Automated Essay Grading With Neural Networks

Ye Tian
Georgia Institute of Technology
Atlanta, GA, US
ytian307@gatech.edu

ABSTRACT
The previous or existing automated essay grading systems used traditional text based machine learning models. The results highly rely on the crafted extracted features. The performances are unstable when grading different essays from various topics. In this paper, I propose a deep learning based (recurrent neural networks) method to grading essays automatically without any feature engineering. By training multiple recurrent neural networks models, the system can perform a better result compare to traditional methods. The system designed to be user-friendly, after input the required selection and paste the essay in text-box, the scoring will show up with three optional different scoring ranges.

Author Keywords
Automated Essay Grading; Deep Learning; Machine Learning; Neural Networks; LSTM; Tokenization; Name Entity Recognition;

ACM Classification Keywords
I.2.1 Artificial Intelligence: Applications and Expert Systems - Natural language Interfaces

INTRODUCTION
Grading essays is a prolonged and laborious process for the schools that only have a few raters with the same domain knowledge. As a teacher, spend many hours on grading essays could potentially affect the teaching preparation or other interacting activities with students. Different raters are also with various personal opinions and various objective bias. The manually grading process might also be influenced by the above reasons even with the same grading metrics. Several training schools are also providing the service of evaluating student's writing skills by grading their essays. With a large amount of essays, grading essays become very time-consuming and almost impossible by human raters. If there exist an automated grading system can reduce the bias and minimize the cost. Schools, organizations, teachers can all benefit from it. The grading standard can also be unified among different schools. This grading tool can potentially become more accurate after receiving more and more essays from various sources.

In general, automated essay grading system refers to a tool that can grade an essay automatically based on one or more pre-trained machine learning or deep learning models. With the same essay, the system should return the same numeric scoring by reflecting the content and quality. Traditional machine learning models highly rely on the designed features, and to find the most useful features is very difficult. With that desire, I propose a deep learning based model to replace previous. Along with the development of deep learning area, neural networks can now provide a more accurate prediction on a larger dataset with more complex patterns. Specifically, the recurrent neural networks can perform a very decent result for the text input data (essays) since it can capture the order of the sentences. The details of implementation will be revealed in the rest of the paper.

Overall, the intention of this project is to apply the latest technologies to the automated essay grading system and provide a user-friendly interface for all users.

In this paper, the order of each section is as follows: the dataset itself, pre-processing, neural networks models, user interface, and some experiments discussions.

PROBLEM STATEMENT
The aim is to build an automated essay grading system with an enhanced neural network model. The system should automatically generate a score in a specific range for any given essays written by the students in grade 7 to grade 10. The neural network model should be improved over time by comparing the automated generated score with the human assigned standard score.

DATASET
The dataset I used for training models is provided by Hewlett Foundations in the Automated Student Assessment Prize Kaggle competition https://www.kaggle.com/c/asap-aes/data. There are totally 8 different sets of essays, each set was generated from a single prompt. Each essay was written in English by a student from middle school. Selected essays range from an average length of 0 to 1200 words. The essay type is known and the exist scores generated by multiple raters from the same domain. Each essay has been manually graded by at least two or three different raters. Scoring mechanism is in below table.

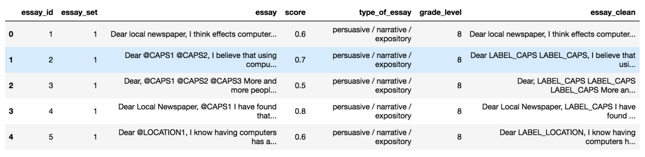
Prompt	#Essays	Avg length	Scores
1	1,783	350	2–12
2	1,800	350	1–6
3	1,726	150	0–3
4	1,772	150	0–3
5	1,805	150	0–4
6	1,800	150	0–4
7	1,569	250	0–30
8	723	650	0–60

Table 1. Essay Sets

Since only the training set and validation set is publicly available, I used the combination of two datasets as my entire essay set. I split the data into the training set, validation set, and testing set follows a proportion of 6:2:2.

DATA PROCESSING

The raw dataset stored in tsv format, each row represents one single data point. The columns include the essay id, essay set id, the essay text, score, and essay type. Figure 1 shows five samples of the data. Since in different essay set, the range of score is different, I converted all essay into the same score range at the beginning. In most of the essays, the personally identifying information replaced by the Named Entity Recognizer from the Standford Natural Language Processing group and a variety of other approaches. The relevant entities are identified in the text and replaced with a string such as "@PERSON1". To better fit the data into the model, I added one more step to replace the string with an underline. The tokenization part will benefit by not splitting different words by the @ signal.



	essay_id	essay_set	essay	score	type_of_essay	grade_level	essay_clean
0	1	1	Dear local newspaper, I think effects computer...	0.6	persuasive / narrative / expository	8	Dear local newspaper, I think effects computer...
1	2	1	Dear @CAPS1 @CAPS2, I believe that using compu...	0.7	persuasive / narrative / expository	8	Dear LABEL_CAPS LABEL_CAPS, I believe that usi...
2	3	1	Dear, @CAPS1 @CAPS2 @CAPS3 More and more peopl...	0.5	persuasive / narrative / expository	8	Dear, LABEL_CAPS LABEL_CAPS LABEL_CAPS More an...
3	4	1	Dear Local Newspaper, @CAPS1 I have found that...	0.8	persuasive / narrative / expository	8	Dear Local Newspaper, LABEL_CAPS I have found ...
4	5	1	Dear @LOCATION1, I know having computers has a...	0.6	persuasive / narrative / expository	8	Dear LABEL_LOCATION, I know having computers h...

Figure 1. Dataset

Below figure shows the word length distribution among all essays. Most of the essays have the word length between 100 words and 400 words. The longest is 1118 words. The shortest is 1 word. In all 12977 essays, the average length of words is 226 words.

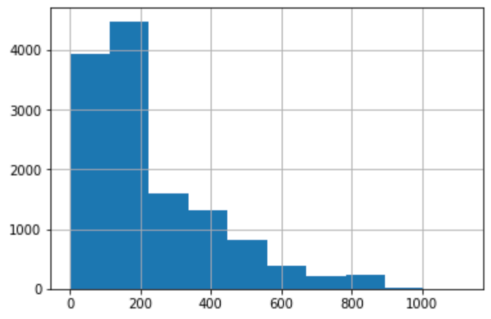
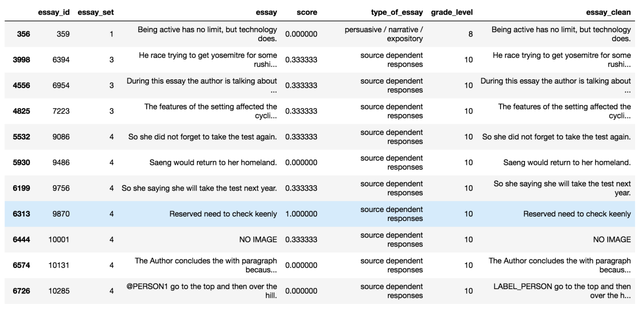


Figure 2. Word Length Distribution

The Figure 3 shows some essays with less than 10 words. The model filters out those records as outlier during the training phase.



	essay_id	essay_set	essay	score	type_of_essay	grade_level	essay_clean
356	359	1	Being active has no limit, but technology does.	0.000000	persuasive / narrative / expository	8	Being active has no limit, but technology does.
3998	6394	3	He race trying to get yosemitre for some rushi...	0.333333	source dependent responses	10	He race trying to get yosemitre for some rushi...
4556	6954	3	During this essay the author is talking about ...	0.333333	source dependent responses	10	During this essay the author is talking about ...
4825	7223	3	The features of the setting affected the cycli...	0.333333	source dependent responses	10	The features of the setting affected the cycli...
5532	9086	4	So she did not forget to take the test again.	0.333333	source dependent responses	10	So she did not forget to take the test again.
5930	9486	4	Saeng would return to her homeland.	0.000000	source dependent responses	10	Saeng would return to her homeland.
6199	9756	4	So she saying she will take the test next year.	0.333333	source dependent responses	10	So she saying she will take the test next year.
6313	9870	4	Reserved need to check keenly	1.000000	source dependent responses	10	Reserved need to check keenly
6444	10001	4	NO IMAGE	0.333333	source dependent responses	10	NO IMAGE
6574	10131	4	The Author concludes the with paragraph becaus...	0.000000	source dependent responses	10	The Author concludes the with paragraph becaus...
6726	10285	4	@PERSON1 go to the top and then over the hill.	0.000000	source dependent responses	10	LABEL_PERSON go to the top and then over the h...

Figure 3. Outlier

DEEP LEARNING APPROACH

Feed-forward Neural Networks

The feed-forward neural network is the foundation of all other types of neural networks. Despite the name of this methodology is very artificial intelligent, the algorithm is not working with the same principle as human-brain neural networks. The feed-forward neural networks can be classified as a non-linear statistical method for either classification or regression task. Below Figure 1 represents a very basic three layers feed-forward neural network.

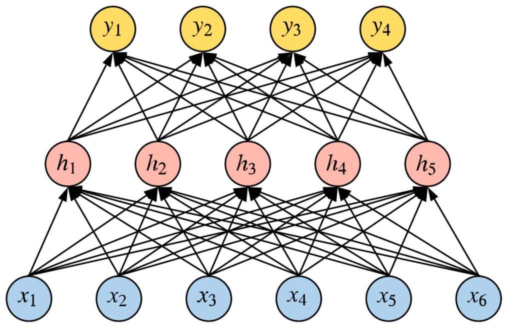


Figure 4. Feed-forward Neural Network

Each circle represents one neuron in the corresponded layer. The data flow follows the arrow direction, both incoming and outgoing. In each arrow, it carries a numeric weight links two neurons between two different layers. The bottom layer represents the input layer, each character from every essay passed from here as input data. One-hot encoding is required for sparse data such as an essay. In the middle is the hidden layer. Every neural network can have as many as hidden layers and each neuron received weight from every input data point. The top layer is the fully connected layer, the size determined by the number of target of the model. Since the arrows always go upwards, there are no links between any neurons at the same layer. Although my model for this project primarily used recurrent neural network, the fully connection layer will be the same. The matrix operation transformed the data from highdimensions to low-dimensions. Here the activation function will assign a probability as the corresponding score.

Recurrent Neural Networks

Since we treat the textual data as a single vector, a regular single feed-forwarding neural network might not perform very well. Such a model actually doesn't have any memory of any previous ingested data point. Each data point processed independently and no relationships. This is not a good way to automated grading essays. Plus the order of words is very important.

Recurrent neural networks can behave somehow like a human when reading a text by iterating through each word and considering all relative information from previous. In practice, if we implement an internal loop and feeding the previous element into current layer, each step will be viewed by the next layer.

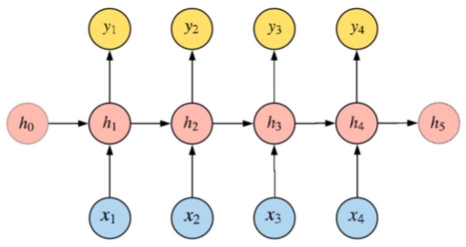


Figure 5. Recurrent Neural Network

From above figure, the x represents input word and y represents the predicted word. In the hidden layer, each h holding all information from the previous layer, which means the latest output actually contains information from all inputs. Since there is no data before x1 and h0, this initial state typically set to zero.

In theory, each h holds all information from previous timestamps. However, in practice, the model couldn't handle the long essays since the distance is too far away. This issue is referred as the vanishing gradient problem. Basically, this problem occurs because the neural network propagated over so many layers and the information tends to vanish. Instead of holding all words, the solution is using long short-term memory network. The LSTM actually computed by setup some gate architectures.

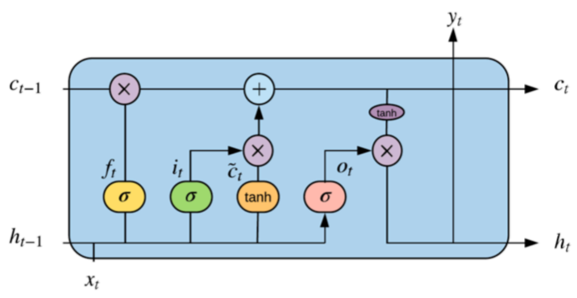


Figure 6. Long Short-Term Memory

In above figure, it shows how the LSTM takes steps to create new state using the old state and current input. LSTM splits the state vector into two components: state component h and memory component c. Other gates referred to input gate i, forget gate f, output gate o. The forget gate decides how much of previous memory should be used in the computation. The input gate determines how much input x and previous state should be preserved. The new memory cell c is produced by using the results from three other gates. Finally, the new state created.

LSTM can help us to produce a better result from the sequential textual data and skip the problem of vanishing gradient. A regular recurrent neural network model does not change the interpretation of previous words. But as human, we constantly changing the interpretation based on the next sentence. One of the goals is also to make our automated essay grading system can grading essays like a human: reading essays and understanding it.

To summarize, I used LSTM layer in order to make a better prediction. Hopefully, the automated essay grading system can identify the topic of the essay and make accurate predictions among all essays.

Over-fitting

In a multi-layer neural network, the number of parameters is huge. The neural networks can learn any information from the training data and apply it to various tasks in both classification and regression. On the other hand, this kind of neural networks with many parameters could potentially over-fit, or fail to make generalizations of the result. When the model gets too attached to the training data but the training data doesn't cover all different kind of data point, the over-fitting will happens. This issue has been researched from many different areas not limited to neural networks. With the limited number of essays, my neural networks will have to with some level of over-fitting. The model could potentially mimicking part of essays and retrieve important information as features and assign scores to other unseen essays.

To solve this problem, I used a method named dropout. This works by randomly dropping some of the neurons on each layer to avoid the neural network only rely on some specific neurons significantly. Therefore, forcing the neural network to not only using any certain weights. The dropout implemented by applying a probability mask on each neuron and determines whether the parameters will be used.

Loss Function

Since the objective of my automated essay grading system is to create a model that can map all essays to the correct correspondent scores. A function is required to measure the loss suffered when comparing the predicted score to the original human assigned scores. This is so-called loss function. The training process is equal the process of minimize the loss function by updating the value of all model parameters. The categorical cross-entropy loss is one of the most popular loss function:

$$L_{cross-entropy}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{i} \boldsymbol{y}_i \log(\hat{\boldsymbol{y}}_i).$$

It follows the goal: minimize the cross-entropy loss. In fact, minimize the negative log likelihood is equal to maximizing the log likelihood of the model.

Model Training

Minimize the loss function can often be solved by using some gradient-based method, such as gradient descent. The major difference between the traditional linear models and neural networks is that the nonlinearity. Most neural networks make loss functions non-convex. To guarantee the global optimum in loss function, the initial weights of all neurons need to be set up correctly.

Using a regular gradient descent might lead to a very slow computation period since the calculation is based on the entire training dataset. To shrink the computation time, I used a more effective method called stochastic gradient descent (SGD). The principle of SGD is similar to the standard gradient descent but only use a subset from the entire training dataset at every iteration. This makes the algorithm more efficient. After each epoch is completed, the model starts over a new training iteration with a new subset. The number of iteration is also one of the hyper-parameter which required to define at the beginning. During the SGD process, the learning rate is also pre-defined as a scalar to determine how quickly the model will update the parameters.

Besides the basic SGD, other optimization algorithms could also potentially speed up the computation time and produce a better performance. One of the algorithms is adam optimizer. Also known as adaptive learning rate algorithm. Since all hyper-parameter values require pre-defined, I use grad search approach to find the best combination and apply it into the final model.

Word Embedding

Since our training is all textual data, one major concern is how to transform the data into feature space. Either one-hot vectors or embedding vectors are commonly used. There are several successful methods for word embedding which do not require the data with a label. The pre-trained word embedding learned from an implemented skip-gram approach in Word2Vec. The goal of skip-gram model is to predict the context surrounding a given word. The output model will return a probability for each unique word in the vocabulary dictionary of being the nearby word. By ingesting the word pairs, the number of surrounding words with a fixed size can represent as a sequence of words. Word pairs are extracted from the sentences and fed into the neural network models as a data neuron. After the training stage is completed, inputting again the same word could receive get a higher probability of a similar word. Map the embedding to the words in student essays is necessary. The pre-trained package usually covers a majority of the words in the training data in English.

To make the automated essay grading system be able to understand and optimize the grading process, the embedding layer is very important and necessary. The first layer should be the embedding layer. In the later experiments, the paper will show the difference between embedding models and non-embedding models.

SYSTEM ARCHITECTURE

Below figure represents the automated essay grading system architecture. The data-flow will start from the user side. After the user pasted their essay in text box and selected the correspondent information. The text essay data and parameters will be delivered to the server side (model side). The submit button will trigger the main function. Since the raw data has noise. The pre-processing section will clean up the essay with a better format and pass essay into a pre-trained model, the model will fit the input text data and return a numeric result as a predicted score. At each time, except return score as result, this system will also copy the essay into a document-oriented database and update current model. Other analysis can be performed at this phase from output file.

To improve the computing efficiently, the training process will only occur by a manually commend, but existing model parameters can always provide a predicted score.

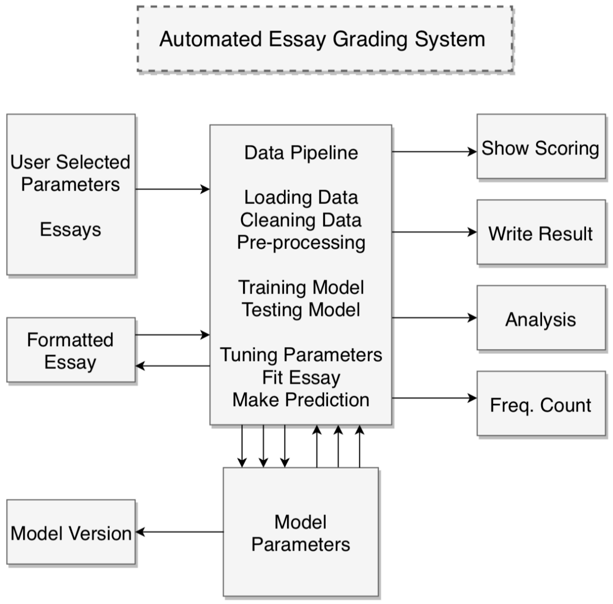
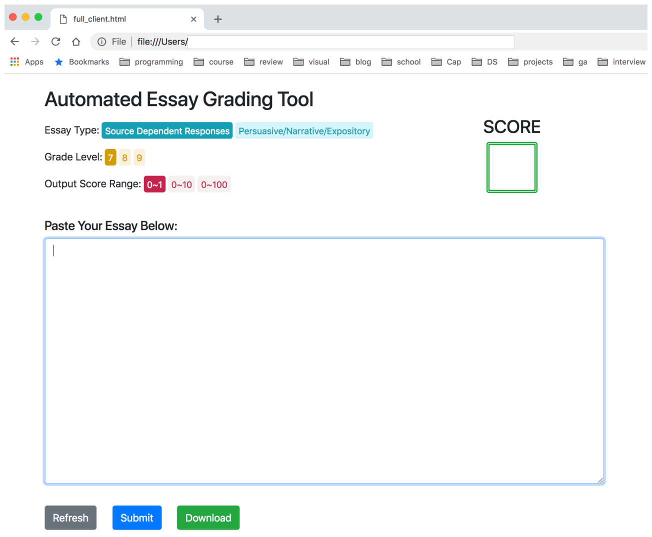


Figure 7. System Architecture

USER INTERFACE

The user interface was designed to be clean and concise. Any user can get a predicted score directly from this tool without any background knowledge of how the deep learning model works. On the webpage, there are three required inputs listed: essay type, grade level, and output scoring range. The current pre-trained neural network models can make a prediction based on different type of essays: source dependent responses, persuasive, narrative, or expository. The grade level will also be counted as the input data and fit into the deep learning model, ideally, the same essay should get different scores with different grade level as input. The result of essay grading will be a numeric score between 0 and 1. This result can be converted into different range like zero to ten or zero to one hundred. The final score should show up in the score box with less than one second response time after the user pastes their essay into an essay input box.



Automated Essay Grading Tool

Essay Type: Source Dependent Responses Persuasive/Narrative/Expository

Grade Level: 7 8 9

Output Score Range: 0~1 0~10 0~100

SCORE

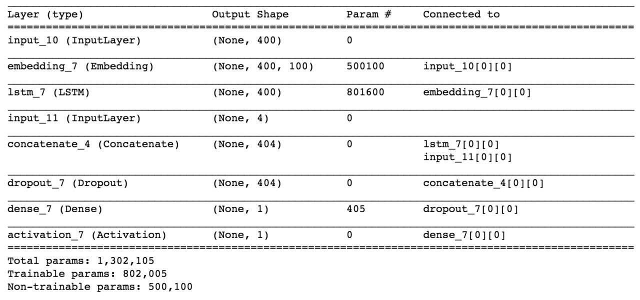
Paste Your Essay Below:

Figure 8. User Interface

The three buttons provide three functions for user: refresh the page, submit their essay, or download their result. The refresh button and submit button always available to use. The download button will only available after the model returns a predicted score. After getting the essay score, the user can download their result as a txt format output file. This can be used for analysis purpose or storage purpose.

EXPERIMENT

This part will present the results of four different neural network architectures that have been trained. As I mentioned previously, embedding layer is one of the hyper-parameters that need to be experimented. Below figure is one of the neural network architecture:



```
Layer (type)                     Output Shape          Param #     Connected to
==================================================================================================
input_10 (InputLayer)            (None, 400)           0
__________________________________________________________________________________________________
embedding_7 (Embedding)          (None, 400, 100)      500100      input_10[0][0]
__________________________________________________________________________________________________
lstm_7 (LSTM)                    (None, 400)           801600      embedding_7[0][0]
__________________________________________________________________________________________________
input_11 (InputLayer)            (None, 4)             0
__________________________________________________________________________________________________
concatenate_4 (Concatenate)      (None, 404)           0           lstm_7[0][0]
                                                                   input_11[0][0]
__________________________________________________________________________________________________
dropout_7 (Dropout)              (None, 404)           0           concatenate_4[0][0]
__________________________________________________________________________________________________
dense_7 (Dense)                  (None, 1)             405         dropout_7[0][0]
__________________________________________________________________________________________________
activation_7 (Activation)        (None, 1)             0           dense_7[0][0]
==================================================================================================
Total params: 1,302,105
Trainable params: 802,005
Non-trainable params: 500,100
```

Table 2. Neural Network Architecture

The first layer is the input layer that ingests the cleaned essay text data. The second layer is the embedding layer that processes the textual data with pre-trained embedding model. The third layer is the long short-term memory layer with all weighted parameters. The fourth layer is adding other input parameters like grade level and essay type. The fifth layer connects all information together and trains data as a whole. The dropout layer and activation layer was added at last. Other commonly used hyper-parameters are:

- The number of epoch: 100
- Batch size: 2048
- Validation and testing percentage: 0.2
- Function used for avoid over-fitting: dropout
- Activation function: sigmoid function
- Loss function: cross-entropy loss and mean square error quadratic loss.
- Optimize function: adam
- Result measurement: mean absolute error

RESULTS

Since the mean absolute error is used to measure the model performance and the loss function is mean square error quadratic loss. The loss function trending can also represent the model over time performance.

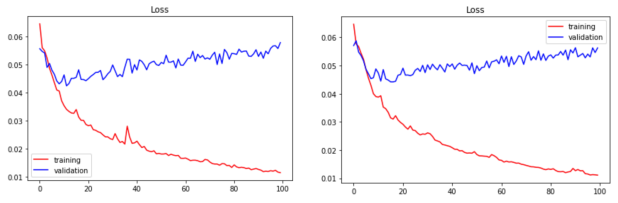


Figure 9. Model Results

Above figures extracted from two neural networks without the embedding layer. The left neural network includes the input parameters (grade level, essay type) and the right neural network without input parameters. Both figures show that the loss is getting less and less over 100 epochs but the validation error is getting larger and larger, the model is over-fitting.

Below figures extracted from two neural networks with the embedding layer. The left neural network includes the input parameters (grade level, essay type) and the right neural network without input parameters. Both figures show a similar trend over time. With more and more training data and over 100 epochs, this model should perform better and better without over-fitting.

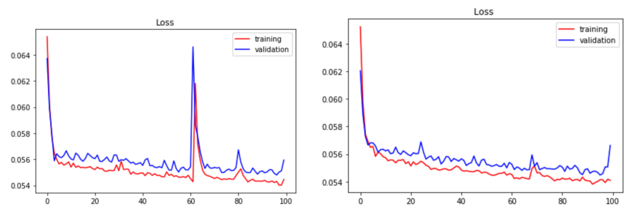


Figure 10. Model Results

CONCLUSION

In this project, I proposed an automated essay grading system with recurrent neural networks as the back-end algorithm approach and a user-friendly interface. Based on my model, this system can make a prediction on any input essays in English. The algorithm itself only requires a certain amount of training data without extracting any pre-defined features from the textual data. I also explored four different architectures as the experiment. The embedding layer can significantly improve the neural networks' performance without over-fitting. Some other hyper-parameters can be tuned as the future analysis.

ACKNOWLEDGMENTS

Thanks the course instructor Dr. David Joyner and my mentor Lingxi Xiao for their guidance and support. Lingxi gave me many helps during this my final project. I appreciate all the TAs and working stuff for the OMSCS-6460 Education Technology class.

REFERENCES

1. Burstein, J., Chodorow, M., & Leacock, C. (2004). Automated essay evaluation: The Criterion online writing service. *Ai Magazine*, *25*(3), 27.
2. Rudner, L. M., & Liang, T. (2002). Automated essay scoring using Bayes' theorem. *The Journal of Technology, Learning and Assessment*, *1*(2).
3. Burstein, J., Chodorow, M., & Leacock, C. (2003, August). CriterionSM Online Es- say Evaluation: An Application for Automated Evaluation of Student Essays. In *IAAI* (pp. 3-10).
4. Wang, H. C., Chang, C. Y., & Li, T. Y. (2008). Assessing creative problem-solving with automated text grading. *Computers & Education*, *51*(4), 1450-1466.
5. McNamara, D. S., Crossley, S. A., Roscoe, R. D., Allen, L. K., & Dai, J. (2015). A hi- erarchical classification approach to automated essay scoring. *Assessing Writing*, *23*, 35-59.
6. Landauer, T. K. (2003). Automatic essay assessment. Assessment in education: Principles, policy & practice, 10(3), 295-308.
7. Chen, H., & He, B. (2013). Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp. 1741-1752).
8. Phandi, P., Chai, K. M. A., & Ng, H. T. (2015). Flexible domain adaptation for auto- mated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 431-439).
9. Burstein, J., Leacock, C., & Swartz, R. (2001). Automated evaluation of essays and short answers.
10. Reilly, E. D., Stafford, R. E., Williams, K. M., & Corliss, S. B. (2014). Evaluating the validity and applicability of automated essay scoring in two massive open online courses. *The International Review of Research in Open and Distributed Learning*, *15*(5).
11. Zhao, Siyuan, Yaqiong Zhang, Xiaolu Xiong, Anthony Botelho, and Neil Heffer- nan. 2017. "A Memory-Augmented Neural Model for Automated Grading": 189–192.
12. Young, Tom, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2017. "Recent trends in deep learning based natural language processing." *arXiv preprint arXiv*:1708.02709.
13. Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, *34*(1), 1-47.
14. Johnson, V. E. (1996). On Bayesian analysis of multirater ordinal data: An applica- tion to automated essay grading. *Journal of the American Statistical Association*, *91*(433), 42-51.
15. Kakkonen, T., Myller, N., Sutinen, E., & Timonen, J. (2008). Comparison of dimen- sion reduction methods for automated essay grading. *Journal of Educational Tech- nology & Society*, *11*(3).