Automated Essay Scoring: A Neural Network Ensemble Approach

Sison, Kevin Anthony S. [a], Yu, Van Arloe M. [a]

[a] *Master of Science in Data Science, Aboitiz School of Innovation, Technology, and Entrepreneurship, Asian Institute of Management, Philippines*

ARTICLE INFO

Keywords:
Automated Essay Scoring
Artificial Intelligence
Text Classification
Neural Network Ensemble
Machine Learning
Natural Language Processing

ABSTRACT

Abstract here.

Automated essay scoring is cost-effective and a rapid way of delivering student feedback. It's implementation would provide a standard scoring mechanism for global examinations and reduce examiner bias. The dataset used in this study was from the Automated Student Assessment Prize (ASAP) downloaded from Kaggle. Essays from this dataset were marked by two human raters and this study created a model that would predict the total score for each essay in the dataset. Quadratic Weighted Kappa (QWK) was used to measure the agreement between human and machine ratings. Feature engineering created 32 features (textual and non-textual). Non-textual features include count and collocation of n-grams, while textfual features cover document embeddings. Different machine learning models were implemented to predict the total essay scores. The best performing models are the Gradient Boosting Regressor (QWK = 0.9848 training time = 6 min) and the Fully Connected Deep Network and LSTM ensemble (QWK = 0.9854 training time = 33 min). Further, the GBR via SHAP revealed that the top 3 most important features are the essay set, number of unique words, and the number of sentences. Improvements can be done on the model by using better document embedding techniques, training separate models for each essay set, extracting other features such as Latent Semantic Analysis (LSA) and sentiment scores.

1. Introduction

Automated Essay Scoring (AES) allows grading of student essays without human interference. It takes in an essay for a given prompt and outputs a numeric score reflecting its quality, based on its content, grammar, and organization. This would provide a standard scoring mechanism for global examinations like GRE, GMAT, TOEFL, and IELTS. An automated essay scoring system would reduce examiner bias when checking student essays and as a result it would liberate more time for teachers and so they can devote more time in improving their lessons and delivery. This would also allow teachers to give quick feedback to work on areas of weakness.

Since February 1999, the Educational Testing Service has utilized E-rater, a system for automated essay scoring particularly used for the Graduate Management Admission Test Analytical Writing Assessment (GMAT® AWA). The system has two components, a scoring application and Critique, composed of programs that can evaluate and identify grammatical errors. This system was also trained with a sample of essays on the same topic that was rated as well by human readers. The new e-rater system was trained with a new feature set that includes errors in grammar, usage, mechanics, and style, organization and development, lexical complexity (ratio of number of word types to tokens, measure of vocabulary level based on Breland's standardized frequency index value, and average word length), prompt-specific vocabulary usage (score point value and cosine correlation), and essay length (Attali and Burstein, 2005).

One potential problem of AI based essay graders is that the algorithm may not be able to distinguish between quality and drivel. Such inabilities were tested and described in an article published in Forbes magazine. For example switching the words "the" in the essay to "chimpanzee" gave a score of 6 out of 6. These automated scoring programs cannot distinguish between a well-written essay and something that is meaningless. A research affiliate from MIT developed a system that creates essays with no sense but captures top scores from AES to reveal AES' weaknesses. This is taken advantage of by people as they try to game the system by putting all the things an AES focuses on and putting it in their essays.

2. Literature Review

The success of an Automated Essay Scoring or Automated Text Scoring System lies on the features engineered by humans. AES systems are usually based on regression (Srivatsa et al) or deep learning approaches. The group of

ORCID(s):

Alikaniotis (2016) has shown that combining SSWE with LSTM creates a system that can score essays very similar to a human. Further they found out that unlike other works, text length has a low correlation with the gold scores, i.e. text length is not a strong predictor of the overall score. Their mode was also capable of generating score-specific word embeddings. Their architecture was able to represent the local contextual and usage information captured by essay scoring. It has also been shown by the group of Taghipour and Ng (2016) , that LSTM performs significantly better than other systems. Interestingly, their method was not reliant on any feature engineering and allows the system to learn automatically the representations needed in the task. They have reported outperforming the open-source baseline by 5.6% (quadratic weighted Kappa). Contreras, et al. took a different approach in essay scoring utilized OntoGen (which utilizes Support Vector Machines) for creating domain ontology and applied NLP algorithms via Natural Language Toolkit (NLTK) to create an AES. The problem of having a small number of target domain data was explored by the research of Phandi, Chai, and Ng (2018). Their group proposed using domain adaptation to apply the AES system from one essay prompt to another essay prompt. Memory networks can also be applied to the AES problem. Using memory networks, the key factors that determine performance are reliable representation and memory components, and the model was able to automatically grade assignments from other subjects (Zhao et al, 2017). Another deep learning approach was the use of non-linear regressor deep neural networks that outputs holistic score3s of 10-60 (Boulanger and Kumar, 2018). The group of Cozma, Butnaru and Ionescu (2018) tried using string kernels and word embeddings to automatically score essays. They also tried combining high-level semantic feature representation particularly bag-of-super-word-embeddings and were able to beat current deep learning approaches. The performance of AES systems usually involves minimizing classification, regression or pairwise classification loss. however , Chen and He (2013), argues that a listwise learning approach that incorporates human-machine agreement into the loss function improves the AES performance.

Text classification can be done via fasText (Joulin et. al, 2017), BLSTM (Zhou et. al, 2017) Naive Bayes (Kim et. al, 2006), Support Vector Machines (Tong and Koller, 2002), kNN, Corner classification network (Ikonomakis et. al, 2005). To make the machine learning task more accurate, feature selection is often performed. In a paper by Forman, he tried twelve feature selection methods, such as information gain and found that Bi-Normal Separation (BNS) outperformed other methods. Another approach was an ontology-guided feature engineering that led to the development of information-theoretic techniques built on the Unified Medical Language System (UMLS) (Garla and Brandt, 2012). Naive Bayes via the Multi-class Odds Ratio and Class Discriminating Measure can also be used in feature selection (Chen et. al, 2008). Aside from Naive Bayes, kNN and SVM can also be used as methods for feature selection. (Rogati et.al, 2002). Kim et. al suggested that to include two empirical heuristics in Naive Bayes: per-document text normalization and feature weighting method. Appropriate combinations of preprocessing can impact significant improvement on classification accuracy (Uysal and Gunal, 2014).

Zhou et. al improved text classification by combining Bidirectional LSTM with 2D Max pooling. Their study also considered BLSTM-2DCNN, which outperforms RecNN, RNN, CNN and even BLSTM-2Dpooling. The effectiveness of a text classification system lies on how good effectiveness is defined, tuning the systems and the ability to estimate if new data is introduced and processed (Lewis 1995). Howard and Ruder proposed applying transfer learning to the text classification problem. Universal Language Model Fine-Tuning serves as an effective tool for fine-tuning a language model.

3. Data Description

Data Description In 2012, the William and Flora Hewlett Foundation sponsored the Automated Student Assessment Prize (ASAP) competition in Kaggle. The competition was created to develop effective automated grading of student-written essays (The Hewlett Foundation: Automated Essay Scoring, 2012). The task of automated essay scoring has an input of the student essay written and the essay set identifier and an output of a predicted numeric score reflecting its quality, based on its content, grammar, and organization (Taghipour et al., 2016).

As seen in Table 1, The dataset contains 8 essay sets, with each set corresponding to a different prompt or topic. It contains 12,976 student-written essays. Each essay was marked by two human raters. The sum of the two scores given by rater 1 and rater 2 makes up the total score. The task is to predict the total score given for each essay in the dataset (The Hewlett Foundation: Automated Essay Scoring, 2012).

To get a better idea of the difference between the essay sets. We did an exploratory data analysis on the difference between essay lengths and scores for each essay set.

From Figure 1, by visual observation, we can see three patterns of distributions. The essay sets 1 and 2 have similar

Table 1
ASAP Dataset

essay set	essay	rater1	rater2	total score
1	Dear local newspaper, I think effects computer...	4.0	4.0	8.0
1	Dear @CAPS1 @CAPS2, I believe that using compu...	5.0	4.0	9.0
1	Dear, @CAPS1 @CAPS2 @CAPS3 More and more peopl...	4.0	3.0	7.0
1	Dear Local Newspaper, @CAPS1 I have found that...	5.0	5.0	10.0
1	Dear @LOCATION1, I know having computers has a...	4.0	4.0	8.0

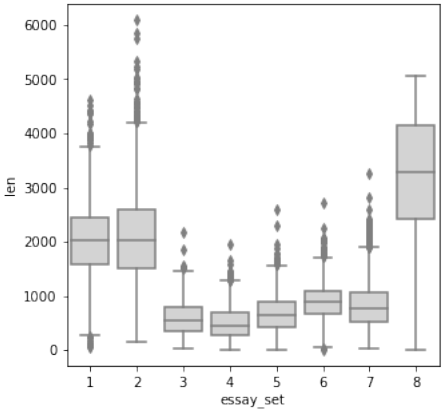

Figure 1: Essay length boxplot by essay sets

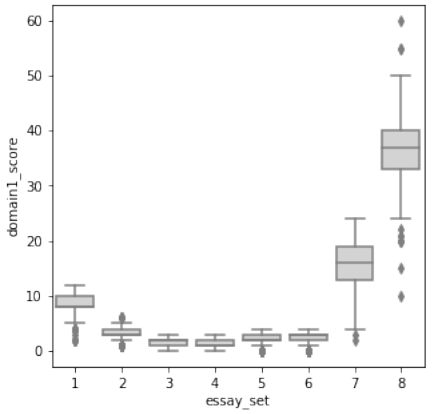

Figure 2: Scores boxplot by essay sets

medians and length ranges. Both sets 1 and 2 also have numerous outliers beyond the maximum. We see that essay sets 3, 4, 5, 6, 7 have narrower distributions than set 1, 2, 8. We see that essay set 8 is left-skewed and has fewer outliers than the other essay sets.

In Figure 2, we wanted to see whether the essay set identification would be a good feature for our model. We applied statistical analysis with a null hypothesis that the essay set is not significant in predicting the score with a p-value of 5%. Using a t-test independent test with Bonferroni correction, we saw that all essay sets are significant with p-values of <= 1.00e-04%. This shows that this feature will be useful for our analytical dataset.

4. Methodology

The main task of AES is to convert this unstructured text data into a structured dataset that we can plug into our models. To do this, we first normalized the data by removing stopwords and removing special characters. Then we developed a feature engineering pipeline to extract both syntactic and semantic features of the essays. We then inputted the data into different machine learning architectures and compared the results.

4.1. Evaluation Metric

The evaluation metric prescribed in the ASAP competition was the Quadratic Weighted Kappa (QWK). The QWK is a measurement of agreement between two ratings. It ranges from 0 to 1, with 0 indicating a random agreement between two ratings and 1 indicating a complete agreement between two ratings (Arora, 2019).

QWK is calculated by first getting the matrices $W_{i,j}$, $O_{i,j}$, and $E_{i,j}$. W is an NxN weighted matrix calculated by Equation 2, where i is the target rating in our case the total score given by human raters, j is the output of our machine learning model, and N is the number of possible ratings (Arora, 2019; Taghipour et al., 2016).

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

$$\kappa = 1 - \frac{\sum_{i,j} W_{i,j} O_{i,j}}{\sum_{i,j} W_{i,j} E_{i,j}} \tag{2}$$

An NxN histogram matrix $O_{i,j}$ is constructed denoting the number of essays that receive a rating i by the human raters and a rating j by the machine learning model (Arora, 2019).

An NxN histogram matrix of expected ratings $E_{i,j}$ is calculated by taking the outer product of the histogram vectors of the human raters and machine learning model's ratings. It is normalized such that the sum of elements in the matrix $E_{i,j}$ and $O_{i,j}$ are the same (Arora, 2019). QWK can finally be calculated using the equation in Equation 1 (Arora, 2019; Taghipour et al., 2016).

We implemented this metric using cohen_kappa_score from sklearn.metrics with hyperparameter weights set to 'quadratic' (Pedregosa et al., 2011). We chose to stick with this metric to be able to compare our results to published baselines. To set our gold standard baseline, the marks given by rater 1 and rater 2 in the dataset have a QWK of 0.9692. In 2012, the winning entry had a QWK of 0.8141.

4.2. Feature Engineering

We extracted two main types of features. We name the two types of features as non-textual and textual features. Non-textual pertains to the numeric features that we derive from the syntactic analysis of the text. Textual features pertain to the vectorized representation of the text. Here we used a document embedding approach.

4.2.1. Non-Textual Features

The non-textual features consist of 16 features that can be found in Figure 3. It is inspired by the baseline model used by Taghipour et al. in their paper using the open-source library EASE (Enhanced AI Scoring Engine) that extracts features from texts (Taghipour et al., 2016). The non-textual features are extracted from the normalized essays except for the number of stop words feature. First, we got length based features of the essay. We got the number of words, number of sentences, number of characters, average length of words, and average sentence length. The intuition of these features is to extract the basic structure of the essay in terms of length.

We also extracted features reflecting the quality of the words used by the student. We extracted the number of stopwords, the number of unique words or the size of vocabulary, and the number of lemmas. The intuition of these features is to extract information on the richness or diversity of the student's vocabulary. The features intend to extract whether the essay is made up of meaningful words or just stop words and/or repeating root words.

As a proxy to the quality of writing, we extracted the number of spelling errors in the essay. The intuition behind this is to extract a feature that indicates the grammar of the essays. We obtained spelling errors using the library pyspellchecker (Barrus, 2018).

We also got the counts of different POS tags of the essays. We got the number of nouns, adjectives, verbs, adverbs, and binned together other parts of speech. These features intend to capture the structure of the essay as well.

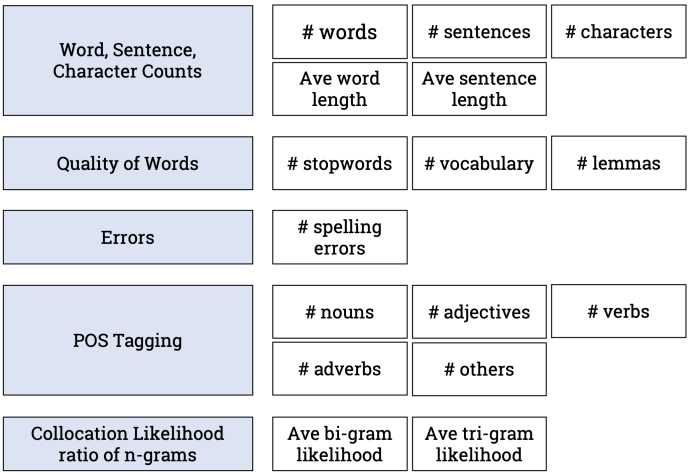

Figure 3: Non-Textual Features

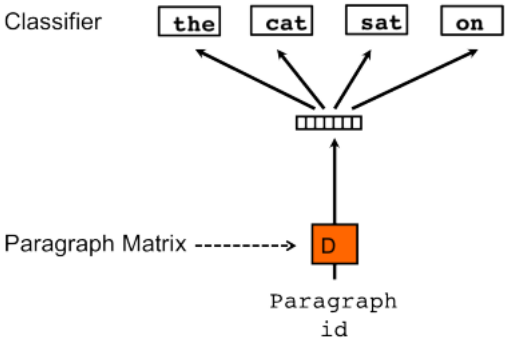

Figure 4: Distributed Bag of Words Paragraph Vectors (Le Mikolov, 2014)

Given the text "I am at home. I am not in school. I am presenting my project. I am explaining.", we wanted to extract a feature that derives the repetitiveness of n-gram usage of the writer. To do this, we derived the collocation likelihood ratio of n-grams. We computed the feature by getting the average of all likelihood ratios of all n-grams in the essays. We did this for both bi-grams and tri-grams in the essays.

4.2.2. Textual Features

Traditional fixed-length vector representations of text are done using bag-of-words. However, this approach does not preserve the semantics of the text in the vector representation. They addressed this issue with their Paragraph Vector Distributed Bag of Words (PV-DBOW), which predicts the context word from the paragraph vector (Le Mikolov, 2014).

We used gensim's doc2vec in our implementation of the document embedding (Řehůřek Sojka, 2010). For the window, we used a window of two which means we take into consideration two words before and two words after. For the vector size, we followed the general rule of thumb discussed by TensorFlow by getting the 4th root of the vocabulary size as the vector size (TensorFlow Team, 2017).

$$vectorsize = \sqrt[4]{vocabularysize} \tag{3}$$

We used the essay sets feature as is. We hypothesize that advanced machine learning architectures will be able to distinguish and separate the evaluation of the essay given the information on what essay set it belongs to. This

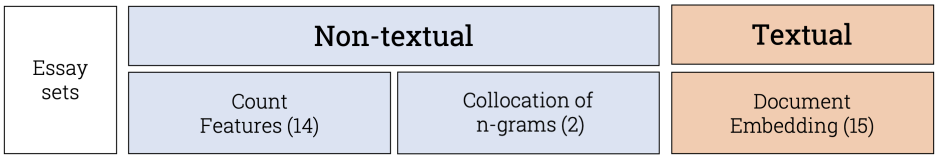

Figure 5: Extracted Features

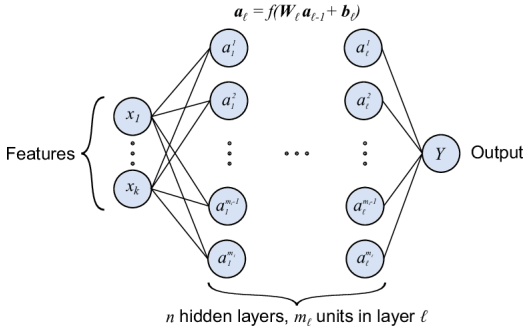

Figure 6: Schematic of a standard, fully connected Deep Neural Network (DNN). In this instance, the output is a scalar (Teichert, 2018)

is important because different essay sets have different prompts. They have different score ranges and have different rubrics for scoring. using this as a feature This gives us a total of 32 features.

4.3. Machine Learning

For the discussion of traditional machine learning methods, we start by defining notations. Let $\mathbf{X} \in \mathbb{R}^{12976\times32}$ be the matrix of features, $\mathbf{y} \in \mathbb{R}^{12976}$ be the scores and $\mathbf{f} \in \mathbb{R}^{12976}$ be the predictions made by the models.

We then tested out deep learning models with fully connected deep networks, Long Short-term Memory (LSTM) networks, and a multiple data input ensemble of fully connected deep networks and LSTM. We also tested traditional machine learning models to establish a baseline using Gradient Boosted Regressor (GBR).

To train a model, we split $\mathbf{X}$ into $\mathbf{X}_{train} \in \mathbb{R}^{9732\times32}$ and $\mathbf{X}_{test} \in \mathbb{R}^{3244\times32}$, $\mathbf{y}_{train} \in \mathbb{R}^{9732}$ and $\mathbf{y}_{test} \in \mathbb{R}^{3244}$, i.e. 75-25 train test split. To evaluate the model, we get the QWK between the R^{3244} and $\mathbf{f} \in \mathbb{R}^{3244}$ to get the agreement between the human raters and the machine learning model.

4.3.1. Fully Connected Deep Networks

Fully connected deep networks are one of the most basic types of neural networks. It is "structure agnostic", which indicates that it does have any assumption on the input to the network. This feature makes it easy to generalize through different use cases (Ramsundar et al., 2018). The structure of the model can be seen in Figure 6 (Teichert, 2018). We implemented our model through Keras. The Keras package allowed us to benefit from the optimized parameter tuning of Keras and Tensorflow (Chollet, 2015).

The input data into our fully connected deep network regressor are the 32 features generated. We implemented multiple architectures of the models. We tested wider and deeper neural networks to see which architecture would achieve the best accuracy. We increased the number of hidden units as the number of layers increased such that the hidden units will decrease for every succeeding layer. The number of hidden units was determined by 2^n.

4.3.2. Long Short-Term Memory Networks

LSTMs are used for learning. It is a special kind of recurrent neural network that are capable of learning long-term dependencies. It was specially designed to avoid vanishing and exploding gradient problems. The model is built upon a memory cell that contains three types of information gates, the input, output and forgets gates (Mikolov et al., 2014). Among the different types of LSTM applications, our implementation is a many-to-one LSTM network, which takes many features as input and outputs a single value (Karpathy, 2015).

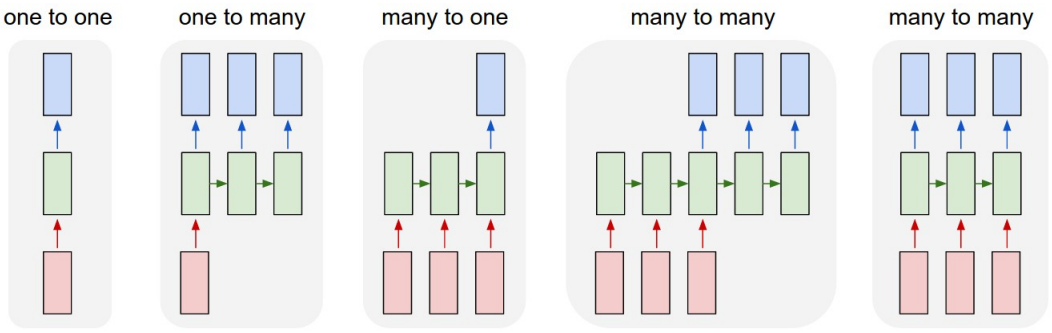

Figure 7: Types of RNN Networks (Karpathy, 2015)

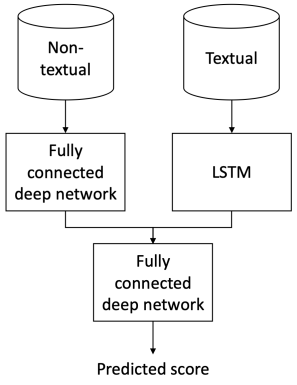

Figure 8: Neural Network Ensemble: A multi-input model to handle mixed data

The input data into our fully connected deep network regressor are only the essay set and textual features. The input data of an LSTM network is always 3-dimensional, so we reshape our features into a 3D shape given by: (samples, features, 1).

We implemented multiple architectures of the models. We tested wider and deeper LSTM networks to see which architecture would achieve the best accuracy. We increased the number of hidden units as the number of layers increased such that the hidden units will decrease for every succeeding layer. The number of hidden units was determined by 2^n.

4.3.3. Fully Connected Deep Networks and LSTM Network Ensemble

Since there are mixed data types that are inputted into our model, we tried to use a multi-input Fully Connected Deep Networks and LSTM Network Ensemble or Neural Network Ensemble (NNE) as seen in Figure 8. We separated the non-textual and the textual features as two different inputs. We concatenated the output of the two models and inputted it into another fully connected deep network. Refer to Figure 9 for the implemented NNE architecture of the model. This architecture will allow the error to be backpropagated to the different models in the ensemble during training. We implemented multiple architectures of the models. We tested wider and deeper networks to see which architecture would achieve the best accuracy. We increased the number of hidden units as the number of layers increased such that the hidden units will decrease for every succeeding layer. The number of hidden units was determined by 2^n.

4.3.4. Optimizer

To optimize the learning, we used Adam optimizer. Adam or Adaptive Moment Optimization is one of the most popular gradient descent based algorithms. It combines both RMSprop and Stochastic Gradient Descent with momentum (Valkov, 2019).

Given by its name, it computes the learning rates for each parameter from estimates of first and second moments of the gradients. Wherein the first moment pertains to the mean and the second moment is the uncentered variance

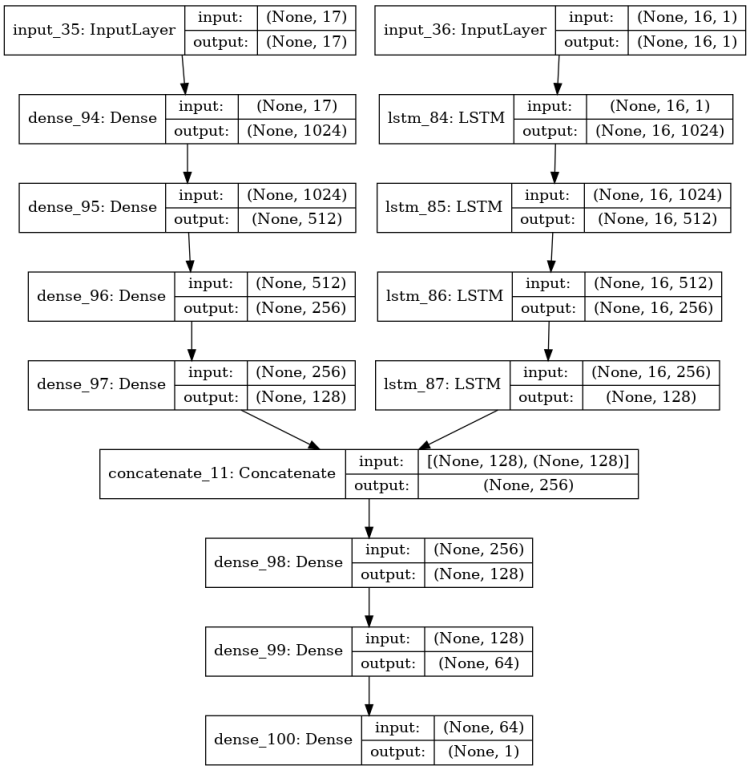

Figure 9: Fully Connected Deep Network and LSTM Network Ensemble Architecture

(Bushaev, 2018).

4.3.5. Training the Deep Learning Model

We then used 100 epochs to train our model using 10-fold cross validation to select the best model. Similar to the implementation used in GBR, the metric we use for performance is r^2 score defined by

$$r^2 = 1 - \frac{\sum_{i=1}^{973}(y_i - f_i)^2}{\sum_{i=1}^{973}(y_i - \bar{y})^2}. \tag{4}$$

Note here that we only use 973 data points as the validation set in cross validation since we take 10 percent in 10-fold CV. After obtaining the model with the best hyperparameters for each model, we use them to predict on the validation set and compute the resulting r^2 score as a measure of performance.

4.3.6. Gradient Boosted Regressor (GBR)

We wanted to try a traditional machine learning as well to establish a baseline. Friedman introduced a new regression technique called "Gradient Boosting Machines" (GBM) in 2001 (Friedman, 2001). This machine learning technique uses weak learners, typically decision trees, and combines them into a single strong learner iteratively as seen in Figure 10 (Shoaran, 2018). Weak learners are able to generalize and reduce overfitting that may occur in decision trees. In our implementation, we minimized the cost function, least squares (LS). In our implementation we used grid search to search through a given hyperparameter space.

We chose to tune the parameters max_depth, min_samples_split, max_features, and learning rate to get the best performing GBR model using a grid search. The max_depth is the maximum depth of the trees that we tune to control overfitting (Jain,2016). A lower number would result in a more generalized model. The min_samples_split is the minimum number of samples required for a node to split into branches (Jain,2016). This is tuned as well to control overfitting. Controlling this would make sure that the model will only learn relations that are evident in the samples.

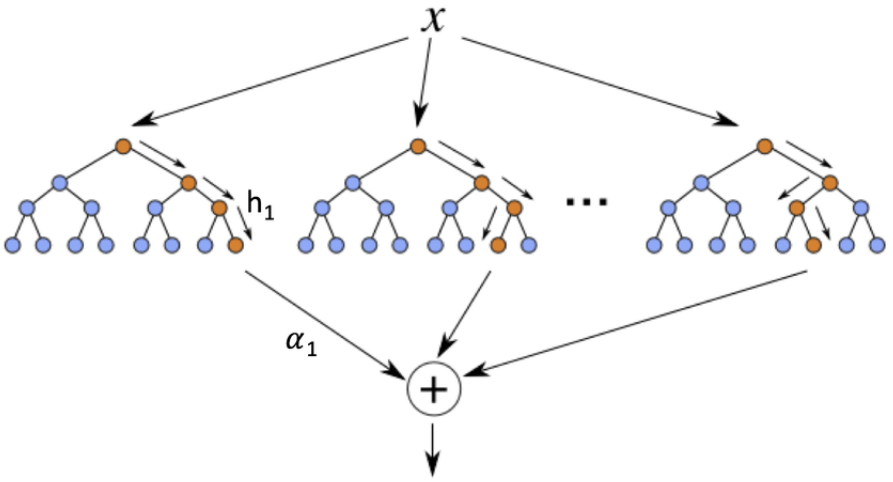

Figure 10: Schematic diagram of a boosted ensemble of decision trees (Shoaran, 2018)

The max_features is the maximum number of features to consider when searching for the best split (Jain,2016). These features are randomly selected just like in Random Forest Decision Trees. Controlling this would control overfitting and only learn from a percentage of the features. This also serves as an embedded feature reduction. The learning rate is able to indicate the contribution of each new base model and allows regularization (Hastie et al.,2001).

We then apply a grid search from a set of hyperparameters using 5-fold cross-validation to select the best model. The metric we use for performance is r^2 score defined by

$$r^2 = 1 - \frac{\sum_{i=1}^{1946}(y_i - f_i)^2}{\sum_{i=1}^{1946}(y_i - \bar{y})^2}. \tag{5}$$

Note here that we only use 1946 data points as the validation set in cross validation since we take 20 percent in 5-fold CV. After obtaining the model with the best hyperparameters for each model, we use them to predict on the validation set and compute the resulting r^2 score as a measure of performance.

To interpret the decision making of the model, we get the feature importance summary plot using SHAP (SHapley Additive exPlanations). The summary plot plots the distribution of SHAP values for every feature for every sample. Higher SHAP values towards the right indicate a more positive relationship, and lower values to the left indicate a more negative relationship. The features are ranked from the most important feature to the least. The color represents the feature value, with higher values as red and lower values as blue (Lundberg, 2020).

5. Discussion of Results

The summary of our results can be seen in Table 2 2. The NNE had the highest QWK at 0.9854. This beats the winner of the Kaggle competition in 2012, the published SSWE BiLSTM (Alikaniotis et al, 2016), and our human benchmark.

The results show that the NNE, GBR, and Fully Connected Deep Network were able to beat the human benchmark and other published state of the art models.

The results show that we can achieve the best accuracy using the multi-input data approach. This is a more complex approach by using an ensemble of two different types of learners and then combining them into one. It shows that this neural network architecture is able to learn patterns better when the input data is more uniform. This method can be useful in any other text classification tasks because there are multiple data inputs that can be extracted from text data.

However, in terms of training time, the NNE took 33 minutes to train. The GBR method which had a QWK that was only 0.06 lower only needed 6 minutes to train. In terms of computational resources, the 0.06 QWK improvement is almost negligible considering the 550% increase in training time. This is one big consideration that is required when choosing the model to be deployed.

Table 2
Model Accuracy

Method	Test Accuracy	Training Time
NNE	0.9854	33 mins
GBR	0.9848	6 mins
Fully Connected Deep Network	0.9774	2 mins
Human Benchmark	0.9693	NA
LSTM	0.9658	17 mins
SSWE BiLSTM (Alikaniotis et al, 2016)	0.96	NA
Kaggle (2012)	0.82	NA

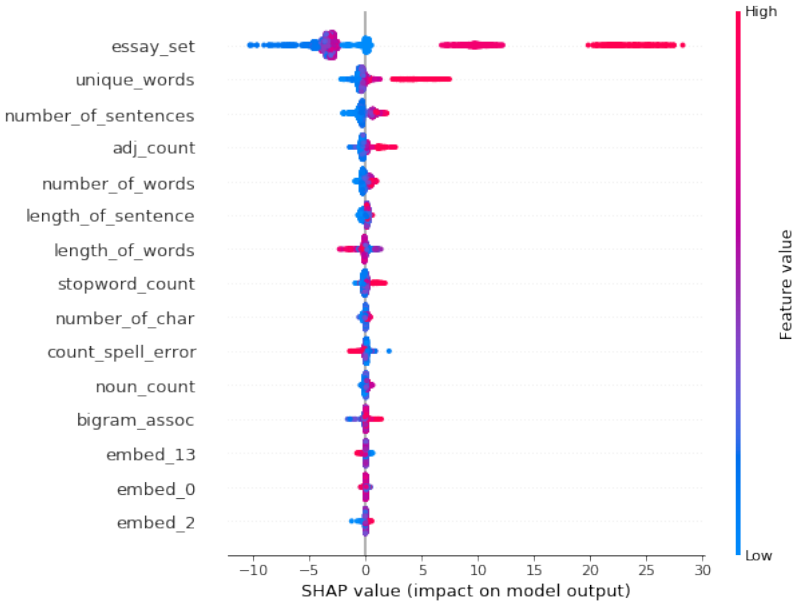

Figure 11: GBR SHAP Summary Plot

In terms of model explicability and interpretability, the NNE model cannot be interpreted easily because of the lack of open-source libraries interpreting multi-input models. The GBR method again appears to be the better choice because of its easy feature importance explicability using SHAP. Using SHAP, we were able to see the most important features in the GBR model and the polarity of the relationship for each feature.

In Figure 11, we can see that the top 3 most important features are the essay set, the number of unique words or vocabulary size, and the number of sentences. The essay set importance agrees with our EDA that the feature is indeed significant. Not only does this show the significance of the feature but it also shows the effectiveness of the model to split the data using the essay set feature. It is evident that there are some essay sets with higher feature values that have a negative impact on the prediction.

The summary plot shows that the extracted non-textual features ranked higher in terms of importance. We can see that a higher number of unique words resulted in higher scores. We also saw that a higher adjective count resulted in higher scores, which can show the vividness of the storytelling skill of the student.

Another important consideration for this study is AI ethics. The application of this model can have a large implication on education and society. Using the BNAJE framework, we can evaluate the Beneficence, Non-maleficence, Autonomy, Justice, and Explicability of the model (Floridi and Cowls, 2019). In its further development stage, it is important that the beneficence of the objective of the model is clear. Risk assessment should be done to ensure the non-maleficence of the model. Written consent must be given by the students before their essays are graded by the model. The model must be assessed to have no unintended bias and must be fair and just. And lastly, the model must

be explicable with the use of feature importance plots and other similar methods.

6. Conclusion

In this paper, we were able to develop an automated essay scoring (AES) model that gives a numerical score to unstructured text data. We did this by extracting syntactic and semantic features from the essay. We tested different machine learning algorithms to predict the score of the essays and evaluated the predictions using Quadratic Weighted Kappa (QWK). A balance between effectivity, efficiency, and explicability must be achieved, so we also evaluated the training time and explicability of the model.

Our best model, the Fully Connected Neural Network and LSTM Network Ensemble (NNE) obtained the highest QWK of 0.9854, beating the human benchmark and other published state of the art models. This demonstrates the power of using a multi-input data approach that allows the neural network to learn patterns in the data better.

In terms of computational efficiency and interpretability, we found that the GBR model was 5.5x faster to train and had better explicability by using SHAP. With the SHAP summary plot, we are able to see the impact of each feature from each sample to the trained model which makes the model interpretable and explicable to users and stakeholders.

We also discussed some AI ethics considerations that must be taken when designing and deploying the model into production. This can be done using the BNAJE framework in AI ethics.

Future improvements can be done on the model by using better document embedding techniques, training separate models for each essay set, extracting other features such as Latent Semantic Analysis (LSA) and sentiment scores. The non-textual features can also be normalized relative to the essay length and the essay scores from different essay sets can be normalized. Further research can be done on the explicability of the multi-input neural network architectures to improve the explicability of the NNE method.

Acknowledgments

We would like to thank the William and Flora Hewlett Foundation (Hewlett) and Kaggle for the dataset that was used in this study. We wish to express our gratitude to our Natural Language Processing professor at the Asian Institute of Management, Madhavi Devaraj, PhD.

References

Arora, Aman. (2019). Quadratic Kappa Metric explained in 5 simple steps. Retrieved from https://www.kaggle.com/aroraaman/quadratic-kappa-metric-explained-in-5-simple-steps

Attali, Burstein, J. (2004). Automated Essay Scoring With E-Rater® V.2.0. ETS Research Report Series, 2004(2), I-21. doi:10.1002/j.2333-8504.2004.tb01972.x

Attali, Y., Burstein, J. (n.d.). Automated Essay Scoring With E-rater® v.2.0. Retrieved from https://www.ets.org/Media/Research/pdf/RR-04-45.pdf

Barrus, Tyler. (2018). Pure Python Spell Checking. Retrieved from https://github.com/barrust/pyspellchecker

Bharath Ramsundar and Reza Bosagh Zadeh. 2018. TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning (1st. ed.). O'Reilly Media, Inc.

Boulanger, D., Kumar, V. (2018). Deep Learning in Automated Essay Scoring. Intelligent Tutoring Systems Lecture Notes in Computer Science, 294-299. doi:10.1007/978-3-319-91464-$0_3$0

Burstein, J., Chodorow, M., Leacock, C. (n.d.). CriterionSM Online Essay Evaluation: An Application for Automated Evaluation of Student Essays. Retrieved from https://www.ets.org/Media/Research/pdf/erater$_i$aai03$_b$urstein.pd f

Bushaev, V. (2018, October 24). Adam-latest trends in deep learning optimization. Retrieved from https://towardsdatascience.com/adam-latest-trends-in-deep-learning- optimization- 6be9a291375c.

Chen, H., He, B. (2013). Automated Essay Scoring by Maximizing Human-Machine Agreement. EMNLP.

Chen, Huang, H., Tian, S., Qu, Y. (2009). Feature selection for text classification with Naïve Bayes. Expert Systems with Applications, 36(3), 5432-5435. doi:10.1016/j.eswa.2008.06.054

Chollet, François. (2015). Keras. Retrieved from https://github.com/fchollet/keras

Cozma, Butnaru, A., Ionescu, R. T. (2018). Automated essay scoring with string kernels and word embeddings. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). doi:10.18653/v1/p18-2080

Floridi, L., Cowls, J. (2019). A Unified Framework of Five Principles for AI in Society. Harvard Data Science Review, 1(1). https://doi.org/10.1162/99608f92.8cd550d1

Forman, G. (2003). An Extensive Empirical Study of Feature Selection Metrics for Text Classificatio. JMLR.org.

Friedman, J. (2001). Greedy boosting approximation: a gradient boosting machine. Ann. Stat. 29, 1189–1232. doi: 10.1214/aos/1013203451

Garla N., Brandt, C. (2012). Ontology-guided feature engineering for clinical text classification. Journal of Biomedical Informatics, 45(5), 992-998. doi:10.1016/j.jbi.2012.04.010

Greene. (2018, July 04). Automated Essay Scoring Remains An Empty Dream. Retrieved April 07, 2020, from https://www.forbes.com/site essay-scoring-remains-an-empty-dream/31221a1574b9

Hastie, T., Tibshirani, R., Friedman, J. (2001). The Elements of Statistical Learning. New York, NY, USA: Springer New York Inc..

Howard, Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). doi:10.18653/v1/p18-1031

Ikonomakis, E., Kotsiantis, S., Tampakas, V. (2005). Text Classification Using Machine Learning Techniques. WSEAS Transactions on Computers.

Jain, Aarshay. (2016). Complete Machine Learning Guide to Parameter Tuning in Gradient Boosting (GBM) in Python. Retrieved from https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/

Joulin, Grave, E., Bojanowski, P., Mikolov, T. (2017). Bag of Tricks for Efficient Text Classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. doi:10.18653/v1/e17-2068

Karpathy, A. (2015). The Unreasonable Effectiveness of Recurrent Neural Networks

Kim, Han, K., Rim, H., Myaeng, S. H. (2006). Some effective techniques for naive bayes text classification. IEEE Transactions on Knowledge and Data Engineering, 18(11), 1457-1466. doi:10.1109/tkde.2006.180

Le, Quoc Mikolov, Tomas. (2014). Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14). JMLR.org, II–1188–II–1196.

Lewis, D. D. (1995). Evaluating and optimizing autonomous text classification systems. Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '95. doi:10.1145/215206.215366

Lundberg, S.M., Erion, G., Chen, H. et al. From local explanations to global understanding with explainable AI for trees. Nat Mach Intell 2, 56–67 (2020). https://doi.org/10.1038/s42256-019-0138-9

Mikolov, Tomas Joulin, Armand Chopra, Sumit Mathieu, Michael Ranzato, Marc'Aurelio. (2014). Learning Longer Memory in Recurrent Neural Networks.

Pedregosa et al. (2011). Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830, 2011.

Phandi, Chai, K. M., Ng, H. T. (2015). Flexible Domain Adaptation for Automated Essay Scoring Using Correlated Linear Regression. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. doi:10.18653/v1/d15-1049

Řehůřek, Radim Sojka, Petr. (2010). Software Framework for Topic Modelling with Large Corpora. 45-50. 10.13140/2.1.2393.1847.

Rogati, Yang, Y. (2002). High-performing feature selection for text classification. Proceedings of the Eleventh International Conference on Information and Knowledge Management - CIKM '02. doi:10.1145/584792.584911

Smith. (2018, June 30). More States Opting To 'Robo-Grade' Student Essays By Computer. Retrieved April 07, 2020, from https://www.npr.org/2018/06/30/624373367/more-states-opting-to-robo-grade-student-essays-by-computer?utm$_s ource = facebook.comutm_m socialutm_campaign = nprutm_term = nprnewsutm_content =$ 20180630

Srivatsa S., Thyagarajan A., Bhomick P. (n.d.). Regression based Automated Essay Scoring.

Shoaran, Mahsa Haghi, Benyamin Taghavi, Milad Farivar, Masoud Emami, Azita. (2018). Energy-Efficient Classification for Resource-Constrained Biomedical Applications. IEEE Journal on Emerging and Selected Topics in Circuits and Systems. PP. 1-1. 10.1109/JETCAS.2018.2844733.

Taghipour, Kaveh Ng, Hwee. (2016). A Neural Approach to Automated Essay Scoring. 10.18653/v1/D16-1193.

TensorFlow Team. (2017). Introducing TensorFlow Feature Columns. Retrieved from https://developers.googleblog.com/2017/11/introdu tensorflow-feature-columns.html

Teichert, Gregory Natarajan, A. Ven, A. Garikipati, Krishna. (2018). Machine learning materials physics: Integrable deep neural networks enable scale bridging by learning free energy functions.

The Hewlett Foundation: Automated Essay Scoring. (2012). Retrieved from https://www.kaggle.com/c/asap-aes/

Tong S., Koller D. (2002). Support Vector Machine Active Learning with Applications to Text Classification. JMLR.org.

Toth. (n.d.). Retrieved April 07, 2020, from http://journalofwritingassessment.org/article.php?article=69

Uysal K., Gunal, S. (2014). The impact of preprocessing on text classification. Information Processing Management, 50(1), 104-112. doi:10.1016/j.ipm.2013.08.006

Valkov, V. (2019, November 17). Demand Prediction with LSTMs using TensorFlow 2 and Keras in Python - Adventures in Artificial Intelligence: Venelin Valkov. Retrieved from https://www.curiousily.com/posts/demand-prediction-with-lstms-using-tensorflow-2-and-keras-in-python/

Zhao, Zhang, Y., Xiong, X., Botelho, A., Heffernan, N. (2017). A memory-augmented neural model for automated grading. Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale - L@S '17. doi:10.1145/3051457.3053982