Automated Essay Grading using Neural Networks

Abhinav Gupta
Computer Science, NYU
New York, NY, USA
abhinav.gupta@nyu.edu

Harsh Wani
Computer Science, NYU
New York, NY, USA
hsw268@nyu.edu

Abstract

In this paper, we explore and propose three Neural Network models to automate the process of essay grading. We utilize the Hewlett Foundation's AES Dataset to train and evaluate our model. We use Kappa score (QWK) score to compare and evaluate our model. We present three models that use word2vec, LSTM, BLSTM, and word embeddings. We also draw comparisons with various existing models and provide insights and knowledge. We also explore the effects of feature size on the model. We implement and provide models with satisfactory performance with updated models.

Keywords—NLP, automated essay grading, AES, LSTM, BLSTM, word2vec, Kappa, QWK, neural networks

1 Introduction

What are Essays? Most of academia consider essays as one of the most important tools that facilitate the assessment of learning conclusions, inferring the capability to recall, establish, and bring together ideas. It is a form of expressing yourself in writing. The importance of grading essays is not lost on the researchers today, who unfortunately feel that therein lies a lot of obstacles in the grading of essays. Mainly, the perceived subjectivity of the grading process, it is believed that the subjective nature of essay grading causes variation in grading done by various humans. This for students on many levels may be unfair. Not only this, but the sheer effort of manual grading takes a substantial amount of human grader's time and in-turn is an expensive process.

One viable option talked about the most is the automated assessment which would be consistent in its grading style, will be cost-effective and time-saving, and comparing it with human scores will make it more genuine. The need for automated grading has further increased in the recent future due to increased demands of Online courses and online tests being conducted worldwide. Computers have helped us in our writings for more than 30-40 years now. The very basic computer can help authors in their writing like processing of words and updating their text.

The research tells us that computer systems can be a more effective cognitive tool. In this paper, we present three methodologies using neural networks which would help us to automatically grade essays. This will provide a system of unified measure and standardization to human given marks. We use the dataset provided for the Hewlett Foundation's Automated Student Assessment Prize competition on Kaggle []. We evaluate our model using QWK or Kappa score [4]. The evaluation of the essays varies for different papers. We evaluate our models to two such given techniques.

In section 3, we discuss the dataset we use and in section 4, the methods or the process employed to train the model. Section 5 discusses the results and evaluations of various models proposed.

2 Prior Work

Over the years, the space of automated essay grading has been explored extensively and a lot of progressive work has been carried out. *Johns et al.* [1] proposed an approach to obtain features like count per essay, no. of long words, POS and sentence counts, etc. from the training set essays. The paper implements a linear regression model to learn and come up with parameters for testing and validation. It applies 5-fold cross-validation and used the error metric used in the paper is Quadratic Weighted Kappa. The paper goes on to hypothesize that by using better-enhanced features such as N-grams or k-nearest neighbors in a bag of words can improve performance.

Kaveh et al. [2] developed a recurrent neural network approach to learn or establish the correlation between the essay and the human assigned scores without using any feature engineering. The authors tested various neural network models for this task of automated essay grading and conducted some examinations to understand the workings and the outputs of these models. The results of these models tell us that the best model used in this paper is a long short-term memory network, which betters the strong baseline by 5.6% in

terms of quadratic weighted Kappa, without any feature engineering. The paper discusses and compares the performance of this task across various neural models such as CNN, RNN, LSTM, etc.

Dimitrios et al [3] proposed a model that forms word representations by learning the extent of how some specific words contribute to the text's score. The authors proposed LSTM networks to represent the meaning of texts and show that a fully automated framework is better than similar approaches. Paper proposed to apply these network models to enhance the performance of ATS systems and learn required feature patterns automatically without using the manual tuning. The paper uses a different combination of techniques such as using two-layer LSTM, word2vec LSTM, word2vec 2 layer LSTM, etc., word2vec 2 layer BSTM, pre-trained word2vec LSTM, SSWE (score specific word embedding) LSTM, SSWE 2 layer LSTM, SSWE 2 layer BLSTM, etc. All these models achieved a good Cohen Kappa. Pre-trained word2vec and SSWE with 2 layers LSMTs achieved over 0.90 kappa with the best model in the paper was SSWE 2 layer BLSTM with 0.96 kappa. However, they have not mentioned or given results for individual kappa score for each essay. Therefore, we assume that they have calculated the kappa score for all essays together collectively.

Huyen et al. [4] in 2018 believed that models built to date just used predefined features without exploiting the power of deep learning features. The methodology proposed is as follows - using three various techniques (g averaged pre-trained GLoVe word vector, Word vector training, Tf-IDf technique) to vectorize the essays; fed these features to various neural network models (Two Layer Feed Forward Neural Network, LSTM, Bidirectional LSTM). The neural network models easily beat the baseline of 0.81407 with the best score achieved in the paper is 0.9447875, using a 2-layer neural network that trains word vectors together with the weights. However, they have not mentioned or given results for individual kappa score for each essay. Therefore, we assume that they have calculated the kappa score for all essays together collectively.

Later *Guoxi et al* [5] argued that the existing work only considers the essay itself and not the rating criteria behind the essay. Paper implemented system called Siamese Bidirectional Long Short-Term Memory Architecture (SBLSTMA) and can capture semantic features and also the rating criteria knowledge behind the essays and gives the best performance of average kappa score of 0.81 with the SBLSTMA model overall essays.

Higgins et al [6], in the paper, propose an algorithm that identifies an off-topic essay without actually requiring a set of topic-specific essays for training. In 2014, *Pennington et al* [8], implemented a model that leverages statistical information by not training on the entire sparse matrix in a large corpus, but in the non-zero elements in a word-word co-occurrence matrix. *Larkey et al* [9], applied text-categorization techniques to achieve automated essay grading. The paper used Bayesian independence classifiers and k-nearest-neighbor classifiers for training and scores were combined using linear regression. Also, in 2002 *Rudner et al* [10], used two Bayesian models; the first one used words and the other-phrases and arguments.

3 Dataset

We use the dataset provided for the Hewlett Foundation's Automated Student Assessment Prize competition on Kaggle []. This dataset consists of 8 different sets of essays that are generated from the same prompt. All the essays across all the sets are of an average length of 150 to 550 words per response. Each essay was graded manually by human instructors. There are 12686 essay samples in total in the dataset. Within the original dataset from Kaggle, the validation and testing sets were not available. We divided the dataset into training, validation and testing sets and used 60% for training, 20 % for the validation and 20% for testing.

The dataset consists of columns: essay_id, essay_set, essay, rater1_domain1, rater2_domain1, rater3_domain1 and the domain1_score. For the project, only the columns essay and domain1_score are of interest. The dataset also includes features of domain 2 and domain 3, but we drop those as we are only going to consider domain1 scores.

Table 1: Permitted Events cleaned dataset schema

Essay Set	Number of Essays	Score Range
1	1783	2-12
2	1800	1-6
3	1726	0-3
4	1726	0-3
5	1772	0-4
6	1805	0-4
7	1569	0-30

4 Methods

In this paper, we implemented and compared several neural net models and their performance with the

benchmark and state of the art models. For model 1 described in section 4.2.1, we employed a 4-Fold cross-validation model with 3 Layer LSTM with Word2Vec. For model 2, we used LSTM with the Embedding layer for each unique word in the vocab as in section 4.2.2. Model 3 uses BLSTM with the word embedding layer, described in section 4.2.3. We trained these models with different parameters and feature dimensions to analyze and compute the best model.

4.1 Data Pre-processing

We store the data to keep relevant information. The rest of the columns and data is dropped. Each essay set contains essays on one topic and is scored on a different scale.

The new filtered dataset is a textual corpus that cannot be fed into the machine learning or the deep learning models. Also, traditional representations of words fail to capture knowledge about a word's context and meaning. This is where we convert the essay corpus into the corresponding vector representation that is representing each essay as a vector. The word vectors signify words as multidimensional continuous floating-point numbers, which maps semantically similar words to proximate points in the geometric space.

The input to these methods that vectorize the corpus is not wholesome raw text but is clean processed data in the form of word tokens. We achieve this by preprocessing the text by getting rid of irrelevant characters like special characters, hashtags, etc. We also remove all the stop words before tokenizing the essay.

We generate a feature vector in mainly 4 different methods.

1. Pre-trained GLoVe word vectors.
2. Pretrained word2vec vectors
3. Train word2Vec for each essay
4. Create an Embedding layer to generate a feature vector for each word to be fed to LSTM and BLSTM.

To train the word2vec model, we first break the corpus to sentences and then words. We then remove the unwanted tokens such as stop words, numbers, some punctuation, etc. This is then fed to the Word2Vec model to train to get a feature vector. This algorithm/model maps words with similar meanings to similar positions in the vector space. This, in turn, means that words are represented in the space model to expose semantic relations among various words that help us to achieve many useful results. Embedding layer takes as inputs, the number of unique vocabulary words and the embedding dimension to generate the embedding matrix.

This above process generated training and testing features in the form of a matrix which we can then feed into various machine learning or deep learning models.

4.2 Models

We applied and implemented several machine learning models and to test against the benchmarks and observed their performance. We here, talk about three approaches that provide the best results. We test the implementation with 50, 200, and 300 dimension feature vectors.

4.2.1 Four-Fold Cross-Validation with 3 layer LSTM and Word2Vec

Instead of taking the traditional route of training the entire dataset as a whole at once, we implemented cross-validation with 4 folds. This model could be trained quickly and generated confidant results. Hyperparameters are as follows -

1) Activation Function – relu
2) Loss – mean_squared_error
3) Optimizer – RMSprop
4) Initial Learning rate – default
5) Dropout – 0.4

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 1, 300)	721200
lstm_2 (LSTM)	(None, 1, 200)	400800
lstm_3 (LSTM)	(None, 64)	67840
dropout_1 (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65

Fig: 1: Model structure LSTM with Word2vec

4.2.2 1 layer LSTM with Word Embeddings

Here, the model is a single layer LSTM with a layer of word embeddings. We here generate a feature vector for all the words in the vocabulary of the given corpus to get the embedding matrix. This is then fed to the LSTM. The training time is increased significantly. The model performs well with only a single layer of LSTM. The pre-trained word embedding layer did not

give satisfactory results. Hyperparameters are as follows -

6) Activation Function – sigmoid
7) Loss – mean_squared_error
8) Optimizer – RMSprop
9) Initial Learning rate – 1e-3
10) Dropout – 0.6

4.2.3 1 layer BLSTM with Word Embeddings

This model employs a Bi-direction LSTM with a word embedding layer. The embedding layer generates an embedding matrix. This is then evaluated using a Bidirectional LSTM (BLSTM) layer. This is also run using 50, 200, and 300 features. Hyperparameters are as follows -

11) Activation Function – sigmoid
12) Loss – mean_squared_error
13) Optimizer – RMSprop
14) Initial Learning rate – 1e-3
15) Dropout – 0.5

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 50)	200000
bidirectional_1 (Bidirection	(None, 600)	842400
dropout_1 (Dropout)	(None, 600)	0
dense_1 (Dense)	(None, 1)	601

Fig: 2: Model structure BLSTM

4.3 Evaluation Metric - Quadratic weighted Kappa

The error metric to evaluate essay score predictions is the Quadratic weighted Kappa(QWK)[4]. QWK is an indication of how much the two raters are in agreement. QWK is based on a quadratic weight matrix and this metric ranges from 0 and 1 were going towards 1 means an increase in the agreement between raters. We calculate QWK between the scores attributed by the human raters and the scores we predict using our automated machine learning or deep learning systems. As given in[4], the QWK is calculated as below:

$$W_{i,j} = \frac{(i-j)^2}{(N-1)^2}$$

Where s1 and s2 are the scores assigned by the humans and the models respectively; N defines the number of viable ratings. Next, matrix O is produced over the essay ratings, where Os1,s2 is the number of essays that got a rating of a by human and rating b by the model. The E matrix is generated as the outer product between each rater's histogram of ratings where there is no correlation between the ratings. Using these matrixes, the equation below calculates the metric:

$$k = 1 - \frac{\sum W_{i,j}\, O_{i,j}}{\sum W_{i,j}\, E_{i,j}}$$

5 Results

We draw a comprehensive comparison between the models implemented and the effect of changing the dimensions of the feature vector or the embedding matrix, i.e, changing the number of features that can be used to define the word. Various research papers employed various evaluation techniques for evaluating the model. We here implement and compare using the two techniques we came across.

5.1 Comparision - Combined Essay Sets

Table 2 shows the QWK score obtained for different dimensions, that were generated by the word2vec model. This score was calculated using the 4-fold LSTM with the Word2Vec model. The QWK score was calculated on all the essay sets in the corpus from the divided training set. The model was trained for 50 epochs

Table 2: QWK scores for different feature vector size

Dimensions	QWK Score
50	0.9555
200	0.9626
300	0.9636

Table 3, compares the model with the existing model in [Dimitri] and [Huyen]. We manage to surpass the various existing approaches using the same evaluation metric of calculating the QWK score on the whole essay set.

Table 3: QWK scores compared with other models

Model	QWK Score
SSWE + Two-layer BLSTM	0.96
SSWE + Two-layer LSTM	0.94
LSTM with TDIDF	0.91
2 - layer NN	0.944

3 – layer LSTM with word2vec (our model)	**0.963**

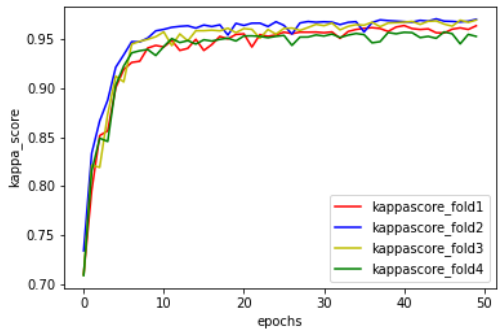

Fig: 4: Kappa score vs Epoch for 4 folds

Fig 4 shows the plot for QWK scores for each fold the model is run for. We see that the best model is obtained in the 4th fold. *Fig 5* compares the mean squared error loss for every epoch for 4 folds and *fig 6* compares the training loss for each epoch for 4 folds. The plots are created using the 300 dimension feature vector for word2Vec.

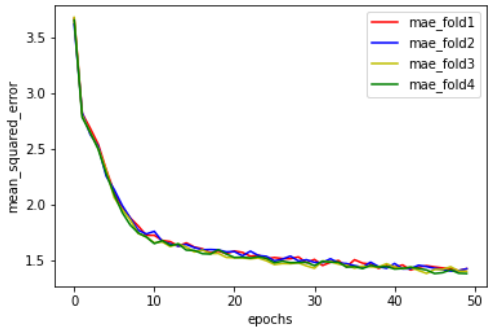

Fig: 5: mean-squared-error vs Epoch for 4 folds

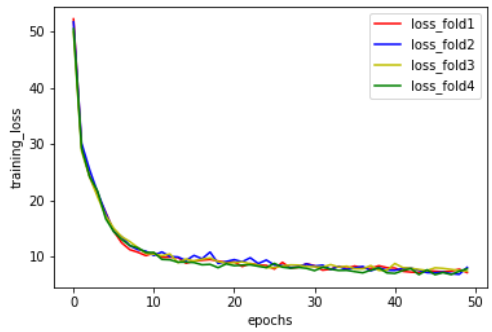

Fig: 6: training loss vs Epoch for 4 folds

5.2 Comparison - Each Essay set compared Individually

Another evaluation that is done calculating the QWK score for each essay set individually and then apply Fisher Transformation to normalized Kappas and calculate the mean quadratic weighted kappa. The size of the dataset was fairly small which was a challenge to train on Neural networks with requiring large datasets to learn the feature vectors. Table 4 shows the best QWK score for various dimensions with Single layer LSTM with Word Embedding.

Table 4: QWK scores compared with other models

Dimensions	QWK Score
50	0.70
200	0.63
300	0.64

In Table 4, we see that increasing the number of dimensions does not help with the prediction.

Table 5 compares the QWK scores when the scores for each essay set are considered individually This provides a comprehensive analysis of our model. Our single layer LSTM with word embeddings competes with the state of the art model LSTM which achieves a Kappa score of 0.74.

Fig 7 shows the loss and mae increase and decrease with the epoch count. Fig 8 shows the mean quadratic weighted average QWK score calculated after applying Ficher Transformation for each epoch. We see that the best model was obtained at epoch 70.

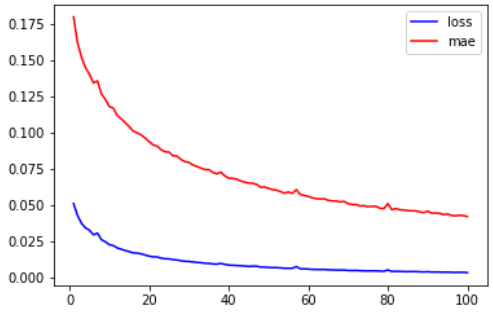

Fig: 7: mean-squared-error and loss vs Epoch for Single Layer LSTM with Word Embeddings

6 Discussion

The overall objective of our paper was to implement multiple models using various techniques and analyze and compare the performances. When the essay sets are evaluated independently the best model is bidirectional LSTM using the word-embedding layer.

On the other hand, evaluating all the essay sets combined the 3 layer LSTM with the Word2Vec model using 4-fold cross-validation gave the best results. In this paper, we drew several key insights and exciting findings through our analysis. Working on the

Model	Essay Set								Mean QWK Score
	1	2	3	4	5	6	7	8	
RNN	0.687	0.633	0.552	0.744	0.732	0.757	0.743	0.553	0.675
0.698	0.616	0.591	0.668	0.787	0.795	0.800	0.752	0.573	0.698
EASE (SVR)	0.781	0.621	0.630	0.749	0.782	0.771	0.727	0.534	0.699
EASE (BLLR)	0.761	0.606	0.621	0.742	0.784	0.775	0.730	0.617	0.705
Single Layer BLSTM	**0.57**	**0.51**	**0.58**	**0.76**	**0.70**	**0.71**	**0.65**	**0.49**	**0.648**
Single Layer LSTM	**0.66**	**0.59**	**0.61**	**0.76**	**0.75**	**0.78**	**0.67**	**0.59**	**0.699**

Table 5: QWK scores compared for each essay set

Word2Vec model, we interpret that dimension of the vector representation of the dataset affects the performance of the model. Although, instead of using pre-trained word vectors, feeding word vectors to the neural network models in a form of an embedding layer also gave satisfactory results.

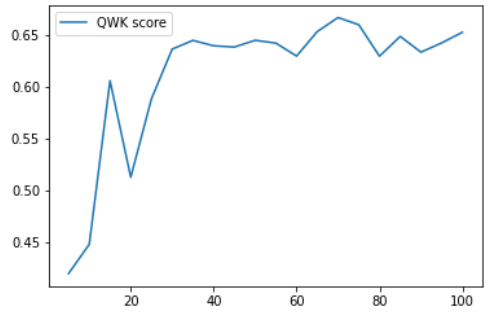

Fig: 8: mean quadratic weighted average QWK score vs Epoch for Single Layer LSTM with Word Embeddings

Further, all the essay sets have different grading scales. Hence, before extracting the features, it is better to normalize the labels. This would give better and ideal results, otherwise, models interpret the data wrongly. Approaches for which we normalize the labels (grading scale), we activation function as sigmoid (as it is between 0 and 1), otherwise we opt for relu.

In terms of models, Bidirectional LSTMs perform well compared to the normal LSTMs across all the techniques as it preserves knowledge both from past and future which enhances the quality of the extraction of semantic information among words.

7 Conclusion

We successfully implemented various Neural Net models to score the essays automatically. This model was trained over the AES dataset and then compared with various evaluation techniques employed by different researchers. Out of our several models, we represent here the best we obtained for the two evaluation techniques.

We obtained a score equivalent to the best model, ie. of 0.96 when taking calculating the QWK score for all the essay sets together. This model was trained and obtained best results on a 300 dimension feature vector, for word2Vec and 3 layer LSTM. We also compared our single Layer LSTM with word embedding and single layer BLSTM with word embedding with various other models. We obtained satisfactory performance by beating several legacy models and competing with the currently available techniques.

We also provide a comprehensive analysis of our models and the effect of dimensions on models and their predictions. We compare the models and scores obtained with different dimensions.

8 References

1. Mahana, Manvi, Mishel Johns, and Ashwin Apte. "Automated essay grading using machine learning." Mach. Learn. Session, Stanford University (2012)
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. Liang, Guoxi, et al. "Automated essay scoring: A siamese bidirectional lstm neural network architecture." Symmetry 10.12 (2018): 682
5. Nguyen, Huyen, and Lucio Dery. "Neural networks for automated essay grading." (2018)
6. Higgins, Derrick, Jill Burstein, and Yigal Attali. "Identifying off-topic student essays without topic-specific training data." Natural Language Engineering 12.2 (2006): 145-159.
7. Ronit Mankad, Github repository, http://github.com/mankadronit

8. Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014
9. . Larkey, L.S. Automatic essay grading using text categorization techniques. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 24–28 August 1998; pp. 90–95.
10. Rudner, Lawrence M., and Tahung Liang. "Automated essay scoring using Bayes' theorem." The Journal of Technology, Learning and Assessment 1.2 (2002).
11. Dataset provided for the Hewlett Foundation's Automated Student Assessment Prize competition on Kaggle https://www.kaggle.com/c/asap-aes/data