Improving Automatic Short Answer Scoring Task Through a Hybrid Deep Learning Framework

Soumia Ikiss[1], Najima Daoudi[2], Manar Abourezq[3], Mostafa Bellafkih[4]
RAISS Laboratory, National Institute of Posts and Telecommunications (INPT), Rabat, Morocco[1, 4]
LyRica Laboratory, School of Information Sciences, Rabat, Morocco[2, 3]

***Abstract*—An automatic short-answer scoring system involves using computational techniques to automatically evaluate and score student answers based on a given question and desired answer. The increasing reliance on automated systems for assessing student responses has highlighted the need for accurate and reliable short-answer scoring mechanisms. This research aims to improve the understanding and evaluation of student answers by developing an advanced automatic scoring system. While previous studies have explored various methodologies, many fail to capture the full complexity of response text. To address this gap, our study combines the strengths of classical neural networks with the capabilities of large language models. Specifically, we fine-tune the Bidirectional Encoder Representations from Transformers (BERT) model and integrate it with a recurrent neural network to enhance the depth of text comprehension. We evaluate our approach on the widely-used Mohler dataset and benchmark its performance against several baseline models using RMSE (Root Mean Square Error) and Pearson correlation metrics. The experimental results demonstrate that our method outperforms most existing systems, providing a more robust solution for automatic short-answer scoring.**

Keywords—Student answer; automatic scoring; BERT language model; LSTM neural network; Natural Language Processing

I. Introduction

Assessment in an educational setting is an essential and fundamental aspect of measuring learners' knowledge and understanding of a subject. In a typical classroom, whether through tests, assignments, or quizzes, teachers provide grades and feedback on students' answers to questions. However, with the increasing number of students enrolling in online platforms and universities, manually evaluating all these answers has become a complicated and expensive procedure. This increase in the number of students highlights the critical requirement for more effective techniques for student assessment. Automated assessment systems can alleviate this burden by offering a scalable and consistent solution to evaluating student performance.

Exams typically consist of a variety of question forms, which are broadly classified as objective and subjective. True/false, multiple-choice, and fill-in-the-blank questions are examples of objective questions that are intended to evaluate specific knowledge and may be evaluated quickly and accurately [1]. Conversely, subjective questions (short answer/essay) need a long or short response and are intended to assess a deeper understanding in addition to the ability to integrate concepts and present them in a more sophisticated way. Because the answers to objective questions are precise and unambiguous, developing an automated assessment system is rather straightforward. On the other hand, creating a system of that kind for subjective questions is more difficult because it needs to analyze text and comprehend the answers' semantic meaning. When a teacher asks his student in an exam setting the following question: “What is the definition of Artificial Intelligence?” For example, " computers that can carry out difficult jobs that humans have historically only been able to complete" might be the answer of one student. In contrast, another student could respond with “Is the technology that makes it possible for computers and other devices to mimic human intelligence and problem-solving skills.” Both answers are accurate, even though they are expressed in different ways using different words. This illustrates the complexity of treating subjective questions automatically, as the system must be capable of recognizing the correctness of varied but equivalent answers. Addressing this complexity requires a system that can comprehend and assess the various ways in which students may present their answers. This calls for the capacity to recognize synonyms, understand context, and evaluate the relevance and accuracy of the content provided in student responses.

An automatic short answer scoring (ASAS) system aims to evaluate and assign scores to short textual responses based on one or more optimal answers. Since the responses of both student and reference are written in natural language, sophisticated Natural Language Processing (NLP) methods and machine learning models have been required to accurately understand and assess what is written. In various NLP tasks, including ASAS, language models (LMs) have demonstrated significant success. These models assess the probability of word sequences and can predict subsequent words based on the preceding words within a sequence [2]. Traditional language models, such as n-gram language models, employ count-based methods to evaluate and understand text. These models typically rely on the frequency of word sequences (n-grams) to predict the likelihood of subsequent words and to determine the overall structure and coherence of a text. In the context of automatic answer scoring, vector-space models that count n-grams have been widely applied due to their simplicity and effectiveness. For instance, [3] conducted a comparative study where they evaluated the efficacy of bag-of-n-gram representation against bags of semantic annotations for the ASAS task. Their findings highlighted the strengths and limitations of count-based models in capturing the nuances and subtleties of human language, emphasizing the need for more sophisticated approaches that can understand the deeper semantic meaning of text. In modern approaches, language models (LMs) are trained using neural networks, which address several limitations inherent in traditional count-based methods. Firstly, they significantly expand the context taken into account, allowing for a more comprehensive understanding of text beyond the fixed-length context of n-grams. Secondly, these models exhibit a generalization capability across different contexts, which enhances their ability to handle diverse linguistic patterns and structures. The initial neural models were based on recurrent neural networks (RNNs), which are well-suited for sequential data. Among these, long short-term memory networks (LSTMs) became particularly popular due to their ability to capture long-range dependencies in text. LSTMs address the vanishing gradient problem in standard RNNs, enabling the model to retain information over longer sequences. The most recent advancements in neural models for language modeling, such as BERT (Bidirectional Encoder Representations from Transformers) introduced by [4], are based on the transformer architecture. This architecture represents a significant shift from previous models like RNNs and LSTMs, leveraging self-attention mechanisms to process and understand text. The transformer architecture enables these models to capture intricate dependencies and contextual relationships within text more effectively, making them highly suitable for a wide range of Natural Language Processing (NLP) tasks.

In this work, we introduce a novel automatic answer-scoring framework that combines the strengths of a pre-trained BERT model through fine-tuning with the capabilities of an LSTM network. BERT is a multi-layer bidirectional Transformer encoder designed for Natural Language Processing (NLP). Developed by Google, the pre-trained BERT model leverages a vast amount of unlabeled data, including 800 million words from books and 2.5 billion words from Wikipedia. By fine-tuning an additional classification layer along with all the pre-trained parameters, BERT can be adapted for specific NLP tasks. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) designed to effectively capture long-range dependencies and temporal patterns in sequential data. It deals with long-term dependencies, by incorporating memory cells and gating mechanisms to control the flow of information. These features enable LSTM networks to remember and utilize information from earlier time steps, making them well-suited for tasks involving sequential data. This hybrid approach leverages the advanced contextual understanding of BERT and the sequential processing power of LSTM to enhance the accuracy and efficiency of scoring short textual responses.

The findings of our research have two significant implications for both science and society. From a scientific perspective, we demonstrate the effectiveness of combining large language models like BERT with recurrent neural networks to improve text representation and enhance the accuracy of automated short-answer scoring. On the other hand, our research could have a significant implicant on society by reducing the workload on educators, allowing them to focus more on interactive and personalized teaching. Furthermore, this system could be widely implemented in online educational platforms, ensuring consistent and fair assessment for a growing number of students worldwide.

The remainder of this paper is structured as follows. Section II provides a brief review of related works. Section III introduces and elaborates on the proposed approach. Section IV outlines the experimental details, including the dataset, metrics, and implementation settings. Section V presents the results and the corresponding discussions. Finally, Section VI summarizes the study and suggests potential directions for future improvement.

II. Literature Review

Research in grading natural language responses with computational methods has a history dating back to the early work of [5], However, it is only in the current decade that these systems are achieving the level of accuracy necessary for practical use in educational settings. For end users to have confidence in these systems, the challenge lies in developing robust and accurate assessment mechanisms that closely mirror human evaluators. Several methods have been introduced in this area.

Early methods for automatic grading relied heavily on pattern matching, which required significant expert intervention to extract relevant patterns and features from student responses. The study in [6] explored the use of concept mapping techniques, mapping the related concepts in student answers to those in desired answers [7], [8]. Further developed the concept of information extraction from student answers through pattern matching. These researchers utilized regular expressions and parse trees to identify and extract relevant patterns within the text [9]. Involved comparing eight knowledge-based text similarity measures alongside two prominent corpus-based measures: Latent Semantic Analysis (LSA) and Explicit Semantic Analysis (ESA). These measures were trained on both domain-specific and generic corpora.

Later on, the use of machine learning techniques for automatic scoring has become popular [10]. Integrating machine learning into their work [9] to improve the performance involves graph alignment and lexical semantic similarity features using SVM and term frequency-inverse document frequency (TF-IDF). In a similar work, an approach was presented by[11], that suggested a short text similarity-based short answer grading method. They extracted multiple features such as text alignment, vector similarity, TF-IDF, and length ratios. In a similar context [12] combined sentence-level and token-level features for their approach. For sentence-level features, they employed InferSent, a pre-trained sentence embedding model that makes use of a Bi-LSTM network, to get sentence embeddings for the question, the reference answer, and the learner's answer. Semantic representations of the text are also extracted using deep learning-based word embeddings. [13] Employed standard NLP embeddings, such as Word2Vec, GloVe, and FastText, to extract the semantic and distributional properties. The study in [14] Propose a recurrent neural network to resolve the task, by staking three layers of Siamese Bi-LSTMs layer, a pooling layer using earth-mover distance (EMD), and an output layer with regression the predict the score [15]. Then enhanced the RNN-based technique by leveraging LSTM along with sense vectors and Manhattan distance in place of the pooling layer.

While the aforementioned deep learning techniques accomplish end-to-end grading and scoring, they are dependent on a substantial volume of labeled corpus for model training, which is not present in the majority of ASAG corpora. Several pre-trained transfer learning models are used to tackle this challenge [16]. Some works use these pre-trained models by extracting embeddings directly from language models such as BERT, GPT, and ELMO[17], other works incorporate fine-tuning paradigms, such as those by [18], [19].

III. Proposed Method

The automatic short answer grading can be approached both as a regression and as a classification problem [20]; where in a regression task the model is trained to predict a continuous grade for a student's answer. The prediction is typically based on the similarity between the reference answer (the correct answer) and the student's answer. The goal is to minimize the difference between the predicted grade and the actual grade assigned by human graders. The problem can be also a classification task; the model classifies the student's answer into discrete categories such as "correct," "partially correct," or "incorrect." More fine-grained classes can also be defined, depending on the specific requirements of the grading system. The present research implements the regression task to allow for the assignment of continuous scores and provide a finer granularity in grading.

The core concept of our model is to harness the advantages of both the transformer architecture and classical neural networks to accurately interpret the given text. Specifically, we use BERT for its capabilities in understanding the context and semantics of the input text. To capture the sequential dependencies and long-range relationships in the text we also employ LSTM (Long Short-Term Memory).

First, the stacked transformer encoders in BERT aid in conserving computational resources, by using a weighted sum of all other words' embeddings to encode the hidden state of a word [16]. Although this method makes good use of the connections between every word in a phrase, it falls short in terms of taking word order and spacing into account [21]. The LSTM network is ideally suited to produce more precise global context data because of its memory cells and gate architecture. By incorporating temporal information, LSTM can compensate for the shortcomings of BERT’s encoding.

Second, compared to conventional static embeddings like Glove, BERT's dynamic word embeddings, which are produced via extensive unsupervised pretraining and refined on downstream tasks, offer substantial advantages over traditional static embeddings like Glove[4]. These dynamic embeddings provide more comprehensive and flexible general-purpose information by adapting to various settings. This feature enhances the training and convergence of upper-layer neural networks, enabling classical neural networks on BERT to attain impressive performance even with fewer datasets.

Empirical studies support the effectiveness of combining recurrent neural networks with fine-tuned BERT models, particularly in specialized tasks with limited training data. For instance, [22] demonstrated improved aspect-category sentiment analysis by integrating RoBERTa with a specialized CNN, leveraging the strengths of both architectures. [23] Explored the potential of combining BERT with neural networks for sentiment analysis purposes to classify students' reviews on Moocs. They specifically add an LSTM on top of BERT and then a CNN as a local feature extractor.

Given these insights, our proposal is a newly designed model that combines the strengths of the fine-tuning BERT model along with the LSTM network to properly understand as well as evaluate student's responses. For that, we incorporate an LSTM layer on the top of the fine-tuned BERT. The LSTM network extracts fine global context from BERT outputs, providing a richer understanding of temporal relationships. “Fig. 1” illustrates the framework of the proposed approach which is described below in detail.

A. *Fine-Tuning Bert Component*

BERT (Bidirectional Encoder Representations from Transformers) [4] is a groundbreaking pre-trained language representation model developed by Google AI Language[4]. The training is conducted on a massive corpus comprising the Book Corpus with 800 million words and English Wikipedia with 2.5 billion words. The masked language model and next-sentence prediction are two unsupervised tasks that were used to train the initial BERT model. These tasks involved layers for language model decoding and classification. However, for fine-tuning our model for the sentence pair classification task, we do not utilize these specific layers.

A text pair containing the student's answer and reference one is what our model receives as input. Every sequence starts with a unique classification token (CLS), as illustrated in "Fig. 2". To distinguish between the input pair, a special token (SEP) is inserted at the end of each input to help the model understand the end of an answer and the beginning of another. Wordpiece embeddings are used as the token input by BERT. For every token, BERT employs positional embeddings and segment embeddings in addition to token embeddings. Token positions in sequence are indicated by the information included in positional embeddings. When the model input contains sentence pairs, segment embeddings come in helpful. Tokens belonging to the first sentence will have a segment embedding of 0, whereas tokens belonging to the second sentence will have a segment embedding of 1. BERT incorporates embedding of the input by summing up the three embeddings, namely token, position, and segment embeddings, creating a rich representation for each token in the sequence. These representations are then fed to a multilayer bidirectional transformer encoder “Fig. 2” which is the core component of Bert’s architecture, leveraging a multi-head attention mechanism (Formula 1) to focus on data from many representation subspaces at different points in the input sequence simultaneously. After the multihead attention, each transformer layer additionally has a fully connected feed-forward network. Twelve transformer layers are stacked in the model's basic version (BERT_base). The output contextualized embeddings of Bert are then fed to an Lstm Layer to capture more information from input answers. BERT incorporates an attention mechanism to focus on different parts of the input sequence. This mechanism uses Formula (1) to compute the attention weights:

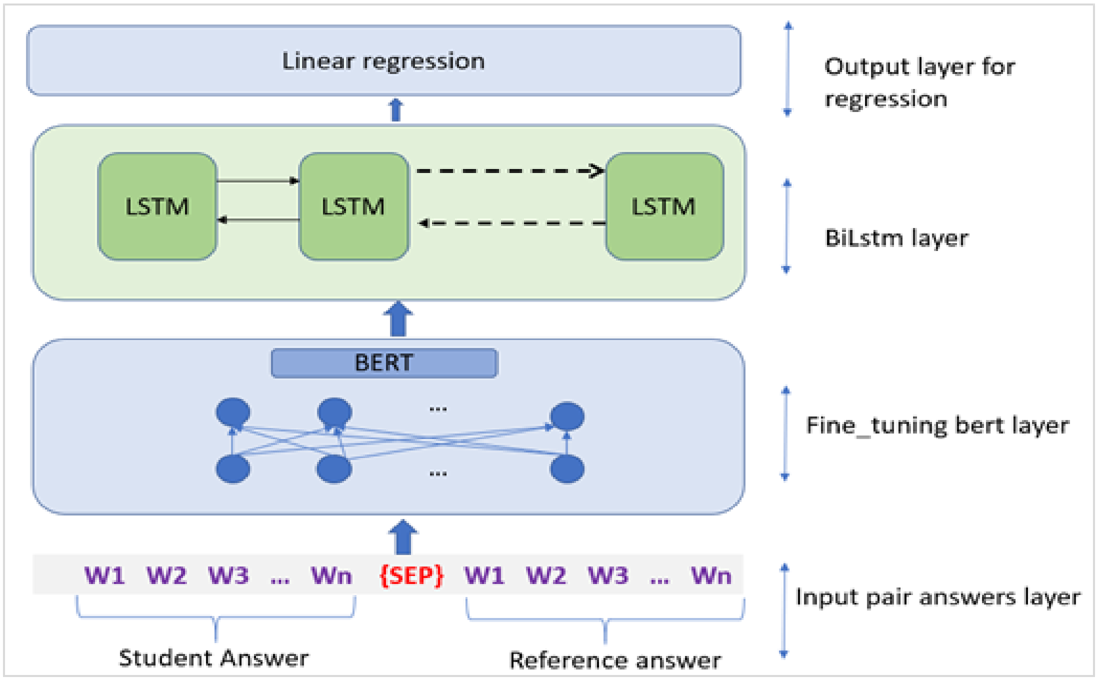

Fig. 1. The general architecture of the proposed model.

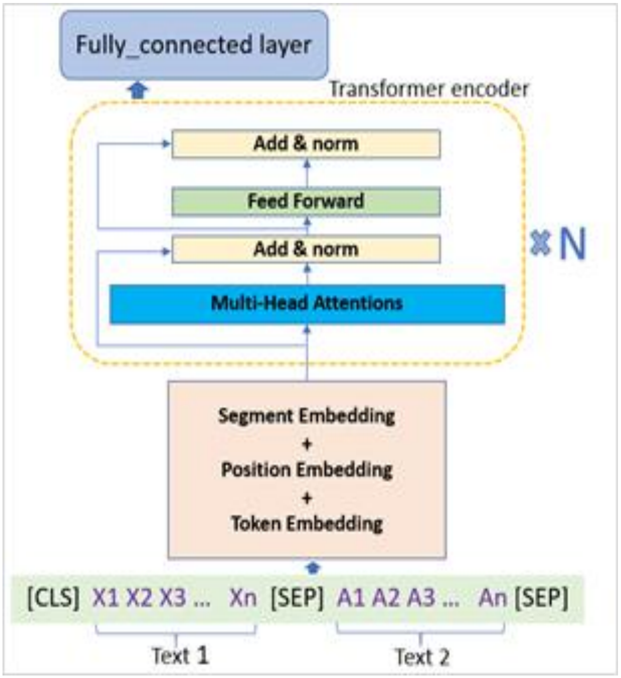

Fig. 2. The overall structure of the finetuned bert layer.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

Where Q(query) is the matrix of queries, representing the current state or the part of the input we are focusing on, K(key) represents the matrix of keys, representing the entire input sequence, V(value) is the dimension of the key vectors, used for scaling the dot product, and d_k the dimension of the key vectors, used for scaling the dot product.

B. LSTM Component

Aiming to augment an already outstanding model into an even more proficient automatic answer-scoring framework, we delved into the concept of incorporating an LSTM layer into the final fully connected layer of the transformers within BERT.

Recurrent neural network (RNN) architectures with Long Short-Term Memory (LSTM) networks are intended to simulate sequences and their dependence upon them over time more accurately than RNNs with typical architectures. LSTMs were introduced by Hochreiter and Schmidhuber in 1997[24] and have since become a fundamental building block for many sequential data processing tasks. LSTMs are composed of units called LSTM cells, which replace the simple neurons in standard RNNs. Each LSTM cell maintains a cell state (C_t) defined in Formula (6) and three gates that control the flow of information: the input gate (I_t), the forget gate (F_t), and the output gate (O_t) defined in Formulas (3), (2) and (4) respectively. Forget Gate eliminates data that is no longer helpful to the LSTM, which has a sigmoid layer for decision-making. *tanh* and sigmoid layers are used by the input gate, which is in charge of adding pertinent data to the existing cell state. With the use of a sigmoid layer, the output gate displays the pertinent data from the current cell. "Fig. 3" displays the construction of the LSTM.

$$F_t = \sigma\left(W_f \,.\, [h_{t-1}, x_t] + b_f\right) \quad (2)$$

$$I_t = \sigma(W_i \,.\, [h_{t-1}, x_t] + b_i) \quad (3)$$

$$O_t = \sigma(W_o \,.\, [h_{t-1}, x_t] + b_o) \quad (4)$$

$$\tilde{C}_t = tanh(W_c \,.\, [h_{t-1}, x_t] + b_c) \quad (5)$$

$$C_t = F_t \,.\, C_{t-1} + I_t \,.\, \tilde{C}_t) \quad (6)$$

$$h_t = O_t \,.\, \tanh(C_t) \quad (7)$$

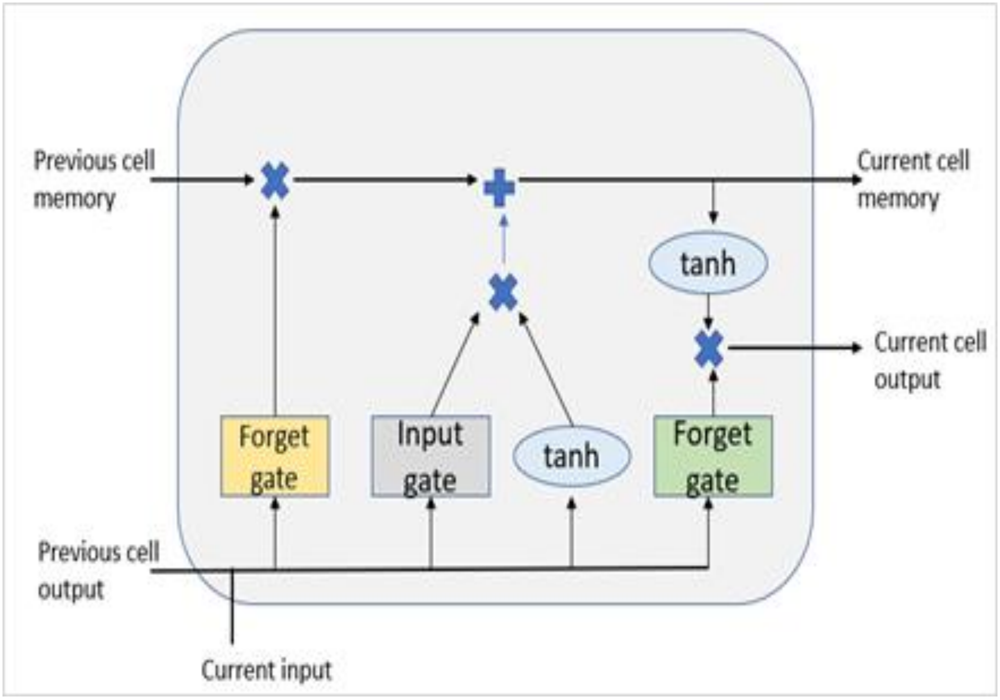

Fig. 3. A single unit of long short-term memory (LSTM) neural networks.

IV. Experiments

A. Dataset

This study leverages the computer science dataset created by [10], which is a comprehensive collection of student responses from the University of North Texas, specifically designed to evaluate the effectiveness of automatic scoring systems. The dataset contains 2273 student answers associated with 80 questions, sourced from 10 different assignments and two tests within the field of computer science. Each answer provided by the students was independently scored by two teachers, utilizing an integer-based grading scale that ranges from 0 to 5, where 0 represents an incorrect answer and 5 indicates a fully correct response. The true score of the student's answer was determined by taking the average of the two scores that were labeled, which resulted in 11 scoring grades ranging from 0 to 5 with 0.5 intervals between each grade. The shape of the dataset is (2273, 7). It has 2273 rows and seven columns. The columns have questions, desired answers, student answers, and scores.

B. Evaluation Measures

To evaluate our model, we adopt the standard metrics used by the previous automatic scoring systems. As explained above, we model the problem as a regression task, specifically using the Pearson correlation coefficient (Pearson's r) and root mean square error (RMSE) as metrics to measure the performance of the proposed approach. Below, we report the different metrics with their mathematical expressions:

Root-Mean-Squared Error (RMSE): The use of RMSE is very common, and it is considered an excellent general-purpose error metric for numerical predictions. RMSE is defined in Formula (8).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\widehat{y_i} - y_i)^2} \tag{8}$$

Where n is the number of samples, $\widehat{y_i}$ is the predicted value and y_i represent the actual value.

1) *Pearson's r[25]*: measures the strength and direction of the linear relationship between two continuous variables. In the context of evaluating model performance, Pearson's r can be used to assess the correlation between predicted scores generated by a model and the actual scores assigned by human evaluators. Pearson's r is defined in Formula (9) :

$$r = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\widehat{y_i} - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2 \sum_{i=1}^{n}(\widehat{y_i} - \bar{\hat{y}})^2}} \tag{9}$$

Where y_i the actual value, $\widehat{y_i}$ the predicted value, $\bar{y}$ the mean of actual values and $\bar{\hat{y}}$ is the mean of predicted values.

C. Implementation Details

Our experiments were conducted in the Google Collaboratory environment, which provides access to high-performance GPUs, facilitating efficient and accelerated model training and evaluation. For the implementation of our models, we utilized Python as the primary programming language, leveraging powerful libraries such as NLTK (Natural Language Toolkit) for text preprocessing and PyTorch for building and training our neural network models. The dataset used in our study, consisting of student answers and corresponding questions, was divided into two distinct sets: 80% of the data was allocated for training the models, while the remaining 20% was reserved for testing their performance.

The implementation of our method consists first of a preprocessing step to prepare data in the best format for training.

For that, we consider two functions based on their effect on training results. The first function removes stop words from the texts. The second function is removing words that appear in the question text from both the reference answer and the student response to prevent the model from unfairly rewarding a student response that simply repeats words from the question. The texts are then tokenized using Bert tokenizer, which converts them into tokens. These tokens are further processed into input IDs, attention masks, and token type IDs, forming the input features for the BERT model. We use the Bert_base_uncased version as the pre-trained model of our Bert layer with 12 transformer layers, 768 hidden units per layer, and total parameters of ¼ 110M. The "uncased" nature of this model converts text to lowercase and removes accent markers, simplifying vocabulary handling. It employs WordPiece embeddings with a 30,000-token vocabulary, supporting input sequences up to 512 tokens. This model provides dynamic, context-sensitive word embeddings, significantly improving our model's ability to accurately evaluate student answers through fine-tuning our specific dataset. The output of the fine-tuned BERT model, which consists of contextualized embeddings of the paired answers, is then fed into an LSTM layer followed by a linear output layer for regression. The specific parameter settings for the model are detailed in Table I. This configuration allows the model to capture both the intricate context provided by BERT and the sequential dependencies effectively modeled by the LSTM.

TABLE I. PARAMETER SETTINGS OF THE PROPOSED MODEL

Parameter	Value
Batch size	16
Epochs	10
Bert's finetuned learning rate	5e-5
Lstm_hidden_size	256
Number lstm layers	2
Lstm learning rate	1e-3
Optimize	ADAM
Loss function	Mean Square Error

V. RESULTS AND DISCUSSION

A. Ablation Study

In our proposed model, we conducted an ablation study to evaluate the impact of specific components on performance. The study focused on two key variations:

1) BERT fine-tuning with and without question demotion: We examined the effect of removing question-related words from the student's answer before feeding it into the model. This step aims to reduce noise and focus on the unique content of the student's response. By comparing the performance of the model with and without question demotion, we aimed to assess its contribution to the overall accuracy.

2) BERT fine-tuning with and without adding an LSTM layer: To determine the added value of incorporating a Long Short-Term Memory (LSTM) layer, we compared the results of the fine-tuned BERT model both with and without the LSTM layer. The LSTM layer is designed to capture sequential dependencies and provide additional context to the BERT representations. This comparison helps to understand whether the LSTM layer enhances the model's ability to accurately score the answers.

As illustrated in Table II, removing question demotion results in a higher RMSE (0.931 vs. 0.785) and a lower Pearson correlation (0.723 vs. 0.761). This indicates that question demotion significantly contributes to the model's ability to accurately score answers. Question demotion likely helps the model focus on the core content of student answers without being misled by repetitive or irrelevant information from the questions, leading to better alignment with the desired answers.

TABLE II. RESULTS OF THE ABLATION STUDIES

Model Variant	RMSE	Pearson Correlation
Without Question Demotion	0.931	0.723
Without LSTM Layer	0.819	0.741
Full Model (with all components)	**0.785**	**0.761**

Adding the LSTM layer to the model improves performance, reducing the RMSE from 0.819 to 0.785 and increasing the Pearson correlation from 0.741 to 0.761. The LSTM layer likely helps capture sequential dependencies and fine-grained contextual information that the BERT layer might not fully encode, resulting in better performance.

The full model, which includes both question demotion and the LSTM layer, performs the best with the lowest RMSE (0.785) and the highest Pearson correlation (0.761). This demonstrates that both components are essential for achieving optimal performance in automatic answer scoring.

B. Comparison with Baseline Models

We compare the performance of our model with various baseline models based on RME and Pearson correlation scores. The comparison results are illustrated in Table III.

As can be seen from the experimental findings, systems that are based on handcrafted features are relatively yield low to moderate accuracy. Among these methods, the BOW (Bag of Words) approach combined with SVMRank [10] exhibited the best performance, yielding a Pearson's correlation coefficient of 0.480 and an RMSE of 1.042. This indicates that while feature engineering-based models can capture some relevant aspects of the answer-scoring task, their performance is limited compared to more advanced deep-learning models. The moderate correlation and relatively high RMSE suggest that these methods might struggle with capturing the deeper semantic relationships and nuances present in the text. Combining semantic network approaches using Glove and Word2Vec embeddings along with an SVM model slightly improves the performance metrics, achieving a Pearson's correlation coefficient of 0.631 and an RMSE of 0.834 [26]. This enhancement suggests that integrating semantic information from pre-trained embeddings can better capture the underlying meaning and context of the text, leading to more accurate scoring. However, the improvement is still moderate, indicating that these traditional machine learning methods, even when augmented with semantic embeddings, may not fully exploit the complexities of the language as effectively as more advanced deep learning techniques. Using dynamic embeddings only, without fine-tuning pre-trained models such as ELMo, GPT, and BERT, performs poorly in similarity regression tasks. For instance, the results show that traditional word embeddings (e.g., Word2Vec, GloVe) yield better performance metrics compared to contextual embeddings (e.g., ELMo, BERT) [17]. This observation highlights that merely leveraging the powerful pre-trained models without task-specific fine-tuning can lead to suboptimal results, as these models may not fully align with the specific requirements and nuances of the target task.

The experiments show that the fine-tuned BERT model performs very well, achieving RMSE and Pearson correlation values of 0.819 and 0.741, respectively. This indicates that task-specific fine-tuning significantly enhances the model's ability to capture the nuances and intricacies of the dataset, resulting in improved scoring accuracy and correlation with the target metrics. Finally, the results show that adding an LSTM layer on top of the fine-tuned BERT model improves the results, achieving a Pearson correlation of 0.761 and an RMSE of 0.785. This significantly surpasses the results of all baseline systems, demonstrating the effectiveness of combining BERT's powerful language representation with LSTM's ability to capture long-term dependencies. The experiments highlight the limitations of feature engineering-based and dynamic embedding-only models. Fine-tuning pre-trained models, especially when combined with additional layers like LSTM, significantly improves performance. Our proposed model, BERT Fine-Tuned Based LSTM, achieves the best results, establishing a new benchmark for automatic answer scoring on the Mohler dataset.

TABLE III. COMPARISON RESULTS ON THE MOHLER DATASET

System	description		RMSE	Pearson correlation
[10]	BOW (Bag of Words) approach with SVMRank		1.042	0.480
	BOW (Bag of Words) approach with SVR		0.999	0.431
	Tf-idf with SVR		1.022	0.327
[11]	tf-idf with LR (Logistic Regression) and SIM (Semantic Information)		0.887	0.592
[12]	HoPSTags + Sentence Embedding features		0.921	0.542
[17]	Dynamic embeddings (not fine-tuned + cosine similarity feature	ELMO	0.978	0.485
		GPT	1.082	0.248
		BERT	1.057	0.318
		GPT_2	1.065	0.311
[26]	Semantic network with SVM		0.834	0.631
(In this work)	Word2vec & mean_pooling with cosine similarity feature		1.005	0.405
	Bert(embedding only) & mean_pooling with cosine similarity feature		1.021	0.367
	Fine_tuned Bert_base		0.819	0.741
(Proposed model)	BERT Fine-Tuned Based LSTM		**0.785**	**0.761**

VI. Conclusion

In this paper, we introduce a new method for automatic answer scoring by leveraging the strengths of both transformer-based and classical neural network architectures. The proposed model contains a fine-tuned layer of the pre-trained BERTbase model for contextualized embedding extraction, followed by an LSTM layer to benefit from its sequence modeling capabilities for more improvement. The model was trained using the Mohler dataset, a benchmark corpus widely used for automatic scoring tasks. In the experiments, we compared our model with several state-of-the-art models to evaluate the performance. The results demonstrated that our approach shows significant improvement regarding both RMSE and Pearson correlation measures. These improvements underscore our model's enhanced capability to understand and evaluate the semantic content of both student and reference answers, leading to more accurate grading outcomes. In future work, we can improve such an automatic scoring system so it can face the problem of different distributions that can arise due for example to differences in question types between the current question answers (training set) and the new question answers (test set). Strategies based on domain adaptation and transfer learning can be employed to address this case.

References

[1] V. Salvatore, N. Francesca, et A. Cucchiarelli, « An Overview of Current Research on Automated Essay Grading », J. Inf. Technol. Educ., vol. 2, janv. 2003, doi: 10.28945/331.

[2] Y. Goldberg, « Neural Network Methods for Natural Language Processing », Synth. Lect. Hum. Lang. Technol., vol. 10, p. 1-309, avr. 2017, doi: 10.2200/S00762ED1V01Y201703HLT037.

[3] J. G. A. Mantecon, H. A. Ghavidel, A. Zouaq, J. Jovanovic, et J. McDonald, « A Comparison of Features for the Automatic Labeling of Student Answers to Open-Ended Questions », International Educational Data Mining Society, juill. 2018. Consulté le: 11 juillet 2024. [En ligne]. Disponible sur: https://eric.ed.gov/?id=ED593101

[4] J. Devlin, M.-W. Chang, K. Lee, et K. Toutanova, « BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding », 24 mai 2019, arXiv: arXiv:1810.04805. doi: 10.48550/arXiv.1810.04805.

[5] E. B. Page, « Computer Grading of Student Prose, Using Modern Concepts and Software », J. Exp. Educ., vol. 62, no 2, p. 127-142, janv. 1994, doi: 10.1080/00220973.1994.9943835.

[6] L. James, « CAA of Short Non-MCQ Answers », 2001.

[7] L. F. Bachman et al., « A Reliable Approach to Automatic Assessment of Short Answer Free Responses », in COLING 2002: The 17th International Conference on Computational Linguistics: Project Notes, 2002. Consulté le: 12 juillet 2024. [En ligne]. Disponible sur: https://aclanthology.org/C02-2023

[8] P. Thomas, « The evaluation of electronic marking of examinations », sept. 2003, doi: 10.1145/961511.961528.

[9] M. Mohler et R. Mihalcea, « Text-to-Text Semantic Similarity for Automatic Short Answer Grading », in Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), A. Lascarides, C. Gardent, et J. Nivre, Éd., Athens, Greece: Association for Computational Linguistics, mars 2009, p. 567-575. Consulté le: 12 juillet 2024. [En ligne]. Disponible sur: https://aclanthology.org/E09-1065

[10] M. Mohler, R. Bunescu, et 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, Portland, Oregon, USA: Association for Computational Linguistics, juin 2011, p. 752-762. [En ligne]. Disponible sur: https://aclanthology.org/P11-1076

[11] M. A. Sultan, C. Salazar, et 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, San Diego, California: Association for Computational Linguistics, 2016, p. 1070-1075. doi: 10.18653/v1/N16-1123.

[12] S. Saha, T. I. Dhamecha, S. Marvaniya, R. Sindhgatta, et B. Sengupta, « Sentence Level or Token Level Features for Automatic Short Answer Grading?: Use Both », in Artificial Intelligence in Education, vol. 10947, C. Penstein Rosé, R. Martínez-Maldonado, H. U. Hoppe, R. Luckin, M. Mavrikis, K. Porayska-Pomsta, B. McLaren, et B. du Boulay, Éd., in Lecture Notes in Computer Science, vol. 10947. , Cham: Springer International Publishing, 2018, p. 503-517. doi: 10.1007/978-3-319-93843-1_37.

[13] T. D. Metzler, P. G. Plöger, et G. Kraetzschmar, « Computer-assisted grading of short answers using word embeddings and keyphrase extraction », PhD Thesis, Master's thesis, Hochschule Bonn-Rhein-Sieg, Germany, 2019.

[14] S. Kumar, S. Chakrabarti, et S. Roy, « Earth Mover's Distance Pooling over Siamese LSTMs for Automatic Short Answer Grading », in Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia: International Joint Conferences on Artificial Intelligence Organization, août 2017, p. 2046-2052. doi: 10.24963/ijcai.2017/284.

[15] C. N. Tulu, O. Ozkaya, et U. Orhan, « Automatic Short Answer Grading With SemSpace Sense Vectors and MaLSTM », IEEE Access, vol. 9, p. 19270-19280, 2021, doi: 10.1109/ACCESS.2021.3054346.

[16] A. Vaswani et al., « Attention Is All You Need », 5 décembre 2017, arXiv: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[17] S. K. Gaddipati, D. Nair, et P. G. Plöger, « Comparative Evaluation of Pretrained Transfer Learning Models on Automatic Short Answer Grading », 2 septembre 2020, arXiv: arXiv:2009.01303. Consulté le: 13 mai 2024. [En ligne]. Disponible sur: http://arxiv.org/abs/2009.01303

[18] L. Camus et A. Filighera, « Investigating Transformers for Automatic Short Answer Grading », in Artificial Intelligence in Education, I. I. Bittencourt, M. Cukurova, K. Muldner, R. Luckin, et E. Millán, Éd., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, p. 43-48. doi: 10.1007/978-3-030-52240-7_8.

[19] C. Sung, T. I. Dhamecha, et N. Mukhi, « Improving Short Answer Grading Using Transformer-Based Pre-training », in Artificial Intelligence in Education, S. Isotani, E. Millán, A. Ogan, P. Hastings, B. McLaren, et R. Luckin, Éd., Cham: Springer International Publishing, 2019, p. 469-481. doi: 10.1007/978-3-030-23204-7_39.

[20] R. Dadi et S. Sanampudi, « An automated essay scoring systems: a systematic literature review », Artif. Intell. Rev., vol. 55, p. 1-33, mars 2022, doi: 10.1007/s10462-021-10068-2.

[21] A. Rogers, O. Kovaleva, et A. Rumshisky, « A Primer in BERTology: What We Know About How BERT Works ».

[22] W. Liao, B. Zeng, X. Yin, et P. Wei, « An improved aspect-category sentiment analysis model for text sentiment analysis based on RoBERTa », Appl. Intell., vol. 51, no 6, p. 3522-3533, juin 2021, doi: 10.1007/s10489-020-01964-1.

[23] A. Baqach et B. Amal, « A new sentiment analysis model to classify students' reviews on MOOCs », Educ. Inf. Technol., p. 1-28, févr. 2024, doi: 10.1007/s10639-024-12526-0.

[24] S. Hochreiter et J. Schmidhuber, « Long Short-Term Memory », Neural Comput., vol. 9, no 8, p. 1735-1780, nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[25] D. G. Bonett et T. A. Wright, « Sample size requirements for estimating pearson, kendall and spearman correlations », Psychometrika, vol. 65, no 1, p. 23-28, mars 2000, doi: 10.1007/BF02294183.

[26] N. H. Hameed et A. T. Sadiq, « Automatic Short Answer Grading System Based on Semantic Networks and Support Vector Machine », Iraqi J. Sci., p. 6025-6040, nov. 2023, doi: 10.24996/ijs.2023.64.11.44.