# Deep Learning Models for Short Answer Scoring

**[1]Saurav Kumar, [2]Dr. Ahtesham Farooqui, [3]Sachin Sahu**

Sam College of Engineering and Technology
Bhopal, Madhya Pradesh.

***Abstract*: Automated scoring of descriptive answers is a critical component in educational assessment, leveraging advancements in Natural Language Processing (NLP). Recent years have witnessed substantial growth in NLP, primarily attributed to the transformative impact of deep learning. This research explores the application of deep learning techniques, emphasizing their efficacy in automated scoring, particularly within the realm of short answer scoring tasks. In this study, we systematically compare various common deep learning models for the Short Answer Scoring (SAS) task. The outcomes shed light on the strengths and weaknesses of these models, providing valuable insights for the advancement of automated scoring systems in educational settings.**

***Keywords*: NLP, Deep Learning, Short Answer Scoring (SAS) Task, Educational Assessment, Automated Scoring.**

## 1. INTRODUCTION

The prowess of deep learning in Natural Language Processing (NLP) tasks such as Machine translation, question answering, text summarization etc. [1]. Traditional machine learning requires heavy feature engineering which mostly involves extraction of handcrafted features using techniques like regular expression matching, lemmatization/stemming, tokenization etc.

Which can be time-consuming and cumbersome, to achieve significant results. They are generally trained on high dimensional sparse features and can be computationally expensive. With successes in feature embedding's, which are low dimensional dense representations of textual data, deep learning models are robust and easier to train. They outperform most approaches in NLP tasks with minimal feature engineering.
The computational requirements that most deep learning architectures bring with them are well accommodated with the use of GPUs. Further, transfer learning has greatly helped overcome several common challenges in deep learning including lack of datasets of sufficient size and lack of computational resources to train very deep networks. Recently, some works [29] [30] [31] have shown the success of transfer learning in NLP.

In short answer scoring, we assign scores for brief answers to the corresponding question prompts. We train our model on multiple individual responses for each question prompts using the scores assigned by annotators as a target. The responses typically consist of one or few sentences. The question can be from multiple or single domains based on the assessment. The prompts might also include open-ended questions. In short answer scoring, we mainly consider the content of the responses over grammar, spelling, and vocabulary. For our task, spelling and grammar are not considered for scoring but might affect the clarity of the responses.

Deep learning approaches suit SAS tasks better because of their ability to learn a complex hierarchical representation of data automatically. In this work, we investigate some common deep learning approaches to short answer scoring. The dataset and evaluation metric used are discussed in the following sections followed by the methods and results.

## 2. DATASET

**Dataset and Evaluation Metrics:** To conduct our investigation into short answer scoring, we employed the Automated Student Assessment Prize - Short Answer Scoring dataset provided by the Hewlett Foundation [2]. This dataset comprises 10 question prompts, offering a diverse range of assessment scenarios. In our training set, we meticulously curated 17,000 examples, ensuring a balanced distribution with approximately 1,700 examples per question prompt. The test set, designed for robust evaluation, encompasses 5,100 examples. This set maintains variability with 300 to 600 examples per question prompt, providing a thorough assessment of the generalization capabilities of our model.

**Response Characteristics:** The responses within the dataset are characterized by brevity, typically consisting of one or a few sentences. This aligns with the nature of short answer scoring tasks, where succinct and focused answers are essential.
The questions in the dataset span multiple domains, reflecting the diverse nature of assessments. Additionally, open-ended questions are incorporated into the prompts, further challenging the adaptability and discernment of our short answer scoring model.

**Model Training Approach:** To train our short answer scoring model, we adopted a supervised learning approach. Each response in the training set is paired with scores assigned by annotators, serving as our target variable. The utilization of multiple individual responses for each question prompt ensures a comprehensive learning process, capturing the nuances and variations in student answers.

**Evaluation Metrics:** We employed standard evaluation metrics to assess the performance of our short answer scoring models. Precision, recall, and F1 score are used to measure the model's ability to correctly identify and score relevant content in the responses. Additionally, we consider overall accuracy and, where applicable, domain-specific metrics to provide a nuanced understanding of our model's effectiveness across various assessment domains.

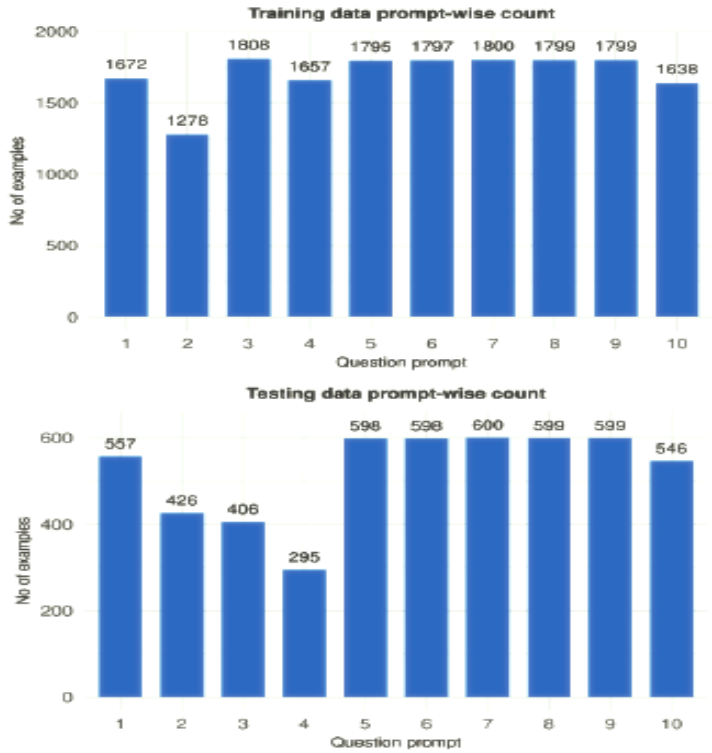


**Figure 1: Frequency of Responses Across Question Prompts**

The distribution of responses across the ten question prompts is illustrated in Figure 1. Each question prompt exhibits a distinctive frequency distribution, reflecting the diverse nature of the dataset.
The responses, characterized by an average length of approximately 50 words, exemplify the conciseness inherent in short answer scoring tasks. Notably, high variability is observed among question sets, showcasing the dataset's comprehensive coverage of a broad range of disciplines.

## 3. EVALUATION

We use the quadratic weighted kappa error as the Fig. evaluation metric as per the competition guidelines. It measures the agreement between 2 raters. We average the quadratic weighted kappa across the question sets using fisher transformation as instructed by the Kaggle competition guidelines.

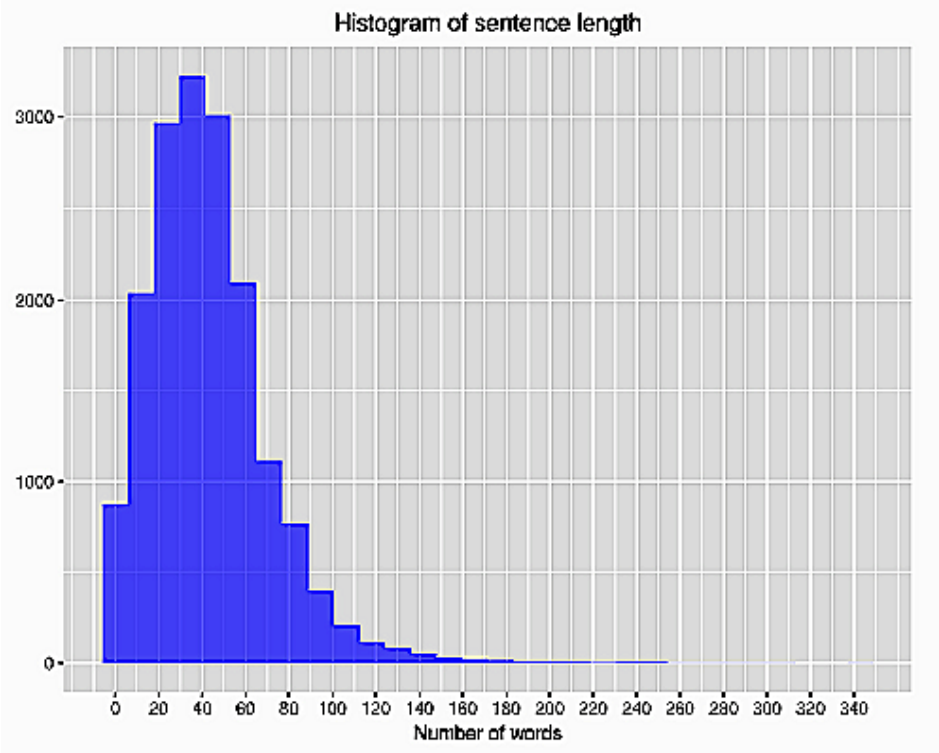


**Fig. 2 Histogram of sentence length**

## 4. LITERATURE REVIEW

Some attempts to develop scoring engines for short answers include c-rater [3] by ETS technologies. Microsoft's Power grader [5] uses a learned similarity metric to cluster answers for a question. The grader can score the responses in each cluster collectively greatly reducing the effort required for grading.

It normalizes responses into a canonical form based on variations among sentences for concept matching. The model is constructed by hand by content experts. [7] also use an unsupervised approach and explore knowledge-based (WorldNet) and corpus-based (LSA and ESA) measures. They also propose a feedback technique to improve the performance of the system. [4] evaluates n-gram and word level matching and Doc2Vec based similarity methods. [6] use Maximal Marginal Reference to obtain reference answers and find similarity between the reference and student answer using GAN-LCS. [8] extract features from the question, answer and student models and evaluate on 6 different algorithms. They show better performance using Deep Belief Networks [9] and significant improvement by using composite features from the question and student model.

The winners of the asap-sas competition [10] use heavy feature engineering and use ensemble methods on different types of models. (Tandalla, 2012) used the Boruta algorithm to determine the relevant words, bigrams and trigrams that helped to predict the score. This was a key step in the model's performance. Further, he used regular expressions to look for acceptable responses. He trained multiple models using random forest and gradient boosting machine and averaged their predictions. (Zbontar, 2012) used stacking with ridge regression, SVM, K-Nearest neighbor, Random Forests and Gradient Boosting Machines as base learners to train. Features were created using character level four-grams and 6-grams and latent semantic indexing. (Conart, 2012) used 6 ensembles of about 81 individual models, trained per answer set.

The models included GLM, SVM, RF used with different tokenization, scaling and reduction techniques. They were stacked at multi-levels with GLM and Generalized additive models at second-level and ordinary least squares at the third-level. 3 other ensembles based on Nelder-Mead optimization to obtain coefficients. The best of the 6 ensembles were selected as a final model (Jesenky, 2012) extracted features from several bag-of-word and string matching and different models were run of subsets of the features. They use genetic algorithm on the weak learners to find the best performing subset.
The final predictions were selected through voting. (Peters & Jankewicz, 2012) extracted features using unsupervised algorithms along with some text statistics, compression-based text similarity. They stacked models like SVM, GBM,

cubist and Sofia using a GBM model. [11] proposed an improvement to Tandalla's approach by automating the generation of patterns. They extracted the content tokens and structure information using word-order graphs.

They group the semantically similar related words to facilitate alternative responses. Their performance of their approach is on par with that of Tandalla's.

## 5. ARCHITECTURE

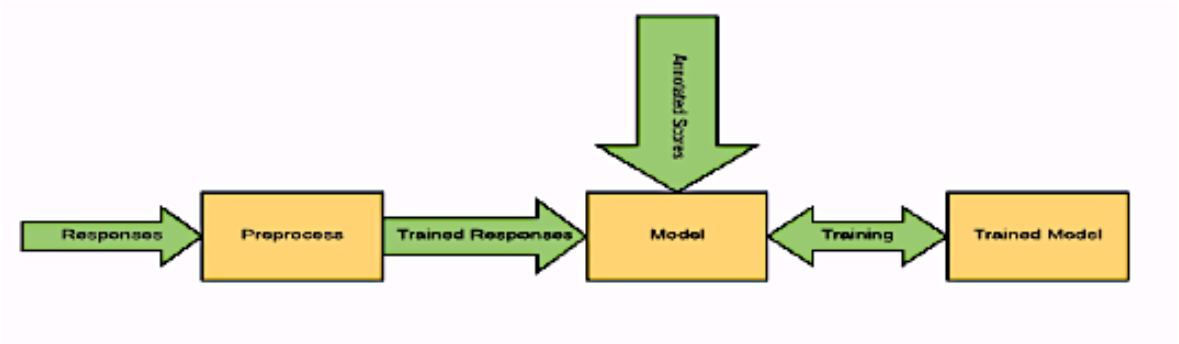


**Figure 1: Training architecture**

We adopt a typical supervised learning architecture. Models were trained on preprocessed response text with the annotated scores as the target. Some models also use question set as input.


**Figure 2: Testing architecture**

Models trained were saved and then used for prediction on the test data. Inputs were preprocessed similar to that in the training process. The model then outputs the predicted score.

## 6. METHODOLOGY

Minimal preprocessing was applied to the dataset including converting all characters to lower cases. Other method-specific preprocessing methods are mentioned in their corresponding sections. All methods use a batch size of 32 for training. Learned word embedding's were used over pre-trained word embedding's as they showed slightly better performance. The outputs of regression models are rounded to an integer value.

**Character level CNN** We use the character level CNN from (Zhang, X. and LeCun, Y. 2019). Char CNN [18] learns to predict a target without the use of any knowledge embedding's or syntactic and semantic structure of the language. We use a maximum input length of 1800 characters with zeroes padded at the end. We use the model design described in the paper with the eighth layer replaced by a fully- connected layer that predicts the score. Particularly, we use the small version of the Conv Net. We merge the question set in the final layer. We use Adam [19] optimizer with an initial learning rate of 0.001. We use a multi-step linear learning rate decay. The learning rate is reduced by 0.1 times once every 3 epochs. The model converged in about 15 epochs.

**Word level CNN** for this method punctuations and stop words were removed and the text was padded and truncated at the end before passing as input. We use a maximum input sequence length of 90. We use a 1D Convolution layer with 64 filters of window size 5 and a max pooling layer with pool size 4 followed by a dropout [20] layer with dropout probability 0.5 and batch normalization [21]. The question set value is merged with the feed-forward layer which is l2 regularized and outputs a single value. We use RMS Prop [22] optimizer with an initial learning rate of 0.01. We use an exponential learning rate decay at a rate 0.1 applied once every 3 epochs. The model converged in about 15 epochs.

Word level bi-LSTM The text is pre-processed similar to that in the CNN model. We use a maximum input sequence length of 90. We use a 250-dimensional bi-LSTM [23] [24] model with learned50-dimensional embedding's. The bi-LSTM is followed by a global average pooling and dropout layer with dropout probability 0.5 and batch normalization. The question set value is merged with the feed forward layer which is l2 regularized and outputs a single value. We use the RMS Prop optimizer with learning rate and learning rate decay similar to the CNN model. We clip the gradients at 10 to prevent exploding gradients. The model converged in about 15 epochs. We use the cuDNN [25] implementation of the LSTM to speed up the training on GPU.

BERT BERT [26] is a bidirectional language representation model pre-trained with deep Bidirectional Transformers. They can be easily fine-tuned for many downstream tasks. We use the BERT base model. We add a fully-connected layer for classification with softmax activation. We use a maximum input sequence length of 90. We use Adam optimizer with a learning rate of 5e-6 and an epsilon value of 1e-9. We clip the gradients by their global norm using a threshold of 1. The model converged in about10 epochs. Training was faster compared to other methods and it achieved better performance in lesser epochs.

## 7. RESULTS AND ANALYSIS

The results of evaluation on the test data are shown in table 1. All experiments were performed on an NVIDIA 940mx GPU. The models took about 15 to 20 minutes to train on the same**.** BERT performs better than the other models which can be attributed to its ability to learn deep bidirectional representations and the rich knowledge representations in its pre-trained language model.

The LSTM model performs better over CNN models as RNN models encode long-range context dependency [27] which help them handle information across multiple sentences better. (Yin et al., 2019) show that RNNs perform better than CNNs when the input sequence is long. We observed that using pre-trained sentence embedding's like USE [28] did not produce satisfactory results. We observe that all models perform poorly on question prompt 3. This is because of the fact that it is an open-ended question and assesses the student's conceptual understanding and interpretation skills. It's also common to find less agreement between scorers for such type of questions which affects the data annotations itself. We further performed simple experiments to test the reliability of the models by modifying a few examples from the data that has a full score and was scored correctly by all the models.

**We performed the following changes in the responses:**

1. Introducing spelling mistakes
2. Paraphrasing the documents
3. Replacing few words with a synonym

We found that the RNN model performed better than the other models in the case where minor spelling mistakes were introduced and, in the case, where words were replaced with their synonyms.
BERT performed significantly better when the responses where paraphrased. This is because of BERT's deep contextual representations conditioned in both directions which allow the context to be preserved despite the change in structure. The performance of BERT was still poor compared to its performance on the test data.

| Prompt | CharCNN | CNN | LSTM | BERT |
|---|---|---|---|---|
| 1 | 0.68 | 0.68 | 0.70 | 0.79 |
| 2 | 0.58 | 0.67 | 0.66 | 0.70 |
| 3 | 0.29 | 0.27 | 0.28 | 0.37 |
| 4 | 0.56 | 0.55 | 0.59 | 0.69 |
| 5 | 0.72 | 0.75 | 0.78 | 0.75 |
| 6 | 0.80 | 0.74 | 0.74 | 0.84 |
| 7 | 0.55 | 0.58 | 0.60 | 0.66 |
| 8 | 0.43 | 0.50 | 0.54 | 0.60 |
| 9 | 0.67 | 0.64 | 0.70 | 0.80 |
| 10 | 0.61 | 0.67 | 0.71 | 0.74 |
| QWK | 0.70 | 0.73 | 0.75 | 0.79 |
| Mean QWK | 0.60 | 0.62 | 0.65 | **0.71** |

**TABLE I METRICS ON TEST DATA**

## 8. CONCLUSIONS

We use different types of deep learning models to compare their performance among themselves and demonstrate the simplicity and efficiency of the architectures over non-neural approaches which require extracting several features from the text data manually as can be observed in the methods used by the winners of the Kaggle competition. They also train faster and are resilient to minor variations within the data. Several challenges still exist in adopting automated scoring systems including lack of robust and generic approaches and digitized data for training, lack of simple tools and resources that can help adopt the system without requiring expertise in NLP.

## REFERENCES:


[1] Saha, S., Dhamecha, T. I., Marvaniya, S., Foltz, P., Sindhgatta, R., & Sengupta, B. (2019). Joint multidomain learning for automatic short answer grading. arXiv preprint arXiv:1902.09183.

[2] Gong, T., Yao, X.: An attention-based deep model for automatic short answer score. Int. J. Comput. Sci. Softw. Eng. 8(6), 127–132 (2019)

[3] Grandini, M., Bagli, E., Visani, G.: Metrics for multi-class classification: an overview. arXiv preprint arXiv:2008.05756 (2020)

[4] Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)

[5] Riordan, B., Horbach, A., Cahill, A., Zesch, T., Lee, C.: Investigating neural architectures for short answer scoring. In: Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 159–168 (2017)

[6] Wang, Z., Lan, A.S., Waters, A.E., Grimaldi, P., Baraniuk, R.G.: A meta-learning augmented bidirectional transformer model for automatic short answer grading. In: EDM (2019)

[7] Xu, C., Zhou, W., Ge, T., Wei, F., Zhou, M.: Bert-of-Theseus: compressing BERT by progressive module replacing. arXiv preprint arXiv:2002.02925 (2020)

[8] Xue, L., et al.: mT5: a massively multilingual pre-trained text-to-text transformer. arXiv preprint arXiv:2010.11934 (2020)

[9] Burrows, S., Gurevych, I., & Stein, B. (2015). The Eras and Trends of Automatic Short Answer Grading. International Journal of Artificial Intelligence in Education, 25(1), 60-117.

[10] Mohler, M., & Mihalcea, R. (2009). Text-to-text semantic similarity for automatic short answer grading. Paper presented at the Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009).

[11] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., et al. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683

[12] Menini, S., Tonelli, S., De Gasperis, G., Vittorini, P.: Automated short answer grading: a simple solution for a difficult task. In: CLiC-it (2019)

[13] Surya, K., Gayakwad, E., & Nallakaruppan, M. (2019). Deep learning for short answer scoring. Int. J. Recent. Technol. Eng.(IJRTE), 7(6)

[14] Sasi, Nair, D., & Paul. (2020). Comparative Evaluation of Pretrained Transfer Learning Models on Automatic Short Answer Grading. arXiv pre-print server.

[15] Gomaa, W. H., & Fahmy, A. A. (2020). Ans2vec: A Scoring System for Short Answers (pp. 586-595): Springer International Publishing

[16] Hassan, S., A, A., & El-Ramly, M. (2018). Automatic Short Answer Scoring based on Paragraph Embeddings. International Journal of Advanced Computer Science and Applications, 9(10).

[17] Ghavidel, H. A., Zouaq, A., & Desmarais, M. C. (2020). Using BERT and XLNET for the Automatic Short Answer Grading Task. Paper presented at the CSEDU (1).