Automated Essay Grading using Long Short Term Memory Networks

Anmol Preet Singh
2K18/SE/027
Delhi Technological University
Delhi, India
anmolpreetsingh_2k18se027@dtu.ac.in

Chitra Singla
2K18/SE/048
Delhi Technological University
Delhi, India
chitrasingla_2k18se048@dtu.ac.in

***Abstract* — Automatic essay grading is the process of automatically evaluating essays and giving them a certain score or grade. Our project aims to build a system that can evaluate the essays in a very efficient way based on various aspects such as vocabulary, tense, active and passive voice, grammatical and spelling errors, and sentence lengths. The dataset is taken from a past competition held on Kaggle.com sponsored by William and Flora Hewlett Foundation in 2012, named Automated Student Assessment Prize (ASAP). It contains 12975 essay samples in total divided into 8 sets based on the type of essays. We extracted only those words (tokens) from the essays that were not present in the stop words and identifiers for NLP (Natural Language Processing). After that, we tokenized the essays into sentences then further into words and made feature vectors (Word Embeddings) from them using different word embedding models. Hold-out cross-validation is used as a validation technique because of the large size of our dataset and Long Term Short Memory (LSTM) network is used to train and test the model. Quadratic mean average kappa score is used as a performance measure to find errors between the actual scores and predicted scores of the essays. Our model has given the best kappa score of 0.97207 till now which is significantly more than the earlier 0.81407 kappa score of the ASAP competition of Kaggle.**

Keywords — embedding, variance, corpus, neural network, LSTM, NLP, RNN, CNN

I. INTRODUCTION

Automated essay grading can be defined as a system that evaluates an essay using specialized computer programs and assigns a score or grade for the essay. The systems access some aspects of writing skills like vocabulary, tense, the active and passive voice of essay, grammatical and spelling errors, and sentence lengths, and syntactic and semantic meanings of an essay and evaluate them.

Automated Essay Grading is used in educational assessment and it is an application of machine learning and natural language processing. The main objective of this system is to classify a large set of textual entities into a small number of discrete categories, and these categories correspond to the assigned scores for the essay, for example, the numbers 1-5. Accurate models of automated essay grading will help to reduce the human effort and error in checking and grading essays and will provide valuable feedback easily to the authors. Every year, thousands of students in schools and universities write essays on the same topics and it becomes very difficult and time consuming for the teachers to evaluate every essay and then provide feedback to the students. Feedback is very important for the students to improve their writing skills. So this system will help teachers to efficiently evaluate the essays in very less time and also to give quick feedback to the students.

II. RELATED WORK

In this section, we described many approaches used in automated essay grading in the past.

In 2012, Kaggle organized an Automated Student Assessment Prize (ASAP) competition which was sponsored by William and Flora Hewlett Foundation (Hewlett), which aims to find a system that can find better automated essay scoring systems. The dataset released contains around 13000 essays written by american school students, and the same essays are used in our system [1].

Alikaniotis, Yannakoudakis, Rei (2016) used Long-Short Term Memory networks to represent the meaning of texts and demonstrated that a fully automated framework can achieve good results. They introduced a method for identifying the regions of the text that the model has found more discriminative. Their system achieved a kappa score of 0.96 [2].

Taghipour, Tou Ng (2016) developed an approach based on Recurrent Neural Networks to learn the relation between an essay and its assigned score, without any feature engineering. In their model, LSTM outperforms the RNN network in terms of quadratic kappa score [3].

Nguyen, Dery also used the dataset from Kaggle past competition and two-layer neural networks and three different Long-Short Term Memory networks and their model was able to achieve the best kappa score of 0.9447875 using 300-dimensional GloVe word embedding [4].

Madala, Gangal, Krishna, Goyal proposed an approach for evaluating the essays based on feature selection and ranking techniques, some surface level, and deep linguistic features, and 4 text classification algorithms [5].

Song, Zhao evaluated the essays using Regression Tree, Linear Regression, Linear Discriminant Analysis, and SVM and among all these, Regression Trees achieved the best results with a Kappa score of 0.52 [6].

III. EXPERIMENT DESIGN

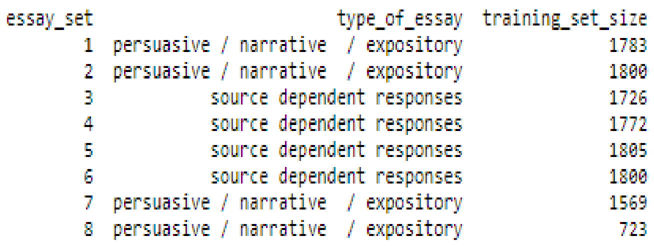
A. Dataset

The dataset used in this system is taken from Kaggle.com. In 2012, Kaggle held a competition Automated Student Assessment Prize (ASAP) which was sponsored by William and Flora Hewlett Foundation (Hewlett).

There are 8 sets of essays present in the data set and the average length of the essays varies from 150 to 550 words. It contains around 13000 essays in total and all essays are written by American students of grades 7 to 10. All the essays are hand graded and are double scored i.e. scores from 2 checkers are provided for each essay.

The Hewlett Foundation removed the personal information from the essays using the Named Entity Recognizer (NER) from the Stanford Natural Language Processing group. They removed “LOCATION”, “DATE”, “NAME”, “TIME”, etc and added a string such as “@LOCATION1” in place of those entities.

The essays in the different sets are of different types and also the count of essays differs in each set. The following image shows the type and count of essays.



```
essay_set                        type_of_essay  training_set_size
        1  persuasive / narrative  / expository               1783
        2  persuasive / narrative  / expository               1800
        3           source dependent responses                1726
        4           source dependent responses                1772
        5           source dependent responses                1805
        6           source dependent responses                1800
        7  persuasive / narrative  / expository               1569
        8  persuasive / narrative  / expository                723
```

Table - 1 (Essay Descriptions)

The dependent variable in our model is only one - domain1_score. It is the score given for each essay in every set. Independent variables include tokenized and cleaned essays, minimum and maximum scores given to a certain essay set to define the range of scores for evaluation. The maximum score for a certain essay is 60 and the minimum score is 0.

The whole dataset is divided into two parts training and testing data in the ratio 7:3 which means 70% of the data is used for training of the model and the rest 30% of the data is used for testing the system.

B. Data Pre-Processing

The data taken from the dataset is clean and has no outliers present. It also has some special characters to identify personally-identifying information.

In data preprocessing, we performed attribute reduction by removing empty columns and those columns whose values for all essays were not given. Then we converted essays to feature vectors by word tokenizing them so that they can be fed to RNN.

For converting essays into feature vectors, we removed the tagged labels and word tokenized the sentences, then removed stopwords using the NLP library to enhance the results and tokenized the essay into sentences and then each sentence into words. After that Feature Vectors are made from the words list of an Essay using the traditional Word2vec model.

For data analysis, we used principal component analysis and it is used to simplify data of higher dimensions into lower dimensions. It searches for a linear combination of variables so that maximum variance can be extracted from variables. From the data, a correlation matrix was made. Then we computed eigenvalues and eigenvectors, sorted them in descending order, and took maximum 2 values as our new features as they explained maximum variance for projecting.

Variance Of Components:
[14.01, 10.88, 7.82, 6.86, 5.66]
Cumulative Variance:
[14.01, 24.90, 32.72, 39.58, 45.25]

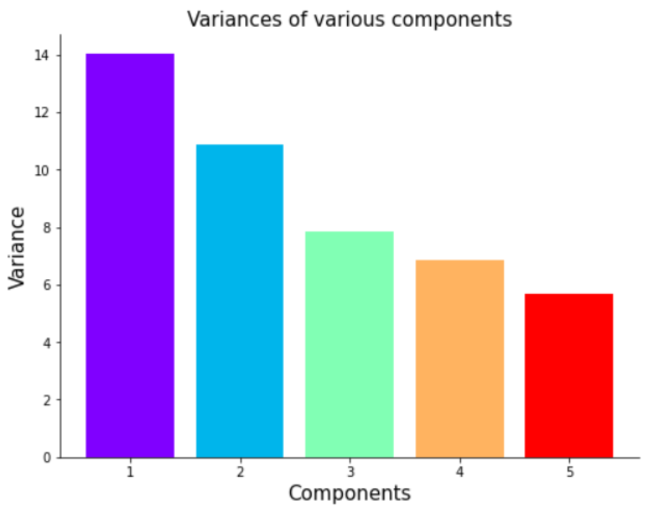


Fig - 1 (Variance of top 5 components)

C. Word Embedding

Word Embedding is a way to represent words such that similar words can be identified easily by machine learning algorithms syntactically or semantically. They help in feature generation, document clustering, text classification, and natural language processing tasks.

Some types of word embeddings are Word2vec, GloVe, FastText, Tf-Idf, Bert, Elmo.
Gensim Python library is used for word embeddings.
We have used 5 types of word embedding models, they are as follow:
Word2vec, pre-trained Word2vec, GloVe, FastText, and pretrained FastText.

Word2Vec
Word2vec is a two-layer neural network used to create word representation in vector space. It captures syntactic and semantic word relationships and reconstructs the linguistic context of words.
There are two main types of Word2Vec models that are used for producing word representation:
Continuous Bag of Words (CBOW) - In this type of word2vec word embedding, the order of the words does not contribute to the result as this model uses the current word to predict the results.
Continuous skip-gram - In this model, more weightage is given to nearby context words than distant context words, and each of the context vectors is weighed and compared independently.
Pre-Trained Word2Vec word embeddings are trained on Google news dataset and it consists of 100 billion words.

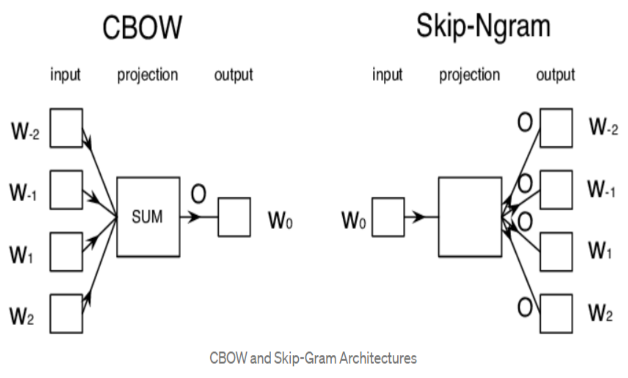


Fig - 2 (CBOW and Skip-N gram)

GloVe
Word2Vec word embedding only takes local contexts into account to predict, not global context. GloVe (Global vectors for word representation) embeddings look for context terms in some defined area and give less weight to distant words.
It is a type of pre-trained word embedding model trained on the Wikipedia dataset which consists of around 6 billion words.

FastText
FastText words embedding is similar to word2vec by the Facebook research team but besides, it also contains embeddings for n-grams that help in data sets without vocabulary words. Pre-trained FastText is trained on a wiki news dataset and contains around 1 million word vectors.

We have shown how the concept of transfer learning can result in more efficient and faster results. We have used 300 as a vector dimension because the accuracy remains constant after 300 even if we increase the vector dimension. So 300 acts as a threshold.

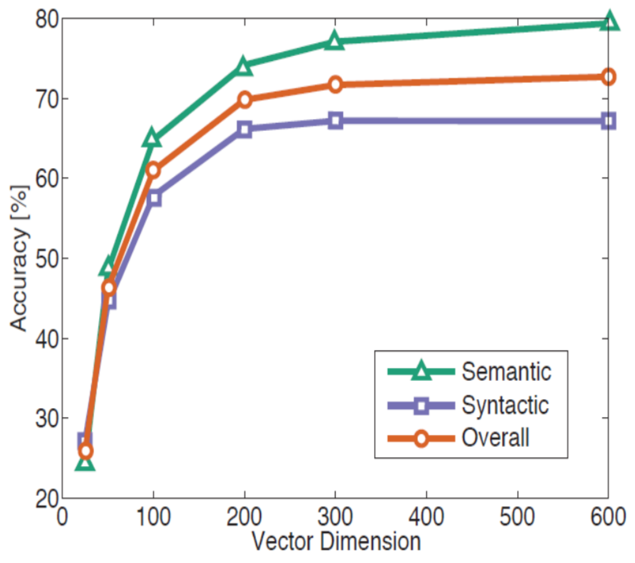


Fig - 3 (Vector Dimension vs Accuracy)

IV. RESEARCH METHODOLOGY

A. Performance Measure

For validation, hold out cross-validation is used because of the large size of the dataset, and the data is divided into training and testing or validation data in the ratio 7:3 that means 70% of the data is used for training of the model and the rest 30% is used for testing the model.
Quadratic mean average kappa score is used as a performance measure. It provides a score that compares the actual scores and the predicted scores of the essays. It is defined as follows:

$$\kappa \equiv \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e},$$

Fig - 4 (Kappa Score)

Here, po is the relative observed agreement of scores provided by raters and pe is the probability of the predicted scores.
Quadratic mean average kappa takes two equal-length lists, one is of actual scores and the second is of predicted scores of the essays and outputs a value between -1 and 1, where -1

denotes complete disagreement, 1 complete agreement, and 0 random agreement. It is very efficient to measure inter-rater reliability.

B. *Why MLP is not used*

The Multilayer perceptron is a feed-forward artificial neural network that uses backpropagation as a supervised learning technique. MLPs learn fixed-function approximation and are unaware of the temporal structure of the input. They have messy scaling and the size of the sliding window in it is fixed and must be imposed on all inputs to the network. The size of the output is also fixed and any outputs that do not conform must be forced, so their rigid behavior makes them more useful in normal classification problems instead of text classification.

C. *Why RNN is not used*

Recurrent Neural Networks (RNN) suffer from short-term memory and if the sequences are very long, they do not carry earlier information to the next time steps and may leave out the important information while processing.
They also suffer from vanishing gradient problems during the backpropagation, which means that the gradient shrinks as the network back propagates. Gradients are those values that are used to update the weights in the network and if they become very small, they will not make any changes in the weights and hence do not contribute much.

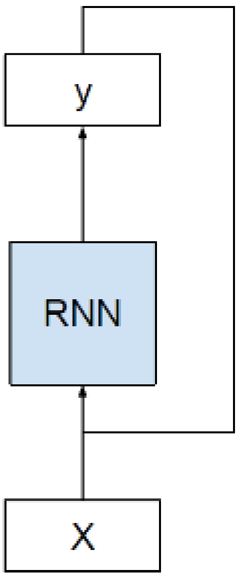


Fig - 5 (RNN)

D. *LSTM*

The Long Short-Term Memory (LSTM) network is a type of Recurrent Neural Network (RNN) that is designed for solving sequence-related problems.

An essay is a sequence of words and the meaning of a sentence is defined by its sequence only, if the sequence of words changes then the meaning will also change. So, we are supposed to classify sequences here and because of that LSTM model is chosen as the best alternative.

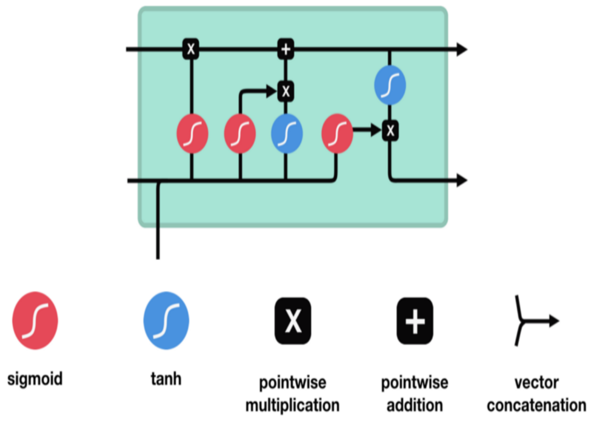


Fig - 6 (LSTM) [7]

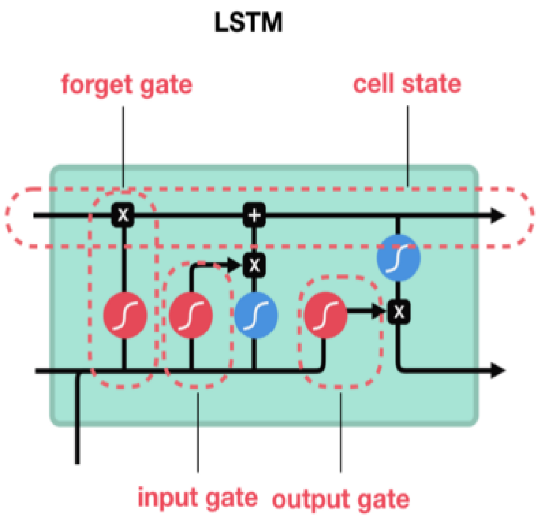


Fig - 7 (LSTM Gates) [7]

LSTM consists of 3 gates - a forget gate, cell state, input, and output gate.

Forget Gate - This is the first gate that is encountered by the input and it decides whether the information should be kept or thrown away. After that, the information from the current input and the previously hidden gate is passed through a sigmoid function and it returns a value between 0 and 1, if the value is closer to 0 then the gate forgets it and if closer to 1 then keeps it.

Input Gate - This gate helps to update the cell state. After passing through the sigmoid function, the current state, and the hidden state through the tanh function to get the output between -1 and 1 which helps in regulating the network. Then the output from tanh is multiplied with the output of the sigmoid function and it will help to decide whether the information is important or not.

Cell State - The cell state is first multiplied to the forget vector and then the output of the input gate is added to it which gives the new cell state.

Output Gate - The last gate is the output gate which decides the next hidden state. The output we get after the input gate is the hidden state and the new cell state and hidden state are carried to the next time step.

E. Working of LSTM

LSTM is a supervised learning algorithm in which the networks backpropagate. In this algorithm, we modify the weights of the neural network to minimize the error in the outputs corresponding to the given inputs. The algorithm is as follows:

1. Fed the inputs to the network and gets output corresponding to it.
2. Calculate the error by comparing the outputs to the expected outputs.
3. After that calculate the derivative of the error for weights of the network.
4. Then adjust the current weights to minimize the error.
5. Repeat the above steps

We have used a Long Short Term Memory (LSTM) network of 3 types – Dual-layer LSTM, Bidirectional LSTM, and CNN LSTM.

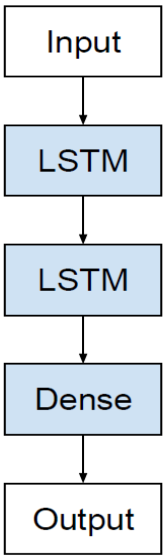


Fig - 8 (Dual-Layer LSTM)

The Bi-directional LSTM model is an extension of the traditional LSTM network. It is used for evaluating the input in both sequential orders and where timestamps are available.

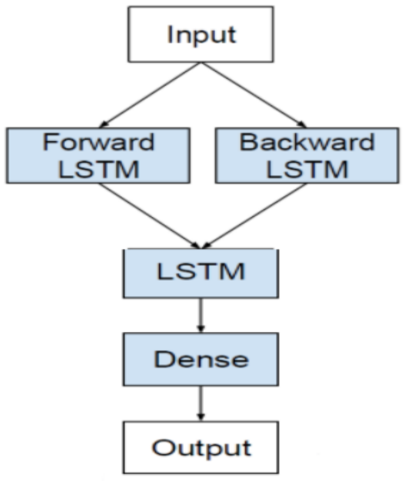


Fig - 9 (Bidirectional LSTM)

The CNN LSTM model involves Convolutional Neural Network (CNN) layer before the LSTM layer and the CNN layer is used for feature extraction on the input data and the LSTM network is used to support sequence prediction.

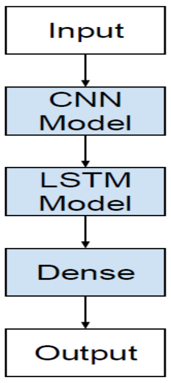


Fig - 10 (CNN LSTM)

Relu is used as an activation function because we have not normalized training labels in the data.
It is a piecewise linear function in which the output is equal to the max of input and 0, which means if the input is greater than 0 then output is equal to input and if the input is smaller than 0 then output is equal to 0.

A dropout of 40% is used after every layer. This is done to prevent overfitting of models to our training dataset.

V. RESULTS AND ANALYSIS

A. Dataset

The dataset is taken from a past competition ASAP of Kaggle.com. Around 13000 essays are present in the dataset and all are graded by humans.
All the essays are divided into 8 sets and their average length varies from 150 to 550 words and the score ranges from 0 to 60.

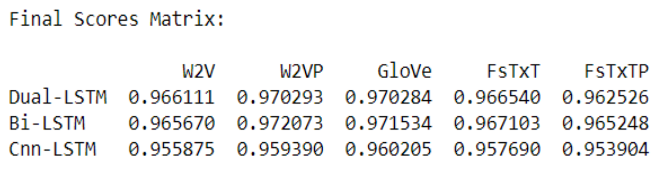
B. Techniques Used

In our system, essay grading is done by using Long short term memory networks in which feature vectors made by word tokenizing the essay are provided as input, and a score is generated for that essay.

C. Comparison

The comparison is done by using a quadratic mean average kappa score.

The following image shows the quadratic mean average kappa scores of all the 5-word embeddings – W2V, W2VP, GloVe, FsTxT, FsTxTP, and 3 LSTM models Dual-LSTM, Bi-LSTM, and Cnn-LSTM.



```
Final Scores Matrix:

                W2V      W2VP     GloVe     FsTxT    FsTxTP
Dual-LSTM  0.966111  0.970293  0.970284  0.966540  0.962526
Bi-LSTM    0.965670  0.972073  0.971534  0.967103  0.965248
Cnn-LSTM   0.955875  0.959390  0.960205  0.957690  0.953904
```

Table - 2 (Final Scores Matrix)

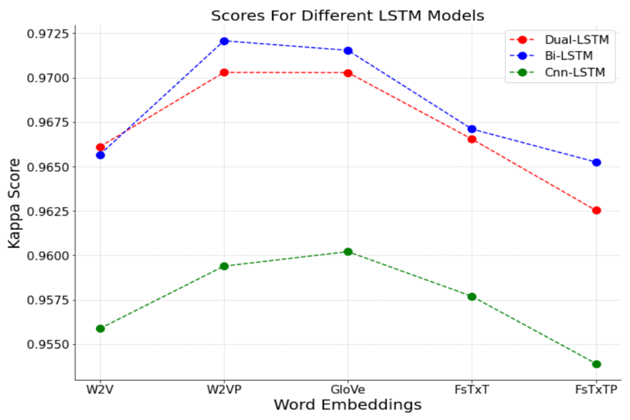


Fig - 11 (Word Embedding vs Kappa Score)

The above plot shows the values of the Quadratic mean average kappa score for all the word embeddings and 3 LSTM models. Our model is giving the best Quadratic Mean Average Kappa Score as 0.97207 when pre-trained Word2Vec word embedding is used in the Bi-Directional LSTM model. During the Automated Student Assessment Prize (ASAP) competition, the best Kappa score achieved was 0.81407.

VI. CONCLUSION AND FUTURE WORK

Automated essay grading is a very important application of machine learning and natural language processing. It has been studied many times in the past and also a competition was held on Kaggle.com in 2012. The approach in our current model uses various language features such as grammatical correctness, vocabulary, tense, the domain information content of the essay, and the comparison is done using a variety of word embeddings and long short term memory (LSTM) models.

Our model has achieved a higher value of quadratic mean average kappa score than the kappa score of the Automated Student Assessment Prize (ASAP) Kaggle Competition. The system gives the best score of 0.97207 on the Quadratic mean average kappa metric and the best kappa score of the winner of the competition is 0.81407. This score is achieved by the Bi-directional LSTM model using 300 dimensional pre-trained word2vec embedding layer and it shows the vast potential of neural networks to solve natural language processing problems.

The future scope of automated essay grading can extend in various dimensions. Good semantic and syntactic features can be taken into account while grading and for this, various semantic parsers, and sentence embeddings like Doc2Vec, SentenceBERT, InferSent, and Universal Sentence Encoder by Google can be used.

Moreover, the weights of our current model can be stored and used in real-time Automated Essay Scoring applications such as a website.

VII. REFERENCES

[1] Automated Student Assessment Prize (ASAP) Kaggle Competition - Develop an Automated Scoring Algorithm for Student-written Essays 2012. https://www.kaggle.com/c/asap-aes

[2] "Automatic Text Scoring Using Neural Networks", Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei, University of Cambridge Cambridge, UK, 2016

[3] "A Neural Approach to Automated Essay Scoring", Kaveh Taghipour, and Hwee Tou Ng, National University of Singapore 13 Computing Drive Singapore 117417, 2016

[4] "Neural Networks for Automated Essay Grading ", Huyen Nguyen, and Lucio Dery, Stanford University

[5] "An empirical analysis of machine learning models for automated essay grading ", Deva Surya Vivek Madala, Ayushree Gangal, Shreyash Krishna, Anjali Goyal, and Ashish Sureka, Ashoka University, 2018

[6] "Automated Essay Scoring Using Machine Learning", Shihui Song, and Jason Zhao, Stanford University

[7] Long Short Term Memory (LSTM) network image https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf2

[8] RNN and LSTM models images https://machinelearningmastery.com/lstms-with-python/

Contribution

Both of us have studied 3-4 research papers individually and decided on word embeddings, models, and data of the project. In the coding part, both have contributed equally. Coding was done using a git repository as a version control system.