Detection of Sarcastic News Headlines

Pulkit Singal (2023AIB2064), Ajay Tomar (2023AIB2071), Shashwat Bhardwaj (2023AIY7528)
Yardi School of Artificial Intelligence, Indian Institute of Technology Delhi
Hauz Khas, New Delhi-110016

Abstract

This report investigates the efficacy of fine-tuning BERT and RoBERTa for detecting sarcastic news headlines, juxtaposed with a baseline BiLSTM model. Employing a curated dataset tailored for sarcasm detection, we conduct comprehensive training and evaluation to gauge the models' capabilities in capturing the intricate linguistic features inherent to sarcastic discourse. By comparing the performance of transformer-based architectures with traditional recurrent neural networks (BiLSTM), we aim to elucidate their potential for advancing automated sarcasm detection systems, offering valuable insights for the development of more accurate and robust natural language understanding models.

1 Introduction

Sarcasm detection in news poses a formidable challenge owing to the intricate and often subtle nature of sarcastic language. However, recent advancements in natural language processing (NLP), notably the emergence of pre-trained transformer models such as BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa (Robustly optimized BERT approach), have sparked considerable interest in leveraging these powerful architectures for sarcasm detection tasks. This report delves into the exploration and fine-tuning of BERT and RoBERTa architectures, tailored specifically for the detection of sarcasm in news headlines. Moreover, we conduct a comprehensive benchmarking exercise, juxtaposing the performance of these cutting-edge models against a traditional baseline approach employing a Bidirectional Long Short-Term Memory (BiLSTM) network. Through this comparative analysis, our objective is to assess the effectiveness of transformer-based models in capturing the nuanced linguistic cues indicative of sarcasm, thereby shedding light on their potential to augment sarcasm detection in news headlines.

Our work aims to contribute to the advancement of natural language understanding by addressing this crucial aspect of automated sarcasm detection. By evaluating the performance of state-of-the-art transformer models alongside a baseline BiLSTM model, we endeavor to provide insights into their respective strengths and weaknesses in discerning sarcasm in news headlines. Ultimately, our endeavor is to foster the development of more robust and context-aware NLP systems capable of deciphering the intricate nuances of human communication, thereby facilitating more accurate and reliable interpretation of textual data in real-world applications.

2 Bidirectional LSTM (Baseline)

Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) is a type of recurrent neural network (RNN) architecture designed to overcome the vanishing gradient problem, which is common in traditional RNNs. A Bidirectional LSTM (BiLSTM) consistes of LSTMs in both directions. Unlike traditional LSTMs, which process input sequences in one direction (either forward or backward), bidirectional LSTMs process the input sequence in both directions simultaneously.

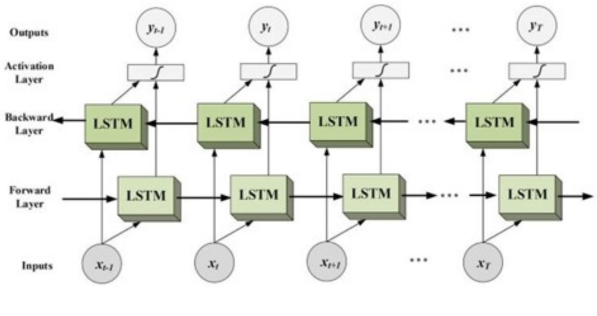


Figure 1: Bidirectional LSTM

Bidirectional LSTMs are particularly useful when the context of each element in the sequence depends on both past and future elements. For example, in natural language processing tasks such as named entity recognition or part-of-speech tagging, understanding the context of a word often requires considering both preceding and following words.

By processing the sequence in both directions, bidirectional LSTMs can capture dependencies and patterns that may not be apparent from considering the sequence in just one direction. This makes them well-suited for tasks where context plays a crucial role in understanding the input data.

2.1 Architecture

BiLSTM architecture consists of:

1. Memory Cells: BiLSTMs contain memory cells, which are responsible for retaining information over time. These cells have self-connections, allowing them to maintain their state over long sequences.

2. Gates: BiLSTMs have three types of gates that control the flow of information into and out of the memory cells:

    - Forget gate: Controls what is kept versus forgotten, from the previous cell state.
    - Input gate: Controls what parts of the new cell content are written to the cell.
    - Output gate: Controls what parts of the cell are output to the hidden state.

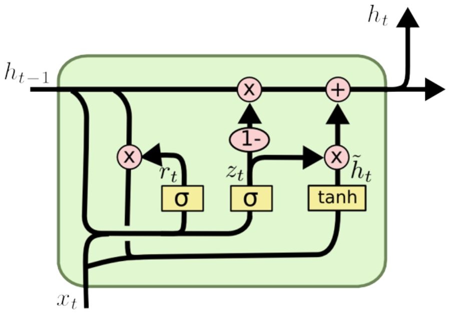


Figure 2: BiLSTM Memory Cell with Gates

3. Hidden State: BiLSTMs have a hidden state that is passed from one time step to the next. It carries information from previous time steps and influences the current prediction.

During training, the BiLSTM learns the parameters of its gates and memory cells using backpropagation through time (BPTT). The gradient flow is preserved through the use of the gates, allowing the network to learn long-range dependencies without suffering from the vanishing gradient problem.

Speaking end to end, there is an embedding layer converting input tokens into dense vectors, followed by a BiLSTM layer capturing both past and future context in the sequence. The BiLSTM layer is stacked with multiple layers to enhance its capacity in learning long-term dependencies. The bidirectional nature allows the model to consider context from both directions, enriching its understanding of the input sequence. The concatenated output of the BiLSTM is then passed through a linear layer for dimensionality reduction, followed by a sigmoid activation function to produce a probability distribution over the output classes.

The setup employs a 'basic_english' tokenizer, and embedds tokens into a 100-dimensional vector space, with a vocabulary size of 27,235 unique tokens. These embeddings are then processed by a bidirectional LSTM network consisting of two layers, each with a hidden state dimension of 256. Dropout regularization with a probability of 0.5 is applied within the LSTM layers to mitigate overfitting.

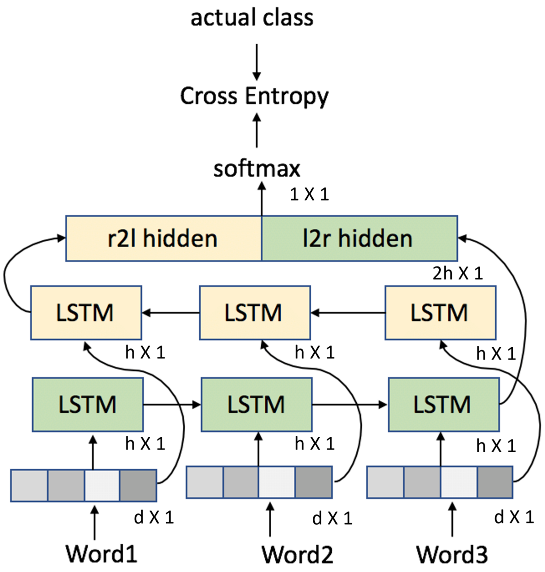


Figure 3: Working of BiLSTM

2.2 Fine-tuning of Model

The training dataset is split into 70:30 ratio for training and validation of model. Binary Cross Entropy Loss is used to define the loss function. The model is trained using gradient descent with Adam Optimizer, with a learning rate of 0.001 for 15 epochs.

2.2.1 Results on Training Dataset

Figure 4 shows the training loss vs epochs plot as a function of epochs. The loss starts at around 0.5 and decreases steadily to around 0.15 by epoch 15. This indicates that the model is effectively learning from the training data.

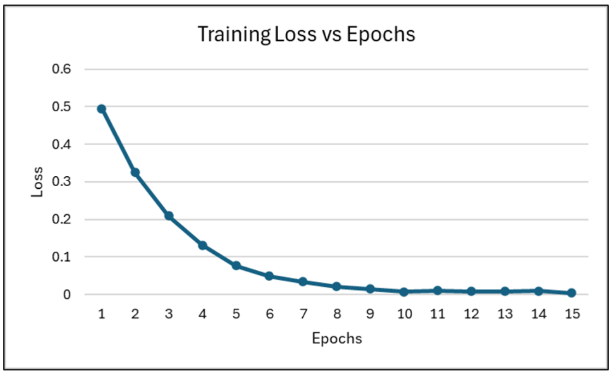
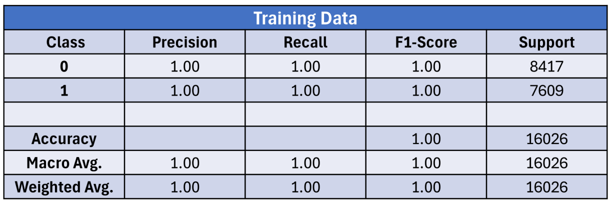


Figure 4: Training Loss vs Epochs (BiLSTM)

The classification report for the fine-tuned model is shown in Figure 5, which indicates the precision, recall, f1-score and support of both the classes, along with the accuracy achieved.



Training Data				
Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	1.00	1.00	1.00	8417
1	1.00	1.00	1.00	7609
Accuracy			1.00	16026
Macro Avg.	1.00	1.00	1.00	16026
Weighted Avg.	1.00	1.00	1.00	16026

Figure 5: Classification Report (BiLSTM: Training)

Figure 6 shows the Receiver Operating Characteristic (ROC) curve for the model, which is plotted for different true positive rates and false positive rates obtained by varying the value of threshold.

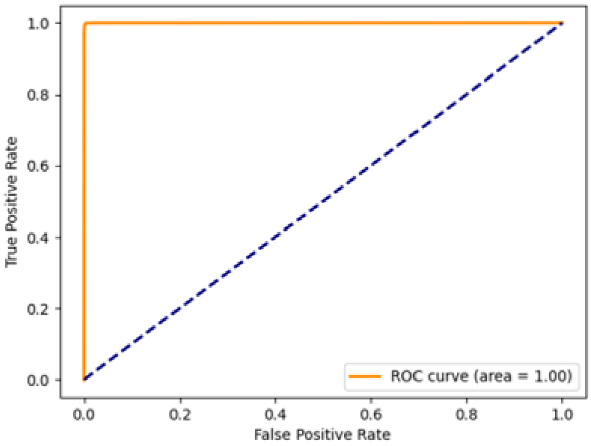


Figure 6: Receiver Operating Characteristic (ROC) Curve (BiLSTM: Training)

2.2.2 Results on Validation Dataset

Figure 7 shows the validation loss vs epochs plot as a function of epochs. The loss starts at around 1.2 and decreases steadily to around 0.4 by epoch 15. This indicates that the model is effectively learning from the training data.

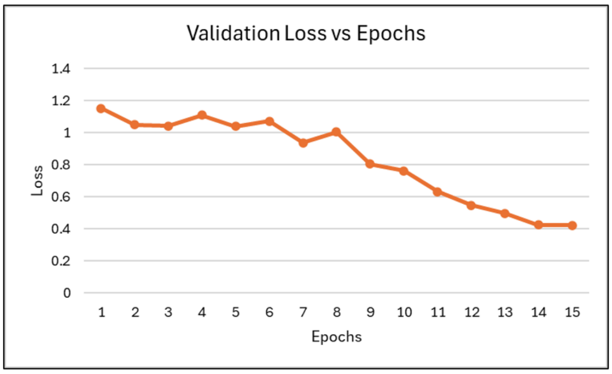
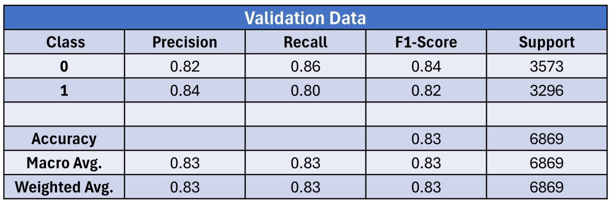


Figure 7: Validation Loss vs Epochs (BiLSTM)

The classification report for the fine-tuned model is shown in Figure 8, which indicates the precision, recall, f1-score and support of both the classes, along with the accuracy achieved.



Validation Data				
Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.82	0.86	0.84	3573
1	0.84	0.80	0.82	3296
Accuracy			0.83	6869
Macro Avg.	0.83	0.83	0.83	6869
Weighted Avg.	0.83	0.83	0.83	6869

Figure 8: Classification Report (BiLSTM: Validation)

Figure 9 shows the Receiver Operating Characteristic (ROC) curve for the model, which is plotted for different true positive rates and false positive rates obtained by varying the value of threshold.

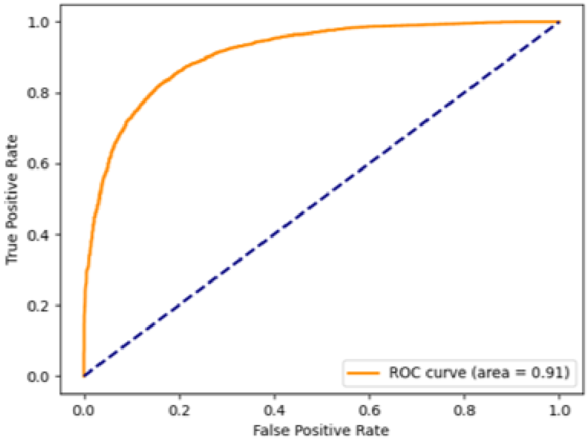


Figure 9: Receiver Operating Characteristic (ROC) Curve (BiLSTM: Validation)

2.3 Testing of Model

The performance of the model is tested on the testing dataset, giving a kaggle score of **0.94409**.

3 BERT (Bidirectional Encoder Representations from Transformers)

BERT (Devlin et al., 2018), short for Bidirectional Encoder Representations from Transformers (Vaswani et al., 2017), is a pre-trained language representation model introduced by researchers at Google in 2018. It has sparked significant interest and advancements in natural language understanding tasks. BERT employs a transformer architecture, which allows it to capture bidirectional contextual information from large amounts of unlabeled text data. This bidirectionality enables BERT to understand the context of a word based on both its preceding and succeeding words, leading to state-of-the-art performance on various natural language processing tasks such as text classification, question answering, and named entity recognition.

3.1 Fine-tuning of Model

The training dataset is split into 70:30 ratio for training and validation of model. Binary Cross Entropy Loss is used to define the loss function. The model is trained using gradient descent with Adam Optimizer, with a learning rate of 1e-05 for 15 epochs.

3.1.1 Results on Training Dataset

Figure 10 shows the training loss vs epochs plot as a function of epochs. The loss starts at around 0.006 and decreases steadily to around 0.0001 by epoch 15. This indicates that the model is effectively learning from the training data.

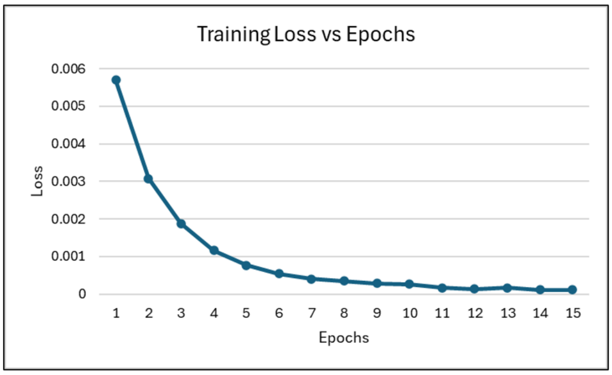
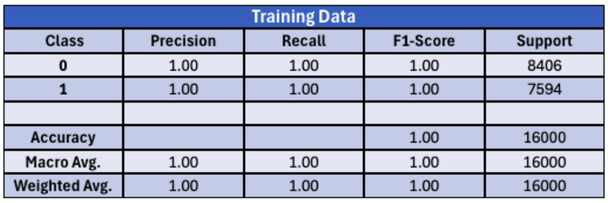


Figure 10: Training Loss vs Epochs (BERT)

The classification report for the fine-tuned model is shown in Figure 11, which indicates the precision, recall, f1-score and support of both the classes, along with the accuracy achieved.



Training Data				
Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	8406
1	1.00	1.00	1.00	7594
Accuracy			1.00	16000
Macro Avg.	1.00	1.00	1.00	16000
Weighted Avg.	1.00	1.00	1.00	16000

Figure 11: Classification Report (BERT: Training)

Figure 12 shows the Receiver Operating Characteristic (ROC) curve for the model, which is plotted for different true positive rates and false positive rates obtained by varying the value of threshold.

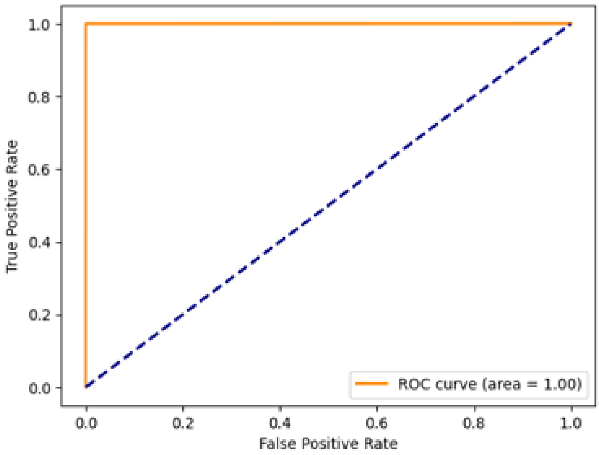


Figure 12: Receiver Operating Characteristic (ROC) Curve (BERT: Training)

3.1.2 Results on Validation Dataset

Figure 13 shows the validation loss vs epochs plot as a function of epochs. The loss starts at around 0.106 and decreases steadily to around 0.101 by epoch 15. This indicates that the model is effectively learning from the training data.

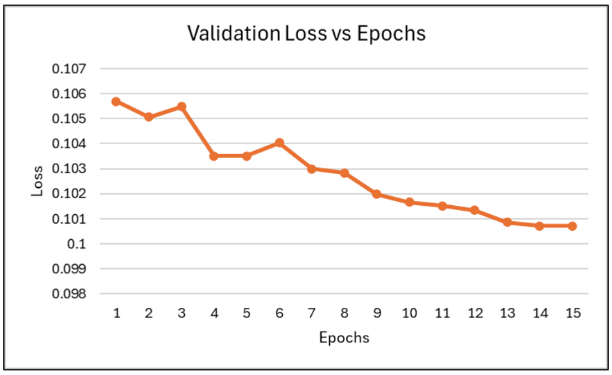
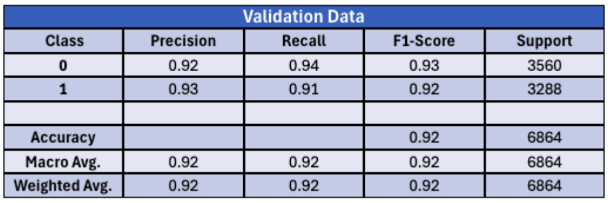


Figure 13: Validation Loss vs Epochs (BERT)

The classification report for the fine-tuned model is shown in Figure 14, which indicates the precision, recall, f1-score and support of both the classes, along with the accuracy achieved.



Validation Data				
Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.92	0.94	0.93	3560
1	0.93	0.91	0.92	3288
Accuracy			0.92	6864
Macro Avg.	0.92	0.92	0.92	6864
Weighted Avg.	0.92	0.92	0.92	6864

Figure 14: Classification Report (BERT: Validation)

Figure 15 shows the Receiver Operating Characteristic (ROC) curve for the model, which is plotted for different true positive rates and false positive rates obtained by varying the value of threshold.

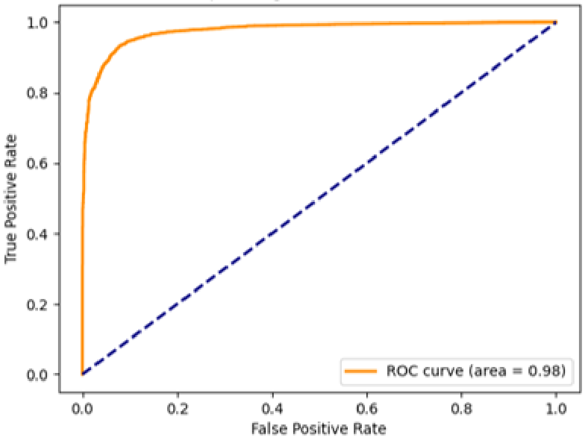


Figure 15: Receiver Operating Characteristic (ROC) Curve (BERT: Validation)

3.2 Testing of Model

The performance of the model is tested on the testing dataset, giving a kaggle score of **0.97012**.

4 RoBERTa (Robustly optimized BERT approach)

RoBERTa (Liu et al., 2019) is a reimplementation of BERT with some modifications to the key hyperparameters and tiny embedding tweaks, along with a setup for RoBERTa pre-trained models. In RoBERTa, we don't need to define which token belongs to which segment or use token_type_ids. With the aid of the separation token tokenizer.sep_token (or), we can easily divide the segments. The Facebook AI and the University of Washington researchers found that the BERT model was remarkably undertrained, and they suggested making several changes to the pretraining process to improve the BERT model's performance. RoBERTa is trained with FULL-SENTENCES without NSP loss, dynamic masking, large mini-batches, a larger byte-level BPE (Byte Pair Encoding).

The key differences between RoBERTa and BERT can be summarized as follows:

- RoBERTa is a reimplementation of BERT with some modifications to the key hyperparameters and minor embedding tweaks. It uses a byte-level BPE as a tokenizer (similar to GPT-2) and a different pretraining scheme.
- RoBERTa is trained for longer sequences, too, i.e. the number of iterations is increased from 100K to 300K and then further to 500K.

- RoBERTa uses larger byte-level BPE vocabulary with 50K subword units instead of character-level BPE vocabulary of size 30K used in BERT.

- In the Masked Language Model (MLM) training objective, RoBERTa employs dynamic masking to generate the masking pattern every time a sequence is fed to the model.

- RoBERTa doesn't use token_type_ids, and we don't need to define which token belongs to which segment. Just separate segments with the separation token tokenizer.sep_token.

- The next sentence prediction (NSP) objective is removed from the training procedure. Larger mini-batches and learning rates are used in RoBERTa's training.

4.1 Architecture

Figure 16 shows the architecture of RoBERTa. The model is based on the Transformer architecture, which is explained in the paper 'Attention is All You Need' (Vaswani et al., 2017). The Transformer architecture is a type of neural network that is specifically designed for processing sequential data, such as natural language text.

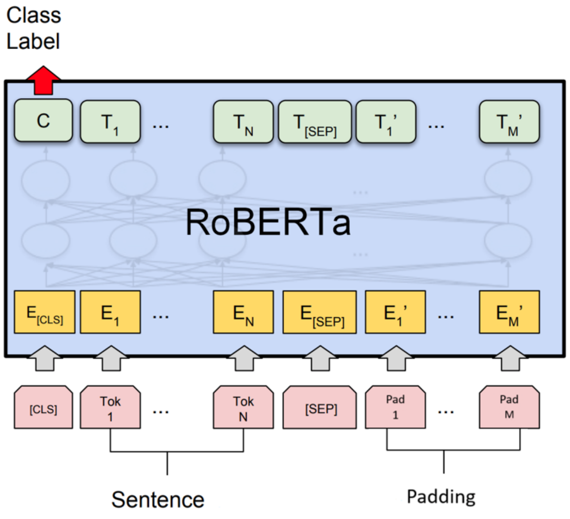


Figure 16: Architecture of Roberta

The architecture is nearly comparable to that of BERT, with a few slight modifications to the training procedure and architecture to enhance the results in comparison to BERT. The RoBERTa model consists of a series of self-attentional and feed-forward layers. The self-attention layers allow the model to weigh the importance of different tokens in the input sequence and compute representations that consider the context provided by the entire sequence. The feed-forward layers are used to transform the representations produced by the self-attention layers into a final output representation.

4.2 Working

RoBERTa works by pre-training a deep neural network on a large corpus of text. Here's a high-level overview of how it works.

Pre-Training: RoBERTa must be pre-trained first on a significant text corpus before being used. A fraction of the tokens in each phrase are randomly masked during pre-training, and the model is trained to predict the masked tokens based on the context provided by the unmasked tokens.

Fine-tuned: The model can be fine-tuned for specific NLP tasks, including named entity recognition, sentiment analysis, or question answering, after pre-training. During fine-tuning, the model is trained on a smaller dataset that is specific to the task at hand, utilizing the pre-learned weights as initialization.

Inference: After fine-tuning, the model can be used for inference on a new text, by inputting the text into the network and using the learned representations to make predictions.

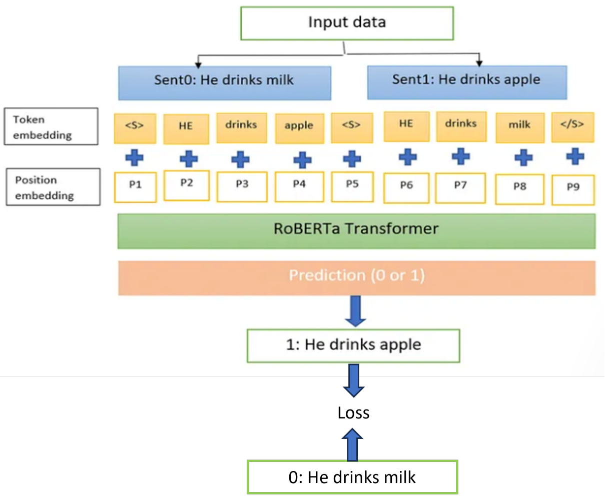


Figure 17: Working of Roberta

4.3 Fine-tuning of Model

The training dataset is split into 70:30 ratio for training and validation of model. Binary Cross Entropy Loss is used to define the loss function. The model is trained using gradient descent with

Adam Optimizer, with a learning rate of 1e-05 for 15 epochs.

4.3.1 Results on Training Dataset

Figure 18 shows the training loss plotted as a function of epochs. The loss value starts from 0.018 and it reaches around 0.001. This indicates that the model is effectively learning from the training data.

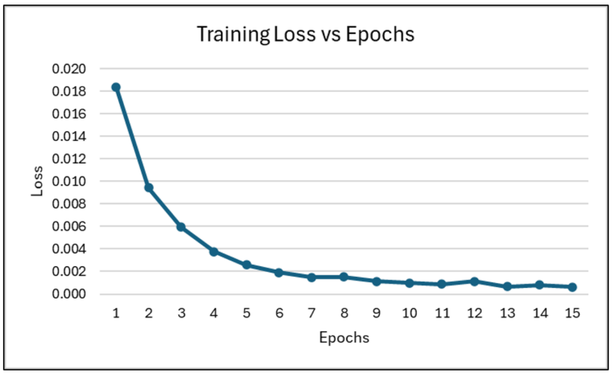
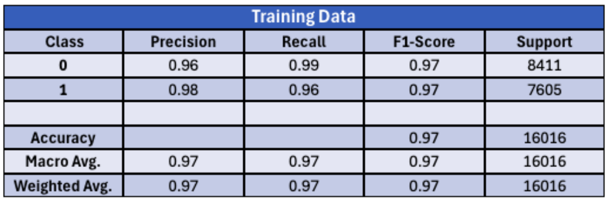


Figure 18: Training Loss vs Epochs (RoBERTa)

The classification report for the fine-tuned model is shown in Figure 19, which indicates the precision, recall, f1-score and support of both the classes, along with the accuracy achieved.



Training Data				
Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.96	0.99	0.97	8411
1	0.98	0.96	0.97	7605
Accuracy			0.97	16016
Macro Avg.	0.97	0.97	0.97	16016
Weighted Avg.	0.97	0.97	0.97	16016

Figure 19: Classification Report (RoBERTa: Training)

Figure 20 shows the Receiver Operating Characteristic (ROC) curve for the model, which is plotted for different true positive rates and false positive rates obtained by varying the value of threshold.

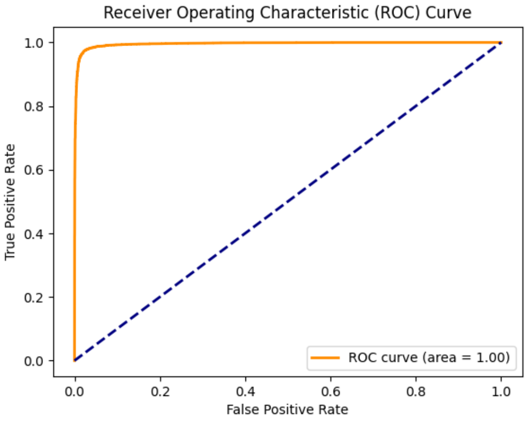


Figure 20: Receiver Operating Characteristic (ROC) Curve (RoBERTa: Training)

4.3.2 Results on Validation Dataset

Figure 21 shows the validation loss vs epochs plot as a function of epochs. The loss starts at around 0.092 and decreases steadily to around 0.09 by epoch 15. This indicates that the model is effectively learning from the training data.

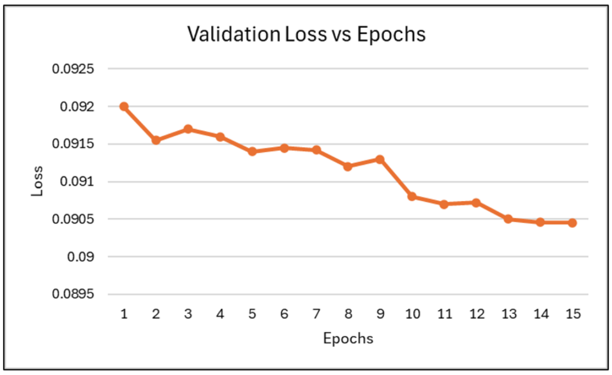
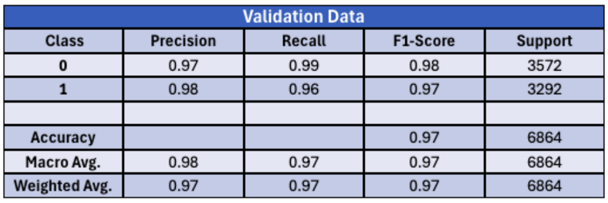


Figure 21: Validation Loss vs Epochs (RoBERTa)

The classification report for the fine-tuned model is shown in Figure 22, which indicates the precision, recall, f1-score and support of both the classes, along with the accuracy achieved.



Validation Data				
Class	**Precision**	**Recall**	**F1-Score**	**Support**
0	0.97	0.99	0.98	3572
1	0.98	0.96	0.97	3292
Accuracy			0.97	6864
Macro Avg.	0.98	0.97	0.97	6864
Weighted Avg.	0.97	0.97	0.97	6864

Figure 22: Classification Report (RoBERTa: Validation)

Figure 23 shows the Receiver Operating Characteristic (ROC) curve for the model, which is plotted for different true positive rates and false positive rates obtained by varying the value of threshold.

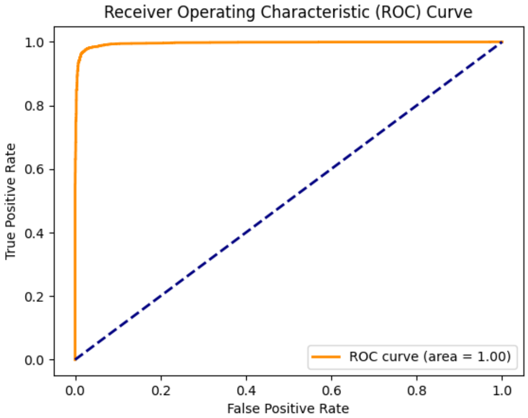


Figure 23: Receiver Operating Characteristic (ROC) Curve (RoBERTa: Validation)

4.4 Testing of Model

The performance of the model is tested on the testing dataset, giving a kaggle score of **0.97012**.

5 Comparison of BiLSTM, BERT, and RoBERTa

Table 1: Comparison of BiLSTM, BERT, and RoBERTa on validation data

Metric	**BiLSTM**	**BERT**	**RoBERTa**
Class 0 Precision	0.82	0.92	0.97
Class 1 Precision	0.84	0.93	0.98
Class 0 Recall	0.86	0.94	0.99
Class 1 Recall	0.80	0.91	0.96
Class 0 F1-Score	0.84	0.93	0.98
Class 1 F1-Score	0.82	0.92	0.97
Class 0 Support	3573	3560	3572
Class 1 Support	3296	3288	3292
ROC-AUC	0.91	0.98	1.00
Accuracy	0.83	0.92	0.97

6 Model Performance Comparison

Figure 24 shows the Kaggle Score obtained for different models.

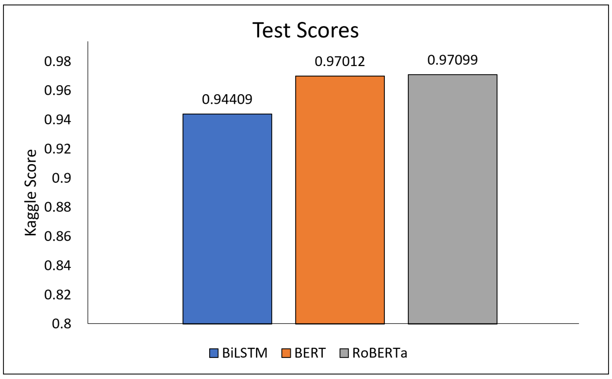


Figure 24: Test Scores (Kaggle)

It is observed that:

- RoBERTa achieves the highest accuracy among the three models, with a score of **0.97099**.
- BERT follows closely behind RoBERTa with an accuracy of **0.97012**, indicating its effectiveness in multi-label classification tasks.
- BiLSTM, while still performing well, lags behind RoBERTa and BERT with an accuracy of **0.94409**.
- The results suggest that transformer-based models like RoBERTa and BERT outperform traditional sequence models like BiLSTM in multi-label classification tasks.
- Considerations for model selection should prioritize RoBERTa or BERT for higher accuracy and better performance.

.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.