A Memory-Augmented Neural Model for Automated Grading

Siyuan Zhao
Worcester Polytechnic Institute
Worcester, MA 01609, USA
szhao@wpi.edu

Yaqiong Zhang
Worcester Polytechnic Institute
Worcester, MA 01609, USA
yzhang19@wpi.edu

Xiaolu Xiong
Worcester Polytechnic Institute
Worcester, MA 01609, USA
xxiong@wpi.edu

Anthony Botelho
Worcester Polytechnic Institute
Worcester, MA 01609, USA
abotelho@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
Worcester, MA 01609, USA
nth@wpi.edu

ABSTRACT

The need for automated grading tools for essay writing and open-ended assignments has received increasing attention due to the unprecedented scale of Massive Online Courses (MOOCs) and the fact that more and more students are relying on computers to complete and submit their school work. In this paper, we propose an efficient memory networks-powered automated grading model. The idea of our model stems from the philosophy that with enough graded samples for each score in the rubric, such samples can be used to grade future work that is found to be similar. For each possible score in the rubric, a student response graded with the same score is collected. These selected responses represent the grading criteria specified in the rubric and are stored in the memory component. Our model learns to predict a score for an ungraded response by computing the relevance between the ungraded response and each selected response in memory. The evaluation was conducted on the Kaggle Automated Student Assessment Prize (ASAP) dataset. The results show that our model achieves state-of-the-art performance in 7 out of 8 essay sets.

Author Keywords

Automated grading; neural networks; memory networks; word embeddings; natural language processing

INTRODUCTION

Automated grading is a critical part of Massive Open Online Courses (MOOCs) system and any intelligent tutoring systems (ITS) at scale. Some standard tests, such as Test of English as a Foreign Language (TOEFL) and Graduate Record Examination (GRE), assess student writing skills. Manually grading these essay will be time-consuming. Moreover, as massive open online courses (MOOCs) become widespread and the number of students enrolled in one course increases, the need for grading and providing feedback on written assignments are ever critical.

As part of the automated grading system, AES has employed numerous efforts to improving its performance. AES uses statistical and Natural Language Processing (NLP) techniques to automatically predict a score for an essay based on the essay prompt and rubric. Most existing AES systems are built on the basis of predefined features, e.g. number of words, average word length, and number of spelling errors, and a machine learning algorithm [1]. It is normally a heavy burden to find out effective features for AES. Moreover, the performance of the AES systems is constrained by the effectiveness of the predefined features. Recently another kind of approach has emerged, employing neural network models to learn the features automatically in an end-to-end manner [5]. By this means, a direct prediction of essay scores can be achieved without performing any feature extraction. The model based on long short-term memory (LSTM) networks in [5] has demonstrated promise in accomplishing multiple types of automated grading tasks.

Memory Networks (MN) [6, 4] have been recently introduced to deal with complex reasoning and inferencing NLP tasks and have been shown to outperform LSTM on some complex reasoning tasks [4]. MN is a class of models which contains an external scalable memory and a controller to read from and write to that memory.

To our knowledge, no study has been conducted to investigate the feasibility and effectiveness of MN applied in automated grading tasks. In this study, we develop a generic model for such tasks using Memory Networks inspired by their capability to store rich representations of data and reason over that data in memory. For each essay score, we select one essay exhibiting the same score from student responses as a sample for that grade. All collected sample responses are loaded into the memory of the model. The model is trained with the rest of student responses in a supervised learning manner on these

L@S 2017, April 20–21, 2017, Cambridge, MA, USA.

http://dx.doi.org/10.1145/3051457.3053982

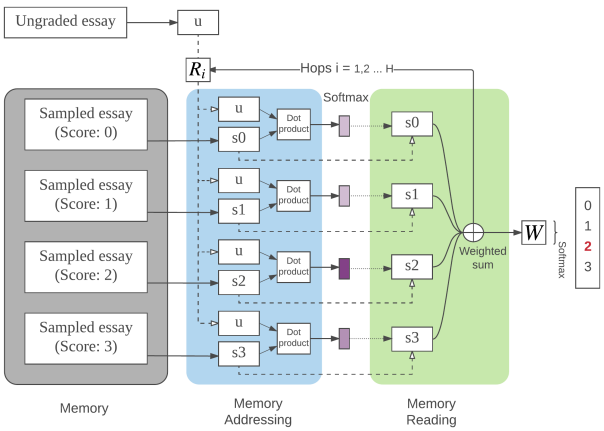

Figure 1. An illustration of memory networks for AES. The score range is 0 - 3. For each score, only one sample with the same score is selected from student responses. There are 4 samples in total in memory. Input representation layer is not included.

data to compute the relevance between the representation of an ungraded response and that of each sample. The intuition is that as a part of a scoring rubric, a number of sample responses of variable quality are usually provided to students and graders to help them better understand the rubric. These collected responses are characterized with expectations of quality described in the rubric. The model is expected to learn the grading criteria from these responses. We evaluate our model on a publicly available essay grading data set from the Kaggle Automated Student Assessment Prize (ASAP) competition (https://www.kaggle.com/c/asap-aes). Our experiments show that our model achieves state-of-the-art results on this dataset.

MODEL

An illustration of our model is given in Figure 1, which is inspired by the work of memory networks applied in question answering [4]. Our model consists of four layers: input representation layer, memory addressing layer, memory reading layer, and output layer. Input representation layer is responsible for generating a vector representation for a student response. Memory addressing layer loads selected samples of student responses to memory, and assigns a weight to each memory piece. Afterward memory reading layer gathers the content from memory by taking weighted sum of each memory piece based on the weights calculated from previous layer, and produces a resulting state. Finally the output layer makes the prediction on the basis of the resulting state. Neural networks models are usually featured with multiple computational layers to learn a more abstract representation of the input. Our model is extended to have the structure of multiple layers (hops) by stacking memory addressing layer and memory reading layer repeatedly.

Input Representation

Each student response is represented as a vector in our model. Given a student response $x = \{x_1, x_2, x_3, ..., x_n\}$, where n is the length of the response, we map each word into a word vector $w_i = Wx_i$. All word vectors come from a word embedding matrix $W \in R^{d \times V}$, where d is the dimension of word vector and V is the vocabulary size. To represent an essay in a vector, we selected position encoding (PE) described in [4]. By the scheme of PE, the vector representation of a response is calculated by $m = \sum_j l_j \cdot Wx_{ij}$, where $\cdot$ is an element-wise multiplication. l_j is a column vector with the structure $l_{kj} = (1 - j/J) - (k/d)(1 - 2j/J)$ (assuming 1-based indexing), where J is the total number of words in the response, d is the dimension of word vector, and k is the embedding index. PE is a simple and efficient way to represent a response, and does not need to learn extra parameters.

Memory Addressing

After generating the representation of the responses, we select a sample from student response for every possible score, which is graded with the same score. The selected samples work as a representation of the criteria in the rubric for all possible scores. Expert knowledge can be used here to choose most representative sample for each score or even generate a number of ideal samples. For our experiment, we randomly pick a sample from student responses for each score, which is graded with that score.

All sampled responses are loaded into the memory as an array of vectors $m_1, m_2, ..., m_h$, where h is the total number of sampled essays. An ungraded response is denoted as x. The basic idea of memory addressing is that it assigns a weight/importance to each sampled response m_i by calculating a dot product between x and m_i followed by a softmax.

$$p_i = Softmax(xA^T \cdot m_i B^T) \quad (1)$$

where $Softmax(y_i) = e^{y_i} / \sum_j e^{y_j}$, A is a $k \times d$ matrix and so is B. Defined in this way p is a weight vector over all sampled responses. A and B are learned matrices used to transfer the response representation to a d-dimensional features space.

Memory Reading

After weight vector p is calculated, the output of the memory is computed as a weighted sum of each piece of memory in m:

$$o = \sum_i p_i m_i C^T \quad (2)$$

where C is a $k \times d$ matrix used to transfer the response representation to the feature space. The $k \times d$ matrix C may be identical to A, but from our experiment, we found that training a separate C leads to a better performance. From the equation, we can see that weight vector p controls the amount of content that is read from each memory piece.

Multiple Hops

The success of neural networks is due to its ability of learning multiple layers of neurons and each layer can transform the representation at previous level into a higher level of abstract representation. Inspired by this idea, we stack multiple memory addressing steps and memory reading steps together to handle multiple hops operations.

Set	# Essays	Avg len	Max len	Min score	Max score
1	1,783	350	911	2	12
2	1,800	350	118	1	6
3	1,726	150	395	0	3
4	1,772	150	383	0	3
5	1,805	150	452	0	4
6	1,800	150	489	0	4
7	1,569	250	659	0	30
8	723	650	983	0	60

Table 1. Selected Details of ASAP dataset

After receiving the output o from equation 2, the ungraded response u is updated with:

$$u_2 = Relu(R_1(u+o)) \tag{3}$$

where R_1 is a $k \times k$ matrix, $u = xA^T$ and $Relu(y) = max(0,y)$. Then memory addressing step and reading memory step are repeated, using a different matrix R_j on each hop j. The memory addressing step is modified accordingly to use the updated representation of the ungraded response.

$$p_i = Softmax(u_j \cdot m_i B) \tag{4}$$

Output Layer

After a fixed number H hops, the resulting state u_H is used to predict a final score over the possible scores:

$$\hat{s} = Softmax(u_H W + b) \tag{5}$$

where W is $k \times r$ matrix, r is the number of possible scores and b is the bias value. Note that the number of output nodes equals to the length of score range. We calculate a distribution over all possible scores and select most probable score as the prediction. The matrices A, B, C, W and $R_1, ..., R_H$ are learned through backpropagation and stochastic gradient descent by minimizing a standard cross entropy loss between the predicted score $\hat{s}$ and the actual score s.

EXPERIMENTAL SETUP

Dataset

Dataset used in this study comes from Kaggle Automated Student Assessment Prize (ASAP) competition sponsored by William and Flora Hewlett Foundation (Hewlett). There are 8 sets of essays and each set is generated from a single prompt. All responses collected in the dataset were written by students ranging from grade 7 to grade 10. Score range varies on essay sets. All essays were graded by at least 2 human graders. The average length of the essays differs for each essay set, ranging from 150 words to 650 words. Selected details for each essay set is shown in Table 1.

Evaluation Metric

Quadratic weighted Kappa (QWK) is used to measure the agreement between the human grader and the model. We choose to use this metric because it is the official evaluation metric of the ASAP competition. Other work such as [1, 5, 3] that uses the ASAP dataset also uses this evaluation metric.

QWK is calculated using

$$k = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \tag{6}$$

where matrices O, w and E are the matrices of observed scores, weights, and expected scores respectively. Matrix $O_{i,j}$ corresponds to the number of student responses that receive a score i by the first grader and a score j by the second grader (the model in our experiment). The weight matrix are $w_{i,j} = (i-j)^2/(N-1)^2$, where N is the number of possible scores. Matrix E is calculated by taking the outer product between the score vectors of the two graders, which are then normalized to have the same sum as O.

Implementation Details

We used the publicly available pre-trained Glove word embeddings [2], which was trained on 42 billion tokens of web data, from Common Crawl (http://commoncrawl.org/). The dimension of each word vector is 300.

5-fold cross validation was used to evaluate our model. For each fold, the data was split into two parts: 80% of the data as the training data and 20% as the testing data. The sampled response for each score is randomly selected from the training data. A model was trained on each essay set due to the fact that score range varies among 8 essay sets.

Baselines

Similarly to [5], our model is compared with Enhanced AI Scoring Engine (EASE), an open-source AES system, to demonstrate the improvements on performance. The reason we use this system as baseline is that it achieved best QWK scores among all open-source systems participated in ASAP competition. [3] described a set of reliable features and reported the results of two models using these features: support vector regression (SVR) and Bayesian linear ridge regression (BLRR).

[5] examined several neural networks models, e.g. RNN and Convolutional Neural Networks (CNN), on ASAP dataset. We also compared our model with their models

To verify the efficacy of GloVe word embeddings and external memory, we developed a simple multi-layer forward neural networks (FNN) model, which is similar to our model with respect to the model structure, but without an external memory. We refer this model as FNN for the rest of paper for convenience. As shown in Figure 2, each word of a student response is first converted to a continuous vector using GloVe word embeddings and the vector representation for the response is obtained by applying PE on all word vectors. Afterward the representation is fed into 4 hidden layers, each of which has 100 hidden nodes. FNN is properly defined by the equations below:

$$h_0 = Relu(A^T x) \tag{7}$$

$$h_i = Relu(R_i h_{i-1}),\ for\ i \geq 1 \tag{8}$$

$$\hat{s} = Softmax(h_H W) \tag{9}$$

where x is the representation generated by GloVe with PE for a student response. h_i is the output of hidden layer i. H is

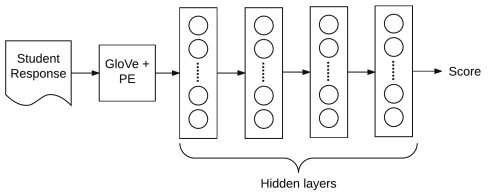

Figure 2. An illustration of baseline FNN. Use GloVe with PE to represent a student response. The representation is fed into 4-layer networks and each layer has 100 hidden nodes.

Set	MN	FNN	EASE	LSTM
1	**0.83**	0.75	0.76	0.78
2	**0.72**	0.7	0.61	0.69
3	**0.72**	0.7	0.62	0.68
4	**0.82**	0.8	0.74	0.8
5	**0.83**	0.8	0.78	0.82
6	**0.83**	0.79	0.78	0.81
7	0.79	0.73	0.73	**0.81**
8	**0.68**	0.63	0.62	0.59
Avg	**0.78**	0.74	0.71	0.75

Table 2. QWK scores on ASAP dataset.

the total number of hidden layers. A, R_i ,and W are weight matrices. The bias vectors are omitted in the equations.

RESULTS

In this section, we describe the results of our experiments on the ASAP dataset and compare these results with baselines mentioned above. Column MN of Table 2 presents the QWK scores of our model. Column EASE contains the results from EASE. We also compare our model to other neural models in [5] and pick the best performance achieved by a single model (LSTM) from their paper. The results are listed in Column LSTM of Table 2.

As indicated in Table 2, our model outperforms in 7 out of 8 sets (except for set 7) and improves the average QWK score by 4.0% compared to the baseline LSTM. As expected, our model surpasses EASE in all 8 sets and improves average QWK score by 10%.

The results from the FNN model mentioned above is presented in column FNN of Table 2. When comparing these results to the best results from EASE, we find that this basic model outperforms EASE in 7 out of 8 sets of essays (except for essay set 1) and is even comparable with the complex model (LSTM). This proves that using Glove word embeddings with PE to represent a student response is able to capture important features useful for grading the response. The effectiveness of the external memory is demonstrated by the fact that MN accomplishes better performance on 7 sets (set 4 is equal) than FNN does.

DISCUSSION AND CONCLUSION

In this study, we develop a generic model for automated grading tasks using memory networks. To our best knowledge this is the first study that memory networks are applied for this kind of task. Our model is tested on ASAP dataset and achieves state-of-the-art performance in 7 out of 8 essay sets.

Our model can be generalized to automatically grade assignments from other subjects. As shown above, there are two key factors to the performance: reliable representation and memory component. In order to apply our model to other kinds of assignment, learning a good vector representation for the assignment is the first step. The next step is to select characterized samples and store these samples to memory. The purpose of this step is to teach the model to understand the grading strategy and eventually associate a vector representation to a score.

However, we only test our model on one dataset. There is a need to explore our model with more datasets that contain various formats of assignments. Furthermore, the representation of the assignment and the mechanism for measuring relevance among assignments is still elementary. Future work should therefore focus on these two areas to improve the generalizability of the model.

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.

REFERENCES

1. Hongbo Chen and Ben He. 2013. Automated Essay Scoring by Maximizing Human-Machine Agreement. In *EMNLP*.
2. Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*, Vol. 14. 1532–1543.
3. Peter Phandi, Kian Ming Adam Chai, and Hwee Tou Ng. 2015. Flexible Domain Adaptation for Automated Essay Scoring Using Correlated Linear Regression. In *EMNLP*.
4. Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems 28*, C Cortes, N D Lawrence, D D Lee, M Sugiyama, and R Garnett (Eds.). Curran Associates, Inc., 2440–2448.
5. Kaveh Taghipour and Hwee Tou Ng. 2016. A Neural Approach to Automated Essay Scoring. In *EMNLP*.
6. Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory Networks. *CoRR* abs/1410.3916 (2014).