Deep Learning + Student Modeling + Clustering: a Recipe for Effective Automatic Short Answer Grading

Yuan Zhang, Rajat Shah, and Min Chi
North Carolina State University
{yzhang93, rshah6, mchi}@ncsu.edu

ABSTRACT

In this work we tackled the task of Automatic Short Answer Grading (ASAG). While conventional ASAG research makes prediction mainly based on student answers referred as Answer-based, we leveraged the information about questions and student models into consideration. More specifically, we explore the Answer-based, Question, and Student models individually, and subsequently in various combined and composite models through feature engineering. Additionally, we extend the exploration of machine learning methods by utilizing Deep Belief Networks (DBN) together with other five classic classifiers. Our experimental results show that our proposed feature engineering models significantly out-performed the conventional Answer-based model and among the six machine learning classifiers, DBN is the best followed by SVM, and Naive Bayes is the worst.

1. INTRODUCTION

Developing effective Computer-based assessment has been increasingly gaining its importance over years and it is widely believed that open-ended problems are more effective to access student knowledge than multiple choices. The former require students to generate free text and communicate their responses and thus student answers are relatively immune to test-taking shortcuts like eliminating improbable answers. On the other hand, grading student's free text answers is often time-consuming and challenging. Therefore, much research has focused on how to automatically grade student free text answers. Generally speaking, research to date has concentrated on two sub-tasks: grading student essays, which includes checking the style, grammar, and coherence of an essay [13], and grading student short answers [16, 18, 19], which is the focus of this work. More formally, [7] defined short answers as those: 1) in the form of natural language; 2) requiring students to recall external knowledge that is not provided by the question; 3) of which the length ranges between one phrase to one paragraph; 4) focusing on the correctness of the content rather than the style; and 5) and are closed, which means that the answers have to match the specific facts corresponding to questions. The goal of this work is to explore effectiveness of various Machine Learning (ML) approaches on Automatic Short Answers Grading (ASAG). An ASAG system is one that automatically classify student answers into, correct or incorrect, based on the referred correct one(s).

Much of the prior research on ASAG is answer-based which involves applying various Natural Language Processing (NLP) techniques to extract a wide variety of text-based features directly from student answers. These features include various measurements of text similarities between student answers and the referred correct ones. Often time, the shorter the student answers, the harder to classify them into correct or incorrect because the limited text provides fewer lexical features. Many classic NLP approaches such as bag-of-words or keyword matching often fail to work. For example, Table 1 shows an example of student short answer extracted from our training corpus. In this example, using text similarities alone would fail to recognize that the student's answer is correct because it looks quite different from the referred correct answer.

Table 1: An Example of Student Short Answer.

Tutor: Why are there no potential energies involved in this problem?
Student: There is no second object that is massive and can have gravitational energy. **(Correct)**
Correct Answer: Because the rock is the only object in the system, there are no potential energies involved.

On the other hand, information about question and student knowledge can be handily used to improve the effectiveness of existing answer-based ASAG model. For example, in the example above if we know that the question is about "potential energy" and the student's knowledge on "potential energy" is very high, it is more likely that the student will answer the question correctly even though his/her answer looks quite different from the correct one. Thus in this paper we will investigate whether the effectiveness of ASAG can be further improved if we leverage question model, student model, or both into the answer-based model. To the best of our knowledge, this is the first comprehensive study exploring the effectiveness of feature space from all three models on the task of ASAG. For simplicity reasons, in the following we will refer the three models as Answer(Ans), Question(Ques), and Student (Stu) models respectively.

Prior research on ASAG has explored several classic ML classifiers such as Naïve Bayes and Decision Tree. In re-

cent years, Deep Belief Network (DBN) [5] has been successfully implemented and applied in a wide variety of real-world tasks [15,17]. DBN enables the automatic extraction of representative features via an unsupervised pre-training and it can learn the latent complex relationship among features. Given the potential complex connections among the features from Ans, Ques and Stu models, we investigated on leveraging DBN to exploit the more discriminative feature space to facilitate automatic grading. As far as we know, this is the first study to apply DBN to the task of ASAG.

To summarize, we investigated on improving ASAG by utilizing DBN together with five classic ML methods and by extending existing answer-based approaches to leverage a wide range of state features which are either based on or generated from Ans, Ques, and Stu models.

2. RELATED WORK

Popular Natural Language Tutors like AutoTutor [11] and BEETLE II [12] have extensively studied how to automatically understand student Natural Language inputs so that the system can respond to student's responses adaptively. Pulman and Sukkarieh used manually crafted patterns in the part-of-speech tagged answers for pattern matching with the correct answer [19]. Their approach is question-specific in that they applied Naïve Bayes and Decision Tree to automatically generate patterns for each question using a set of marked answers. Results showed their approach can achieve an average accuracy of 84%.

Mohler and Mihalcea developed an unsupervised approach using Knowledge-based and Corpus-based text-to-text similarity measures [18]. They used Latent Semantic Analysis coupled with domain specific corpus built from Wikipedia. Their resulted measures outperformed other similarity measures in that the former obtained Pearson correlation $r = 0.463$ between the computer assigned grades and average of human assigned grades.

Recently, Microsoft's Power Grading [2] took a semi-automated approach based on the observation that similar answers get similar grades. Thus, instead of directly grading student answers, Power Grading builds a hierarchy of short-answer clusters and lets human grader either grade the entire cluster with same score or manipulate the clusters as needed. Inspired by their work and promising results, we borrowed some of the features such as length and tf-idf from previous research into this work.

Our approach differed from previous research in that: 1) unlike relying solely on answer-based methods, we explored features from Ans, Ques and Stu models individually and combined; 2) our models are trained across all questions, that is, it is question-general instead of building question-specific classifiers in previous research; 3) previous approaches mainly involved two or three ML methods while we used a total of six including the state-of-the-art DBN together with five other traditional ML approaches.

3. METHODS

In this section, we will briefly describe the features involved in this study and the ML classifiers applied. For the latter, we will focus on DBN.

3.1 State Features

To investigate the impact of state features on the task of ASAG, we compare the effectiveness of various features from Ans, Ques and Stu models *individually* and *combined*. We also *composite* new features generated within or across different models.

3.1.1 Answer (Ans) Model

In [7], Burrows et al. identified two categories of answer-based approaches: corpus-based approaches are based on mapping the concepts in student answers to those in the reference correct answers [16], while alignment-based approaches are based on clustering student answers by some quality similarity estimates among student answer representations regardless of the correct answers. Our Ans model includes both corpus-based features and alignment-based ones.

Based on [2] and [18], we defined five Ans-based features by measuring the text similarity between student answer and the correct answer(s). The latter consist of the referred correct answer and the correct answers generated by students. More specifically, we have:

- *length difference*: the length difference (in words) between the student and the correct answers.
- *max-matched idf*: the maximum value of idf of matched words in a student answer. The idf of each word is calculated based on the Bag-Of-Word(BOW) generated from the word-answer matrix. This is a good measure to reflect whether prominent keywords in correct answers show up in the student answer.
- *cosine similarity* is calculated using tf-idf vectors of the student answer and the referred correct answers.
- *weighted text similarity*: Wu & Palmer similarity is a knowledge-based measure for text similarity [18], which is based on word similarities. More specifically, we formalize the text similarity between the student answer **s** and the correct answers **c** as sentences. We construct a domain specific word list d for the specific domain by assigning higher weight to domain specific words. Then the text similarity is calculated by weighting the similarities of general words $sim_w(s, c)$ and those of domain specific words $sim_d(s, c)$.
- *Latent Semantic Analysis* (LSA, Landauer and Dumais, 1997): is a computational method which aims to represent a corpora of natural text using the latent subspace. This subspace reflects the weight of each word in each answer so that similar correct answers share similar weight vector of words.

3.1.2 Question (Ques) Model

In domains such as math and science, it is commonly assumed that the relevant knowledge is structured as a set of independent but co-occurring Knowledge Components (KCs). A KC is "a generalization of everyday terms like concept, principle, fact, or skill, and cognitive science terms like schema, production rule, misconception, or facet" [21].

In many Intelligent Tutoring Systems (ITSs) such as Cordillera, completion of a tutor question requires students to apply multiple KCs. By including KCs in our model, we wish to guide the learning process in distinguishing between different

types of questions. Moreover, utilizing KCs is helpful for exploiting the homogeneity among questions. The central idea of Ques model is to build a *Q-matrix* to represent the relationship between individual questions and KCs. Q-matrices are typically encoded as a binary 2-dimensional matrix with columns representing KCs and rows representing questions. Previous researchers have focused on the task of generating or tuning Q-matrices based upon a dataset [1, 20]. For the present work we employ a static Q-matrix manually generated from domain experts.

Additionally, for each question we also include a feature named *questionDifficulty*. It has consistently been selected as one of the important features in our previous work on exploring various state features for modeling student learning [9]. *questionDifficulty* is defined as difficulty level of a question and its value is roughly estimated from the training corpus based on the percentage of answers that were correct on the question in the training dataset.

3.1.3 Student (Stu) Model

Student modeling is an important component for any interactive e-learning environment so that the system can adapt its behaviors based on student needs and knowledge [3]. There are many techniques for generating student models and among them, Bayesian Knowledge Tracing (BKT) [10] is the most widely used. Fundamentally, the BKT model can be seen as a Hidden Markov Model with two hidden states: learned and unlearned. They are defined based on whether a student has mastered the target knowledge or not. BKT keeps a running assessment of the probability that a student is in the learned state based on the student's past history of performance (e.g. *correct*, *incorrect*). BKT assumes that student learning process is a Markov Chain in that at each time $t+1$, the probability of a student has learned the knowledge p^{t+1} is only dependent on his learning state at time t.

Our Stu model used the outputs of the BKT, that is the probability that a student is in the learned state after answering n questions, denoted as $p(S^n = learned)$ as state features. Moreover, our Stu model is KC-specific in that for each of domain KCs, our model will include one probability of being in the learned state on the corresponding KC in the Stu model. Our goal is to use these KC specific probabilities to predict whether the student will answer the next question correctly. Additionally, we also included student KC-specific pretest scores which measures student initial incoming competence.

Therefore, our final Stu model includes a combination of KC-specific learning probabilities calculated from BKT and the student KC-specific pretest scores.

3.1.4 Composite Feature Space

In this part we will explore state features representing the underlying connections between the Ques and the Stu models. As described above, KCs are involved in both Ques and Stu models and thus we hypothesized that a student's performance on a problem should depend on the KCs involved in the problem and the student's performance on corresponding KCs. Hence, we conduct the Cartesian product (CP) using the Ques and Stu models. Additionally, we applied the clustering on the Stu model based on their learning states and pretest scores. Compared with the original features in the Stu model, using student clustering can be seen as more compact representation. Here we used Gaussian Mixture Model, which is a type of soft-clustering methods. Similarly, we hypothesized that the students with similar patterns in Stu clusters may have similar performance on certain types of questions and thus we also conduct the Cartesian product using the student clustering features and Ques vector.

3.2 Six Classifiers

Prior research on ASAG successfully explored several classic ML methods which included: Naive Bayes (NB), Logistic Regression (LR), Decision Tree (DT), Artificial Neutral Network (ANN), and Support Vector Machine (SVM). In recent years deep learning model has been widely used in computer vision and image processing. In this paper, we will compare Deep Belief Networks (DBN) [5] against those five classic ML methods. Given the space constraints, we only briefly describe DBN in the following paragraphs.

DBN is one of the most widely implemented deep learning models. Through the unsupervised pre-training in the first stage, DBN is able to extract the latent features that are more representative than the original input features. Given the input features, DBN first utilizes the stacked Restricted Boltzmann Machine (RBM) layers to automatically extract the high-level features. After the feature extraction in pre-training phase, the weights in these layers are then folded into neural networks for supervised training. Since the capacity of feature extraction mainly lies in the pre-training phase, we now present the mechanism of RBM.

RBM is a restricted version of Markov Random Field. It consists of two layers of variables, visible units V and hidden units H. From the perspective of feature extraction, V stands for the original feature inputs and H denotes the extracted feature representation. The joint distribution of V and H is defined by an energy-based probabilistic model, as follows:

$$P(V,H) = \frac{exp(-E(V,H))}{Z}, \quad Z = \sum_{V,H} exp(-E(V,H)) \tag{1}$$

where the energy function $E(V,H)$ is defined to be:

$$E(V,H) = -V^T W H - B^T V - C^T H. \tag{2}$$

In the above equation, W denotes the weights between V and H. Specifically, $W_{i,j}$ represents the weight between V_i and H_j, and B, C denote the biases for visible units and hidden units, respectively. The denominator Z serves as the normalizer for the probability distribution.

Given that each unit of V or H is independent with other units in the same layer, the conditional distribution is fully factorial and can be easily derived. Due to the intractability of gradient computation brought by the factor Z, the training of RBM (i.e., pre-training phase) follows the Contrastive Divergence algorithm [14], which executes K steps of alternating Gibbs sampling to approximate the gradient. The details can be found in [4].

4. DATA DESCRIPTION

Our training corpus was collected from Cordillera [8, 21], a Natural Language ITS that teaches students introductory college physics. The domain consists of a subset of the physics work-energy domain, which is characterized by eight primary KCs including Kinetic Energy, Gravitational Potential Energy, Spring Potential Energy, and so on. In Cordillera, students interact with tutor by means of natural language entries, and currently the Natural Language understanding module in Cordillera is using human interpreters referred as the language understanding wizard [6]. The *only* task performed by the human wizards is to match student answers to the closest response from a list of potential correct or incorrect responses.

Our training corpus involves 158 students. The data collection consists of the following stages: 1) background survey; 2) studying textbook and prerequisite materials, 3) taking a pretest; 3) training on Cordillera, 4) and taking a post test. In total there are 482 different questions involved in the training corpus and it takes students roughly 4-9 hours to complete the training. Our training corpus includes sequences of tutorial dialogue interactions between students and Cordillera, one sequence per student, and the average number of Cordillera-student interactions is more than 280 per student. For each interaction in a sequence, it consists of a tutor question, a student answer to the question, and two output labels ***correct*** or ***incorrect*** based on human wizards inputs. Thus, *these human manually generated binary labels function as ground truth in our training corpus.*

Based on the definition in [7], our training corpus included **16228** short answers selected from a total of 27868 dialogues. The average length of student answers in our corpus is **7.6** words. **61.66%** of training corpus is labeled as "correct" while the rest are labeled as "incorrect". A series of standard natural language pre-processings including stop word removal, tokenization, punctuation removal and word correction, have been conducted on our training corpus. Additionally, we also conducted domain-specific pre-processing, which includes expanding acronyms to their full forms and removing quantitative questions with equations.

5. EXPERIMENTS

To evaluate the effectiveness of various features from Ans, Ques, and Stu models individually, combined, and/or composite features generated from these three models, we use two ubiquitously implemented classifiers - LR and SVM in Experiment 1. Then in Experiment 2, we will compare DBN against five classic ML classifiers on the best feature model produced in Experiment 1.

5.1 Experiment 1: Exploring Feature Space

For Ans model, we use the five Ans features described in 3.1.1. For Ques model, we include 9 Ques features (one is *questionDifficulty* and the other eight are KC-based Q-matrix features, one feature per KC) and for Stu model, we include 16 Stu features (8 KC-based learning parameters and 8 KC-based pretest scores). Generally speaking, our Experiment 1 can be divided into three stages:

In stage 1, we compare the Ans, Ques, and Stu model individually. Our goal is to investigate whether either Ques or Stu model will be more effective than Ans model for ASAG. In stage 2, we will compare different ways of combining the three basic models. Our results from stage 1 show that Ans-based model alone performs better than either Ques or Stu model (depicted in Section 6.1.1) and thus we mainly explore whether to include the Ques and/or Stu models to the Ans-based model in stage 2. Finally, in stage 3, we will compare different ways of generating new features from the three models (depicted in Section 3.1.4) together with the best model learned from stage 2, which is AQS. Table 2 summarize the types of feature models we explored in each stage.

Table 2: Feature Representations.

Feature	Abbr.	Construction
Stage 1 Basic	*A(ns)*	Ans Model
	S(tu)	Stu Model
	Q(ues)	Ques Model
Stage 2 Combined	*AS*	A + S
	AQ	A + Q
	AQS	A + Q + S
Stage 3 Composite	*CF1*	*AQS* + SC (Student Clustering)
	CF2	*AQS* + SC + CP(Q,S)
	CF3	*AQS* + SC + CP(Q,SC)

⋆ CP denotes Cartesian Product.

To quantitatively evaluate the effectiveness of different feature models, we train LR and SVM with 10-fold cross-validation (CV). LR is widely adopted as the prediction model in industry for its efficiency and robustness. On the other hand, SVM is one of the most popular classifier due to its effectiveness and the capability to incorporate different kernels. Here we adopt RBF kernel for our SVM models.

5.2 Experiment 2: Six Classifiers

In Experiment 2, we evaluate six classifiers with 10-fold cross-validation using the best feature model from Experiment 1, CF3. The six classifiers are NB, LR, DT, ANN, SVM and DBN. As for the DBN, we build three hidden layers, with 74, 34, 10 hidden units respectively and the learning rate is set to be 0.01.

Among the six classifiers, NB assumes the state features are conditionally independent given the output label while the other models do not have such strong assumption and thus are able to combine multiple features to make predictions. Since there exist latent connections among our extracted features, we expect that NB would perform poorly compared to other models. While all five remaining classifiers can make use of combined features to explore latent connections among features, their approaches are different: LR only linearly combines features; DT synthesizes the features at different branches to make predictions; the hidden layers in ANN and the kernel function of SVM can effectively achieve the non-linear feature mapping; while SVM and ANN utilize the relatively fixed pattern for feature combination, DBN enables the extraction of more representative features via a separate unsupervised pre-training procedure. Although the best model CF3 already contains composite features, we expect the DBN can further leverage the latent connections among features that cannot be manually captured in CF3.

6. RESULTS

Five widely used measures, Accuracy, Area Under the Curve (AUC), Precision, Recall and F-measure are used to evaluate how well various classifiers performed. For precision, recall and F-measure, we treat incorrect answers as the positive class because it is more important for the system to know when the student answer is incorrect.

6.1 Experiment 1: Exploring Feature Space

In the following, we will report our results from each stage listed in Table 2. Given that *A(ns)* (Ans model) is the fundamental model studied in previous research, it will be our baseline model for comparisons across three stages.

6.1.1 Stage 1: Three Basic Models

We first compare Ans, Ques and Stu model separately and Table 3 shows the 10-fold cross-validation results. In Table 3, the best performance of corresponding classifier with respect to each measure is in bold and the best value of each measure is marked *.

Table 3: Performance of Basic Models.

Classifier	Evaluation	*A*	*S*	*Q*
LR	Accuracy	**0.646**	0.616	0.633
	AUC	**0.589**	0.499	0.548
	Precision	**0.564**	0.025	0.425
	Recall	0.342	0.001	**0.548**
	F-measure	0.426	0.002	**0.478**
SVM	Accuracy	**0.728**∗	0.540	0.636
	AUC	**0.654**∗	0.546	0.567
	Precision	**0.830**∗	0.422	0.551
	Recall	0.331	**0.572**∗	0.271
	F-measure	0.474	**0.486**∗	0.364

⋆ The majority class is 0.617.
⋆ '∗' is for the highest value of each measure across all models.

Table 3 shows that all three models beat the majority class baseline (0.617) except for the case of applying SVM on Stu model. As expected, when using either LR or SVM, Ans model outperforms Stu and Ques models on Accuracy, AUC and precision. For the other two measures, Stu model provides the best Recall and F-measure when using SVM and Ques model yields the best Recall and F-measure when using LR. Moreover, when comparing LR and SVM, Table 3 shows that SVM classifier seems to be more effective than LR in that the highest values of five measures are all generated by SVM, marked *. More specifically, for Ans model, SVM outperforms LR on all the measures except Recall; for Stu model, SVM outperforms LR on every measure except for Accuracy; finally, for Ques model, SVM outperforms LR on three out of five measures, the exceptions are recall and F-measure.

Overall, while the Ans model generate the best Accuracy, AUC and Precision, the best Recall and F-measure are generated using either the Ques model for LR or the Stu model for SVM. Therefore, we expect combining the Ques and Stu model with Ans model would result in more effective models.

6.1.2 Stage 2: Three Combined Models

To test the effectiveness of combining multiple features, we show the 10-fold CV performance of A, AQ, AS and AQS by applying LR and SVM respectively in Table 4.

Table 4: Performance of Combined Features.

Classifier	Evaluation	*A*	*AQ*	*AS*	*AQS*
LR	Accuracy	0.646	0.719	0.712	**0.768**
	AUC	0.589	0.696	0.690	**0.753**
	Precision	0.564	0.656	0.663	**0.737**
	Recall	0.342	0.591	0.576	**0.671**∗
	F-measure	0.426	0.621	0.616	**0.703**
SVM	Accuracy	0.728	0.784	0.777	**0.822**∗
	AUC	0.654	0.731	0.733	**0.781**∗
	Precision	0.830	0.880	**0.881**∗	0.876
	Recall	0.331	0.505	0.513	**0.615**
	F-measure	0.474	0.641	0.649	**0.723**∗

Table 5: Performance of Composite Features.

Classifier	Evaluation	*A*	*CF1*	*CF2*	*CF3*
LR	Accuracy	0.646	0.786	0.802	**0.810**
	AUC	0.589	0.769	0.784	**0.794**
	Precision	0.564	0.736	0.764	**0.774**
	Recall	0.342	0.692	0.707	**0.720**
	F-measure	0.426	0.713	0.734	**0.746**
SVM	Accuracy	0.728	0.835	0.830	**0.848**∗
	AUC	0.654	0.799	0.824	**0.850**∗
	Precision	0.830	**0.887**∗	0.778	0.769
	Recall	0.331	0.649	0.795	**0.859**∗
	F-measure	0.473	0.750	0.787	**0.811**∗

⋆ $CF1$ AQS + Student Clustering (SC).
⋆ $CF2$ AQS + SC + Cartesian product(Ques, Stu).
⋆ $CF3$ AQS + SC + Cartesian product(Ques, SC).

It is observed that by adding either Ques or Stu model into Ans model, the effectiveness of resulted models is greatly improved on each of five measures. For example, the Accuracy increases from 0.646 for Ans model to 0.719 for AQ model, and 0.712 for AS model under LR. We can observe the same pattern when SVM is applied. For both LR and SVM classifier, it seems that AQ and AS have comparable performance.

AQS, the combination of all three models, outperforms either AQ or AS for both LR and SVM on all five measures except on Precision by SVM where AS has a slightly higher value (0.881) than AQS (0.876). Therefore, it suggests that Stu and Ques model indeed contribute different information to ASAG task. Similarly, across three models, Table 4 shows that the SVM classifier seems to be more effective than LR in that the best of each of the five measures (those marked *) are generated by SVM except for Recall where the best value 0.671 is generated by LR on AQS model.

6.1.3 Stage 3: Three Composite Models

Given that AQS performs as the best model in Stage 2, we explore whether the effectiveness of classifiers can be further improved by adding composite features. Table 5 shows the performance of CF1, CF2 and CF3.

Tables 4 and 5 show that CF1 is more effective than AQS on every measure when using SVM and on four out of five measures except on Precision using LR. It suggests that the using student clustering can indeed further improve the performance of either LR and SVM.

The improvement from CF1 to CF2 and CF3 mainly stems from the power of Cartesian product. Furthermore, the difference between CF2 and CF3 lies in the different choices of features used for Cartesian product. The result shows that there exists stronger association between the latent student clusters and Ques model than that between Stu model and Ques model. Overall, SVM outperforms LR throughout CF1

to CF3 in that the best of five measures (those marked *) are all generated by SVM in Table 5.

To summarize, the performance of SVM dominates LR when using individual feature models, combined models, and composite models. With only one exception, the best of each of the five measures (those marked *) are all generated by SVM across all three stages. Finally across the nine models, the best model for both LR and SVM is CF3 in that CF3 is more effective than the other eight models on every measures using LR and on four out five measures except on Precision using SVM. Therefore, CF3 is selected for Experiment 2.

6.2 Experiment 2: Six Classifiers

Table 6 shows the performance of the six ML classifiers on CF3: $AQS + \text{SC} + \text{CP(Q,SC)}$ using 10-fold cross-validation. From the results, we draw the first conclusion that NB falls behind other classifiers with a large margin of 18% except on Recall. As expected, LR, DT, ANN, SVM and DBN outperform NB in all the evaluations due to the capacity of combining features and NB's strong independent assumption. Table 6 shows that DBN yields the highest Accuracy, AUC, Precision and F-measure while SVM reaches the best recall value of 0.859 closely followed by DBN. For AUC and F-measure, we have the values in the increasing order for NB, LR, DT, ANN, SVM, and DBN. Overall, our results suggest that DBN performs the best among the six classifiers followed by SVM and NB performs the worst.

Table 6: Comparing the Six Classifiers

Evaluation	NB	LR	DT	ANN	SVM	DBN
Accuracy	0.631	0.810	0.825	0.837	0.848	**0.850***
AUC	0.667	0.794	0.813	0.827	0.850	**0.890***
Precision	0.511	0.774	0.775	0.791	0.769	**0.830***
Recall	0.823	0.720	0.765	0.784	**0.859***	0.838
F-measure	0.631	0.746	0.770	0.787	0.811	**0.834***

7. CONCLUSION

In this paper we tackled the task of ASAG through feature engineering and exploration of better ML approaches such as DBN. For feature engineering, we utilized two other models: Ques and Stu models and explored various combined and composite feature representation. Our results showed that by utilizing the composite features, we obtain an AUC improvement of around 35% and 30% and F-measure improvement of around 75% and 72% on LR and SVM respectively as compared with using Answer-based features only. The comparisons among different classification models shows that DBN outperforms all other methods on Accuracy, AUC, Precision and F-measure. On Recall, DBN performs slightly worse than SVM. Furthermore, the experiment has led to some interesting observations: (1) The clustering of student, as a more compact representation, leads to more discriminative features when combined with question features using Cartesian product. (2) While SVM results in better Accuracy, the composite feature representation brings less improvement on SVM than LR probably because we used RBF kernel in our SVM models which allows the classifier to operate in an infinite-dimension of feature space.

8. ACKNOWLEDGMENTS

This research was supported by the NSF Grant 1432156 "Educational Data Mining for Individualized Instruction in STEM Learning Environments".

9. REFERENCES

[1] T. Barnes. The q-matrix method: Mining student response data for knowledge. 2005.

[2] S. Basu, C. Jacobs, and L. Vanderwende. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, 2013.

[3] J. E. Beck and B. P. Woolf. Using a learning agent with a student model.

[4] Y. Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2009.

[5] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 2007.

[6] N. O. Bernsen and L. Dybkjaer. *Designing Interactive Speech Systems: From First Ideas to User Testing*. Springer-Verlag New York, Inc., 1997.

[7] S. Burrows, I. Gurevych, and B. Stein. The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, 2015.

[8] R. Carolyn. Tools for authoring a dialogue agent that participates in learning studies. *Artificial Intelligence in Education: Building Technology Rich Learning Contexts that Work*, 158:43, 2007.

[9] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction*, 2011.

[10] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 1994.

[11] S. K. D'Mello, B. Lehman, and A. Graesser. A motivationally supportive affect-sensitive autotutor. In *New perspectives on affect and learning technologies*. Springer, 2011.

[12] M. O. Dzikovska, A. Isard, P. Bell, J. D. Moore, N. Steinhauser, and G. Campbell. Beetle ii: an adaptable tutorial dialogue system. In *Proceedings of the SIGDIAL 2011 Conference*, pages 338–340. Association for Computational Linguistics, 2011.

[13] D. Higgins, J. Burstein, D. Marcu, and C. Gentile. Evaluating multiple aspects of coherence in student essays. In *HLT-NAACL*, 2004.

[14] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 2002.

[15] G. B. Huang, H. Lee, and E. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *CVPR*, 2012.

[16] C. Leacock and M. Chodorow. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 2003.

[17] A.-r. Mohamed, G. Dahl, and G. Hinton. Deep belief networks for phone recognition. In *NIPs*, 2009.

[18] M. Mohler and R. Mihalcea. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th EACL*, 2009.

[19] S. G. Pulman and J. Z. Sukkarieh. Automatic short answer marking. In *Proceedings of the second workshop on Building Educational Applications Using NLP*.

[20] K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement.*, 1983.

[21] K. VanLehn, P. W. Jordan, and D. J. Litman. Developing pedagogically effective tutorial dialogue tactics: experiments and a testbed. In *SLaTE*. Citeseer, 2007.