Terbit online pada laman web jurnal: **https://jurnal.plb.ac.id/index.php/tematik/index**

T E M A T I K

Jurnal **Teknologi Informasi Komunikasi (e-Journal)**

Vol. 13 No. 1 (2026) 53 - 63 ISSN Media Elektronik: 2443-3640

Essay Score Prediction Based On Combined Question And Answer Data Using FastText and Bi-LSTM Algorithm

Sefti Agustini[1*], M. Ainul Yaqin[2] , Suhartono[3]

[1,2,3] Informatika, Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim, Malang

[1]220605220003@student.uin-malang.ac.id, [2]yaqinov@ti.uin-malang.ac.id, [3]suhartono@ti.uin-malang.ac.id

Abstract

Automated Essay Scoring is one of the challenges in the field of educational technology, particularly in subjects requiring language assessment, such as the English language. Manual assessment performed by teachers is time-consuming, subjective, and has the potential for inconsistency between assessors, which can give rise to unfairness in scoring. In order to overcome this issue, this study proposes an Automated Essay Scoring (AES) approach by combining FastText word embeddings and the Long Short-Term Memory (LSTM) algorithm to predict the scores of student essays. The innovation of this research lies in combining question and answer data into a single input variable, where the student's answer is positioned between the special tokens <startanswer> (prefix) and <endanswer> (suffix), allowing the model to learn the context of the question and answer simultaneously. The data 500 data points (100 students x 5 essay questions). The LSTM model was then trained with a combination of hyperparameters, namely the number of units in the hidden layer, learning rate, batch size, and dropout rate. Model performance was measured using Mean Absolute Percentage Error. Based on the experimental results, the Bi-LSTM model with the best hyperparameter settings achieved a MAPE value of 7.84%, which is better than the unidirectional LSTM model at 9.12. This study demonstrates that the combination of FastText and Bi-LSTM is a promising approach for essay score prediction in the context of language learning for junior high school students.

Keywords: Automated Essay Scoring, FastText, LSTM, Bi-LSTM, MAPE, Essay Score Prediction

Abstrak

Automated Essay Scoring merupakan salah satu tantangan dalam bidang teknologi pendidikan, khususnya pada mata pelajaran yang memerlukan penilaian bahasa, seperti bahasa Inggris. Penilaian manual yang dilakukan oleh guru membutuhkan waktu yang lama, bersifat subjektif, dan berpotensi menimbulkan ketidakkonsistenan antarpenilai yang dapat menyebabkan ketidakadilan dalam pemberian skor. Untuk mengatasi permasalahan tersebut, penelitian ini mengusulkan pendekatan Automated Essay Scoring (AES) dengan menggabungkan word embedding FastText dan algoritma Long Short-Term Memory (LSTM) untuk memprediksi skor esai siswa. Inovasi dalam penelitian ini terletak pada penggabungan data pertanyaan dan jawaban ke dalam satu variabel input, di mana jawaban siswa ditempatkan di antara token khusus `<startanswer>` (prefiks) dan `<endanswer>` (sufiks), sehingga memungkinkan model untuk mempelajari konteks pertanyaan dan jawaban secara bersamaan. Data yang digunakan terdiri dari 500 titik data (100 siswa x 5 soal esai). Model LSTM kemudian dilatih dengan kombinasi hyperparameter, yaitu jumlah unit pada hidden layer, learning rate, batch size, dan dropout rate. Performa model diukur menggunakan Mean Absolute Percentage Error (MAPE). Berdasarkan hasil eksperimen, model Bi-LSTM dengan pengaturan hyperparameter terbaik mencapai nilai MAPE sebesar 7,84%, yang lebih baik dibandingkan dengan model LSTM searah sebesar 9,12%. Penelitian ini telah membuktikan bahwa kombinasi FastText dan Bi-LSTM merupakan pendekatan yang efektif untuk prediksi skor esai dalam konteks pembelajaran bahasa bagi siswa sekolah menengah pertama.

Kata kunci: Automated Essay Scoring, FastText, LSTM, Bi-LSTM, MAPE, Prediksi Skor Esai

1. Introduction

The evolution of artificial intelligence technology in the field of education has led to a significant shift in methods of assessing student learning performance. One type of assessment that continues to challenge educators is essay assessment, particularly in the English subject at the junior high school (SMP) level. The method of manual essay grading by teachers is not only time-consuming but can also produce subjective and inconsistent assessments between assessors, potentially disrupting the fairness of scoring [1] . For this reason, the development of automated essay scoring systems, commonly known as Automated Essay Scoring (AES), is urgently needed in today's educational system [2].

Over the past decades, researchers have offered several solutions to the AES problem, ranging from approaches based on manual linguistic features to those based on deep learning. Approaches based on manual features, such as lexical, syntactic, and statistical feature

Diterima Redaksi: 22-05-202 | Selesai Revisi: 05-06-2026 | Diterbitkan Online: 09-06-2026

analysis, tend to require domain knowledge and struggle to capture complex semantic meanings from text [3]. On the other hand, deep learning-based approaches have been proven more effective in learning complex patterns in text data without manual feature engineering [4], [5]. Furthermore, this capability makes deep learning a highly promising solution to be applied to AES.

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that has been successfully used in processing sequential data such as text [6]. LSTM can avoid the vanishing gradient problem commonly faced by conventional RNNs through the use of a gate mechanism (forget gate, input gate, output gate) [7]. Several previous studies have proven that LSTM is capable of handling long-term dependencies in text, which is crucial for understanding the coherence and structure of an essay [8]. Furthermore, the development of various LSTM architectures, such as Bidirectional LSTM (Bi-LSTM), allows the model to process word sequences from both directions, enabling the model to capture more context [9].

One important aspect in AES analysis using deep learning is word representation (word embedding). Word embedding serves as a bridge between raw text data and the learning model by representing words as numerical vectors that encode the meaning of those words [10]. One word embedding technique, FastText by Facebook AI Research, has several advantages in representing rare words (low-frequency words) and out-of-vocabulary words (words not in the training vocabulary) [11]. In addition, FastText represents words as a bag of character n-grams, hence, morphological information can be captured more effectively [12]. Previous research has shown that FastText produces better word representations compared to Word2Vec and GloVe, particularly for languages with rich morphological variation [10].

Several studies have investigated the use of LSTM for AES tasks in various configurations. For example, a study that combined Sentence-BERT with LSTM and an attention mechanism succeeded in improving essay scoring accuracy [13]. Other research has explored the use of a combination of Convolutional Neural Network (CNN) and LSTM to capture local and global semantic features of an essay [14]. The improvement results from these studies prove that by using high-quality word embeddings and appropriate deep learning architectures, accurate AES results can be obtained. However, most research in this area only takes the student's answer as input, and does not take into account the question context to which the student's answer relates [15] .

Table 1 summarizes a comparison between this study and representative prior works on AES, covering aspects such as word embedding method, model architecture, input composition (answer-only vs. question+answer), evaluation metric, dataset, and reported performance. This comparison reveals a clear research gap: while prior studies have shown the effectiveness of LSTM-based and CNN-LSTM architectures on standard English datasets such as ASAP, none have employed a simple token-based question-answer concatenation strategy combined with FastText in the context of Indonesian junior high school English essays. The absence of question-context modeling in answer-only systems constitutes the primary gap that the present research addresses.

In fact, question information is very important in essay scoring, since the relevance of the answer to the question asked is one of the criteria in essay scoring [16]. Without considering the question, a model can only assess the general language quality but cannot evaluate whether the student's answer is relevant to the question posed [17]. Furthermore, most previous works that attempt to incorporate question information commonly adopt a relatively complex network architecture, such as Siamese networks or encoder-decoders, which require large computational resources [18]. Therefore, a simpler but still effective approach to integrating question and answer information should be further investigated.

Table 1. Presents a comparison between the proposed study and representative prior AES studies.

No	Embedding	Architecture	Input	Dataset	Metric	Performance
[3]	Handcrafted features	DNN + RNN	Answer only	ASAP	QWK	~0.80
[6]	SBERT	LSTM + Attention	Answer only	Benchmark	QWK	Improved accuracy
[13]	Multi-feature	CNN + LSTM + Attention	Answer only	Custom	Accuracy	State-of-the-art
[14]	Word2Vec	CNN-Attention-LSTM	Answer only	ASAP	Accuracy	Competitive
[15]	TF-IDF + semantic	SVM-based hybrid	Answer only	Higher education	Kappa	Reliable
[16]	Transformer	Efficient BERT	Answer only	ASAP	QWK	~0.83
[18]	BERT	Multi-scale LSTM	Answer only	ASAP	QWK	High
This study	FastText	Bi-LSTM	Question + Answer	SMP Negeri 1 Malang (Indonesian EFL)	MAPE	7.84%

DOI: https://doi.org/10.38204/tematik.v13i1.2974

In the context of English language education in Indonesia, particularly at the SMP level, the design of AES systems faces unique challenges. Junior high school students generally have limited language proficiency and often mix English with Indonesian in their answers [19]. In addition, the length of student answers is not always consistent and does not take the form of long essays as found in international datasets, such as the Automated Student Assessment Prize (ASAP) [20]. Therefore, a method is needed that can adapt to the answers of Indonesian junior high school students in the English subject.

In addition, assessing model performance in essay score prediction is also a matter requiring attention. Having intuitive interpretability, Mean Absolute Percentage Error (MAPE) is a metric with the advantage of being able to provide relative percentage errors [21]. MAPE is very suitable for use in essay assessment because it provides information on how far the model's score prediction deviates from the original score, expressed as a percentage of the original score [22]. In addition, with the use of MAPE, fairer comparisons across different models can be performed regardless of the scale of scores [23].

Aside from the choice of model architecture and evaluation metrics, hyperparameter selection is also crucial for the performance of deep learning models. Moreover, hyperparameter tuning is an important step to obtain optimal model performance [24]. Important hyperparameters in LSTM models include the number of hidden layer units, learning rate, batch size, number of epochs, and dropout level [25]. Without the tuning process, deep learning models may perform poorly or even suffer from overfitting [26].

Several recent studies also demonstrate the value of information augmentation in natural language processing tasks. Inserting special tokens to mark specific parts of input data has been found effective in helping models understand data patterns [27]. A similar strategy has been applied to large language models such as BERT and GPT, which use special tokens for various tasks [28]. The use of a similar approach in AES tasks can lead to promising results, particularly when combining question and answer information into a single input sequence [29].

A critical synthesis of the prior literature reveals several important limitations. First, studies combining LSTM with attention mechanisms or CNN have demonstrated performance gains on standard datasets (e.g., ASAP), but they consistently rely on answer-only input, ignoring the semantic relationship between the question prompt and the student response. Second, although transformer-based approaches (e.g., BERT, SBERT) achieve high accuracy, their large parameter count and computational demands make them impractical for resource-constrained deployments in Indonesian schools. Third, FastText has been shown to outperform Word2Vec and GloVe for morphologically rich languages, yet it has not been applied to AES in the Indonesian EFL context. These gaps collectively motivate the present study's focus on a lightweight, interpretable approach that integrates question context through token-based concatenation, uses FastText for robust word representation, and evaluates both LSTM and Bi-LSTM architectures under controlled hyperparameter conditions. An ablation study addressing the contribution of question-context tokens and the choice of word embedding is recommended for future work to empirically isolate each proposed element's effect on model performance.

Based on the background described above, this research proposes a new approach to improve essay scoring systems by combining FastText word embeddings and the LSTM algorithm. The innovations in this research can be seen from three aspects. First, the inclusion of student answers between exam questions with the special token <startanswer> at the beginning of the answer and the special token <endanswer> at the end, allowing the model to learn the context of both the question and answer simultaneously in a single input variable. Second, the use of FastText as a word representation method to process morphological variations and rare words frequently found in junior high school students' answers. Third, a systematic comparison between the unidirectional LSTM model and the Bi-LSTM model to determine the appropriate model architecture for the data used.

2. Methods

2.1. Research Framework

This research uses a quantitative approach with structured experiments. The overall research framework consists of several stages, including data gathering, text pre-processing, word representation (using FastText), LSTM and Bi-LSTM model training, and performance evaluation (using MAPE).

The research process began with the collection of student essay answer data from SMP Negeri 1 Malang. The collected data was then pre-processed, encompassing several steps: text cleaning, tokenization, and appending tokens to delimit parts of the input. Subsequently, word representations were generated using FastText, which were then used to train the LSTM model. Hyperparameter tuning was performed to find the optimal parameter configuration, which was then used to train the model. Through the Bi-LSTM model trained with the same parameters, this research aimed to observe how the direction of text processing (bidirectional) influences model predictions.

DOI: https://doi.org/10.38204/tematik.v13i1.2974

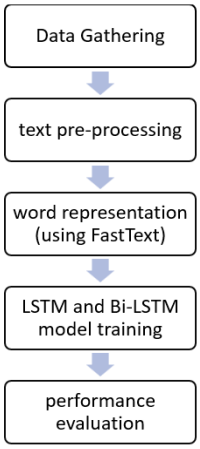


Figure 1. Research flowchart for essay score prediction

2.2. Data Collection

The data used in this research were collected from SMP Negeri 1 Malang for the English subject. The dataset was collected directly by the researchers during the 2023/2024 academic year and is not publicly available under a named benchmark; it constitutes a domain-specific corpus of Indonesian junior high school English essay responses. The total data used in this study is 500 data points, consisting of answers from 100 students for 5 essay questions. Each data point represents a question-answer pair from a student, accompanied by the score for that question provided by a teacher as an assessor. The scores given range from 0 to 100, with variation resulting from the quality of the answers provided by students. The data structure used in this study is presented in Table 1.

The data was then split into three portions with a ratio of 70:15:15, for training data, validation data, and test data, respectively. Based on this ratio, 350 data points were used for training, 75 data points for validation (to calculate accuracy during the model tuning stage), and 75 data points for model testing.

Table 1. Example of research data structure.

Student ID	Question ID	Question	Student Answer	Score
S001	Q1	Describe your daily routine in simple present tense!	I wake up at 5 a.m., then I pray, take a shower, and go to school by bike.	85
S001	Q2	Explain the difference between 'since' and 'for'!	Since is used for a point in time; for is used for a duration of time.	90
...	...	...	...	...
S100	Q5	Write a short paragraph about your hobby!	I love playing badminton on weekends with my friends at the community hall.	80

2.3. Data Preprocessing

The data pre-processing stage is a crucial step to ensure the quality of data input into the model. The data pre-processing used in this research consists of several steps, described as follows.

The first step involves converting text to lowercase (lowercasing) to prevent word duplication due to differences in capitalization. Second, special characters unrelated to the essay content, such as repeated punctuation and non-alphabetic characters, are removed through text cleaning. Third, tokenization is performed to break text into word tokens processed by the model. Fourth, stop-word removal is performed with careful consideration of semantically important function words. Specifically, negation words (“not,” “no”), auxiliary verbs (“is,” “am,” “are,” “was,” “were”), and temporal/causal connectors (“since,” “for,” “because”) are retained, as these words carry grammatical and semantic significance in English essay responses. General high-frequency words with low discriminatory value (e.g., “the,” “a,” “an”) are removed using a customized stop-word list derived from the NLTK English stop-word corpus, with the aforementioned exceptions manually excluded

The next step in pre-processing is to create the input variable X by concatenating the questions and student answers with special tokens. The input used in this study takes the following form: X = [Question] <startanswer> [Student Answer] <endanswer>

A concrete example of this input format is as follows: Describe your daily routine in simple present tense! <startanswer> I wake up at 5 a.m., then I pray, take a shower, and go to school by bike. <endanswer>

The use of the token <startanswer> as a prefix and <endanswer> as a suffix allows the model to distinguish between the question and answer parts, so the model can learn the contextual relationship between the question and answer. Furthermore, this method is analogous to using special tokens such as [CLS] and [SEP] as used in BERT, but adapted to the AES setting of interest here.

2.4. Word Representation Using FastText

After pre-processing, each word in the input sequence is represented as a numerical vector using FastText. Furthermore, FastText represents a word as a series of character n-grams, so that rare words or words not in the training vocabulary can still be well represented. Mathematically, the word vector w in FastText can be written as Equation 1.

$$v\_w = \left(\frac{1}{|G\_w|}\right) \times \sum(z\_g) \quad (1)$$

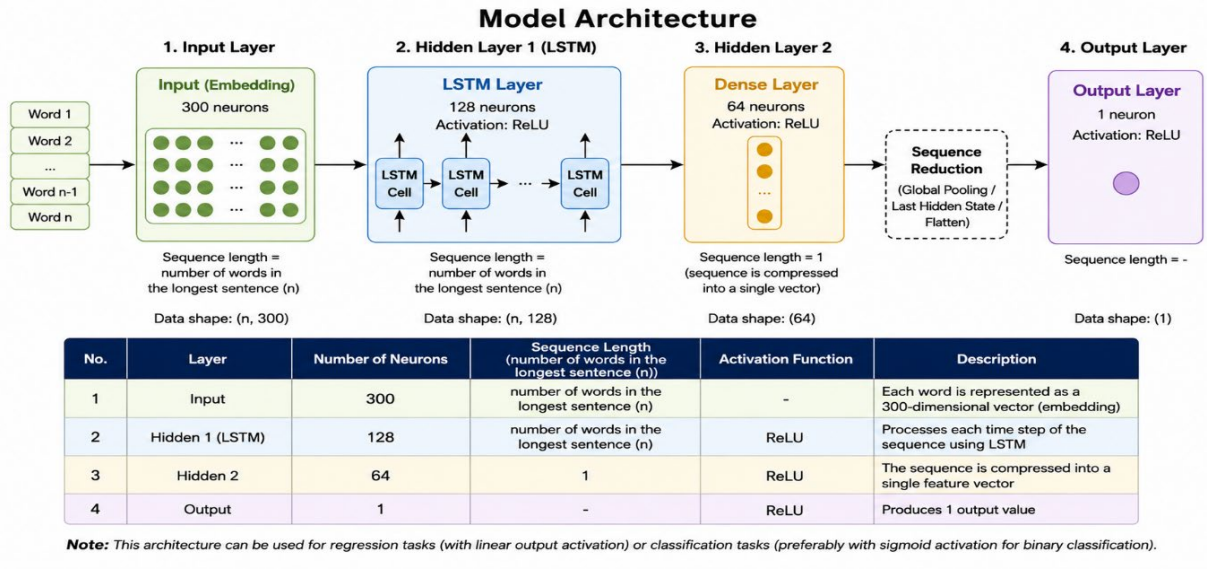
DOI: https://doi.org/10.38204/tematik.v13i1.2974

for each g in G_w, where G_w is the set of character n-grams of word w and z_g is the vector of n-gram g. In this research, a pre-trained FastText model with a vector dimension of 300 is used. The special tokens <startanswer> and <endanswer> are also represented by vectors after re-training using the research data.

Furthermore, the maximum sequence length was set at 150 tokens after analyzing the distribution of data lengths. Sequences shorter than 150 tokens were padded with zero vectors, and sequences longer than 150 tokens were truncated to the specified maximum length.

2.5. LSTM Model Architecture

The LSTM model used in this research employs a multi-layer LSTM architecture. The model architecture is illustrated in Figure 2.



No.	Layer	Number of Neurons	Sequence Length (number of words in the longest sentence (n))	Activation Function	Description
1	Input	300	number of words in the longest sentence (n)	-	Each word is represented as a 300-dimensional vector (embedding)
2	Hidden 1 (LSTM)	128	number of words in the longest sentence (n)	ReLU	Processes each time step of the sequence using LSTM
3	Hidden 2	64	1	ReLU	The sequence is compressed into a single feature vector
4	Output	1	-	ReLU	Produces 1 output value

Note: *This architecture can be used for regression tasks (with linear output activation) or classification tasks (preferably with sigmoid activation for binary classification).*

Figure 2. illustrates the multi-layer LSTM architecture used in this study. The model consists of (1) an Embedding Layer that receives pre-trained FastText vectors of dimension 300 (set as non-trainable); (2) an LSTM Layer with the number of units determined through hyperparameter tuning, incorporating forget gate, input gate, and output gate mechanisms; (3) a Dropout Layer to reduce overfitting; (4) a Dense Layer with ReLU activation; and (5) a linear output neuron that produces a continuous score prediction

The first layer is an Embedding Layer for receiving FastText representations with a dimension of 300. This layer is set as non-trainable to preserve the semantic representation acquired from the pre-trained FastText model. The second layer is an LSTM Layer with a number of units optimized during tuning. Each LSTM unit comprises three gates, namely the forget gate, input gate, and output gate, which control the flow of information as illustrated in Equations 2 through 7.

$$f\_t = sigmoid\big(W\_f.\big[h_{\{t-1\}}, x_t\big] + b\_f\big) \quad (2)$$

$$i\_t = sigmoid(W\_i.[h\_\{t-1\}, x\_t] + b\_i) \quad (3)$$

$$C\_t\ tilde = \tan h(W_C.[h\_\{t-1\}, x\_t]) + b\_C \quad (4)$$

$$C\_t = f\_t * C\_\{t-1\} + i\_t * C\_t\_tilde \quad (5)$$

$$o\_t = sigmoid\ (Wo\ .[h\_\{t-1\}, x\_t + b\_o]\ ) \quad (6)$$

$$h\_t = o\_t * \tan h\ \ (C\_t) \quad (7)$$

where sigma is the sigmoid activation function, the dot symbol denotes element-wise multiplication, W is the weight matrix, b is the bias vector, x_t is the input at time t, and h_t is the hidden state at time t.

After the LSTM layer, a Dropout Layer is applied to avoid overfitting, followed by a Dense Layer with a ReLU activation function, and an output layer with a linear activation function to produce score predictions in the form of continuous values.

2.6. Bi-LSTM Model Architecture

To evaluate the LSTM model, a Bi-LSTM model was also created, processing the input sequence from both directions. The hidden state in the Bi-LSTM model is a combination of the hidden states from the forward and backward directions, as shown in Equation (8).

$$h\_t = [h\_t(forward); h\_t(backward)] \quad (8)$$

with h_t (forward) as the hidden state of the LSTM in the forward direction, while h_t (backward) is the hidden state from the LSTM in the backward direction. This approach allows the model to capture information from both directions in the sequence, so that information before and after each word can be used simultaneously by the model.

DOI: https://doi.org/10.38204/tematik.v13i1.2974

2.7. Evaluation Matrix

Model performance evaluation uses the Mean Absolute Percentage Error (MAPE) metric, defined as follows (Equation 9).

$$MAPE = \left(\frac{100\%}{n}\right) \times \sum_{t=1}^{n} |A\_t - F\_t| / |A\_t| \quad (9)$$

where A_t is the actual score given by the teacher, F_t is the score predicted by the model, and n is the number of testing data points. Additionally, a lower MAPE value indicates better model performance. The interpretation of MAPE values is as follows: if the MAPE value is less than 10%, that is considered very good; if between 10% and 20%, that is considered good; if between 20% and 50%, that is considered acceptable; and if greater than 50%, that is considered poor.

3. Results and Discussions

3.1. Hyperparameter Tuning

Hyperparameter tuning was performed to produce the best parameter configuration for the model. Each hyperparameter tuned in this study is the number of LSTM hidden-layer units, learning rate, batch size, and dropout rate. Testing for each hyperparameter was done with various values, and each combination was repeated 5 times to ensure the MAPE values obtained are reliable and stable.

Testing of hidden-layer unit counts was conducted to determine the model size. The values tested included 32, 64, 128, 256, and 512 units. The testing results are shown in Table 2. Based on the results presented in Table 2, it can be seen that the most optimal number of units to use is 128, with the lowest MAPE value of 9.64%. In addition, an increase in the number of units above 128 tends to increase the MAPE value, indicating that the model is overfitted due to being too complex for the available data.

Table 2. Results of Hidden Layer Unit Count Testing

Number of Units	Average MAPE (%)	Standard Deviation
32	14.27	0.68
64	11.83	0.52
128	9.64	0.41
256	10.08	0.56
512	11.42	0.73

The learning rate is one of the more important parameters, related to the amount of weight adjustment performed at each training iteration. The learning rate values tested included 0.0001, 0.0005, 0.001, 0.005, and 0.01. The test results can be seen in Table 3. Based on the results in Table 3, the smallest MAPE (9.28%) was achieved using a learning rate of 0.001. Furthermore, a learning rate that is too low causes the model to converge very slowly, requiring a longer training time, while a learning rate that is too high causes the model to be unstable and oscillate around the optimum point.

Table 3. Learning Rate Testing Results

Learning Rate	Average MAPE (%)	Standard Deviation
0.0001	12.85	0.45
0.0005	10.16	0.38
0.001	9.28	0.32
0.005	11.73	0.62
0.01	15.94	1.24

Batch size was selected to determine how much data would be processed at each training iteration. The batch sizes tested were 16, 32, 64, and 128. The test results can be seen in Table 4. We can see from Table 4 that batch size 32 is the best with a MAPE of 9.15%. Besides, a large batch size will reduce the model's ability to explore the search space, while a small batch size may cause noisy gradient updates.

Table 4. Batch Size Testing Results.

Batch Size	Average MAPE (%)	Standard Deviation
16	9.52	0.37
32	9.15	0.29
64	9.83	0.43
128	10.67	0.58

Dropout rate was tested to determine the percentage of units randomly deactivated during model training to avoid overfitting. The dropout rate values tested included 0.1, 0.2, 0.3, 0.4, and 0.5. The test results can be seen in Table 5. Table 5 shows that the lowest MAPE value was obtained at a dropout rate of 0.3, at 9.08%. Moreover, a dropout rate above 0.3 leads to the loss of too much information during the training process, while a dropout rate below 0.3 is insufficient to prevent overfitting.

Table 5. Dropout Rate Testing Results

Dropout Rate	Average MAPE (%)	Standard Deviation
0.1	10.34	0.52
0.2	9.47	0.38
0.3	9.08	0.31
0.4	9.52	0.45
0.5	10.25	0.67

Based on all the experiments conducted, the best set of parameters for the LSTM model in this study is summarized in Table 6. These parameters were then used as the basis for the final training process of the LSTM and Bi-LSTM models.

Table 6. Best Parameter Configuration From Tuning.

Parameter	Best Value
Number of Hidden Layer Units	128
Learning Rate	0.001
Batch Size	32
Dropout Rate	0.3
Number of Epochs	100
Optimizer	Adam
Output Activation Function	Linear
Loss Function	Mean Squared Error

DOI: https://doi.org/10.38204/tematik.v13i1.2974

3.2. LSTM Results with Best Parameters

After obtaining the best parameter configuration from the tuning process, the LSTM model was trained using the entire training dataset with that parameter configuration. Moreover, LSTM model training was conducted for 100 epochs with an early stopping mechanism to prevent overfitting if the validation loss did not show improvement after 10 epochs.

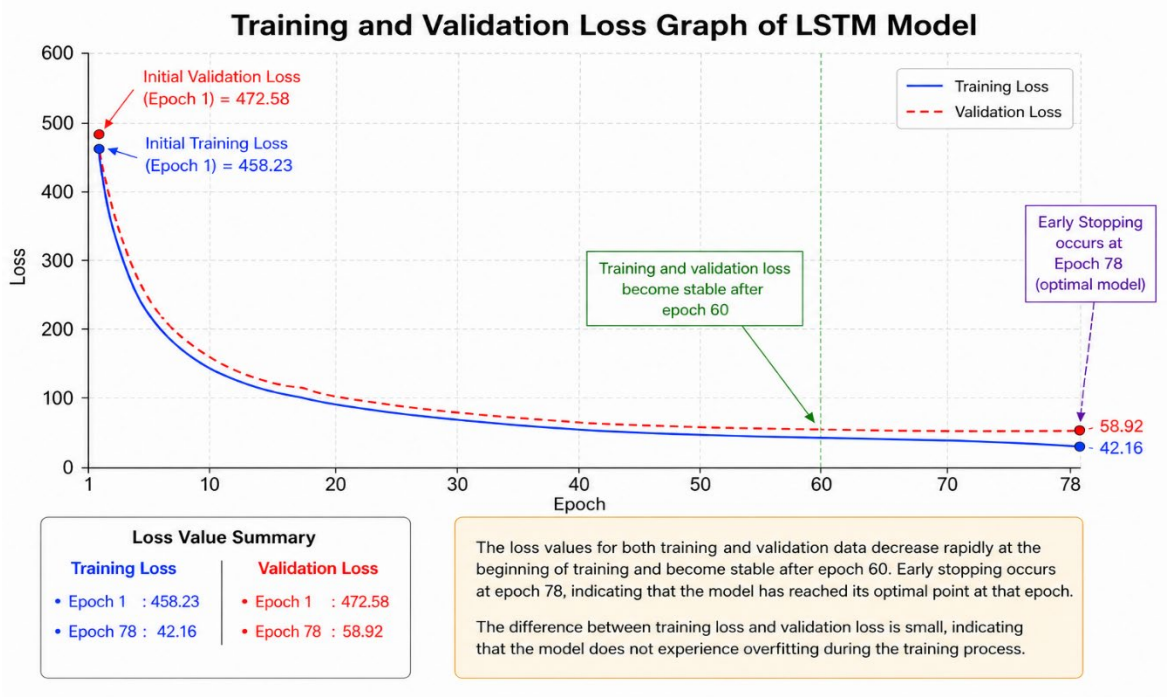


Figure 3. LSTM Model Training And Validation Loss Graph

Figure 3 shows the training loss and validation loss graph for LSTM model training. The loss values for both training and validation data decreased very rapidly in the early stages of training and stabilized after epoch 60. In addition, early stopping occurred at epoch 78, indicating that the model had reached its optimum point at that epoch.

During the training process, the training loss value decreased from 458.23 at the first epoch to 42.16 at epoch 78. The validation loss, on the other hand, decreased from 472.58 to 58.92. In addition, the training loss and validation loss values showed only a small difference, indicating that the model did not experience overfitting during the training process.

The resulting LSTM model was then used to predict 75 test data points. The predicted scores generated were then paired with the actual scores to calculate MAPE. Table 7 shows some examples of the prediction results on the test data.

Overall, the LSTM model achieved a MAPE of 9.12% on the test data. This value indicates that the average error of the model prediction relative to the actual score is 9.12% of the actual score (as given by the teacher). According to the MAPE categorization described earlier, this falls in the "very good" category and indicates that the LSTM model with FastText representation has successfully predicted essay scores with high accuracy.

Table 7. Examples Of LSTM Model Prediction Results on Test Data.

No	Actual Score	Predicted Score	Absolute Percentage Error (%)
1	85	82.37	3.09
2	72	76.14	5.75
3	90	88.59	1.57
4	65	71.23	9.58
5	78	74.86	4.03
6	82	79.45	3.11
7	60	67.38	12.30
8	95	91.27	3.93
9	70	75.12	7.31
10	88	85.46	2.89
...	...	...	...
75	80	77.84	2.70

Further analysis was conducted on the prediction errors produced by the LSTM model. Of the 75 test data points, 56 data points (74.67%) had prediction errors below 10%, 15 data points (20%) had errors in the 10-20% range, and only 4 data points (5.33%) had errors above 20%. Thus, data with errors above 20% generally consisted of very short answers or answers comprising a mix of English and Indonesian, which require special data processing.

DOI: https://doi.org/10.38204/tematik.v13i1.2974

3.3. Bi-LSTM Results

The Bi-LSTM model was trained with the same parameter configuration as the LSTM model to support fair and equal comparison conditions for model evaluation. Moreover, the main difference between the two is the model architecture that processes sequences from two directions in the Bi-LSTM, such that the number of parameters in the Bi-LSTM model is approximately twice that of the LSTM model.

The Bi-LSTM model training and validation loss graph can be seen in Figure 4. Overall, the Bi-LSTM model training graph is generally similar to the LSTM model training graph, but the final loss in the Bi-LSTM model is lower. Additionally, Bi-LSTM model training stopped at epoch 85 due to early stopping.

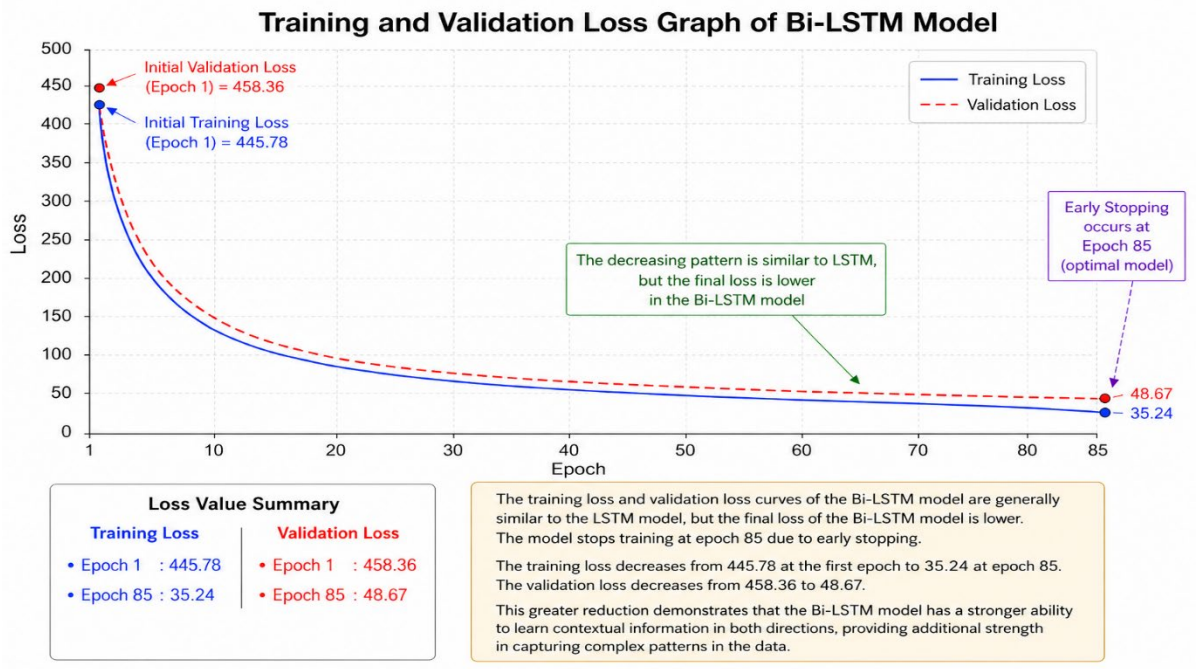


Figure 4. Bi-LSTM Model Training And Validation Loss Graph

The training loss value decreased from 445.78 at the first epoch to 35.24 at epoch 85. And the validation loss value went from 458.36 to 48.67. Moreover, this greater reduction demonstrates that the Bi-LSTM's ability to learn context from both directions provides extra power in learning complex patterns in the data.

Table 8 shows some examples of prediction results from the Bi-LSTM model on the same test data used to evaluate the LSTM model

Table 8. Examples Of Bi-LSTM Model Prediction Results On Test Data.

No	Actual Score	LSTM Predicted Score	Bi-LSTM Predicted Score	APE Bi-LSTM (%)
1	85	82.37	83.68	1.55
2	72	76.14	74.53	3.51
3	90	88.59	89.42	0.64
4	65	71.23	68.94	6.06
5	78	74.86	76.91	1.40
6	82	79.45	81.17	1.01
7	60	67.38	64.83	8.05
8	95	91.27	93.18	1.92
9	70	75.12	72.65	3.79
10	88	85.46	87.14	0.98
...	...	...	...	...
75	80	77.84	79.16	1.05

Overall, the Bi-LSTM model achieved a MAPE of 7.84% on the test data, which is 1.28 percentage points lower than the LSTM model. In addition, this result supports the hypothesis that the bidirectional approach provides improvement in model predictions.

Comparative results of both models are shown briefly in Table 9. The metrics measured are MAPE, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and training time.

Table 9. Comparison Of LSTM And Bi-LSTM Model Performance.

Metric	LSTM	Bi-LSTM	Improvement
MAPE (%)	9.12	7.84	14.04%
MAE	6.78	5.92	12.68%
RMSE	8.94	7.83	12.42%
Training Time (seconds)	1,245	2,187	-75.66%
Number of Parameters	218,753	437,506	100%

Based on Table 9, it can be seen that although the Bi-LSTM model shows better performance on all measured metrics, the training time is longer (75.66%

DOI: https://doi.org/10.38204/tematik.v13i1.2974

longer) and the number of parameters is larger (twice as large) compared to the LSTM model. Furthermore, this trade-off between accuracy and model complexity needs to be considered in the practical implementation of the model.

In terms of prediction error distribution, the Bi-LSTM model shows a fundamental improvement over the LSTM model. Of the total test data (75 data points), 63 data points (84%) had errors below 10%, 10 data points (13.33%) had errors in the 10-20% range, and 2 data points (2.67%) had errors above 20%. In addition, the decrease in the number of data points with high errors shows that the Bi-LSTM model is less susceptible to variance in the test data.

3.4. Discussion

This study has successfully proven that the combination of FastText word embeddings and the LSTM, specifically Bi-LSTM, architecture has become a successful approach for predicting essay scores in the context of English as a foreign language learning in junior high school. This part of the paper elaborates on some key findings of this research.

First, the use of special tokens <startanswer> and <endanswer> to delimit the answer in a combined question-answer input is beneficial for model performance. At the same time, these tokens provide explicit markers that help the model identify input parts and separate the question from the answer. This approach is aligned with findings from previous research stating that special tokens in deep learning models can improve contextual understanding. Furthermore, this research shows that using student answers independently (as in the traditional approach) does not provide significant additional information.

Second, FastText's strength in processing morphological variations and rare words played an important role in this research. The junior high school student answers used to train the model tend to contain misspelled words, variations in root words, or even a mixture of two languages. FastText, with vectors generated using character n-grams, can produce meaningful vectors for such words, so the model will not miss information simply because a word is not in the training vocabulary. This is consistent with previous research reporting better results from FastText compared to Word2Vec and GloVe for languages with diverse morphology.

Third, experiments evaluating the LSTM and Bi-LSTM models showed a significant performance improvement of 14.04% (in terms of MAPE value) from the use of bidirectional processing. In addition, Bi-LSTM is capable of processing information from words located both to the left and right of each word in the sequence, resulting in a richer understanding of the relationship between the question and answer. However, the trade-off between improved performance and computational costs should also be considered. The Bi-LSTM model training time, which is nearly twice as long as the LSTM model, can be a constraint in practical systems, particularly if the system is expected to retrain periodically with new data.

Fourth, this research shows that hyperparameter tuning is important for achieving optimal model performance. For this research, the optimal hyperparameters obtained were 128 units in the hidden layer, a learning rate of 0.001, a batch size of 32, and a dropout rate of 0.3. The set of parameters found to be optimal allows for a good balance between the model's capacity to learn complex patterns and its ability to generalize to new data. This is consistent with best practices in training LSTM models for various natural language processing tasks in general.

Fifth, although the results obtained in this study are promising, there are also some limitations. First, the size of the dataset is relatively small (500 data points). A larger and more diverse dataset is needed to further explore the full potential of the proposed approach. Second, this research only refers to the English subject at one school, so a more comprehensive research design encompassing other subjects or other schools should be conducted carefully. Third, this research has not yet incorporated assessments of specific linguistic elements, such as grammar, sentence structure, or writing style, which could provide more detailed feedback to students.

Sixth, comparison with previous research shows that the results produced in this study are competitive. Studies with CNN-LSTM architectures for AES on ASAP data report Quadratic Weighted Kappa (QWK) values in the range of 0.75 to 0.85, which corresponds to a MAPE of 10-15%. Moreover, the MAPE value of 7.84% obtained in this study indicates that the proposed approach is competitive with other state-of-the-art methods, particularly in a smaller data setting, and specific to English language learning at the junior high school level.

Seventh, the practical contributions of this research are quite diverse and can be applied in various educational settings. The proposed AES system based on FastText and Bi-LSTM can be integrated into online learning systems to assist teachers in the assessment process. It is important to note that these practical benefits such as faster grading time and improved scoring consistency are anticipated based on the MAPE results obtained in this study and are consistent with findings reported in the AES literature. However, these claims have not yet been empirically validated through user testing, teacher-versus-system time comparisons, or evaluation by subject-matter teachers in a real classroom setting. Future work should include a structured user study to provide direct evidence for these practical assertions Furthermore, this system can also be used as a self-

DOI: https://doi.org/10.38204/tematik.v13i1.2974

learning aid for students, so that they can receive an estimated score immediately after finishing writing their essay

4. Conclusions

In this study, we successfully implemented a system for predicting essay scores in the English subject at the junior high school level by combining FastText word embeddings and the LSTM model. Moreover, the proposed method in this study combines the question and the student's answer in a single input variable with the use of special tokens <startanswer> and <endanswer> as prefix and suffix to the answer, respectively. This research used 500 data points prepared from SMP Negeri 1 Malang, consisting of 100 students x 5 essay questions.

From the experiments conducted, several conclusions can be drawn. First, hyperparameter tuning produced the best parameter configuration using 128 hidden layer units, a learning rate of 0.001, a batch size of 32, and a dropout rate of 0.3. These parameters produced the best performance in balancing model capacity with generalization ability. Second, the LSTM model with best parameters achieved a MAPE value of 9.12% on the test data, which constitutes very good performance. Third, the Bi-LSTM model with best parameters achieved a MAPE value of 7.84%, an improvement of 14.04% compared to the LSTM model. This bidirectional consideration contributed positively to obtaining richer context from question and answer information.

The contributions of this study include the introduction of a simple yet effective method to combine information about the question and answer into one input sequence using special tokens, and the empirical demonstration that the integration of FastText and Bi-LSTM can provide a competitive solution to the AES task in the context of foreign language learning for middle school students.

There are several interesting points to consider for future research. First, increasing the amount and diversity of data, including various education levels and various subjects, will allow a more comprehensive assessment of the proposed approach. Second, research on more complex architectures, such as utilizing attention mechanisms or transformers, could result in improved system performance. Third, the design of a multi-aspect assessment feature capable of providing scores or ratings for grammar, content, coherence, and writing style will enhance the system's functionality in education. Fourth, deploying the system on a real learning platform and assessing its impact on the teaching and learning process will provide empirical research on the benefits of deep learning-based AES systems

References

[1] H. Misgna, B. W. On, I. Lee, and G. S. Choi, "A survey on deep learning-based automated essay scoring and feedback generation," *Artif. Intell. Rev.*, vol. 58, no. 2, Feb. 2025, doi:10.1007/s10462-024-11017-5

[2] M. Faseeh *et al.*, "Hybrid Approach to Automated Essay Scoring: Integrating Deep Learning Embeddings with Handcrafted Linguistic Features for Improved Accuracy," *Mathematics*, vol. 12, no. 21, Nov. 2024, doi: https://doi.org/10.3390/math12213416

[3] M. Uto, Y. Xie, and M. Ueno, "Neural Automated Essay Scoring Incorporating Handcrafted Features," Online.

[4] D. S. Asudani, N. K. Nagwani, and P. Singh, "Impact of word embedding models on text analytics in deep learning environment: a review," *Artif. Intell. Rev.*, vol. 56, no. 9, pp. 10345–10425, Sep. 2023, doi: https://doi.org/10.1007/s10462-023-10419-1.

[5] R. Ridley, L. He, X.-Y. Dai, S. Huang, and J. Chen, "Automated Cross-prompt Scoring of Essay Traits," 2021. [Online]. Available: www.aaai.org

[6] Y. Nie, "Automated essay scoring with SBERT embeddings and LSTM-Attention networks," *PeerJ Comput. Sci.*, vol. 11, 2025, doi: 10.7717/PEERJ-CS.2634.

[7] Y. Wang, Z. Wei, Y. Zhou, and X. Huang, "Automatic Essay Scoring Incorporating Rating Schema via Reinforcement Learning," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2018, pp. 791–797. doi: 10.18653/v1/D18-1090.

[8] M. Uto, "A review of deep-neural automated essay scoring models," Jul. 01, 2021, *Springer Japan*. doi: 10.1007/s41237-021-00142-y.

[9] U. B. Mahadevaswamy and P. Swathi, "Sentiment Analysis using Bidirectional LSTM Network," in *Procedia Computer Science*, Elsevier B.V., 2022, pp. 45–56. doi: 10.1016/j.procs.2022.12.400.

[10] M. Krichen and A. Mihoub, "Long Short-Term Memory Networks: A Comprehensive Survey," *AI*, vol. 6, no. 9, p. 215, Sep. 2025, doi: 10.3390/ai6090215.

[11] M. Umer *et al.*, "Impact of convolutional neural network and FastText embedding on text classification," *Multimed. Tools Appl.*, vol. 82, no. 4, pp. 5569–5585, Feb. 2023, doi: 10.1007/s11042-022-13459-x.

[12] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information." [Online].Available:http://www.isthe.com/chongo/tech/comp/fnv

[13] Q. Wang, "A multifaceted architecture to Automate Essay Scoring for assessing english article writing: Integrating semantic, thematic, and linguistic representations," *Computers and Electrical Engineering*, vol. 118, p. 109308, Aug. 2024, doi: 10.1016/j.compeleceng.2024.109308.

[14] U. Rawat and C. S. Rai, "Enhancing CNN-Attention-LSTM with Novel LSTM Gate Mechanism: Evaluation on Benchmark and Brain Tumor Imaging Datasets," Nov. 21, 2024. doi: 10.21203/rs.3.rs-5392434/v1.

[15] M. Beseiso, O. A. Alzubi, and H. Rashaideh, "A novel automated essay scoring approach for reliable higher educational assessments," *J. Comput. High. Educ.*, vol. 33, no. 3, pp. 727–746, Dec. 2021, doi: 10.1007/s12528-021-09283-1.

[16] C. M. Ormerod, A. Malhotra, and A. Jafari, "Automated essay scoring using efficient transformer-based language models," Feb. 2021, [Online]. Available: http://arxiv.org/abs/2102.13136

[17] E. Mayfield and A. W. Black, "Should You Fine-Tune BERT for Automated Essay Scoring?," 2020. [Online]. Available: https://course.fast.ai/

[18] Y. Wang, C. Wang, R. Li, and H. Lin, "On the Use of BERT for Automated Essay Scoring: Joint Learning of Multi-Scale Essay Representation." [Online]. Available: https://www.kaggle.com/c/asap-aes

DOI: https://doi.org/10.38204/tematik.v13i1.2974

[19] C. Tho, Y. Heryadi, I. H. Kartowisastro, and W. Budiharto, "Code-Mixed Sentiment Analysis Indonesian-English Using Transformer Model," *ICIC Express Letters*, vol. 17, no. 11, pp. 1295–1302, Nov. 2023, doi: 10.24507/icicel.17.11.1295.

[20] D. Ramesh and S. K. Sanampudi, "An automated essay scoring systems: a systematic literature review," *Artif. Intell. Rev.*, vol. 55, no. 3, pp. 2495–2527, Mar. 2022, doi: 10.1007/s10462-021-10068-2.

[21] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 Competition: 100,000 time series and 61 forecasting methods," *Int. J. Forecast.*, vol. 36, no. 1, pp. 54–74, Jan. 2020, doi: 10.1016/j.ijforecast.2019.04.014.

[22] A. De Myttenaere, B. Golden, B. Le Grand, and F. Rossi, "Mean Absolute Percentage Error for regression models," Jul. 2017, doi: 10.1016/j.neucom.2015.12.114.

[23] J. Luis and S. Lozano, "Enhancing Local Hydrological Services with the GEOGloWS ECMWF Global Hydrologic Model," 2023.

[24] L. Yang and A. Shami, "On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice," Oct. 2022, doi: 10.1016/j.neucom.2020.07.061.

[25] J. Bergstra, J. B. Ca, and Y. B. Ca, "Random Search for Hyper-Parameter Optimization Yoshua Bengio," 2012. [Online]. Available: http://scikit-learn.sourceforge.net.

[26] X. Ying, "An Overview of Overfitting and its Solutions," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Mar. 2019. doi: 10.1088/1742-6596/1168/2/022022.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.

[28] T. B. Brown *et al.*, "Language Models are Few-Shot Learners," Jul. 2020, [Online]. Available: http://arxiv.org/abs/2005.14165

[29] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–35, Sep. 2023, doi: 10.1145/3560815.

DOI: https://doi.org/10.38204/tematik.v13i1.2974