# Automatic Short Answer Grading using Siamese Bidirectional LSTM Based Regression

Arya Prabhudesai
*Department of Computer Engineering*
*College of Engineering Pune*
Pune, India
prabhudesaiap16.comp@coep.ac.in

Ta N. B. Duong
*School of Information Systems*
*Singapore Management University*
Singapore
donta@smu.edu.sg

***Abstract*—Automatic student assessment plays an important role in education - it provides instant feedback to learners, and at the same time reduces tedious grading workload for instructors. In this paper, we investigate new machine learning techniques for automatic short answer grading (ASAG). The ASAG problem mainly involves assessing short, natural language responses to given questions automatically. While current research in the field has focused either on feature engineering or deep learning, we propose a new approach which combines the advantages of both. More specifically, we propose a Siamese Bidirectional LSTM Neural Network based Regressor in conjunction with handcrafted features for ASAG. Extensive experiments using the popular Mohler ASAG dataset which contains training samples from Computer Science courses, have demonstrated that our system, despite being simpler, provides similar or better overall performance in terms of grading accuracy (measured with Pearson *r*, mean absolute error and root mean squared error) compared to state-of-the-art results.**

***Keywords*—neural networks, automatic grading, short answer, Siamese LSTM, feature engineering**

## I. INTRODUCTION

Assessment of student understanding is an important part of the learning process. An automatic short answer grading system (ASAG) (Mohler et al., 2011 [1], Ramachandran et al., 2015 [2], Sultan et al., 2016 [3], etc.) performs this task by assigning a score to the student answer, given a correct answer to the same question which is used as reference. The ASAG system eases the load on the instructor and is also devoid of the subjectivity that factors in when natural responses are graded by humans. The timely and effective feedback provided by an automated grader can be used by both, students and professors to focus on areas of potential improvements. The ASAG task that we consider in this paper, is focused on Computer Science courses, particularly the Data Structures course taught at an introductory level at universities. Some samples of the data we have used are shown in Table I.

Except some particular forms of automated assessment such as multiple-choice, true/false questions, etc., in many cases, student responses are given in the form of natural text which requires the ability of natural language processing (NLP) [1]. The task of grading short answers is different from the grading of essay-like answers (Taghipour et al., 2018 [4], McNamara et al., 2015 [5], etc.). The focus of the latter is on the coherence of the text, grammatical correctness, and style of writing, while for short answer grading the prime area of focus is the correctness of the student's answer given one or more correct answers to the same question. The ASAG task is also considerably different from paraphrase detection (Agarwal et al., 2018 [6], Issa et al. 2018 [7], etc.), as in ASAG we need to provide a score, not just a binary yes/no decision [1].

TABLE I: Short question-answer pairs in an introductory computer science course [1]

| | | |
|---|---|---|
| **Question 1** | How are arrays passed to functions? | |
| **Reference Answer 1** | by reference. | |
| **Student Answers** | | **Grade** |
| **Answer a** | Arrays are passed by reference. | 5 |
| **Answer b** | specify the array name without brackets. | 3 |
| **Answer c** | function( int [], int length) | 4 |

In recent times, neural networks and distributed representations [8] have showcased immense potential in the field of natural text processing. Neural networks have the capability to exploit the context of natural text to create a single dense vector representation also known as its embedding. This embedding is then used for mathematical models and computations. Neural networks have proven to be more robust than traditional machine learning models which make use of handcrafted features (LeCun et al., 2015 [9], Schmidhuber et al., 2015, etc. [10]).

In this paper, we tackle the ASAG task with the help of neural networks and deep learning techniques. We use the combination of a simple Siamese Bidirectional Long Short Term Memory (LSTM) Neural Network Architecture and feature engineering. The Siamese LSTM system receives the student answer and the reference answer as inputs, and processes both in parallel to give us a final grade as output. While deep learning has provided remarkable results in many NLP applications, there are still tasks in which feature engineering can be beneficial, e.g., resulting in simpler models with comparable performance [11]. Therefore, we make use of feature engineering as an enhancement for the

Authorized licensed use limited to: ULAKBIM UASL - EGE UNIVERSITESI. Downloaded on July 24,2024 at 13:41:17 UTC from IEEE Xplore. Restrictions apply.

base architecture. In particular, we consider the rating criteria behind the grade of a student answer - with respect to the reference answer for a given question.

Our approach of combining deep neural networks and feature engineering, despite being simpler, performs similarly or better compared to existing ASAG systems for the popular Mohler et. al. dataset [12]. More specifically, our approach provides a Pearson correlation coefficient which is slightly better compared to the state-of-the-art score of 0.649 [13]. On top of that, we have also improved other important performance measures such as the mean absolute error (MAE) and root mean squared error (RMSE) compared to existing systems.

The rest of this paper is organised as follows. Section II describes the related work in automatic short answer grading. Section III introduces the architecture of our system. Section IV describes system implementation details. Section V analyses experimental results and findings. Section VI concludes the paper and highlights directions for future work.

## II. LITERATURE REVIEW

One of the earliest research in the task of automatic short answer grading was conducted by Mohler et al., 2009 [1]. They compared a number of corpus-based and knowledgebased measures used for the semantic similarity of texts. The effect of the choice of domain and corpus-size on the result was then tested by applying these measures on different sized corpora belonging to different domains. They also introduced a technique to feedback the model with the correct student answers that do not match with the instructor-provided reference answers.

Mohler et al., 2011 [12] used machine learning techniques to show that the combination of several graph alignment features with lexical semantic similarity measures can grade student answers much more accurately, compared to the case when the semantic features were used separately. They also proposed the alignment of dependency graphs to exploit structural components in ASAG tasks. They reported their best Pearson correlation coefficient as 0.518.

Early short answer grading systems relied on manual extraction of patterns from the answers in order to facilitate grading. Ramachandran et al., 2015 [2] identified important text with the help of word-order graphs from the instructor-provided reference answers and the top-scoring student answers to solve the SemEval Semantic Textual Similarity (STS) task and to obtain a correlation value of 0.61.

Sultan et al., 2016 [3] introduced a fast and easy method for the task to get a correlation coefficient of 0.592. They made use of a supervised model for the purpose which augmented the text similarity features with additional grading-specific constructs to provide high accuracy. The additional features used in their model include question demoting, term weighting, and length ratio.

A comprehensive study of the existing systems for the ASAG task can be found in a survey by (Burrows et al., 2015) [14]. They performed two kinds of analysis on the research conducted in ASAG. Their historical analysis identifies 35 ASAG systems with 5 different themes - these are indicative of the advancement in the automated student assessment process. On the other hand, their component analysis reviews the 6 common dimensions used for student answer grading, from preprocessing to effectiveness.

Galhardi et al., 2018 [15] also conducted a systematic review of the machine learning approaches used so far for the ASAG task. They have conducted a thorough analysis of 44 papers that conform to their set of guidelines and which use machine learning to solve the ASAG task. For a long time, research in the domain of ASAG has mainly concentrated on traditional machine learning approaches - deep learning methods getting more attention only very recently.

Hassan et al., 2018 [16] made use of paragraph embedding for the task to get a Pearson coefficient of 0.569. Both the student and reference answer are converted to their respective embedding after which the cosine similarity is calculated between them to finally train a ridge regression classifier. A comparative analysis is carried out between different vector representations and their effect on the performance of the system. Gomaa et al., 2019 [17] have developed a technique called Ans2Vec using the skip-thought approach to convert reference answers and student answers into contextual vectors, and measures the similarity between them, in particular the cosine similarity.

Kumar et al., 2017 [13] used a Siamese LSTM with Earth Mover's Distance Pooling for the ASAG task. Their system uses a combination of Siamese LSTMs, a pooling layer based on the Sinkhorn distance, and a support vector ordinal output layer. They reported an accuracy of 0.649. Our model has some fundamental differences from this work as follows. We have not used any distance pooling in our LSTM architecture, resulting in a computationally simpler model. Secondly, our architecture uses a combination of feature engineering and deep learning methodologies to achieve the desired result. Our system appears to be more robust, since it achieves a better balance between all the three key measures of performance for the ASAG task: the Pearson coefficient, root mean square error (RMSE) and the mean absolute error (MAE).

## III. ARCHITECTURE

In supervised learning, models are trained on examples and their respective labels. In our experiment, we use a Siamese Bidirectional LSTM neural network architecture to train our model. Fig. 1 shows our basic architecture along with feature engineering.

In the Siamese Bidirectional Architecture without feature engineering, we have two parallel streams of input. Each side accepts one of the answers: reference answer, or the student answers. $R_i$ denotes the reference answer $i$ for question $Q_i$ from the dataset. $A_{ij}$ represents the student answer $j$ for question $Q_i$ corresponding to reference answer $R_i$. As shown in the figure, we send the reference answer $R_i$ and the student answer $A_{ij}$ simultaneously into the system. Both $R_i$ and $A_{ij}$ are

Authorized licensed use limited to: ULAKBIM UASL - EGE UNIVERSITESI. Downloaded on July 24,2024 at 13:41:17 UTC from IEEE Xplore. Restrictions apply.

then converted into their respective vector representations also known as sentence embeddings by the Embedding layer. The embeddings generated by the layer are 300 dimensional.

activation function so as to get a numerical score output instead of a binary or multi-class classifier output. This output is our predicted grade for the given student answer.

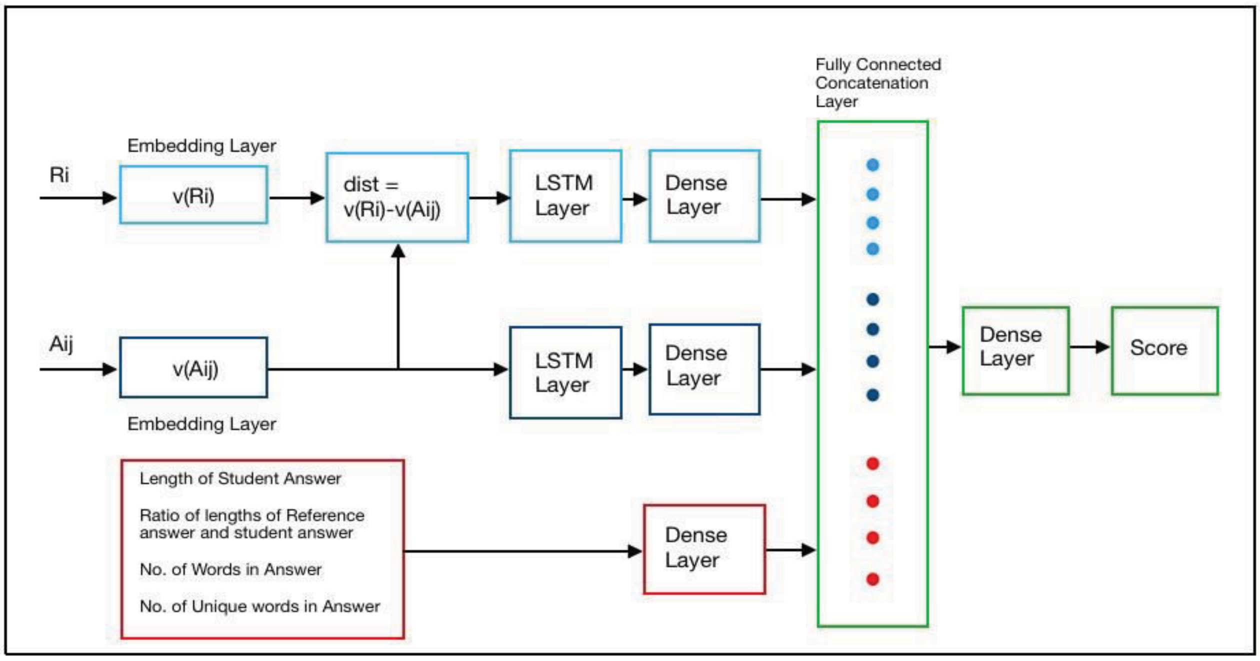

Fig. 1: Siamese Bidirectional LSTM Neural Network Architecture with Feature Engineering

These sentence embeddings v($R_i$) and v($A_{ij}$) are then used for further processing.

The 300-dimensional vector v($A_{ij}$) is sent directly to the LSTM cell whereas in case of the reference answer stream, the mathematical distance between the two vectors:

$$dist = v(R_i) - v(A_{ij}) \tag{1}$$

is calculated and the resultant value stored in the variable dist, is made to pass through the LSTM cell. The reason for this is, we have only a limited number (81) of reference answers in the Mohler dataset while the number of student answers per reference answer is more. Hence, to avoid overfitting the model on the same values of reference answers repeatedly, we calculate the difference between the student answer vector and the reference answer vector and train our model on these values.

The outputs of both the LSTM cells go through a Dense layer with sigmoid activation in order to enhance the feature extraction capabilities of the system. The outputs of both these dense layers are then processed through a fully connected concatenation layer to get a final combined output consisting of features of both, the student answer as well as the reference answer which signifies the rating criteria for evaluation. This combined output is then made to go through a Dense layer with sigmoid activation. The final output layer is also a Dense layer, but has a single unit with a linear

In the next step, we consider feature engineering to complement the basic Siamese Bidirectional LSTM architecture. Fig. 1 highlights the feature engineering part in red boxes. The features we have employed in our system include:

- Length of the student answer.
- Ratio of the length of the reference answer and length of student answer.
- Number of words in the student answer.
- Number of unique words in the student answer.

All these mentioned features, after initial preprocessing, are made to transform through a Dense layer with sigmoid activation. The output of this Dense layer along with the outputs of the Dense layers from the other two streams, i.e, the reference answer and the student answer streams, are then concatenated via a fully connected layer as shown in Fig. 1. The combined output is passed through a Dense layer with sigmoid activation. The final score is obtained via a Dense layer with a single unit which has a linear activation function.

## IV. IMPLEMENTATION

### A. Dataset

The dataset we have used in our experiment is the short answer grading V2.0 dataset[1] originally introduced by Mohler et al., 2011 [12]. The dataset consists of assignments/exams

[1] http://lit.csci.unt.edu/index.php/Downloads

Authorized licensed use limited to: ULAKBIM UASL - EGE UNIVERSITESI. Downloaded on July 24,2024 at 13:41:17 UTC from IEEE Xplore. Restrictions apply.

provided to a computer science course in basic data structures at the University of North Texas.

TABLE II: Samples from the dataset used in this paper

| **Question 1** | Where do C++ programs begin to execute? | |
|---|---|---|
| **Reference Answer 1** | At the main function. | |
| **Student Answers** | | **Grade** |
| **Answer a** | the Function main(). | 5 |
| **Answer b** | At the root | 2.5 |
| **Answer c** | in the testing phase | 0 |
| **Question 2** | How many constructors can be made for a class? | |
| **Reference Answer 2** | Unlimited number. | |
| **Student Answers** | | **Grade** |
| **Answer a** | any number you want | 5 |
| **Answer b** | several | 4.5 |
| **Answer c** | one. | 0 |

The data consists of answers to 81 questions spread across ten assignments and 2 examinations. The original dataset consists of 87 questions, 6 of which have not been included in the authors' work since they were not short answer questions. The answers in the dataset are graded on a scale of 0 (completely incorrect) to 5 (perfect answer). The question, its reference answer / correct answer, and the student answers to the question along with their respective grades form the primary components of the dataset. The student answers are graded by two human instructors. Hence we have three different grades as a part of the dataset, one from each of the human graders and a third grade which is the average of the two. All three of these values lie on a scale of 0 to 5. For our experiment, we make use of the average grade as the gold standard for training the model as has been done by Mohler et al., 2011 [12].

Each question in the dataset consists of between 24 to 31 student responses with an average number of responses per question being 28 for the overall dataset. The total number of responses in the dataset is 2273. Table II shows two samples from the dataset, each sample consisting of the question, its reference answer along with three sample student answers and their respective grades (the average value of the grades assigned by the two human graders).

## B. Data Augmentation

The total number of samples available for training in the Mohler dataset are 2273, which is not sufficient to train a complex neural network. Hence we use data augmentation techniques to increase the training dataset size.

On observing the dataset, it can be deduced that there are many student answers which have been given an average grade of 5.0 by the human instructors. This means that they can be considered equivalent to a reference answer. The assumption we have made for our system is, if $n$ out of $m$ students received the perfect grade ($n < m$) for a given question, then we get $(n-1) * m$ new training pairs for that question. Using this technique, we now have a dataset of approximately 37000 samples which we can use to train the model.

## C. Feature Engineering

In order to implement feature engineering, we build the required set of features from our augmented dataset. The features we have used include: the length of the student answer, ratio of the length of the reference answer and length of the student answer, number of words in the student answer and number of unique words in the student answer.

We use the ratio of lengths of the reference and student answer as a feature, as opposed to solely using the length of reference answer as a feature. This is to avoid overfitting the model by repeated training on the same values since the number of student answers per reference answer is high.

## D. Preprocessing the Dataset

Since our modified dataset includes feature engineered samples, there are two types of data to be preprocessed: text data i.e the student and reference answers and numerical features that we generated in Section IV-C.

The preprocessing tasks carried out for the text data include the standard steps used in text processing: punctuation removal, spellcheck, stopwords removal, and lemmatization. The Embedding layer in the architecture shown in Fig. 1 requires input in the form of a padded sequence of numbers denoting the sentence flow, each numerical element of which is then mapped to its respective word embedding. We have used a maximum padded sequence length of four, considering the text we are dealing with is short answers.

The preprocessing of the numerical data includes scaling each of the four features we generated in Section IV-C using normalization.

## E. Training the Model

We split the dataset into a training set and a test set, with a test set ratio of 0.20. The model is then trained with a mean absolute error loss function.

We have made use of pre-trained GloVe word vectors[2] for our Embedding layer [18]. Hence the Embedding layer maps each element of the padded sequence to a 300-dimensional word embedding.

## F. Evaluating the Model

The results of the model are evaluated using the Pearson correlation coefficient, the root mean squared error (RMSE) and the mean absolute error (MAE) between the predicted scores and the actual grades as metrics. The Pearson correlation coefficient is a measure of the strength of the association between the two variables, the equation for which is given by:

[2] https://nlp.stanford.edu/projects/glove/

Authorized licensed use limited to: ULAKBIM UASL - EGE UNIVERSITESI. Downloaded on July 24,2024 at 13:41:17 UTC from IEEE Xplore. Restrictions apply.

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 (y_i - \overline{y})^2}} \tag{2}$$

where $x_i$ and $y_i$ denote the $i^{th}$ values for distribution of variables X and Y, and $\overline{x}$ and $\overline{y}$ indicate the mean values of X and Y distribution. Here we can consider X to be the distribution of the actual scores and Y to be the distribution of the scores predicted by the model.

The RMSE is a commonly used metric to estimate the mathematical difference between the values predicted by a system and the observed values. The equation for calculating the RMSE is given by:

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(e_i)^2} \tag{3}$$

where n denotes the number of samples, and $e_i$ refers to the squared distance between the actual and predicted scores.

The mean absolute error (MAE) measures the difference between two continuous variables. The MAE is calculated as:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_i| \tag{4}$$

where $e_i$ is the mathematical distance between the actual and the predicted scores and n refers to the total number of samples in the distribution.

## V. Results and Analysis

In this section, we report and compare the results of our proposed system with state-of-the-art approaches, such as [13] and [17]. For a more comprehensive analysis, we have also conducted evaluations on some more basic architectures such as the simple LSTM architecture and the LSTM architecture with feature engineering.

TABLE III: Performance on the Mohler dataset for various approaches evaluated in this paper

| Architecture Used | Pearson Correlation Coefficient | RMSE | MAE |
|---|---|---|---|
| Siamese Bidirectional LSTM Architecture | 0.651 | 0.930 | 0.668 |
| Siamese Bidirectional LSTM Architecture with Feature Engineering | 0.655 | 0.889 | 0.618 |
| Simple LSTM Architecture | 0.381 | 0.982 | 0.959 |
| LSTM Architecture with Feature Engineering | 0.523 | 0.901 | 0.904 |

### A. Comparing with the original LSTM architecture

For the simple LSTM architecture, we used two inputs, the student answer, and the reference answer. We obtain the sentence embeddings for each of these two text inputs using the average of the GloVe pre-trained word embeddings. Then, the cosine similarity between the two vectors is calculated in order to get a single numerical value lying between 0 and 1. We pass these values through an LSTM cell followed by a Dense layer in order to get the final predicted score for a given student answer.

For the LSTM with feature engineering architecture experiment, the steps followed are the same as above but along with the cosine similarity values, we also use the length of the student answer as an additional feature and train the model on a combination of the two features.

As can be seen from Table III, we get a peak Pearson correlation *r* value of 0.651 for the Siamese LSTM Architecture when trained without employing any feature engineering. With the inclusion of feature engineering techniques, there is an improvement in the performance of the system and we get a Pearson *r* of 0.655. We also observe that adding feature engineering improves all other performance measures, i.e. RMSE and MAE. Hence in this case, a combination of feature engineering and deep neural networks boosts the performance of the system as compared to that of a solely neural network based system.

In the case of the Simple LSTM Architecture, we do not get a very impressive result since the correlation coefficient observed is only 0.3811. But again, the inclusion of feature engineering boosts the performance of the system drastically and we get a coefficient of 0.52358 for the LSTM architecture with feature engineering.

### B. Comparing with latest approaches

TABLE IV: Performance reported by existing systems and compared to our approach

| System | Pearson Correlation Coefficient | RMSE | MAE |
|---|---|---|---|
| [Sultan et al., 2016] [3] | 0.63 | 0.85 | - |
| [Gomaa et al., 2019] [17] | 0.569 | 0.797 | - |
| [Kumar et al., 2017] [13], LSTM-EMD-SVOR | 0.550 | 0.830 | 0.490 |
| [Kumar et al., 2017] [13], LSTM-EMD-Logits | 0.649 | 1.135 | 0.657 |
| Siamese Bidirectional LSTM with Feature Engineering | 0.655 | 0.889 | 0.618 |

In order to compare the performance of our system with the state-of-the-art approaches, we have included some benchmark results on the Mohler dataset in Table IV. In particular, we consider the best-performing methods published in IJCAI by Kumar et al., 2017 [13], and most recently by Gomaa et al., 2019 [17], among others. We note that some of the existing methods did not report all performance measures.

As can be observed from the table, our system achieves a good balance between all the three key performance measures

Authorized licensed use limited to: ULAKBIM UASL - EGE UNIVERSITESI. Downloaded on July 24,2024 at 13:41:17 UTC from IEEE Xplore. Restrictions apply.

i.e the Pearson correlation coefficient, the RMSE and MAE. More specifically, our approach combining deep neural networks and feature engineering has a similar or better Pearson correlation coefficient than the other existing systems, and outperforms the best method by Kumar et al., 2017 [13], namely LSTMEMD-Logits, with respect to all the three measures. We also note that another method reported in [13], LSTM-EMDSVOR, performed well in terms of MAE but produced sub-par results in terms of Pearson *r*.

### *C. Summary*

Based on the experimental results, we can observe that our approach of incorporating feature engineering into deep neural networks provides significant performance gain in terms of short answer grading accuracy when compared to most recent, existing methods. On top of that, our approach appears to be offering a good balance between various important accuracy measures. This is important, as we believe that it is not sufficient to use a single measure of grading accuracy. The correlation measure could be used for ordering students' performance, while minimizing absolute deviation results in fairer auto-assessment.

## VI. CONCLUSION AND FUTURE WORK

In this paper, we consider the challenging problem of automatic short answer grading in an introductory computer science course. The primary purpose of the task being, to ease the assessment load on the instructors and engender more focus on feedback and improvement. To this end, we developed a combined approach which incorporates Siamese Bidirectional LSTM Neural Network Architecture and feature engineering techniques to solve the automatic grading task.

The experiments conducted with a real-world, popular dataset have demonstrated the effectiveness of our approach. In summary, our basic system without feature engineering gives a reasonably good Pearson correlation coefficient, while our feature engineered system could further boost the performance. At the same time, our combined approach improves on other important performance measures in auto-grading tasks such as root mean squared error and mean absolute error compared to state-of-the-art systems. It is obvious that a more accurate auto-grading system is of great practical benefit to both learners and instructors.

Our approach has been validated using the most popular dataset obtained from a computer science course. We are currently working to extend the combined approach by conducting more experiments on available ASAG datasets in other application domains, such as the SemEval data (Dzikovska et al., 2013 [19]).

## REFERENCES

[1] M. Mohler and R. Mihalcea, "Text-to-text semantic similarity for automatic short answer grading," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 567–575, Association for Computational Linguistics, 2009.

[2] L. Ramachandran, J. Cheng, and P. Foltz, "Identifying patterns for short answer scoring using graph-based lexico-semantic text matching," in *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 97–106, 2015.

[3] M. A. Sultan, C. Salazar, and T. Sumner, "Fast and easy short answer grading with high accuracy," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1070–1075, 2016.

[4] K. Taghipour and H. T. Ng, "A neural approach to automated essay scoring," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1882–1891, 2016.

[5] D. S. McNamara, S. A. Crossley, R. D. Roscoe, L. K. Allen, and J. Dai, "A hierarchical classification approach to automated essay scoring," *Assessing Writing*, vol. 23, pp. 35–59, 2015.

[6] B. Agarwal, H. Ramampiaro, H. Langseth, and M. Ruocco, "A deep network model for paraphrase detection in short text messages," *Information Processing & Management*, vol. 54, no. 6, pp. 922–937, 2018.

[7] F. Issa, M. Damonte, S. B. Cohen, X. Yan, and Y. Chang, "Abstract meaning representation for paraphrase detection," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 442–452, 2018.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[10] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[11] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in contextdependent deep neural networks for conversational speech transcription," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pp. 24–29, IEEE, 2011.

[12] M. Mohler, R. Bunescu, and R. Mihalcea, "Learning to grade short answer questions using semantic similarity measures and dependency graph alignments," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 752–762, Association for Computational Linguistics, 2011.

[13] S. Kumar, S. Chakrabarti, and S. Roy, "Earth mover's distance pooling over siamese lstms for automatic short answer grading.," in *IJCAI*, pp. 2046–2052, 2017.

[14] S. Burrows, I. Gurevych, and B. Stein, "The eras and trends of automatic short answer grading," *International Journal of Artificial Intelligence in Education*, vol. 25, no. 1, pp. 60–117, 2015.

[15] L. B. Galhardi and J. D. Brancher, "Machine learning approach for automatic short answer grading: A systematic review," in *Ibero-American Conference on Artificial Intelligence*, pp. 380–391, Springer, 2018.

[16] S. Hassan, A. A. Fahmy, and M. El-Ramly, "Automatic short answer scoring based on paragraph embeddings," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 10, pp. 397–402, 2018.

[17] W. H. Gomaa and A. A. Fahmy, "Ans2vec: A scoring system for short answers," in *International Conference on Advanced Machine Learning Technologies and Applications*, pp. 586–595, Springer, 2019.

[18] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[19] M. O. Dzikovska, R. D. Nielsen, C. Brew, C. Leacock, D. Giampiccolo, L. Bentivogli, P. Clark, I. Dagan, and H. T. Dang, "Semeval-2013 task 7: The joint student response analysis and recognizing textual entailment challenge," tech. rep., 2013.

Authorized licensed use limited to: ULAKBIM UASL - EGE UNIVERSITESI. Downloaded on July 24,2024 at 13:41:17 UTC from IEEE Xplore. Restrictions apply.