructors to engage in learning analytics and educational data mining without programming skills.

2. SPECIFIC WORKSHOP OBJECTIVES

Broadly, this workshop offers those in the EDM community an exposure to LearnSphere as a community-based infrastructure for educational data and analysis tools. In opening lectures, the organizers will discuss the way LearnSphere connects data silos across universities and its unique capabilities for sharing data, models, analysis workflows, and visualizations while maintaining confidentiality.

More specifically, we propose to focus on attracting, integrating, and discussing researcher contributions to Tigris, the web-based workflow authoring and sharing tool. The goal of Tigris is to support any custom analysis method that can be applied to the datasets and to produce outputs in a standardized way that facilitates both quantitative and qualitative model comparisons. This workflow feature allows researchers to apply their own analysis methods to the vast array of datasets available in the educational data repository. It affords researchers the advantages of (1) using the built-in learning curve visualizations on the outputs of their own analysis workflows, (2) easily comparing their results both quantitatively and graphically to the outputs of

any other analysis methods that are currently in LearnSphere (e.g., Bayesian Knowledge Tracing [1], Performance Factors Analysis [6], MOOC activity analysis [3], and others) or that have been uploaded to LearnSphere as a custom workflow, and (3) sharing their own analysis workflows with the community of researchers. Without any prior programming experience, researchers can use LearnSphere's drag-and-drop interface to compare, across alternative analysis methods and across many different datasets, model fit metrics like AIC, BIC, and cross validation as well as parameter estimates themselves.

Workshop submissions will involve a brief description of an analysis pipeline relevant to modeling educational data as well as accompanying code. Prior to the workshop itself, the organizers will coordinate with authors of accepted submissions to integrate their code into Tigris. A significant portion of the workshop will be dedicated to hands-on exploration of custom workflows and workflow modules within Tigris. Authors of accepted submissions will present their analysis pipelines, and everyone attending the workshop will be able to access those analysis pipelines within Tigris to a variety of freely available educational datasets available from LearnSphere. The end goal is to generate, for each workflow component contribution in the workshop, a publishable workshop paper that describes the outcomes of openly sharing the analysis with the research community.

Finally, workshop attendees will discuss bottlenecks that remain toward our goal of an easier, more open way to share analytic tools. We will also brainstorm possible solutions. Our goal is to create the building blocks to allow groups of researchers to integrate their data with other researchers we can advance the learning sciences as harnessing and sharing big data and analytics has done for other fields.

3. REFERENCES

[1] Corbett, A.T., & Anderson, J.R. (1995). Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4, 253-278.

[2] Koedinger, K.R., Booth, J.L., & Klahr, D. (2013). Instructional complexity and the science to constrain it. Science, 342(6161), 935-937.

[3] Koedinger, K.R., Kim, J., Jia, J.Z., McLaughlin, E.A., & Bier, N.L. (2015). Learning is not a spectator sport: Doing is better than watching for learning from a MOOC. In *Proceedings of the 2nd ACM Conference on Learning@ Scale*, pp. 111-120.

[4] Koedinger, K.R., Corbett, A.T., & Perfetti, C. (2012). The Knowledge‐Learning‐Instruction framework: Bridging the science‐practice chasm to enhance robust student learning. *Cognitive science, 36*(5), 757-798.

[5] Newell, A. (1990). Unified theories of cognition. Cambridge, MA: Harvard University Press.

[6] Pavlik, P.I., Cen, H., & Koedinger, K.R. (2009). Performance factors analysis – A new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on AIED*, 531–538.

Keyword Index

Author Index