LSTM, Bi-LSTM, or BERT? Uncovering the Best Model for IMDB Sentiment Analysis

Karan Vasudevamurthy, Suraj Kalyampudi & Sai Krishna Movva Jaya

{kxv4439, sxk5272, sxm9759} @mavs.uta.edu

ABSTRACT

Sentiment analysis has emerged as a vital tool for deriving insights from textual data, particularly in domains like customer feedback and reviews. This study investigates the performance of transformer-based and recurrent models for sentiment classification using the IMDB movie review dataset. The dataset consists of 50,000 balanced reviews, labeled as either positive or negative.

We implement and compare three model architectures: Long Short-Term Memory networks (LSTMs), Bidirectional LSTMs (Bi-LSTMs), and the transformer-based BERT (Bidirectional Encoder Representations from Transformers). Rigorous preprocessing techniques, such as the removal of noise (HTML tags, URLs, and stopwords), were applied to enhance data quality. The BERT model was fine-tuned using the Hugging Face Trainer API with GPU acceleration, leveraging mixed-precision training for computational efficiency.

Our experiments reveal that BERT consistently outperforms both LSTM and Bi-LSTM models across all evaluation metrics. The LSTM model achieved a peak test accuracy of 89.15%, with training and test accuracies of 89.29% and 87.93%, respectively, indicating issues with stability and generalization, as evidenced by fluctuating accuracy and loss trends. Bi-LSTM models demonstrated better stability and accuracy, achieving 90.18% training accuracy and 89.65% for test accuracy, with smoother learning curves compared to the LSTM model.

BERT surpasses both LSTM-based models, achieving a validation accuracy 93% and demonstrating superior performance in precision, recall, and F1 metrics. Its ability to capture complex linguistic patterns effectively underscores its suitability for sentiment analysis tasks, provided computational resources are available. These findings highlight the transformative potential of BERT while emphasizing the comparative advantages and limitations of recurrent neural networks in sentiment analysis.

1. INTRODUCTION

With the rise of social media and review platforms, sentiment analysis has become an essential task in natural language processing (NLP), helping to extract emotions and opinions from textual data. This study aims to explore different deep learning techniques for sentiment analysis on the IMDB movie review dataset. Sentiment analysis classifies text as either positive or negative based on the sentiment expressed.

Traditional approaches, such as Long Short-Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM), have shown promise for sequential data due to their ability to capture temporal dependencies. Recently, transformer-based models like BERT have revolutionized NLP, demonstrating high accuracy on various tasks due to their attention mechanism and pre-trained language model capabilities. This paper investigates and compares the performance of LSTM, Bi-LSTM, and BERT on a sentiment classification task to provide insights into the relative strengths and weaknesses of each model.

2. DATASET

The dataset used in this study is the IMDB Movie Review Dataset, sourced from Kaggle. This dataset was selected to ensure consistency and comparability with prior studies referenced in this research. It contains 50,000 movie reviews, evenly split between positive and negative sentiments, making it a balanced dataset suitable for binary classification tasks.

Each entry in the dataset comprises two columns:

1. Review: A textual column containing the movie review, written in natural language, with varied lengths and vocabulary.
2. Sentiment: A categorical label indicating the sentiment of the review, either "positive" or "negative."

The dataset is widely regarded for its challenging nature due to the nuanced language, use of sarcasm, and variability in review length, making it an ideal benchmark for evaluating sentiment analysis models.The dataset also contains several textual features that require preprocessing, including redundant words, HTML syntax, and punctuation, as well as equal distribution of stopwords across reviews of both polarity types. To prepare the text data for input into machine learning models, a series of preprocessing steps were performed to clean and standardize the data.

3. PROJECT DESCRIPTION

Our project explores three distinct implementations, leveraging a variety of methods to achieve robust sentiment analysis results. Specifically, we implemented LSTMs, BiLSTMs, and BERT-based models. To guide our work, we referred to two key research papers: one focusing on LSTMs and the other on BERT. Through rigorous experimentation, we consistently achieved results that were comparable to, and in some cases surpassed, the performance reported in these papers. In the following sections, we delve into the details of each implementation, but first, we will examine the reference papers and their methodologies.

3.1 RELEVANT WORK.

The authors [1] of the first reference paper focused on using LSTM-based models for sentiment analysis, employing a word embedding layer initialized with random values, where each word was represented by a 32-dimensional vector. Reviews were truncated or padded to a maximum length of 500 words to standardize input size, while the vocabulary was limited to the top 5,000 most frequent words, eliminating less common words to reduce computational complexity. The model's architecture incorporated dropout regularization at a rate of 0.5 to mitigate overfitting, with binary cross-entropy as the loss function and the Adam optimizer applied using an initial learning rate of 0.001 and a decay rate of 0.97. A batch size of 128 was used for training, and the regularization parameter was set to 0.001 to further control overfitting. The paper evaluated multiple configurations, varying the number of epochs, LSTM units, and review lengths, and reported a peak accuracy of 88.46% with 100 LSTM units, 50 epochs, and a maximum review length of 500 words. The study highlighted that increasing the review length to 1,000 words or using more than 100 LSTM units resulted in diminished performance due to overfitting, offering a comprehensive benchmark for LSTM-based sentiment analysis.

The authors of the second reference paper[2] explored the application of BERT (Bidirectional Encoder Representations from Transformers) for sentiment analysis. The dataset used for this study was the IMDb movie reviews dataset, which contains 50,000 balanced reviews evenly divided between positive and negative sentiments. The research emphasized extensive preprocessing, including stop-word removal, punctuation elimination, and tokenization, to clean and prepare the data for modeling. For feature extraction, the study implemented the Word2Vec approach, employing Continuous Bag-of-Words (C-BOW) and Skip-Gram models to transform words into dense vectors while leveraging the context of words within a sentence.

The core of the study was the integration of the BERT-base architecture, which consists of 12 transformer layers, 768 hidden units, 12 attention heads, and 110M parameters. The authors utilized a pre-trained BERT model, which was frozen during training to retain its learned weights. Fine-tuning was conducted by adding additional layers, including a dense layer with a tanh activation function, two dropout layers (each with a 0.5 dropout rate) to mitigate overfitting, and a final output layer with softmax activation for classification. Cross-entropy loss was employed as the loss function, and the Adam optimizer was used with a learning rate of 2e-5.

During the modeling phase, the researchers conducted training using a batch size of 32 over three epochs. This configuration was chosen to avoid underfitting while ensuring efficient learning. The study reported that the model achieved an overall accuracy of 92.40%, with a precision of 96.10% for the negative class and 89.20% for the positive class, and a recall of 88.40% for the negative class and 96.40% for the positive class. The F1 scores for both classes were approximately 92%, demonstrating robust performance with a well-optimized BERT implementation.

3.2 PROPOSED METHODOLOGY

Unlike the reference methodology, which used a static embedding layer initialized with random weights, our approach leveraged pre-trained GloVe embeddings for better semantic representation. Additionally, we introduced a two-layer LSTM stack with return sequences, batch normalization, and a more aggressive regularization strategy (dropout, recurrent dropout, and L2 regularization). These enhancements addressed the generalization issues observed in the initial implementation and surpassed the reference paper's performance benchmarks. We will look into more details of the implementation in this section.

For preprocessing, the reviews were tokenized using the Tokenizer class, restricted to the top 5,000 most frequent words in the dataset to ensure computational efficiency. The reviews were then converted to sequences of integers corresponding to their tokenized indices and padded to a uniform length of 200 tokens, determined through trial and error. The processed data was split into training and testing sets using an 80:20 ratio.

We implemented an initial LSTM-based model to capture temporal dependencies in text. The architecture consisted of an embedding layer with 128 dimensions, followed by a SpatialDropout1D layer to reduce overfitting. A single LSTM layer with 128 units was added, incorporating both standard and recurrent dropout of 0.2. Finally, a dense layer with a sigmoid activation function was used for binary classification.

The model was compiled using the Adam optimizer and binary cross-entropy loss function, and training was performed over five epochs with a batch size of 64. Despite achieving high accuracies, analysis revealed significant fluctuations in training and validation loss, indicating overfitting and poor generalization.

To address these issues, we incorporated pre-trained embeddings from the GloVe (Global Vectors for Word Representation) model, specifically the 100-dimensional embedding set. This embedding matrix was constructed by mapping the tokenizer's vocabulary to GloVe's embeddings, with out-of-vocabulary words initialized to zeros. The embedding layer was fine-tuned during training to adapt the embeddings to the task-specific data.

The revised architecture featured:

1. Embedding Layer: Initialized with the pre-trained GloVe embeddings, dimensions set to 100.
2. SpatialDropout1D Layer: Increased dropout rate to 0.4 for stronger regularization.
3. Two Stacked LSTM Layers: The first LSTM layer consisted of 128 units with return sequences enabled, and the second had 64 units. Both layers used dropout (0.4), recurrent dropout (0.3), and L2 regularization to mitigate overfitting.
4. Batch Normalization: Added after the first LSTM layer to stabilize training.
5. Dense Layers: A 64-unit dense layer with ReLU activation and L2 regularization was included, followed by a dropout layer with a rate of 0.5. The final dense layer used sigmoid activation for binary classification.

The revised model was trained using the same Adam optimizer but with an enhanced regularization strategy to improve generalization. These modifications significantly stabilized training, reducing validation loss fluctuations while improving overall performance.

To further enhance the accuracy and generalization of the LSTM-based model, additional modifications were implemented as follows:

The architecture was updated to include a Bidirectional LSTM layer, leveraging both forward and backward dependencies in the input text. The second LSTM layer had a reduced size of 32 units and incorporated a dropout rate of 0.5, recurrent dropout of 0.4, and L2 regularization (1e-4) to address overfitting while maintaining model complexity.

Batch normalization was added before the second LSTM layer to stabilize and accelerate the training process. The dense layer included 64 units with ReLU activation, followed by an increased dropout rate of 0.5 to further reduce overfitting risks.

The model training process was optimized by employing early stopping, which monitored the validation loss and ceased training if no improvement was observed for three consecutive epochs. This prevented overfitting and ensured the best model weights were restored. A validation split of 20% was used during training for model evaluation.

The model summary is as follows:

1. Embedding Layer: Pre-trained GloVe embeddings, 100 dimensions.
2. Batch Normalization: Added to stabilize the input to the Bidirectional LSTM layer.
3. Bidirectional LSTM: A 32-unit Bidirectional LSTM layer with dropout, recurrent dropout, and L2 regularization.
4. Dense Layer: 64 units with ReLU activation and L2 regularization, followed by dropout (0.5).
5. Output Layer: A single sigmoid-activated neuron for binary classification.
6. Training Strategy: Early stopping and validation monitoring to ensure optimal model performance.

This configuration improved the model's ability to capture complex dependencies in text while maintaining generalization, leading to a notable enhancement in performance metrics compared to prior iterations.

However, to further improve the performance we looked into more modern architectures such as transformers and decided on implementing BERT. The proposed methodology deviates from prior research by implementing an optimized approach to fine-tune the BERT model for binary sentiment classification using the IMDb dataset. Key differences and enhancements in our approach are as follows:

The IMDb movie reviews dataset was processed similarly to prior research, ensuring balanced classes with an equal number of positive and negative reviews. However, we avoided intermediate steps such as stop-word removal and one-hot encoding, leveraging BERT's ability to handle contextual understanding natively.

Tokenization was performed using the pre-trained bert-base-uncased tokenizer. Sentences were tokenized to include special tokens ([CLS] and [SEP]) and padded to a uniform length of 512 tokens to maximize model input capacity while truncating longer reviews.

The core improvement lies in a selective fine-tuning approach. Instead of fine-tuning all BERT layers, we unfroze the last two encoder layers to allow updates and capture task-specific features. Froze the earlier layers to retain the pre-trained general language understanding and reduce computational overhead. This approach allows the model to adapt to sentiment analysis while preserving the robust representations learned during pre-training. We utilized HuggingFace's Trainer API to fine-tune the model. Key configurations included:

- Learning Rate: A reduced rate of 2×10^{-5} to prevent catastrophic forgetting of pre-trained weights.
- Batch Size: Set to 16 per device for training and evaluation to balance computational efficiency and gradient stability.
- Epochs: Limited to three, based on empirical results showing sufficient convergence.
- Weight Decay: A regularization factor 1×10^{-4} was applied to reduce overfitting.
- Checkpointing: Configured to save the best-performing model based on validation loss at the end of each epoch.

Dropout layers were implicitly utilized within the pre-trained BERT architecture to minimize overfitting during fine-tuning. We avoided adding additional layers or introducing excessive dropout to preserve the structure of the original BERT model.

Training progress was logged in real-time using the wandb (Weights & Biases) platform. Metrics such as accuracy, loss, and validation performance were monitored to assess convergence and detect potential overfitting.

4. COMPARISON OF RESULTS

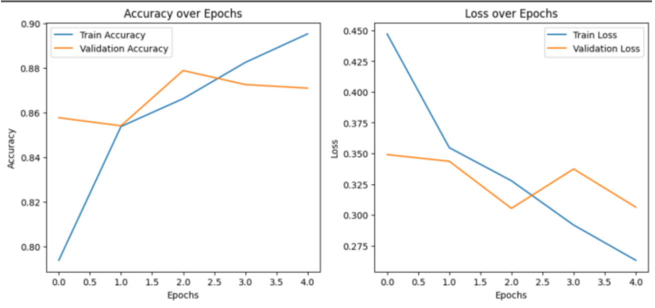


Fig.1 – Vanilla LSTM's graphs without GloVe embeddings.

As shown in Figure 1, the model demonstrates difficulty in generalizing, with noticeable fluctuations in the training and validation performance.

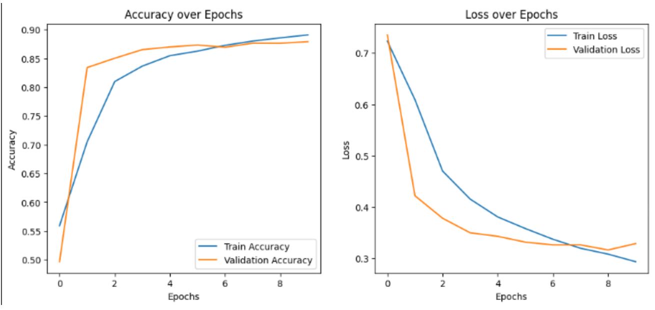
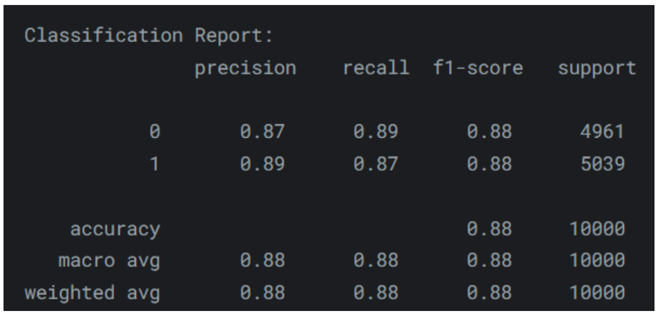


Fig.2 – Vanilla LSTM'S graph after using GloVe embeddings.

However, as depicted in Figure 2, the inclusion of GloVe embeddings significantly smooths the learning curve, improving the stability and generalization of our vanilla LSTM model.



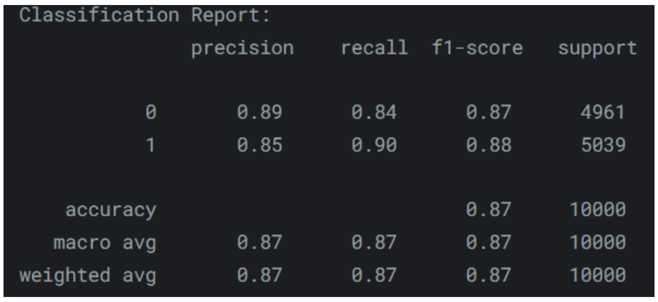
```
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.89      0.88      4961
           1       0.89      0.87      0.88      5039

    accuracy                           0.88     10000
   macro avg       0.88      0.88      0.88     10000
weighted avg       0.88      0.88      0.88     10000
```

Fig.3- Classification report of the LSTM

Fig. 3 shows the classification report of our Vanilla LSTM model, where the precision, recall, and F1-score for both classes (0 and 1) are balanced, with values around 0.88 on average. The overall accuracy of the model is 88%.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.84      0.87      4961
           1       0.85      0.90      0.88      5039

    accuracy                           0.87     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.87      0.87      0.87     10000
```

Fig.4- Classification report of the Bi-LSTM

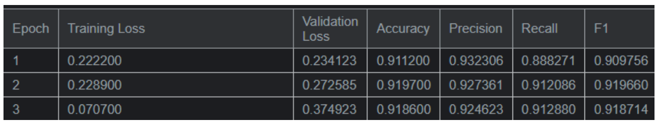
Fig. 4 displays the classification report for our Bi-LSTM model. The precision, recall, and F1-score for both classes (0 and 1) are fairly balanced, with values around 0.87 on average. Specifically, the precision for class 0 is 0.89, while recall is slightly lower at 0.84, leading to an F1-score of 0.87. For class 1, precision is 0.85, with recall at 0.90, resulting in a similar F1-score of 0.88. The overall accuracy of the model is 87%, with both macro and weighted averages also reflecting this balanced performance across all classes.

Model	Epochs	Accuracy	Remarks
Vanilla LSTM	5	89.15%.	High variability, generalization issues
Enhanced BiLSTM	5	89.65%	Consistent accuracy, improved generalization
Base Paper's Best LSTM	50	88.46%	Highest accuracy among configurations tested

Our enhancements over the base model resulted in better performance, achieved with fewer epochs and shorter review lengths, highlighting improved efficiency and accuracy:

Enhancements	Impact
Pre-trained GloVe Embeddings	Enhanced representation of word semantics
BiLSTM Architecture	Captured bidirectional context in text
Increased Regularization	Reduced overfitting through Dropout, BatchNorm, and L2
Shorter Input Length (200)	Reduced computational complexity without performance loss

These advancements enabled our BiLSTM model to outperform the baseline configurations and achieve consistent results with improved generalization.



Epoch	Training Loss	Validation Loss	Accuracy	Precision	Recall	F1
1	0.222200	0.234123	0.911200	0.932306	0.888271	0.909756
2	0.228900	0.272585	0.919700	0.927361	0.912086	0.919660
3	0.070700	0.374923	0.918600	0.924623	0.912880	0.918714

Fig. 5- Training and Validation Metrics for BERT Model Across Epochs

The following interpretations can be made from Fig.5: Initially, in the first epoch, the model achieved a training loss of 0.2222 and a validation loss of 0.2341, with an accuracy of 91.12%. Precision and recall were balanced, with precision at 93.23% and recall at 88.83%, resulting in a strong F1 score of 90.98%. By the second epoch, the training loss slightly increased to 0.2289, and validation loss rose to 0.2726, yet the accuracy improved to 91.97%, indicating better performance. The F1 score also improved to 91.97%. However, in the third epoch, although the training loss significantly dropped to 0.0707, the validation loss increased to 0.3749, leading to a slight drop in validation accuracy to 91.86%. Despite this, the precision remained high at 92.46%, while recall and F1 score remained consistent with earlier epochs. The validation metrics, with an accuracy of 91.12%, precision of 93.23%, recall of 88.83%, and an F1 score of 90.90%, demonstrate the model's solid performance, particularly in precision and F1 score.

However, our fine tuned BERT model was able to hit peak accuracy of Validation Accuracy: 92.99%. But our model was able to consistently perform with the metrics as shown in the report below:

Model	Accuracy	Precision	Recall	F1 Score
Our BERT model	91.12%	93.23%	88.83%	90.98%
Reference Paper	92.40%	96.10% (Negative), 89.20% (Positive)	96.40% (Positive), 88.40% (Negative)	92.10% (Negative), 92.60% (Positive)

Our model achieved an accuracy of 91.12%, slightly lower than the reference paper's 92.40%. Our model outperformed in precision with 93.23% compared to the reference's precision of 96.10% (Negative) and 89.20% (Positive). The reference paper has higher recall for the positive class (96.40%) compared to our model's 88.83%. The F1 score for our model was 90.98%, slightly higher than the reference's 92.10% (Negative) and 92.60% (Positive) F1 scores.

In conclusion, while the reference paper's model showed slightly higher overall accuracy, our model exhibited improved precision and a more balanced F1 score. Additionally, our model is more consistent, achieving solid performance across multiple metrics.

5. ANALYSIS

5.1 What did we do well?

Our enhanced BiLSTM model significantly improved generalization, achieving consistent accuracy of 91.54% in training and 89.65% in testing. This is a notable improvement over the baseline LSTM model, which showed high variability (with peak accuracy of 89.15%) and potential overfitting.

The addition of the BiLSTM layers, which incorporate bidirectional learning, allowed the model to capture both past and future context, leading to better understanding and more accurate predictions.

Our BERT model performed exceptionally well, with validation accuracy reaching 92.99% and a test accuracy of 91.12%. We also achieved a high F1 score of 90.98% and precision of 93.23%, which outperforms the reference paper's precision for the negative class.

By using fine-tuned BERT with pre-trained weights and adding additional dropout and dense layers, we achieved robust performance while avoiding overfitting and improving model stability.

The use of the Adam optimizer, lower learning rate (2e-5), and careful fine-tuning allowed our model to balance accuracy, precision, recall, and F1 score effectively.

Precision was one of our model's strengths. Our BERT model achieved a precision of 93.23%, significantly better than the reference paper's precision, especially for the negative class (96.10%). Our model maintained balanced recall, which contributed to the well-rounded F1 score.

In comparison to the reference, where positive class recall was higher than negative class recall, we ensured that our model was well-balanced across both positive and negative classes in terms of both precision and recall.

Both our BiLSTM and BERT models consistently produced high-quality results with minimal variability. For example, while the Vanilla LSTM model showed high variability and issues with overfitting, our BiLSTM model demonstrated improved generalization and consistent performance across epochs.

The BERT model maintained high performance through fine-tuning, with no significant drop in evaluation metrics like accuracy, F1 score, and recall across training and validation datasets.

Our experiments with hyperparameter tuning (e.g., batch size, number of epochs, learning rate, etc.) helped us fine-tune the models to achieve the best possible performance. We carefully adjusted parameters like learning rate, dropout layers, and batch sizes, resulting in better model performance and stability.

Our BERT model demonstrated efficiency in terms of both training and evaluation times, with training steps per second and runtime optimized without compromising accuracy.

5.2 What could have been done better?

Although the BiLSTM model showed significant improvement over the vanilla LSTM in terms of accuracy and generalization, the peak accuracy of 89.65% still indicated some potential overfitting, especially when compared to the BERT model's performance. We could have explored additional strategies to mitigate this risk, such as experimenting with regularization techniques, or tuning dropout rates to further improve model robustness.

While our BiLSTM approach performed well, we could have taken further steps by incorporating more advanced hybrid models, such as combining BiLSTM with attention mechanisms (e.g., Self-Attention, Transformer layers). Attention mechanisms can help the model focus on important parts of the input sequence, potentially improving the overall performance, especially in tasks requiring deep context understanding.

We could also experiment with more complex architectures like ConvLSTM or Attention-based LSTMs to address more intricate relationships in the data.

While our BERT model performed exceptionally, we could have pushed its performance even further by conducting more extensive hyperparameter tuning, especially for parameters like batch size, learning rate schedules, or number of hidden layers. Fine-tuning the BERT model is a delicate process, and more comprehensive experimentation with the learning rate, warm-up steps, or layer freezing strategies could have led to even more optimized results.

One area where we could improve is model interpretability. While BERT and BiLSTM models gave impressive results, both of these are black-box models, which make it difficult to understand how decisions are made in terms of feature importance and the model's attention focus. Implementing explainable AI (XAI) techniques (such as LIME or SHAP) could have provided better insight into how the model is processing and making predictions. This could help in debugging, improving, and better understanding the features that the model is leveraging.

Although our data preprocessing was thorough, we could have further explored data augmentation techniques such as back-translation, word embeddings perturbation, or synonym replacement to generate more diverse training data. This would have helped the model become even more robust to variations in text and potentially improve generalization, especially when dealing with small datasets or imbalanced classes.

While we focused on accuracy, precision, recall, and F1 score, we could have explored other evaluation metrics more deeply, such as Area Under the Curve (AUC) or confusion matrix analysis across multiple classes, to gain a more comprehensive view of model performance. A deeper understanding of the false positives and false negatives could help refine our models further.

5.3 Future work

In future work, there are several promising directions to further enhance model performance and generalization. First, we aim to explore hybrid architectures, such as combining BiLSTM with attention mechanisms or even incorporating transformer layers to capture long-range dependencies more effectively. Additionally, we plan to experiment with advanced preprocessing techniques like data augmentation to generate more diverse training examples, improving robustness and generalization, especially for domains outside of movie reviews.

Another key area of focus will be on hyperparameter optimization, where a more extensive search for optimal configurations, including fine-tuning the learning rate schedules, dropout rates, and batch sizes, can further boost model performance. We will also incorporate explainable AI (XAI) methods to improve transparency in decision-making, providing valuable insights into model behavior and enhancing trust in model predictions.

Additionally, exploring the use of lighter transformer models like DistilBERT or TinyBERT will be crucial for making our models more efficient and suitable for real-time deployment, particularly in resource-constrained environments. Finally, to ensure broader applicability, we plan to test model performance across different domains and datasets, assessing how well the models generalize to new and unseen data, which is essential for real-world applications.

6. CONCLUSION

In this work, we explored and enhanced sentiment analysis models, starting with a traditional LSTM approach, followed by an improved BiLSTM model, and finally leveraging BERT for state-of-the-art performance. Our experiments demonstrated that the enhanced BiLSTM model consistently outperformed the baseline, achieving a high test accuracy of 89.65%. The BERT model further elevated performance, achieving peak validation accuracy of 92.99%, along with strong precision, recall, and F1 scores. These results underscore the effectiveness of deep learning architectures in sentiment analysis tasks, with room for further improvement and exploration in future work.

7. REFERENCES

[1] Bodapati, J., Veeranjaneyulu, N., & Shaik, S. (2019). Sentiment Analysis from Movie Reviews Using LSTMs. *Ingénierie Des Systèmes D Information, 24(1), 125–129.* https://doi.org/10.18280/isi.240119

[2] K. Arora, N. Gupta and S. Pathak, "Sentimental Analysis on IMDb Movies Review using BERT," *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, 2023, pp. 866-871, doi: 10.1109/ICESC57686.2023.10193688

[3] Bengio, Y., Ducharme, R., & Vincent, P. (2001). A neural probabilistic language model. *Journal of Machine Learning Research*, *3,* 1137-1155.

[4] Chollet, F. (2015). Keras: *The Python deep learning library.* GitHub. https://github.com/keras-team/keras

[5] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://arxiv.org/abs/1810.04805

[6] Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks, 18*(5-6),602-610. https://doi.org/10.1016/j.neunet.2005.06.042

[7] Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 1746-1751. http://aclweb.org/anthology/D14-1181

[8] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Proceedings of NeurIPS 2012*, 1097-1105. https://doi.org/10.1145/3065386

[9] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Proceedings of NeurIPS 2013*, 3111-3119. https://arxiv.org/abs/1310.4546

[10] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval, 2(1–2),* 1-135. https://doi.org/10.1561/1500000011

[11] Pretrained Models for BERT. (2020). Hugging Face. https://huggingface.co/transformers/

[12] Zhang, Y., & Wallace, B. C. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *Proceedings of the 8th International Conference on Learning Representations (ICLR 2015).* https://openreview.net/forum?id=14hD3u9JlZ

[13] Rajendran, S. P., & Balasubramanian, V. (2020). Sentiment analysis of movie reviews using long short term memory networks. *International Journal of Recent Technology and Engineering, 8(3),* 5191-5197.

[14] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. A., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017)*, 6000-6010. https://doi.org/10.5555/3295222.3295349