COMP 551 Assignment 4 Report

Billy Exarhakos — vassilios.exarhakos@mail.mcgill.ca — 261051989
Sapphire Hou — sapphire.hou@mail.mcgill.ca — 261078333

December 28, 2022

Abstract

Machine learning approaches are more important than ever in the domains of Natural Language Processing (NLP) and Text Classification [1]. For this project, we investigate the performance of Bidirectional Encoder Representations from Transformers (BERT) on the IMDB movie review dataset. We achieve a test AUROC of 96.28% and a testing accuracy of 91%. When we compare our BERT implementation to Logistic Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), XGBoost, and K-Nearest Neighbours (KNN), we find that our implementation regularly outperforms theirs. Overall, we discovered that our BERT model excelled over the other models greatly, but it was also noticeably slower to train.

1 Introduction

Throughout this study, we discuss our BERT model application on a well-known dataset, the IMDB reviews. We illustrate our results through statistics and figures, and compare our implementation against scikit-LR, learn's SVM, RF, XGBoost, and KNN algorithms. On the IMDB reviews dataset, we reach a binary classification accuracy of 91% and a test AUROC of 96.28% using a BERT model. We also discovered that our solution outperforms scikit-learn's classifiers in terms of testing accuracy.

The IMDB dataset contains textual movie reviews and corresponding ratings from 1 to 10. It was initially gathered by a team of Stanford University researchers for their work on *Learning Word Vectors for Sentiment Analysis* [2]. In addition to providing our results on the dataset, we examine our model implementation in depth. Specifically, we look at how our model performs in one hidden attention layer and what attributes our models prioritize during classification. Finally, we use the RF model to extract feature significance importance from our trained BERT model.

The overarching goal of this project is to train and fine-tune the BERT model so that it can classify movie reviews as precisely as possible based on the writers' sentiments. We also note that, as a result of the development of attention, the execution of BERT has been substantially enhanced [3]. Following this work, the BERT model arises and is shown to be effective on a wide range of NLP tasks [4].

2 Datasets

The IMDB dataset is a collection of favorable and critical movie reviews. Figure 1 depicts the number of characters in the 2,000 reviews chosen at random from the training set. Then, by sorting the review length, we discovered that many reviews contain between 500 and 1,000 characters (Figure 2).

Then we partition the reviews into individual units called tokens. Tokenization involves converting words, punctuation marks, and spaces in textual reviews into tokens, which are then converted into integer indices ("ids") that correspond to terms in the BERT lexicon. The next phase is masking, which indicates whether tokens should be ignored or addressed by the model. The input sequences are transmitted via a multi-headed self-attention mechanism in this example, allowing the model to attend to multiple sections of the input sequence while simultaneously time.

The attention mechanism is regulated by a mask, which is an array of 0s and 1s that decides which tokens the model should pay attention to. The mask is multiplied by the attention weights element by element, thus setting the masked tokens' attention weights to zero. This keeps the model from paying attention to the masked tokens and allows it to concentrate on the remaining tokens.

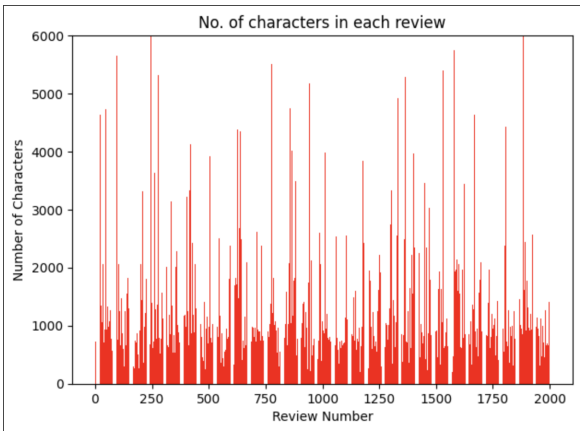

Figure 1: The number of characters in 5,000 randomly selected reviews from the training set

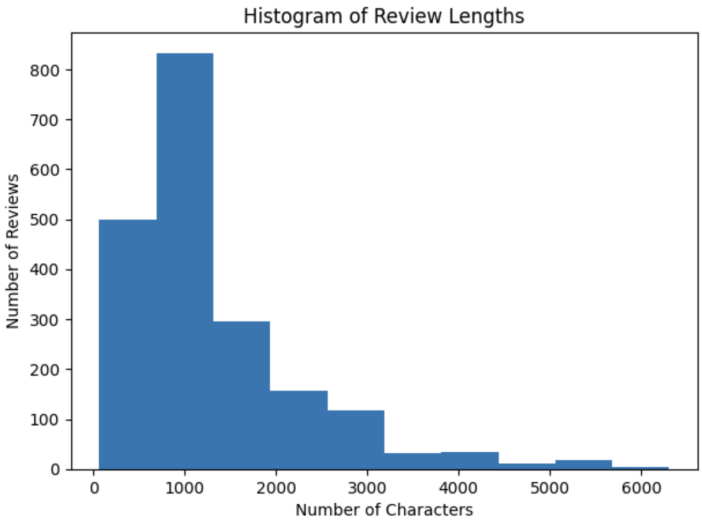

Figure 2: The distribution of lengths of reviews

3 Benchmark

We employ the bert-base-uncased model, which was trained on a huge dataset of lowercase English text. It comprises 12 transformer layers with 12 self-attention heads each, for a total of 144 self-attention heads. The transformer layers are piled on top of each other, and each layer's output is routed through a residual connection and layer normalization before being fed into the next layer. The vocabulary of this BERT model is 30,522 tokens, and it can interpret input sequences of up to 512 tokens in length. It was trained to anticipate masked tokens in the input sequence, so it was fine-tuned for this IMDB sentiment analysis in specific.

The Logistic Regression class from the scikitlearn package is used for the LR model. It uses an L2 regularization penalty with the parameter set to 1.0, resulting in a relatively weak regularization strength, allowing the model to be more flexible. The optimization method has been configured to run for a maximum of 100 iterations.

The SVC class from the scikitlearn package is used for the SVM model. To keep the model's complexity minimal, we employ a linear kernel as the kernel parameter.

The Random Forest Classifier class from the scikitlearn package is used for the RF model. We specify the number of trees to 10, which means the model will train 10 decision trees on the training data. In addition, the Gini impurity measure is used by the RF model to evaluate tree splits.

The Gradient Boosting Classifier from the scikitlearn package is used for the XGBoost model. We employ a decision tree as the weak learner, which is taught in stages, with each learner being trained to rectify the errors caused by the preceding learner via gradient descent.

We also incorporate the scikitlearn library's KNN model for model comparison. We choose 5 neighbors, which indicates that the model will make a forecast based on the majority class of the 5 nearest neighbors. The weights we assign to the neighbors are uniform, which means that each neighbor is given an equal weight.

4 Results

Our first experiment involved training all of the techniques on the IMDB training dataset and comparing their performance on the test set. We trained our model using a random selection of 2,000 reviews from the training set due to BERT's lengthy training period. Looking at the ROC curves of the different models (Figure 3), we can see that, despite the smaller training size, BERT surpassed all of the other approaches significantly. Meanwhile, based on Table 1, we see that our model reach a testing accuracy of 91%.

When the AUROC scores of the models were compared, BERT came out on top, followed by the Logistic Regression model and the SVM model (Figure 4). The higher the AUROC score, the better the model's ability to differentiate between positive and negative samples, with 1 signifying perfect performance and 0.5 suggesting random performance. Our implementation shows that the BERT is the most likely to distinguish between the two classes, followed by the SVM and LR models. However, that improved performance came at a price, since the

BERT model required the greatest time to train (Table 2).

We next dived further into our Random Forest model, extracting the most relevant features based on mean impurity reduction (Figure 5) and feature permutation (Figure 6). Both techniques discovered that the three most relevant attributes for the IMDB dataset were "worst", "poor", and "excellent".

We next investigated one individual attention head to see how the attention layers in our pre-trained multi-headed attention model performed. Figures 7–10 depict the attention weights between the first ten words of the input text in a properly predicted positive review, a correctly predicted negative review, and inaccurately predicted reviews. Each cell's hue in the heatmap represents the attention weight between the relevant input and output word.

Intuitively, we can notice that the attention heatmap focuses on words and phrases that indicate negative feelings, such as "poor," or good sentiments, such as "first" (Figures 7 & 8). However, the model failed to pay attention to emotional terms in mistakenly predicted reviews (Figures 9 & 10). Overall, we can see that one attention head can only give a limited amount of information about how the model arrived at its forecast.

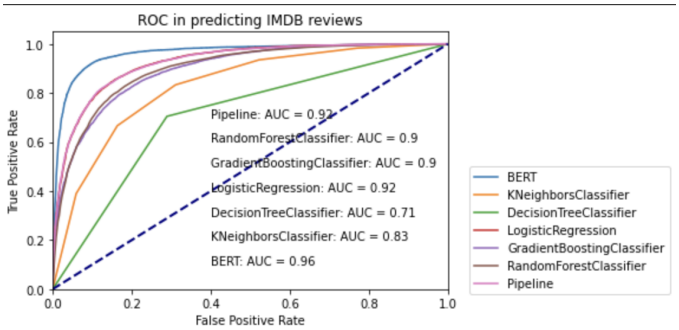

Figure 3: The ROC curves for different models

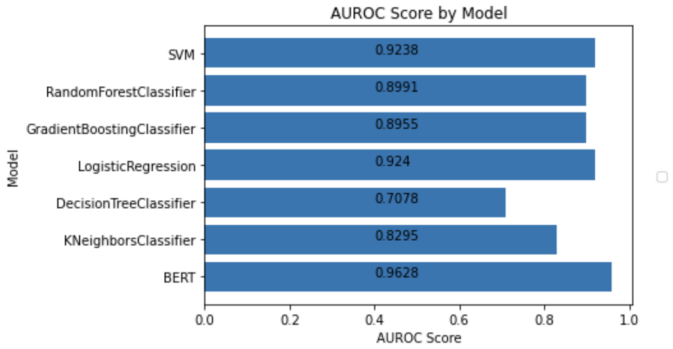

Figure 4: The AUROC scores for different models

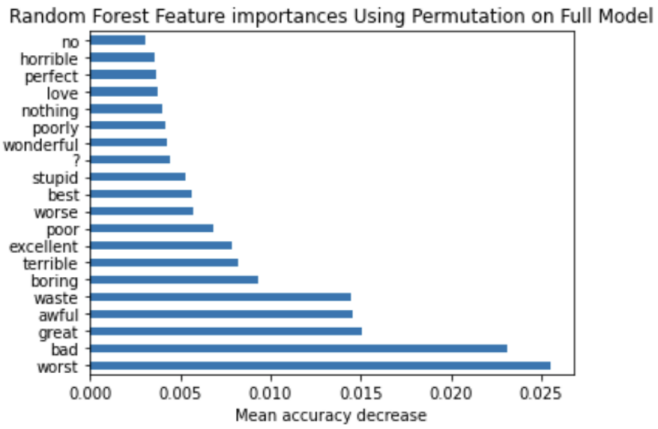

Figure 5: Feature importance on full model

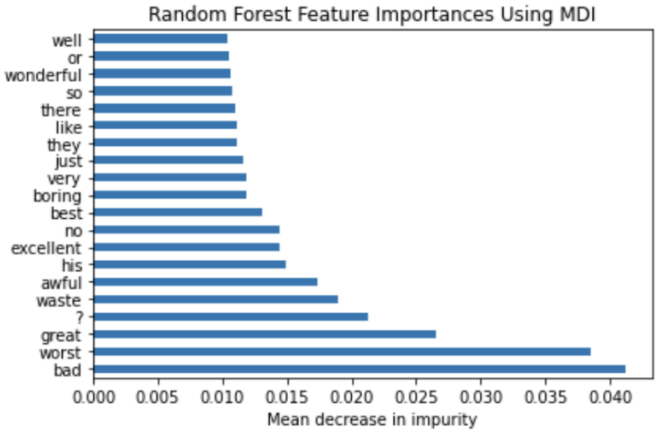

Figure 6: Feature importance using MDI

	Precision	Recall	F1-Score	Support
False	0.95	0.86	0.90	12500
True	0.87	0.95	0.91	12500
accuracy			0.91	25000
macro avg	0.91	0.91	0.91	25000
weighted avg	0.91	0.91	0.91	25000

Table 1: The BERT Model Testing Accuracy

4.1

Classifier	Training Time (s)	Predict Time (s)
BERT (on a sample of 2,000 reviews)	6124.34	1050.23
KNN	0.03	14.67
Decision Tree	0.91	0.017
Logistic Regression	0.14	0.01
XGBoost	7.49	0.07
Random Forest	5.12	0.62
SVM	2842.87	15.56

Table 2: Model Training and Predict Time

4.2

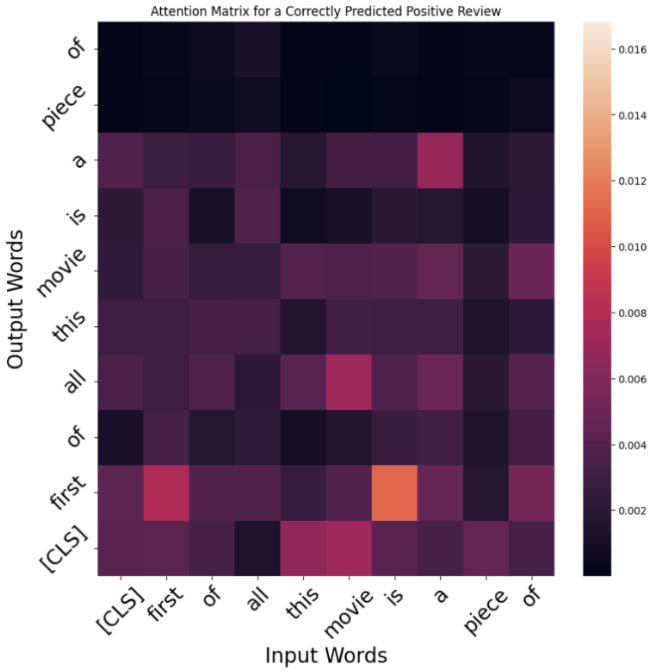

Figure 7: The heatmap on correctly predicted positive review

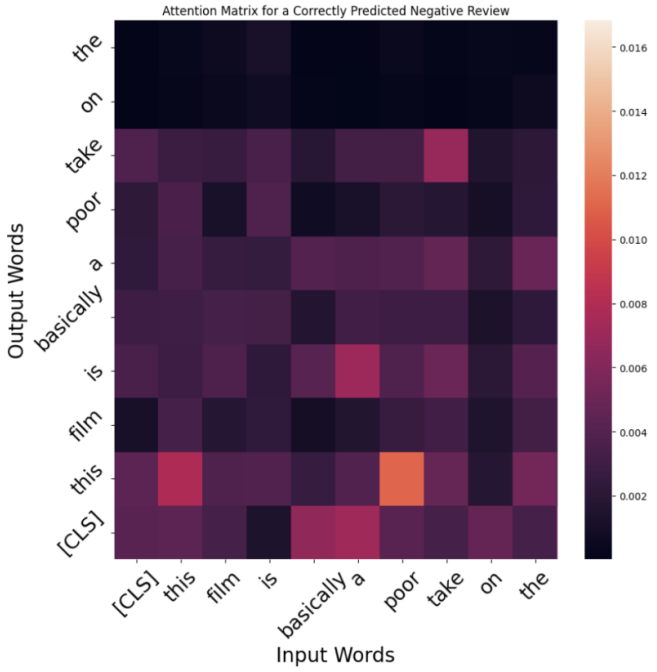

Figure 8: The heatmap on correctly predicted negative review

5 Discussion and Conclusion

Strong natural language processing (NLP) models, such as pre-trained BERT models, may be customized for a wide range of NLP applications, including sentiment analysis. Because it makes use of a large quantity of annotated data, it can do well on this assignment by predicting review sentiment with high accuracy. Tokenization, on the other hand, is one of the major hurdles to using BERT for sentiment analysis, as the model requires the input data to be tokenized in a certain way. Another difficulty with BERT is its high computational cost. In our testing, we only trained it on a subset of the training set, and it still took more than twice as long to train as other models.

Future research topics might include comparing several pre-trained sentiment analysis models, such as RoBERTa or XLNet, and experimenting with other ways for fine-tuning, such as using different optimization techniques or training data. Other research areas might include the implications of various data augmentations on the efficacy of BERT for sentiment analysis, as well as ways for improving the model's performance by incorporating domain expertise.

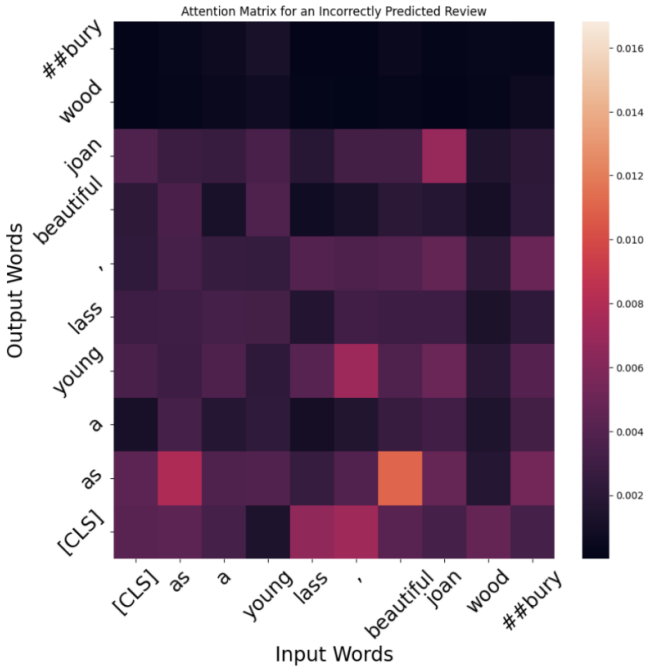

Figure 9: The heatmap on incorrectly predicted positive review

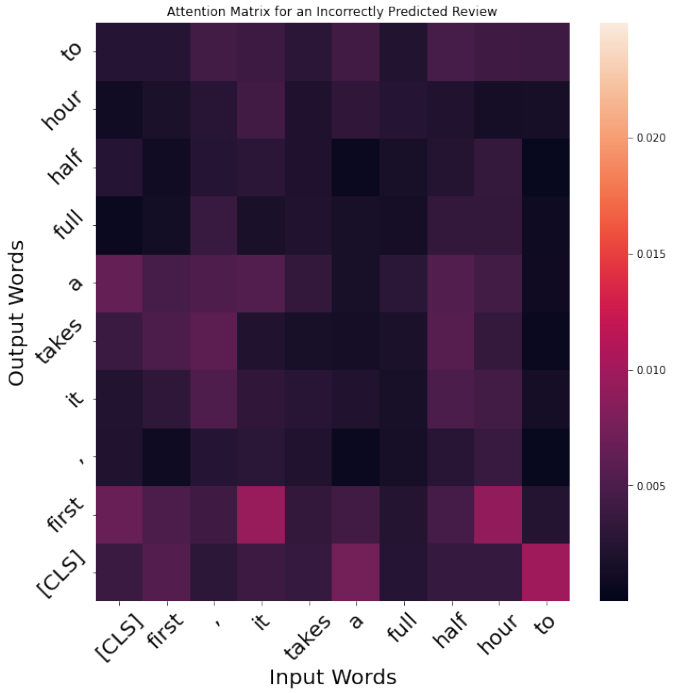

Figure 10: The heatmap on incorrectly predicted negative review

6 Statement of Contributions

Billy and Sapphire contributed equally to both the coding and report writing.

References

[1] Ammar Ismael Kadhim. Survey on supervised machine learning techniques for automatic text classification. *Artificial Intelligence Review*, 52(1):273–292, 2019.

[2] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, page 142–150, USA, 2011. Association for Computational Linguistics.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.