Automatic short passage Grading

Harsit Agarwal, Sai Vineet Reddy

December 1, 2019

harsit.agarwal_asp20@ashoka.edu.in,
saivineetreddy.thatiparthi_asp20@ashoka.edu.in

Abstract

Our goal in this project is to evaluate an essay that is presented to the model instead of a human grader. The essays used to train our model were responses to a standardised test, which each of them graded by human readers. After tweaking the hyper-parameters, the model achieved a peak Cohen Kappa score of 0.85. This number was arrived after employing a combination of many different NLP methods.

Through our project, we wanted to determine the possible benefits and limitations of using such a system to replace human graders in standardised testing.

1 Introduction

Automated essay scoring (AES) has been an important area of research ever since the publication of 'The Imminence of Grading Essays by Computer' by EB Page in 1966. The help that such a system would provide to teachers has been the driving force behind the interest in this area. The rise of MOOCs and improvement in Natural Language Processing techniques has accelerated the interest in this field.

Majority of the work on AES has focused on holistic scoring, which provides a single score based on the quality of the essay. The main reasons behind this has been the presence of datasets that contains essays with holistic scores and the commercial value that such systems provide by automating the the process of grading millions of essays for standardized aptitude tests such as the SAT and the GRE.

Holistic scoring systems cannot be used in classroom settings because of the need to provide feedback to students so that they can understand their mistakes. To address this deficit of the past systems, researchers have begun to score essays on particular dimensions of essay quality such as coherence, technical error, relevance to prompt, etc.

As Ng and Ke note, the scarcity of annotated data for dimension-specific scoring has hindered progress in the field. Due to the scarcity of data and the constraints of time, we attempt to design a model that produces a holistic score for essays.

2 Related Work

Ng and Ke note that all of existing AES systems are learning-based and can be classified on the basis of them employing either supervised, weakly supervised or reinforcement learning. A voting algorithm is used by Chen et al. along with weakly supervised bag-of-words framework to address text scoring.

State-of-the-art approaches to AES systems are all supervised. Researchers following supervised learning have used the following approaches: (1) a regression approach - predicting the score of the essay; (2) A classification approach - classifying essays as belonging to a certain set of classes; (3) a ranking approach - ranking two or more essays based on their quality.

Most approaches use an off-the-shelf learning algorithm for model training. For the regression approach, linear regression, support vector regression and sequential minimal optimization have been typically used. For classification, SMO, logistic regression and Bayesian network classification have been used. For ranking, SVM ranking and LambdaMART have been used.

3 Data Set and Features

We retrieved our data from Kaggle, as well as a medium blog post as they were hosting a format of the dataset that had accounted for some errors present in the data itself. The dataset is called the Automated Student Assessment Prize (ASAP) AES dataset. The dataset can be found at the following link - `https://kaggle.com/c/asap-aes`. The dataset has become widely popular for holistic scoring ever since due to the large number of essays it contains and also because of the large number of essays per prompt. Each of the sets of essays was generated from a single prompt, however, the grading schema used is different for essays belonging to different sets. To account for this, we normalised the scores of all the essays to be between 0 and 1.

One of the drawbacks of this dataset is that it has different score ranges for different prompts. Another drawback is that the essays present in the dataset might not reflect the original version as it does not contain any paragraph information and the essays have gone through an aggressive pre-processing that has expunged name entities and most other capitalized words.

We cleaned the data of incompleteness/non-applicability using the Pandas library (pd.cleanse), and also fixed headers manually. Some values were left empty that were dropped from the final training set to ensure completeness of data. This also helped in allowing us to obtain accuracy loss metrics as incomplete data was corrupting our true loss values.

There were 300 input features for the Neural Network. These 300 features are part of a dense vector representation of a single essay. The representations were created using Word2Vec, a popular method to embed words as numbers before feeding into a neural network. We also tried to train our own embedding layer using the dataset but that gave us a poorer performance. This is probably due to the fact that Word2Vec has been trained on a large corpus and hence is able to find better semantic relationships than what we get using the small corpus available to us.

To make our entire dataset uniform, we removed essays with less than 40 words in them. This helped us increase our accuracies by a small margin. Apart from this, we followed industry standard practices of removing stopwords as well as most punctuation. Without doing this, our Word2Vec model, despite being trained for over 200 epochs, did not seem to derive semantic relationship between the words present in our essay corpus. As mentioned earlier, the ASAP AES dataset had essays belonging to 8 different sets, with each set having its own grading schema. To ensure uniformity, we normalised all the scores to fall between 0 and 1.

Since the data was one large set, we split it on a random basis into two different sets. One was used for testing and another for training. The test-train split was done in a 20:80 ratio.

We use the Cohen's Kappa (also called Quadratic Weighted Kappa) as our evaluation metric. We do so because it is the most widely adopted method for

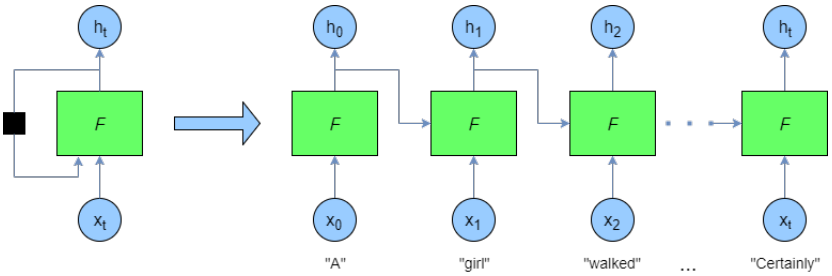


Figure 1: Context preservation in LSTMs

evaluating essay-scoring models and also because it takes into account the possibility of agreement between predicted values and actual values that happens by chance. Cohen's Kappa can range from 0 to 1 and can even be negative if there is less agreement that what is expected by chance. For a detailed discussion on Cohen's Kappa see: `https://www.kaggle.com/c/asap-aes/overview/evaluation`

4 Methods

4.1 Training and Validation

Using a Multi-layered Perceptron, we created two models for prediction of essay scores, based on the techniques employed within the model.

a. **LSTM regressor**

b. **Normal Dense logistic regressor**

Both of these models had a similar set up. The inputs to both being an 300 feature vector as described above, having 3 hidden layers. A dropout of 0.3(30 percent) was also used for both models during training - which proved to have a higher score than what was achieved with a dropout of 0.25. We arrived at the number of hidden layers and the dropout percentage through simple trial and error (a lot of credit has to be given to the literature review that we sifted through as well).

However, the two models differed in the following ways:

- For the LSTM regressor, we swapped out a normal deep neural network filled with simple neurons with something called a LSTM (Long Short Term Memory) units. These units are particularity useful in situations where data from the past determines the final state of the data in the present (even the vanishing gradient problem is taken care off when LSTMs are used). As we are dealing with written text, the situtation is perfect for a LSTM to work (Figure 1).
  In our model, there were two layers of LSTMs (300 and 64 respectively), with each layer having a recurrent dropout of 40 percent.

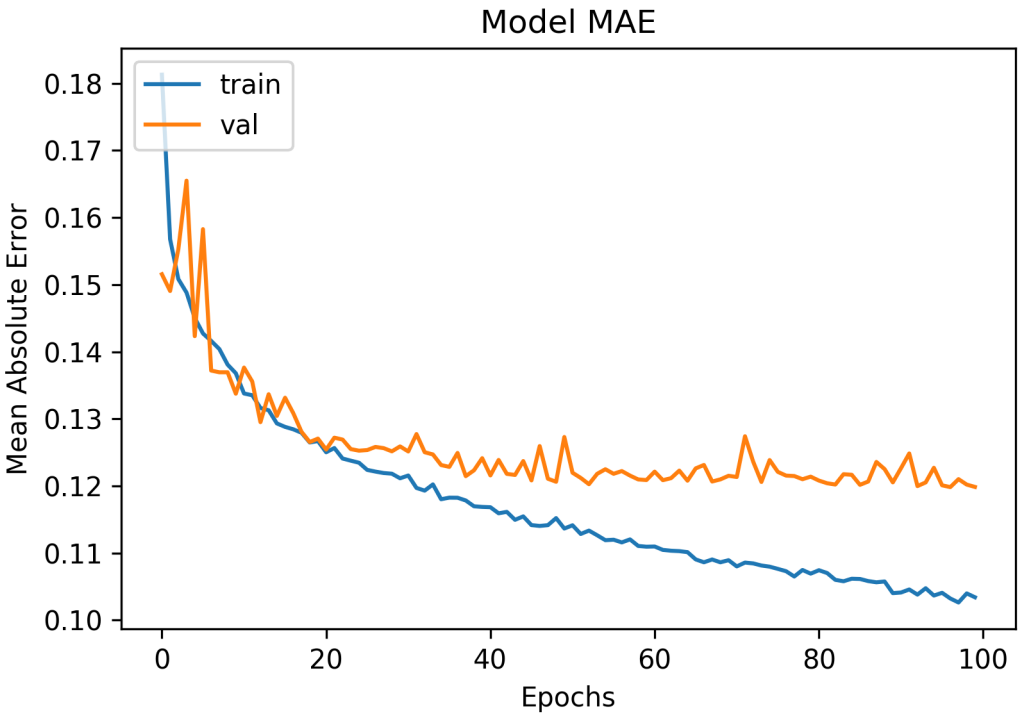


Figure 2: Model MAE over epochs

- For the Normal Dense logistic regressor, The sigmoid function was used as the final activation. This was chosen as the final score was in the range 0 - 1, perfect for outputs generated by sigmoids. In this case, the ReLu activation did not work because ReLu has a range $[0, \infty)$ of outputs, which makes it difficult to spit someting out between just 0 and 1. Additionally, 'Mean Absolute Error' was used to record Error instead of Mean Squared Error (MSE). This somehow produced better scores.

To train the data, we used an 80-20 split and *k-cross validation*, wherein we would train on 80 percent of the data and then validate on the remaining 20.

We noticed was that our model gave the same accuracy when the batch size was 1 as opposed to when the batch size was 300, with batch training being much faster. So, using this method, we were able to cut down on the time it took for the completion of one epoch very quickly.

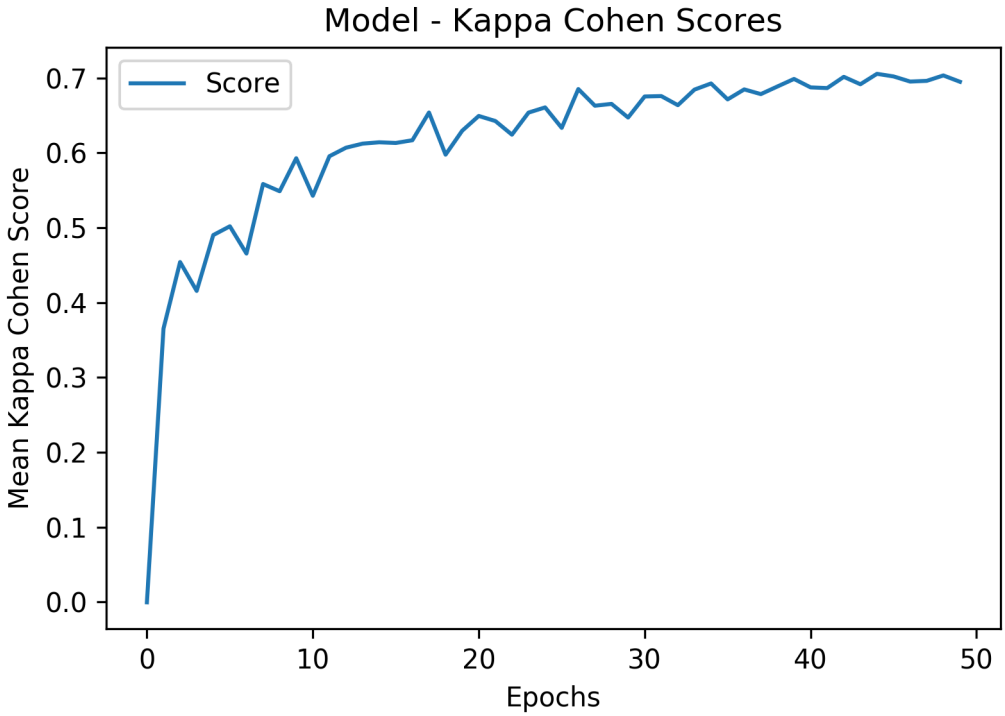


Figure 3: Kappa Cohen Scores over epochs

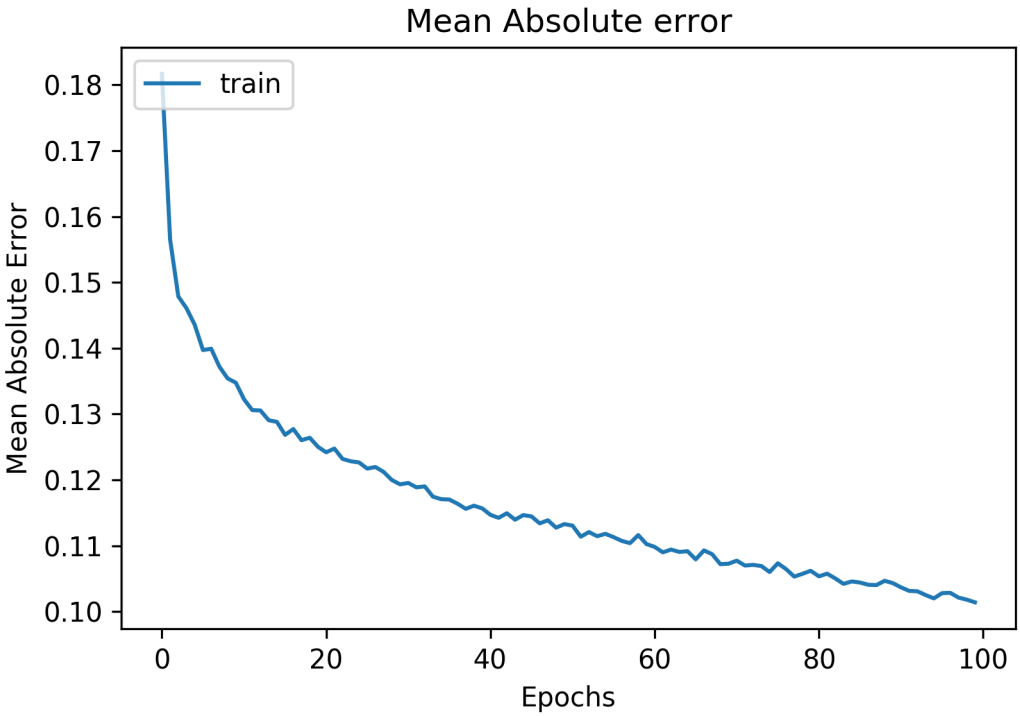


Figure 4: Training loss over epochs

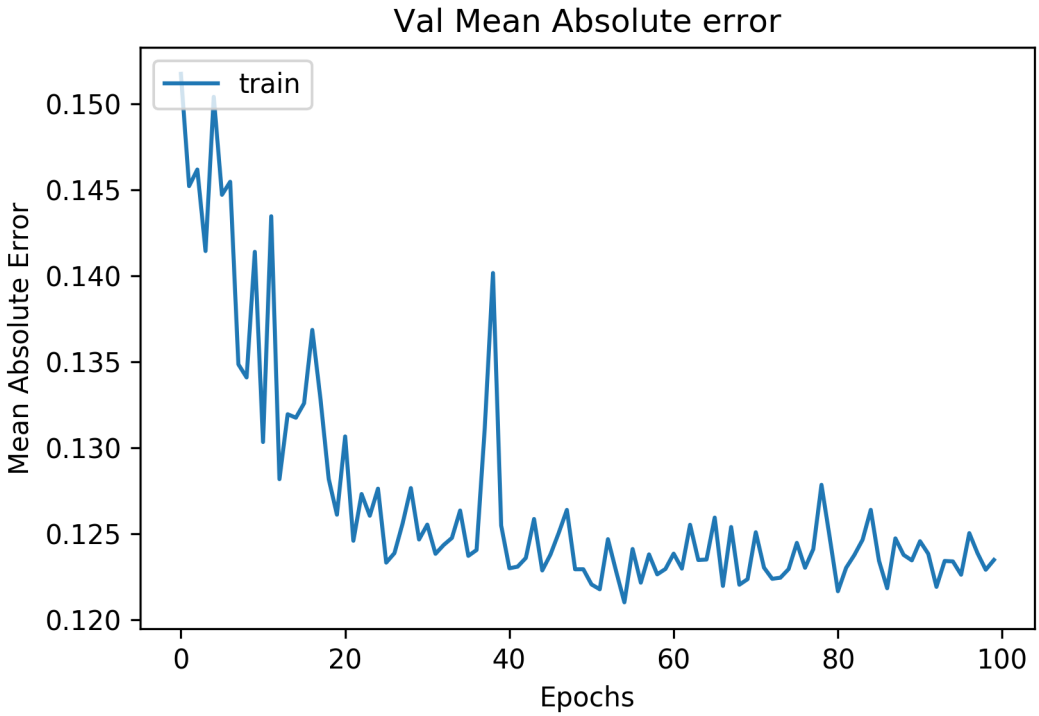


Figure 5: Validation loss over epochs

The MAE obtained for the LSTM model can be found in Figure 4 (Training set). There has been a steady decrease as the number of epochs increased. For the same model, the MAE on the validation set can be found in Figure 4. Note that this graph is a lot more turbulent. This was fixed by increasing the dropout percentage.
For the Neural Network model, the MAE over both the validation as well as training set can be found in Figure 2.

After about 40 epochs, our validation accuracy remained the same in both the models, with an increase in number of epochs not changing the loss significantly. Rather, the model began overfitting. Hence, we found the number of epochs at 40 to be the sweet spot.

We also tried summarizing the essays and then passing them through our model. Our hypothesis was that the most useful/discriminative features of an essay would be preserved by summarization and this would improve the performance of the model as it would learn the most discriminative features. We used LexRank summarization algorithm, an extractive summarization technique to extract the 'top' four senteces from each essay. LexRank used graph-based centrality scoring of sentences to summarize text in an unsupervised manner. We tried summarizing essays before passing them into both of the models stated above. The Kappa Score that was obtained was significantly lesser than when we didn't summarize the essays for both the models. We posit that this might be due to the loss of information that takes place when summarizing the essays.

4.2 Testing

After the training, we used the remaining 20 percent to validate our model. On average, we achieved a Kappa Score (on the LSTM model) of **0.80**. The state of the art networks also produce similar Kappa Scores on the same dataset. The MAE had settled around 0.10 after 100 epochs [as reported by Keras.]

Since we had realised that Kappa Score was quite senistive to how long the model is being trained, we plotted out the Kappa Score of our LSTM model after each epoch to find the sweet spot there too. Figure 3 allowed us to do exactly that.

Table 1 provides a condensed view of our accuracy metrics of the two models.

5 Results

Table 1: Kappa Cohen Scores

LSTM		**Normal Dense**	
Training	Testing	Training	Testing
0.85	0.82	0.71	0.70

In our LSTM model, two layers of 300 and 64 LSTM units respectively gave us a Kappa Cohen Score of 0.82 after training for 100 epochs. Anything more than this overfit the model and decreased the Kappa Cohen Score. The MAE on the validation set was 0.11 at the end of the training phase.

In our neural network model, there were six layers of 600, 200, 200 and 200 neurons respectively. After being trained for 100 epochs, the Kappa Cohen Score achieved was 0.70 on the validation set. The MAE on the same set was 0.14 at the end of the training phase. Interestingly, this model was less susceptible to overfitting, hence, we believe that more epochs might actually help this model.

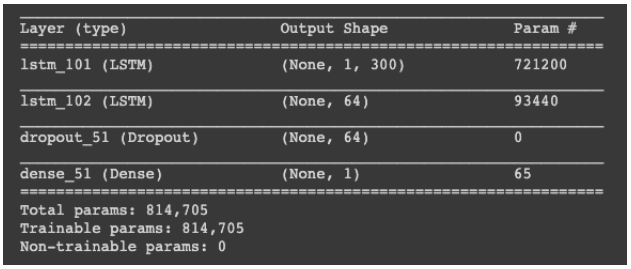
6 Conclusion

In this paper, we successfully demonstrate the ability of a LSTM-based and a dense model to automatically score essays. We find that the LSTM-based model performs better than a simple Feed Forward Neural Network at this task and is able to give us results similar to state-of-the-art results. We also find that extractive summarization of the essays before passing them through the model adversely affects the performance of the model. We note that using a pre-trained library like Word2Vec gives us better performance than trying to learn word-embeddings as part of our model itself.

7 Future Work

- The model might also be useful for dimension-specific scoring (persuasiveness, coherence, etc.).
- Using an abstractive summarization technique might produce better results than an extractive method as the later just prints what it considers the top 'n' number of sentences while the former produces the summary based on an understanding of the text.
- Look into reducing the Kappa Cohen score even further by employing more LSTM units in our network. At the current stage, the computation limitations on contemporary systems prohibit one from doing this.
- Better word embeddings always allow for better performance. Using domain specific methods of generating embeddings will directy translate into neural networks that better perform.

Finally, below is a summary of our LSTM Model:



Layer (type)	Output Shape	Param #
lstm_101 (LSTM)	(None, 1, 300)	721200
lstm_102 (LSTM)	(None, 64)	93440
dropout_51 (Dropout)	(None, 64)	0
dense_51 (Dense)	(None, 1)	65

Total params: 814,705
Trainable params: 814,705
Non-trainable params: 0

8 References

1. Ke, Zixuan, and Vincent Ng. *"Automated essay scoring: a survey of the state of the art."* Proceedings of the 28th International Joint Conference on Artificial Intelligence. AAAI Press, 2019.

2. Taghipour, Kaveh, and Hwee Tou Ng. *"A neural approach to automated essay scoring."* Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 2016.

3. Alikaniotis, Dimitrios, Helen Yannakoudakis, and Marek Rei. *"Automatic text scoring using neural networks."* arXiv preprint arXiv:1606.04289 (2016).

4. Page, Ellis B. *"The imminence of... grading essays by computer."* The Phi Delta Kappan 47.5 (1966): 238-243.

5. https://www.kaggle.com/c/asap-aes

6. *Class notes and discussions* - Professor Ravi Kothari, CS406, Ashoka University.