**SMART GRADING SYSTEM**

**Prachi Channe, Shruti Diwate, Dnyanada Mahajan, Sneha Kharate**

School of Computer Engineering and technology, MIT Academy of Engineering, Pune

**Abstract**

The understanding of problem-solving skills of students is done by textual assignments given by teachers. When professors provide individual remarks on assessment, students can learn from their mistakes. Courses with hundreds of students, on the other hand, put a lot of pressure on professors because delivering individual feedback is largely a manual, repetitive, and time-consuming task. We decided to create a system using the machine learning approach for suggesting computer-assisted grading in various tasks provided by teachers. Topic modelling is used to break down student responses into text segments, which are then transformed using language embeddings. The text segments are then clustered based on their similarity, allowing the same feedback to be applied to all similar segments inside the same cluster

**Keywords:** Machine learning,Neural Networks, LSTM, RNN, Natural Language Processing, Grading.

**1. Introduction**

In learning process, grading and feedback plays an important part. It involves the evaluation and assessment of assignments given by students. Examination have become effective way of assessing the students’ learning. It can be performed using many types of questions like MCQ (Multiple Choice Questions), short answers and essay questions [1]. There are number of papers and projects concerned with assisting grader or teachers in evaluating assignments of their students. Providing manual assessment is much difficult for essay and short answer type questions. Those question require free text responses. It is widely used in a learning process because of its effectiveness in building cognitive skill of students and demonstration of knowledge. Also, scores may vary from marker to marker. Hence, textual understanding and analysis is required in order to provide scores whereas MCQ type question does not require understanding instead selecting the correct option is enough, which is much easier to evaluate [5].

Many a times the students look for feedback along with the graded of their assignment from teachers [1][2]. This helps them in improving their answering skills and avoid mistakes. But evaluating plenty of student’s answers can be burdensome for teachers. Manual assessment of each answer may result in inconsistency. In requires marker to infer meaning from students’ own words, and when it comes to providing feedback to a large group, the assessment can become pretty heavy and time consuming from the teacher’s perspective [1]. At the same time, scoring the same answer can vary from person to person. So, creating a system that is efficient in both grading the performance as well as providing the feedback for student’s assignments and which also takes less time to do it is necessary.[5]

**2. Related Work**

Using machine learning and natural language processing, this study created a simple grading system that achieves a quadratic weighted kappa of 0.70026. It represented the essay and transformed it into its vectorized representation and tokenized sequences using both feature extraction and word vectors. The tokenized sequences are then evaluated using an LSTM neural network, while the vector representation is evaluated using a 2-layer neural network. Another 2-layer neural network is used to forecast the final score after the results are concatenated.

The standard attained during the Automatic Essay Grading Kaggle Competition has been significantly raised by this model. In comparison to the previous best of 0.81407, the top neural network models scored 0.9447875 on the quadratic weighted kappa metric. This demonstrates the enormous potential of neural network topologies to address issues with natural language processing. It was unexpected that the most successful model was a straightforward neural network model employing a 300-dimensional Glove as the embedding layer initialization. A more cautious hyperparameter search using LSTM-based models might perform better than this outcome.

In this study, a novel framework for automated assessment and reporting was suggested and put into practice. It combines supervised deep learning models with an unsupervised MCMC sampling technique. The performances of three models, CNN, CNN+LSTM, and CNN+Bi-LSTM, on AES tasks within the same context were specifically compared in this work. The outcomes showed that of the three algorithms, CNN+LSTM performed the best on the AES tasks. Additionally, on seven out of eight writing tasks, the CNN+LSTM outperformed the baseline model, highlighting the effectiveness of deep learning models with word embeddings for automated essay scoring.

**3. Smart Grading and Feedback System**

**3.1 Task description**

The goal of this research is to create an intelligent, machine learning-based system that could grade essays autonomously. In order to guarantee consistency among the raters, dataset should be choosed carefully that contains a significant number of essays on a given topic that have already been marked by at least two separate qualified rater’s. Once the dataset is selected, separate the features from the essays in order to feed them to the algorithm. In order to extract features such as word count, sentence count, sentence to word ratio, and many more,focused on Python's Natural Language Toolkit (NLTK). The artificial intelligence algorithm will then be used to predict the scores.

In this the first step is to perform pre-processing which contains the following modules:

Pre-processing:

* Spell checker and corrector: Incorrect spellings are identified in this module and counted towards the preliminary score. The words are then replaced with closet correct spellings and passed to the next module. Number of spelling errors is counted and deductions are according to the rubrics given.
* Grammatical errors: This module checks for correctness of grammar in the essay. If a sentence doesn't follow the rule, then it's considered as a grammatical error. One important rule to check here is verb tense agreement, which is a common grammatical mistake among students. Our implementation of grammar checks for such inconsistencies and scores are penalized according to the rubric.
* Punctuations: Here we check the number of punctuation errors. We have considered the following types - full stop, comma, question mark, single and double quotes.
* Lemmatization: This module takes corrected words from the spell corrector and converts them to its root word. This is done to reduce the complexity of processing while maintaining the meaning of the essay. For grammatical reasons, documents are going to use different forms of a word or maybe related words with similar meanings. In many situations, it seems as if it would be useful for a search for one of these words to return documents that contain another word in the set. For implementing lemmatization, we have used the inbuilt NLTK methods which provides the word in its root form based on morphological analysis which is then replaced in the original word in the word vector.
* Removal of stop words: A stop word is a commonly used word such as the, a, an, in which does not add meaning to the sentence. Thus, they can be removed safely without altering the overall meaning. This also helps to reduce the complexity due to decrease in the number of words. We have used NLTK library here for POS tagging. We then perform a linear search on the word to identify words with tags as noun, verb, adjective or adverb. All other tags are referring to stop words and should be purged in the modified word vector.
* The next step is to make a Word2Vec model: In our model, each given essay consists a number of words, and each word represent by a word embedding according to word2vec. The embedding representations are expected to catch the semantic information carried by each word, i.e. the words with similar meanings will near to each other in the vector space. For example, from our corpus, we can find that "computer" similar to "laptop".
* In a sequential manner, perform spell check, then correction were performed. After spell check scores are deducted. After spell corrector, stopword removal followed by lemmatization was performed. The result is a useful cleaned essay which will be used for complex features. In parallel, punctuation checking, grammar checking and identifying sentence proportions was performed all of which will help deduct scores further. These deductions will be combined in the score module and returned to the main application. The product of pre-processing is a clean version of the input essay and a deduced score after considering all features
* The algorithm used is bidirectional LSTM (Long Short Term Memory) which is a sequential model. This is built on top of a word2vec model and the inputs are passed in two ways. The first is in normal order from left to right and the second is reverse order from right to left. We have used ReLU activation function in order to get the outputs which is a score. The model is compiled on the basis of mean squared error loss. We have used five-fold cross validation in order to train the data

**3.2 Activation Function:**

Layers of nodes make up a neural network, which may be trained to map instances of inputs to outputs. The inputs are multiplied by the weights of a node and added together for a certain node. The node's total activation is known as this value. The activation total is then modified using an activation function, which determines the node's specific output or "activation.". The simplest activation function, where no transform is used at all, is known as the linear activation. A network that exclusively uses linear activation functions can be trained relatively quickly, but it is unable to learn complex mapping functions. It is preferable to use nonlinear activation functions since they enable the nodes to understand more intricate data structures. The sigmoid and hyperbolic tangent activation functions are two widely utilised nonlinear activation functions. The sigmoid and tanh functions both saturate, which is a general issue. This indicates that for tanh and sigmoid, high values snap to 1.0 and small values snap to -1 or 0. Additionally, sigmoid and tanh are only really sensitive to input changes around their midpoints, or 0.5 and 0.0 respectively. That’s why we decided to use ReLU activation function in our model.

ReLU is most commonly used activation function in machine learning models. Any negative input causes the function to return 0, but any positive value x causes it to return that value. Thus, it may be expressed as:

f(x)=max (0, x)

**3.3 Evaluation Metrics**

Quadratic Weighted Kappa

A set of predictions and a set of multiclass labels are measured by the Quadratic Weighted Kappa index. It attempts to take into account the similarity between the classes, beyond only the class, rather than just focusing on the precision of the match between predictions and labels. In the context of clinical AI, it is particularly helpful. In addition to label distance, it also considers the likelihood that inter-rater agreement may happen randomly, or by chance. The outer product of the histograms for the actual class labels and predicted class labels is used to compute the agreement by chance. Here is an easy case in point. The difference between an image being graded a 3 and 4 instead of a 0 and 5 when ophthalmologists classify diabetic retinopathy is evident. It would be higher in the former scenario while being lower in the latter.

Mean squared error (MSE)

You may determine how closely a regression line resembles a set of points using the mean squared error (MSE). This is accomplished by squaring the distances between the points and the regression line (also known as the "errors"). The squaring is required to eliminate any unfavourable indications. Additionally, it emphasizes bigger discrepancies. Since you're averaging a collection of errors, this error type is known as the mean squared error. The lower the MSE, the better the forecast.

**4. System Design**

This section introduces the proposed methodology , we have implemented Recurrent Neural Network (RNN) which are most commonly used neural network for problem solving by researchers. Basically, RNN is type of Neural Network which uses the output from previous layer to fed as input to the next layer. Paper consists of implementation and observation of the comparative working of Long Short-Term Memory (LSTM), Bidirectional-LSTM (Bi-LSTM) and Bi-LSTM with attention layer. LSTM is one of the types of RNN which is capable of solving complex problems. This section provides a brief description about the architecture and working of mentioned LSTM models and the training phase.

Word2Vec Model (Word embedding model ) :

In Word2Vec model, each given essay consists a number of words, and each word represent by a word embedding according to word2vec. The embedding representations are expected to catch the semantic information carried by each word, i.e. the words with similar meanings will near to each other in the vector space.

For example, from our corpus, we can find that "computer" similar to "laptop".

The Neural Network can learn to identify the vector of the input word by using the surrounding word of this input. That means if two different words have the same context, the network tends to give them similar word vector. Hence, every word in our dataset has a unique vector containing the latent semantic and the vectors of the words in one essay can combine to an essay matrix which is the input unit of our scoring machine

**4.1 4 -Layer LSTM Model**

The paper, first introduces 4-layer LSTM model. It is a sequential model and works better for chosen dataset which is a corpus of essays in text format. The first layer of model is word embedding layer. The layer takes 300 as first argument which is number of features (output generated from word embedding layer i.e. Word2Vec model ), dropout and recurrent dropout as 0.4 respectively and input size from 1 to 300 that is length of each sentence sequence. The next layer takes 64 features as first argument with recurrent dropout of 0.4. Third layer is also dropout layer with dropout value of 0.5. The final layer is dense layer, it reduces the dimensionality to 1 which is predicted score. Model have uses ReLU activation function in the dense layer so that the score can be predicted correctly since the values of ReLU function ranges form – ∞ to + ∞. For fitting of training data, model have been passed through batch size of 64 and 100 epochs. These epochs are varied according to the size of the test data. The model produces effective results.

**4.2 Bi-LSTM**

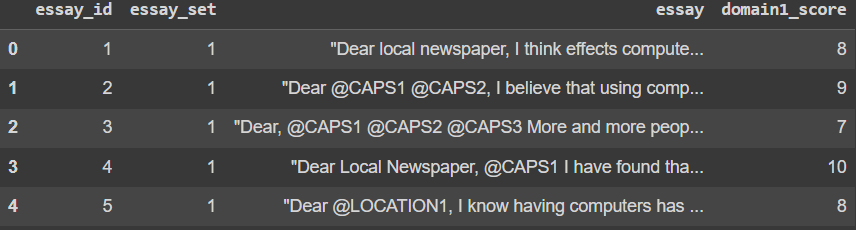
To improve the performance / results got in Unidirectional LSTM, next we implemented the Bi-LSTM model. Unlike standard LSTM, here, the input is allowed to flows in both directions and it is capable of utilizing information from both sides. Again, it is powerful while modelling the sequential data, improving dependencies between words and sentences is both directions. It adds one more LSTM layer to previous model, the direction of information flow is revered. Then model combines the output of both LSTM layers to get the final output. Bi-LSTM is considered to produce more meaningful output when both LSTMs are combined. All other parameters are kept same. Experiments prove that Bi-LSTM model performs better than standard LSTM model.

**4.3 Bi-LSTM with attention layer**

Though, using Bi-LSTM gave better results, in order to improve model performance, attention layer is introduced to the model. Sometimes, basic LSTM gets confused between the words and can predict wrong word. So, in order for the encoder to search for most relevant information, model have been introduced with additional layer of attention mechanism. By applying the attention mechanism, the model will be able to effectively extract the information between essays through inter-sentence alignment and gain better performance.

**5. Experiment Setup**

Dataset used in this paper is “The Hewlett Foundation: Automated Essay Scoring” on Kaggle. The dataset includes 8 essay sets. Each of the essay sets was generated from individual prompt. Average length of essays is in range of 150 to 550 words per response. All essays were hand graded and were double-scored. The training data is in the format of tab-separated value (TSV) file. There are total 3 scores i.e., rater1 score rater2 score and domain score. The Domain score is addition of rater 1 and 2 scores. There are some unwanted empty columns also present in the dataset these columns were dropped using pandas’ libraries. Hence the final dataset has structure something this shown in below.



After generation of dataset the data needed to be preprocessed before passing on to the model. The essays were first cleaned by removing the stop words, punctuation marks and converting all characters to lower case. After cleaning the data next step was to generate feature vector that is to be passed to the word2Vec Model. For models other than neural network word embeddings were generated.

For selecting the best algorithm, various machine learning algorithms were used and the model with best kappa score is selected. following are the models used:

* Linear Regression
* Support Vector Regression (SVR)
* Long Short-Term Memory (LSTM)
* K-Nearest Neighbor
* Decision Tree Classifier

**6. Results and Discussions**

In this section the results of comparison of different machine learning models is explained and the comparison between different LSTM layer is also proposed. Figure 6.1 shows the results of different machine learning models out of which LSTM gives best performance in terms of QWK as well as MSE and variance. Hence the LSTM model was selected for further experiment purpose and predictions of grades.

|  |  |  |  |
| --- | --- | --- | --- |
| **Models** | **Kappa Score** | **Mean Squared Error** | **Variance** |
| Linear Regression | 0.85 | 20.94 | 0.73 |
| SVR | 0.64 | 38.19 | 0.53 |
| LSTM | **0.94** | 7.47 | **0.91** |
| K-Nearest Neighbours | 0.92 | 14.7 | 0.82 |
| Decision Tree Classifier | 0.86 | 22.56 | 0.71 |

Figure no. 6.1: Comparative Study of different machine learning models

Once the model was finalized, various combination of layers in the LSTM were used for better results. The models were modified with different LSTM, Bi LSTM and Attention layers. The combination of layers used are as follows:

* LSTM + LSTM
* LSTM + Bi LSTM + Bi LSTM
* LSTM + Bi LSTM + Attention

Hyperparameter for all of the models were set as constant with batch size of 64, 50 epochs and activation function as ReLU. 5-fold Cross validation was used in each model. The results of QWK, MSE and variance were noted. The results of this comparative study of layers are given in figure 6.2. It is observed that the model that contains the Attention layer performs best amongst all the three models. Further different activation function was used and the results were noted. The results are shown in figure 6.3. For sigmoid and tanh, the QWK is zero and the MSE is very large as compared to the other models. The noted results were not satisfactory and ReLU was decided to be used as activation function for building the final model.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Activation Function** | **Epochs** | **Batch size** | **QWK** | **MSE** | **Variance** |
| LSTM (2) | relu | 50 | 64 | 0.94 | 7.86 | 0.9 |
| LSTM + Bi LSTM (2) | relu | 50 | 64 | 0.95 | 6.66 | 0.92 |
| LSTM + Bi LSTM + Attention | relu | 50 | 64 | 0.96 | 6.2 | 0.92 |

Figure no. 6.2: Comparative Study of different LSTM Layers

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Activation Function** | **Epochs** | **Batch size** | **QWK** | **MSE** | **Variance** |
| LSTM (2) | Sigmoid | 50 | 64 | 0 | 110.45 | 0 |
|  | tan h | 50 | 64 | 0 | 109.33 | 0 |
|  | relu | 50 | 64 | 0.94 | 7.86 | 0.9 |
|  |  |  |  |  |  |  |
| LSTM + Bi LSTM (2) | Sigmoid | 50 | 64 | 0 | 124.36 | 0 |
|  | tan h | 50 | 64 | 0 | 114.17 | 0 |
|  | relu | 50 | 64 | 0.95 | 6.66 | 0.92 |
|  |  |  |  |  |  |  |
| LSTM + Bi LSTM + Attention | Sigmoid | 50 | 64 | 0 | 114.68 | 0 |
|  | tan h | 50 | 64 | 0.95 | 112.23 | 0 |
|  | relu | 50 | 64 | 0.96 | 6.2 | 0.92 |

Figure 6.3: Using Different Activation function

Predictions were made using all the three models generated with selected random essay from the dataset. The results are shown in figure 6.5. It was observed that the model with LSTM, Bi LSTM and Attention layer performs well as compared to the other two models.

|  |  |  |  |
| --- | --- | --- | --- |
| **Essay Id** | **Model** | **Actual Score** | **Predicated Score** |
| 457 | LSTM (2) | 10 | 9 (9.16) |
|  | LSTM + Bi LSTM (2) | 10 | 9(8.558) |
|  | LSTM + Bi LSTM + Attention | 10 | 10(10.113) |

Figure 6.4: Predictions

QWK for prediction of all the selected model were calculated.

|  |  |
| --- | --- |
| **Model** | **QWK** |
| LSTM (2) | 0.924215844 |
| LSTM + Bi LSTM (2) | 0.956398147 |
| LSTM + Bi LSTM + Attention Mechanism | 0.962121614 |

Figure 6.4: Prediction model QWK

**6. Conclusion**

In this paper,

**7. Future Scope**

There is always a scope to increase the project. Proposed Model works good with the typed essays, further it can be modified for handwritten essays as well as essays in different languages. Short answers specific to questions can also be graded.

**References**

1. Automated Essay Grading using Machine Learning Algorithm V.V.Ramalingam 1, APandian2. , Prateek Chetry3 and      Himanshu Nigam4 Department of Computer Science and Engineering, S.R.M. Institute of Science and  Technology, Chennai,  India - 603203
2. Jan Philip Bernius, Stephan Krusche, and Bernd Bruegge. 2021. A Machine Learning Approach for Suggesting Feedback in Textual Exercises in Large Courses. Proceedings of the Eighth ACM Conference on Learning @ Scale. Association for Computing Machinery, New York, NY, USA, 173–182. DOI:https://doi.org/10.1145/3430895.3460135
3. Urs Von Matt, “Kassandra:The Automatic Grading Ssytem”,Institute for Scientific Computing,ETH Zurich,Switzerland,1994
4. Z. Wang, J. Liu and R. Dong, "Intelligent Auto-grading System," 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), 2018, pp. 430-435, doi: 10.1109/CCIS.2018.8691244.
5. Neslihan Süzen, Alexander N. Gorban, Jeremy Levesley, Evgeny M. Mirkes, Automatic short answer grading and feedback using text mining methods, Procedia Computer Science, Volume 169, 2020, Pages 726-743, ISSN 1877-0509,
6. Aldea, A. I., Haller, S. M., & Luttikhuis, M. G. (2020). Towards grading automation of open questions using machine learning. In J. van der Veen, N. van Hattum-Janssen, H-M. Järvinen, T. de Laet, & I. ten Dam (Eds.), *Engaging, Engineering, Education: Book of Abstracts, SEFI 48th Annual Conference University of Twente (online), 20-24 September, 2020* (pp. 584-593). University of Twente.
7. R. Krithika and Jayasree Narayanan. 2015. Learning to Grade Short Answers using Machine Learning Techniques. In Proceedings of the Third International Symposium on Women in Computing and Informatics (WCI '15). Association for Computing Machinery, New York, NY, USA, 262–271. DOI
8. Dr. A. Mercy Rani 2021. Automated Explanatory Answer Evaluation Using Machine Learning Approach. *Design Engineering*. (Jun. 2021), 1181 - 1190.
9. Sanchez, Victor & Nerbonne, John & Verspoor, Marjolijn. (2013). Identifying Important Factors in Essay Grading Using Machine Learning..