Proceedings of the 3rd International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/151/2025.22857

BERT+BiLSTM: Boosting Performance in Movie Review Sentiment Classification

Rongjun Gao

Shanghai Jiao Tong University, Shanghai, China
grj040803@sjtu.edu.cn

Abstract: The demand for movie reviews sentiment analysis is growing rapidly nowadays. This study focuses on the development of advanced text classifiers to address complex classification tasks and proposes three models. The first model utilizes 6 encoder layers to capture information from texts and the second analyzes data using pretrained parameters of bidirectional encoder representations from transformers (BERT) with 12 encoder layers. The third one combines contextual embeddings of BERT with bi-directional long short-term memory's (BiLSTM) sequential modeling capabilities. Specifically, it leverages BERT to extract deep contextual features, which are then fed into a BiLSTM layer followed by a classification head for prediction. Experiments are conducted on a Kaggle Internet Movie Database (IMDB) and the analysis displays a trade-off between classification accuracy and total training time as well as memory consumption. The BERT+BiLSTM model ends up with an accuracy of around 65% but the longest training time per epoch and high memory usage. These results show the model's ability to effectively capture contextual and sequential patterns in text, making it highly suitable for real-world textual sentiment analysis applications.

Keywords: BERT, BiLSTM, Sentiment Analysis, IMDB, Text Classification.

1. Introduction

Sentiment analysis is a field of Natural Language Processing (NLP) that focuses on identifying emotions within a given text. In the modern world, sentiment analysis on movie reviews is playing an increasingly significant role in recommender systems, opinion mining, movie performance evaluation and audience satisfaction analysis [1-3]. In recent years, large amounts of literature are discussing methods to do sentiment analysis on movie reviews using machines and algorithms. With the rising of deep learning, extracting key information among textual data for emotion classification becomes possible.

In fact, this field can be traced back to the beginning of the 21st century, in which collocational clues were used to identify subjectivity from texts [4]. Later, machine learning algorithms came into use, including naive bayes classifiers, maximum entropy, support vector machines for text categorization, and the pointwise mutual information and information retrieval algorithm for measuring semantic orientations of various phrases [5, 6]. In 2004, subjectivity detection was applied to discard objective sentences in movie reviews, and phrase-level analysis was made to disambiguate contextual polarity of different phrases [7, 8]. With the development of deep learning, the Recursive Neural Tensor Networks (RNTNs) were proposed to tackle compositionality of long phrases in

DOI: 10.54254/2755-2721/151/2025.22857

semantic vector spaces [9]. Meanwhile, the Continuous Bag-of-Words Model and the Continuous Skip-gram Model were constructed to provide powerful word embedding tools and decrease computational complexity [10]. In 2017, the multi-head attention mechanism and transformer architecture were designed to solve the gradient explosion problem of RNTNs and better capture dependencies between phrases among long sentences [11]. After 2019, Bidirectional Encoder Representations from Transformers (BERT) achieved multiple breakthroughs among various NLP tasks and unleashed the potential of fine-tuning pre-trained models for classification tasks [12]. Graph Neural Networks (GNNs) are also efficient in analyzing textual information due to their flexibility and interpretability [13].

The main objective of this study is to compare the performance of three different models (the Encoder model, the BERT model and the BERT+ bi-directional long short-term memory (BiLSTM) model) in terms of classification accuracy, training time, computation complexity and stability. First, this paper will introduce the details of three models, including hyperparameters setting, architecture, the loss function and the dataset used. Second, the performance of the three models will be compared. The experimental results demonstrate a trade-off between classification accuracy and training time as well as memory resources. The classification accuracy increases from the Encoder Model to the BERT+BiLSTM model, but a longer training time per epoch is inevitable with a more complicated structure. In this paper, the best training accuracy reached is around 65% using around 60 million parameters.

2. Methodology

2.1. Dataset description and preprocessing

The dataset used in this paper from Kaggle contains 32745 samples for training and 2681 samples for testing in tab-separated value format [14]. All samples in the dataset come from the Internet Movie Database (IMDB) website. Each sample consists of two parts. The first part is a comment on a movie of various lengths. The second part is the category this comment belongs to, where 0 represents negative emotions, 1 represents neutral emotions and 2 represents positive emotions.

Each comment will be transformed into a sequence of word identifiers using tokenizers. The tokenizer used in the encoder model is a word level tokenizer where a whitespace separates each word. It is trained with a set of predefined special tokens. The [UNK] represents unknown tokens, the [PAD] is for sequence padding and the [SOS] and [EOS] mark the start and end of a sentence respectively. It filters out infrequent words (minimum frequency of 2). During the training and inferencing mode, only the first 149 words in a sample will be transformed into word identifiers, and all the rest will be discarded. The tokenizer used in the BERT model and the BERT+BiLSTM model will use the official PyTorch library, which includes special tokens [CLS], [SEP], [PAD] and [MASK].

2.2. Proposed approach

The primary goal of this study is to construct a movie review classification model with the highest possible accuracy. Following the process displayed in Figure 1, the input movie reviews will first be transformed into a sequence of word identifiers with a class token prepended and then be embedded into word vectors. After being positionally encoded, the entire sequence of vectors as well as corresponding masks will be given as the input to the model. The output of the model are logits which will be provided to the SoftMax function to make the final prediction and the SoftMax function output together with the true labels will be given to the loss function for gradient descent in the training process.

DOI: 10.54254/2755-2721/151/2025.22857

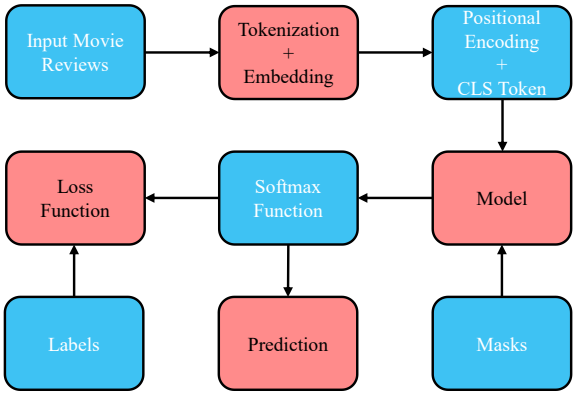

Figure 1: Process for classification and training (picture credit: original)

2.2.1. The encoder model

In the encoder model, one movie review will first be truncated and tokenized into a sequence of word identifiers with a [SOS] token prepended. Each of them is embedded into a word vector of 256 dimensions. Then these vectors will be positionally encoded according to the following formula.

$$PE(pos, 2i) = sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}}) \tag{1}$$

$$PE(pos, 2i+1) = cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}}) \tag{2}$$

where pos represents the which word vector it refers to, and $2i$ denotes the specific dimension in this word vector, $d_{model} = 256$ in this paper [11].

These vectors will be provided to an encoder. The encoder consists of 6 encoder layers, as displayed in Figure 2. In each encoder layer, $X = (x_1, x_2, \cdots, x_{len})$ denote the input matrix to the encoder where each x_i is a word vector, $len = 150$, $h = 8$ denote the number of heads, $d_k = \frac{d_{model}}{h}$ be the dimension of each head, Q, K, V denote the query, key and value. In self-attention, the encoder will first apply linear transforms on them to construct another three matrices and then all three matrices will be separated into h heads. The attention score for each head will be calculated as follows:

$$Attention(q_i, k_i, v_i) = soft\,max(\,mask(\frac{q_i k_i^T}{\sqrt{d_k}}))v_i \tag{3}$$

where mask is used to set the output of padding tokens to be $-\infty$ and these attention scores will be concatenated back to construct a new matrix of shape $len \times d_{model}$ where linear transforms are applied again. After an add and norm layer, the output will be fed into a feed-forward network and be normalized. The output of the [SOS] token will be further linearly transformed into a vector of 3 dimensions, which is the output logits of this model. The dropout rate is set to be 0.1. In this paper, the Encoder Model will be trained for 100 epochs.

DOI: 10.54254/2755-2721/151/2025.22857

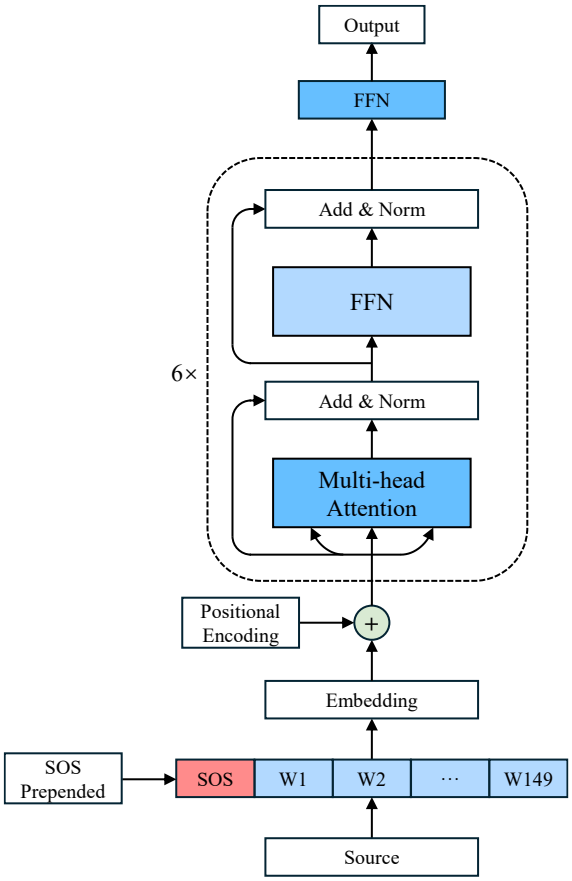

Figure 2: The encoder model (picture credit: original)

2.2.2. BERT

In BERT, the first 148 words of each movie review will be tokenized into word identifiers [12]. A [CLS] token will be prepended and a [SEP] token will be appended. If the length of the movie comment is smaller than 148, [PAD] tokens will be appended after the [SEP] token. In the word embedding layer of BERT, each word identifier will be transformed into word vectors of 768 dimensions. Then positional embeddings and segment embeddings will be applied on the word vectors. Both of two embeddings are learnable parameters for pretraining. Then a normalization and dropout layer will be applied. The entire embedding computation process is given below:

$$Embedding\ Output = Dropout(LayerNorm(WordEmb + PosEmb + SegEmb)) \quad (4)$$

where the dropout rate is set to be 0.2. The output of the embedding layer will be provided to 12 encoder layers. In the encoder layer, the dropout rate is set to be 0.3 in the attention mechanism and 0.2 in the feedforward network. Different from traditional feedforward network, the one in a BERT encoder layer is computed using the function instead of the traditional function, where the formula is given as:

$$GeLU(x) = \frac{x}{2}(1 + erf(\frac{x}{\sqrt{2}})) \quad (5)$$

For the encoder, only the output of the [CLS] token x_{CLS} will be fed into a pooling layer computed using the formula.

$$Pooled\ Output = tanh(W_{pool}x_{CLS} + b_{pool}) \quad (6)$$

DOI: 10.54254/2755-2721/151/2025.22857

and then another dropout layer with dropout rate set to be 0.2 will be applied on the result. In the end, the output of the dropout layer will be linearly transformed into the prediction logits of three dimensions. In this paper, official pretrained parameters for the BERT model will be used (see in Figure 3). The model will be trained for 20 epochs.

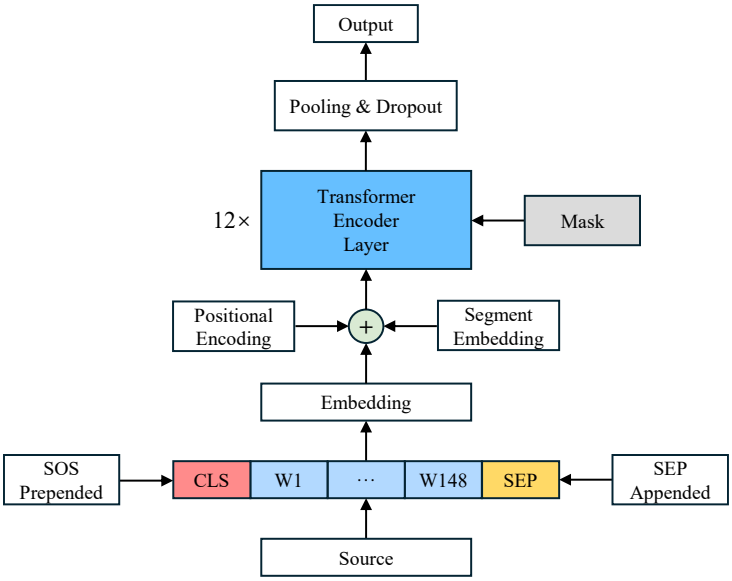

Figure 3: The BERT model (picture credit: original)

2.2.3. BERT+BiLSTM

One of the main drawbacks of the BERT model is that only information from the output of the [CLS] token is used. To solve this problem, in the BERT+BiLSTM model, the first 200 words of one movie review will be tokenized and embedded, and later be fed into the BERT model. The output of the BERT model will be provided to the BiLSTM model, which has only one hidden layer.

The input to the BiLSTM model is a sequence $X = (x_1, x_2, \cdots, x_{len})$ where length $(len) = 200$ in this model and $x_i \in R^{768}$. The BiLSTM produces two sets of hidden states. The first set is the forward hidden states $\overrightarrow{h_t}$, which is computed as the sequence progresses from $t = 1$ to $t = len$. The second set is the backward hidden states $\overleftarrow{h_t}$ computed as the sequence progresses from $t = len$ to $t = 1$. During the computation, the forget gate determines what information to discard from the previous cell state, the input gate controls what new information to add to the cell state, and the cell combines the old state with new information using the tanh function and decides what to output based on the updated cell state. With the updated cell state, the output gate decides what to output with a sigmoid function and the hidden state can be updated accordingly.

The dimension of each hidden vector is 128. The final output at each time step t is the concatenation of the two vectors $[\overrightarrow{h_t}, \overleftarrow{h_t}]$ and the last output $[\overrightarrow{h_{len}}, \overleftarrow{h_{len}}]$ will first be provided to a dropout layer with the dropout rate set to be 0.2 and then a fully connected layer to produce the output logits. The label smoothing rate is 0.1.

DOI: 10.54254/2755-2721/151/2025.22857

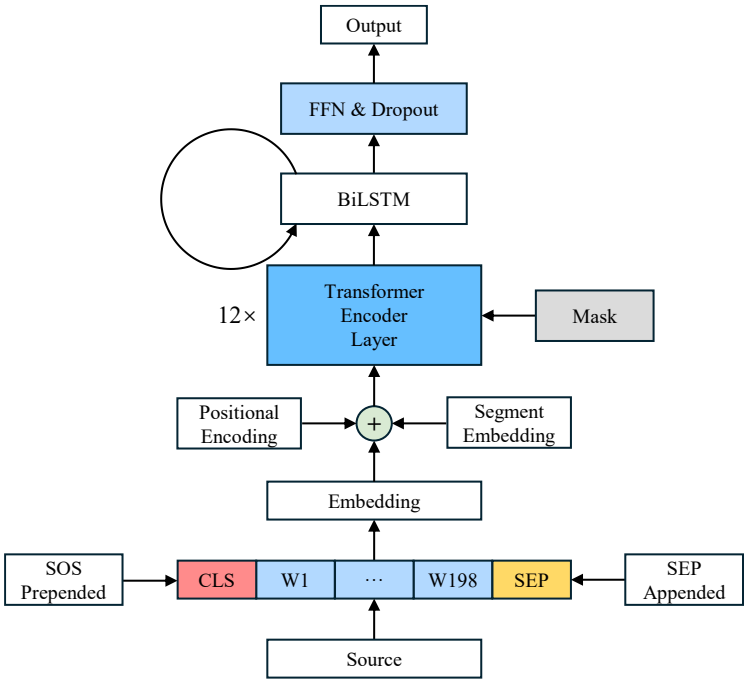

Figure 4: The BERT+BiLSTM model (picture credit: original)

In this paper, official pretrained parameters for the BERT and BiSLTM will be used (see in Figure 4).

2.3. Loss function

The output of the model is a logit vector of three dimensions since there're three classification categories, denoted $x_i = [x_{i,1}, x_{i,2}, x_{i,3}]$ for each sample i. This vector will first be given to the SoftMax function to compute the final probability for each category.

$$y_{i,j} = \frac{e^{x_{i,j}}}{e^{x_1}+e^{x_2}+e^{x_3}} \tag{7}$$

where j=1, 2, 3. Then the cross-entropy loss will be calculated based on the following formula.

$$Loss = -\frac{1}{N}\sum_{k=1}^{N}(y_k, label_k) \tag{8}$$

where $label_k$ is the ground-truth label of the training data and N is the batch size.

2.4. Implementation details

In this paper, the AdamW optimizer will be used with the learning rate set to be 10^{-4} in the Encoder model, 10^{-5} in the BERT model and 3×10^{-5} in the BERT+BiLSTM model. The RTX 4060 laptop GPU will be used for training. The batch size is 64 and models are trained for 10 epochs. A scheduler is also used, with the total steps set to be the product of the number of total training samples and the number of epochs. During the training, the test loss, the training loss, the test accuracy and the training accuracy will be recorded every epoch. The total training time will be recorded at the end of the training.

DOI: 10.54254/2755-2721/151/2025.22857

3. Results and discussion

3.1. Results analysis

3.1.1. The encoder model

The results of the Encoder Model are displayed in Figure 5. The training loss declines sharply at the beginning of the training and stays fixed around 0.62. The testing loss continues growing slowly and then becomes stable after 30 epochs, which shows the potential overfitting problem of the Encoder Model. On the other hand, this can also be verified in the testing accuracy. As displayed in the Figure 5, the training accuracy increases sharply in the initial stage of training and stabilizes after 20 epochs. On the other hand, the testing accuracy does not grow significantly even in the initial stage. It ends up with 50%, lower than expectation. The huge gap between the testing and training accuracy indicates the fact that this model lacks the ability to generalize to testing cases. The reason behind is that this model does not utilize any pretrained parameters and only 6 encoder layers are used, which leads to failure in capturing more generalizable features between the texts.

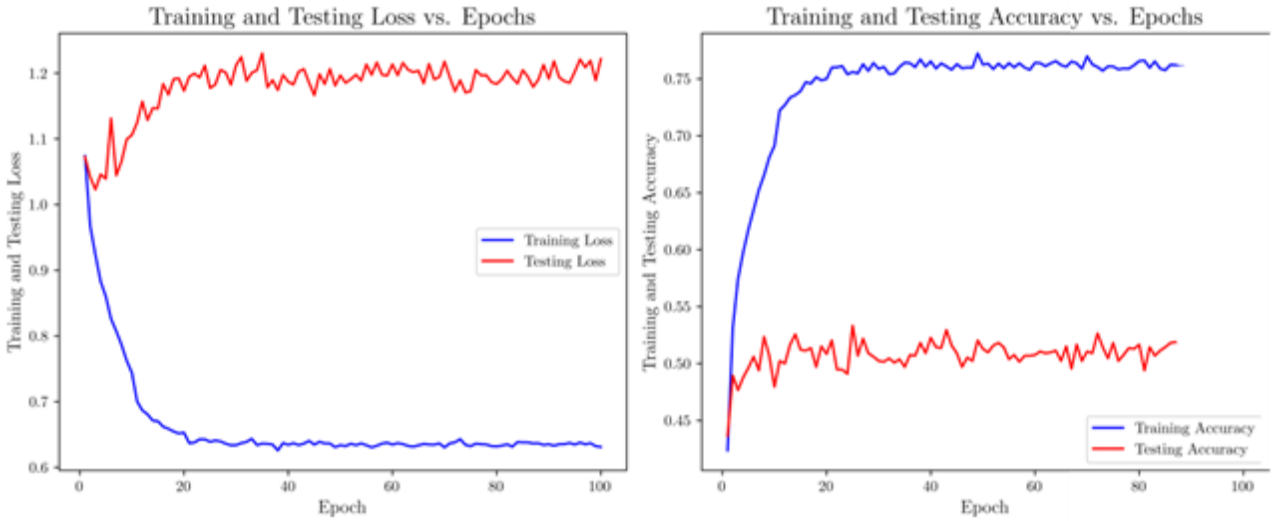

Figure 5: The training results for the encoder model (picture credit: original)

However, the total training time is only 1842.54 seconds, which is satisfying for the cases where the model needs to be trained in a short time without the demand for high identifying accuracy. The total number of parameters is 12419331, which is also acceptable.

3.1.2. BERT

As shown in Figure 6, the training loss continues to decline slowly, but the testing loss also grows slowly after 20 epochs of training. This also indicates the overfitting problem of this model. The training accuracy increases drastically in terms of the overall performance of the model, but the testing accuracy shows a tendency to decline slowly and then becomes around 63% in the end. However, this is far better than the performance of the Encoder model. The reason for this is that this model manages to utilize pretrained parameters and fine-tune the model for the data. Moreover, 12 encoder layers offers a more elaborate architecture for capturing key features between textual data.

DOI: 10.54254/2755-2721/151/2025.22857

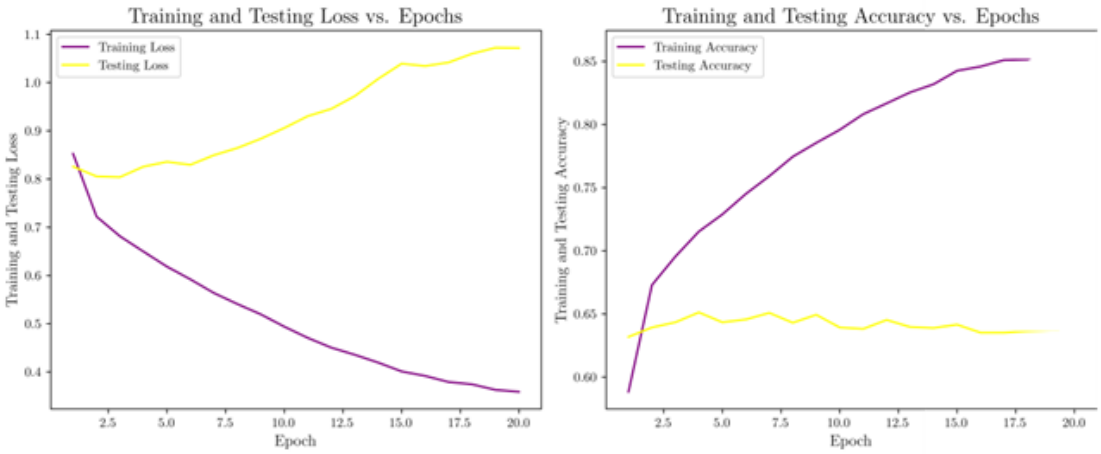

Figure 6: The training results for the BERT (picture credit: original)

Nevertheless, the total training time is 8437.04 seconds, far longer than the one of the Encoder Model. This results from the fact that more encoder layers take more time for forward and back propagation. The total number of parameters is 109484547, far more than the one of the Encoder Model.

3.1.3. BERT+BiLSTM

As displayed in Figure 7, this model performs slightly better than the BERT model, with around 65% of testing accuracy. Very similar to the BERT model, the training loss decreases gradually throughout the entire training process and reaches around 0.4 in the end. The testing loss keeps growing slowly. It is notable that the training accuracy ends up with around 95%, indicating that almost all key features in the training samples are successfully captured in the model. The testing accuracy hovers around 65%, which possibly benefits from the longer input word sequences (*len* = 200) than the length of the BERT model. Furthermore, the BiLSTM model appended after the BERT model manages to capture information not only from the output of the [CLS] token but from all outputs and integrates information more comprehensively. In addition, the label smoothing technique helps the model to generalize slightly better by making the model less certain about its classification decision. However, there still exists a huge gap between the testing accuracy and the training accuracy. The potential reason for this is that analyzing only the first 200 words in the movie review is not sufficient for the model to infer more precise conclusions.

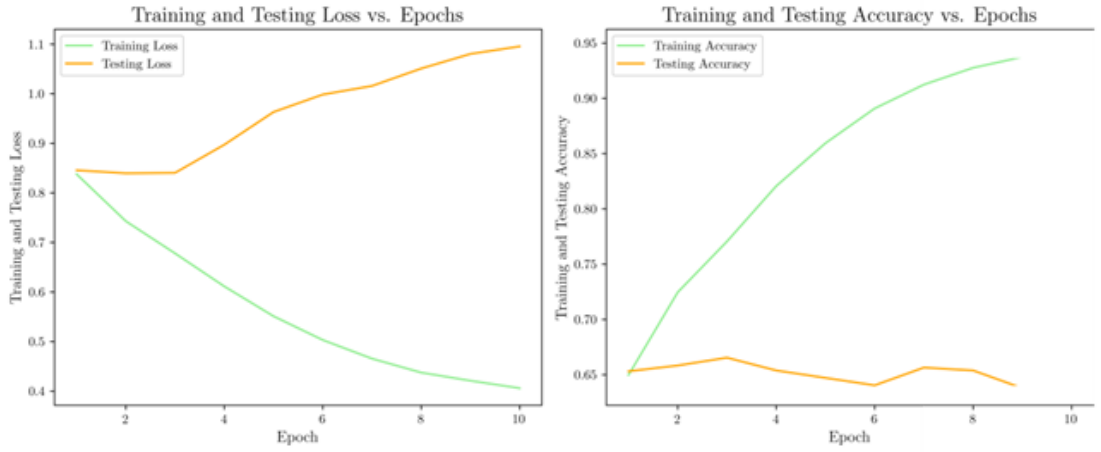

Figure 7: The training results for the BERT+BiLSTM (picture credit: original)

DOI: 10.54254/2755-2721/151/2025.22857

The total training time is 5180.92 seconds, fewer than the BERT model. However, it takes 518.09 seconds per epoch, but the BERT model only takes 421.85 seconds per epoch. This results from the extra computation time and resources needed for the forward and back propagation of BiLSTM. The total number of parameters is 67875331, which is also fewer than the BERT model.

3.2. Results analysis

So far, analysis has been made on the performance of each model in terms of the losses, accuracy, number of parameters and training time. To sum it up, the Encoder Model reaches the lowest accuracy of classification (around 50%) but takes the shortest time for training and consumes the least of memory. This results from its simple architecture since there are only 6 encoder layers in the model. The BERT model achieves better classification accuracy but spends much longer time for training (8437.04 seconds) and consumes the most of memory. In fact, fewer training epochs are needed for this model, and it's estimated from Figure 6 that only ten epochs are needed, which indicates that this model needs around 4000 seconds to finish the training process. The BERT+BiLSTM model reaches the best classification accuracy (around 65%). Nonetheless, it spends the longest training time per epoch and takes a massive amount of memory (12 encoder layers) to offer a more resilient architecture.

All the three models suffer from the problem of overfitting, resulting from the fact that the truncated length of movie reviews is too short for the model to extract key information from the training samples. Thus, a longer truncated length is preferred if a higher classification accuracy is pursued. Alternatively, a new architecture that enables the transformer encoder to read input sequences like an Recurrent Neural Network (RNN) network may be a better option in terms of memory usage efficiency and training time. In fact, there are already some SOTA architectures including the eXtreme Language Modeling (XLNet) that allows encoders to tackle an input sequence of an infinite length. In addition, transformers with sparse attention mechanisms such as Longformer or BigBird that are designed to handle extended contexts more effectively.

Another promising direction is to extend the classifier model beyond text-only inputs by incorporating multi-modal data, such as images, audio, or metadata associated with the text. For instance, combining BERT+BiLSTM with vision models including vision transformers, data-efficient image transformers or audio feature extractors can enable the system to tackle tasks more comprehensively. However, this may require designing a fusion mechanism to effectively integrate multi-modal representations and more computational resources.

Moreover, BERT is known for its large parameter size and high computational cost, which result in long training time, especially when combined with BiLSTM. For resource-constrained environments, variants of BERT can be applied, such as DistilBERT or ALBERT, to reduce memory usage and accelerate processing without significantly compromising performance.

4. Conclusion

This study introduces three text classification models based on transformer encoders, BERT and BiLSTM. In the first model, the only key architecture is 6 layers of encoders and a [SOS] token prepended. The second model utilized pretrained parameters of BERT and fine-tune the architecture. The third model is a combination of BERT and BiLSTM for better information extraction. Multiple experiments are conducted to analyze the performance of these proposed models. The results demonstrate a trade-off between classification accuracy and training time as well as computational memory. The classification accuracy increases from the Encoder Model to the BERT+BiLSTM model, but the training time per epoch increases with the model becoming more complicated. The best training accuracy achieved is around 65% at the cost of around 60 million parameters used.

DOI: 10.54254/2755-2721/151/2025.22857

In the future, there are two main possible directions. The first one is to propose a novel architecture that allows the model to incorporate an input sequence of an infinite length. In theory, the model may need to combine the advantage of transformer encoders and RNN networks, such as the XLNet. The second one is to extract information not only from textual data but also from visional or audio data. However, multi-modal data will require a more carefully designed structure to capture information from multiple sources and balance the trade-off between the training and computational time and classification accuracy. The third direction is to explore variants of BERT to further compress the training time and improve the classification accuracy.

References

[1] *Dang, C.N., Moreno-García, M.N., & Prieta, F.D.L. (2021). An approach to integrating sentiment analysis into recommender systems. Sensors, 21(16), 5666.*

[2] *Nkhata, G., Anjum, U., & Zhan, J. (2025). Sentiment analysis of movie reviews using bert. arXiv preprint 2502.18841.*

[3] *Yessenov, K., & Misailovic, S. (2009). Sentiment analysis of movie review comments. Methodology, 17(17), 1-7.*

[4] *Wiebe, J., Wilson, T., & Bell, M. (2001). Identifying collocations for recognizing opinions. In Proceedings of the ACL-01 Workshop on Collocation: Computational Extraction, Analysis, and Exploitation, 24-31.*

[5] *Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. arXiv preprint, 0205070.*

[6] *Turney, P. D. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. arXiv preprint, 0212032.*

[7] *Pang, B., & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. arXiv preprint, 0409058.*

[8] *Wilson, T., Wiebe, J., & Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In Proceedings of human language technology conference and conference on empirical methods in natural language processing, 347-354.*

[9] *Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, 1631-1642.*

[10] *Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint, 1301.3781.*

[11] *Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.*

[12] *Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, 1, 4171-4186.*

[13] *Lu, G., Li, J., & Wei, J. (2022). Aspect sentiment analysis with heterogeneous graph neural networks. Information Processing & Management, 59(4), 102953.*

[14] *Nguyen, T. (2024). IMDB Movie Reviews for Sentiment Analysis. Retrieved from https://www.kaggle.com/datasets /tuannguyen8531/imdb-movie-reviews-for-sentiment-analysis.*