COMP551 Mini-Project3 Report

Leila Wang, Raphael Wang, Oliver Wang

Abstract

This project compares the performance of three models for IMDB Reviews classification: Gaussian Naive Bayes, Multinomial Naive Bayes, and BERT. The results showed that Multinomial Naive Bayes had an accuracy of 0.84, which was better than Gaussian Naive Bayes with an accuracy of 0.75. However, BERT significantly outperformed both Naive Bayes models with an accuracy of 0.93. This can be attributed to the fact that BERT is pre-trained on a large corpus, enabling it to understand contextual information better than the Naive Bayes models, which can only learn from the given training data. The fact the BERT outperforms than Naive Bayes models indicates that deep learning methods are more effective than traditional machine learning methods.

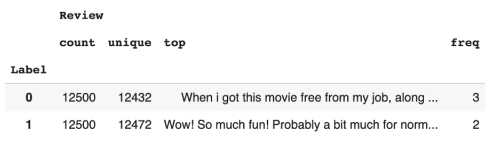
1 Introduction

In this project, the traditional machine learning and deep learning NLP approaches are used to classify reviews of a data set into positive and negative categories. The traditional machine learning and deep learning techniques are compared using the classification performance of the Naive Bayes models and the BERT model on the IMDB Reviews dataset. According to the findings, the DL techniques produced more accurate results than the traditional ML algorithms.

The Large Movie Review Dataset is a popular benchmark dataset for sentiment analysis. It contains 50,000 movie reviews, evenly split between 25,000 reviews for training and 25,000 reviews for testing. The reviews are highly polarized, either positive or negative, and are labeled accordingly. In addition, there are also unlabeled data provided for use in unsupervised learning approaches. The dataset is available in both raw text and pre-processed in bag-of-words formats. For this project, we divided the labeled data into three sets: training, validation, and testing. The training dataset and the testing dataset each contains 50% of the labeled data, whereas the validation dataset is then split from the training dataset for cross validation improvements. It is important to keep in mind that the test dataset is completely independent of the training and validation sets and is only used to evaluate the model's final performance.

2 Datasets

Figure 1 shows the description of the training set we will use for this project, the number of positive and negative reviews have the same counts.



	Review			
	count	unique	top	freq
Label				
0	12500	12432	When i got this movie free from my job, along ...	3
1	12500	12472	Wow! So much fun! Probably a bit much for norm...	2

Figure 1: The description of the training set of the data

Based on Figure 2, it is evident that there is a correlation between review word counts and their label, i.e., positive or negative. Label 1 has a higher word count overall, and reviews that have over 10000 counts are exclusively labeled as positive. This observation suggests that individuals who enjoy a film tend to write longer reviews.

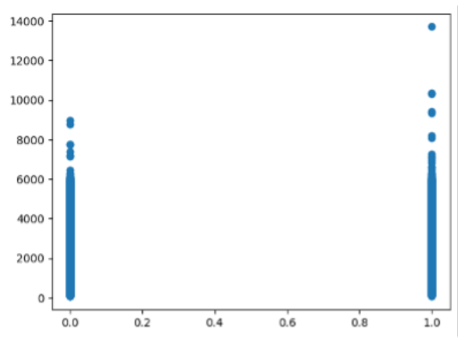
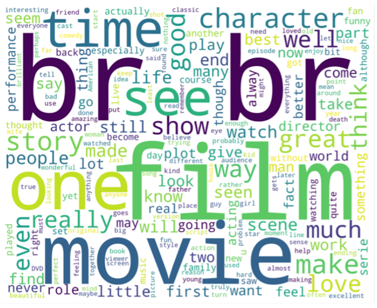


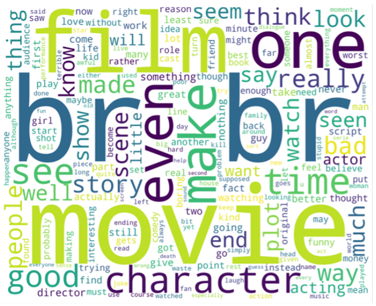
Figure 2: Word counts according to label 0 and 1

Figure 3 shows the word clouds of the most frequent words in each positive and negative dataset before any data preprocessing. We can hardly see any pattern if we just use the unprocessed text data.

Next, we clean and preprocess text data to remove noise and irrelevant elements and standardize the format of the data to improve the effectiveness of NLP models and algorithms. This involves transforming to lowercase, removal of stopwords and punctuation such as HTML, and numbers of words with only 2 letters. With lemmatization, we can effectively reduce the inflectional forms of words to their base or root form, which can provide a clearer representation of the underlying sentiment in a corpus. By performing lemmatization and plotting the word clouds again (Figure 4), we can observe noticeable differences in the composition of the wordclouds between positive and negative reviews. For instance, negative reviews may contain words such as "bad",

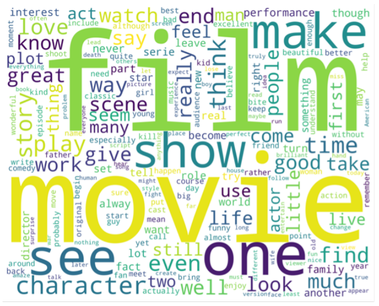


(a) Positive Reviews

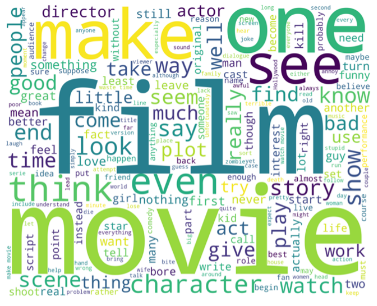


(b) Negtive Reviews

Figure 3: Word clouds of the most frequent words in each positive and negative dataset without data preprocessing



(a) Positive Reviews



(b) Negtive Reviews

Figure 4: Word clouds of the most frequent words in each positive and negative dataset after data preprocessing.

”mean”, and ”poor”, while positive reviews may contain words such as ”best” and ”love”.

Before training a model, it is also a good idea to shuffle the data. This is because the order of the data might occasionally affect the model’s performance. For example, if the data is arranged by label so that all positive instances come before negative examples, the model may learn to predict positive for all examples it meets until it encounters a negative example. When the model is applied to fresh, previously unseen data, this can result in poor generalization and a high error rate. Shuffling the data removes any order or bias from the dataset, allowing the model to acquire a more generalized and robust collection of patterns from it.

3 Results

3.1 Naive Bayes Model

We first implemented a Gaussian Naive Bayes model from scratch and fitted the training data to the model. The model obtained an accuracy of 0.72952, which is not very appealing. A detailed performance report of the Gaussian model is included in section 3.1.2 Improvements.

Figure 5 is the most frequent top five words in the dataset and their distribution, so we can see that each feature is skewed and not normally distributed (Figure 6). Gaussian Bayes is a classification algorithm that assumes the features are normally distributed. If the data is not normally distributed, this assumption may not hold, leading to lower accuracy. Also, since the data are counted so it is discrete, but the Gaussian model works best with continuous data. So we tried to use the multinomial Bayes model next.

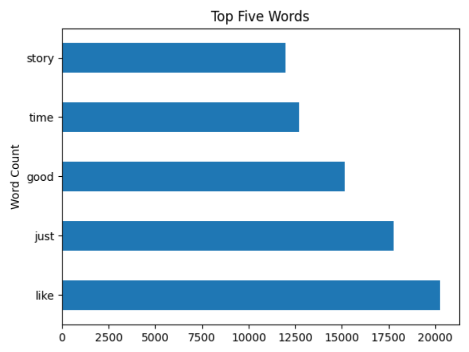


Figure 5: The Top Five Words After Vectorizing the Dataset

In our case, the counts are discrete so we will use multinomial bayes instead.

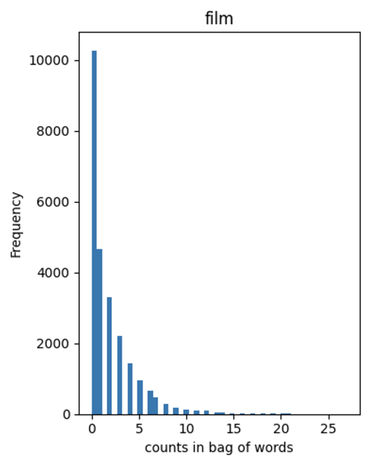


Figure 6: The distribution of the highest occured word

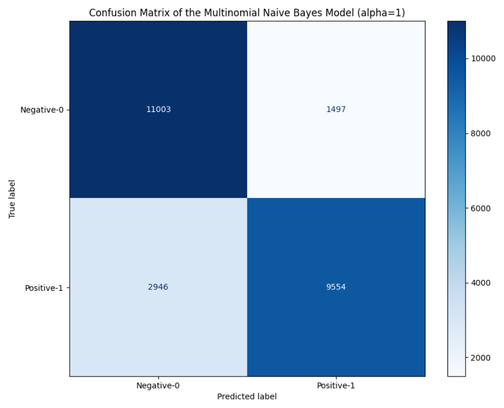


Figure 8: The Confusion Matrix of the Multinomial Naive Bayes Model

3.1.1 Performance of the Multinomial Naive Bayes Model

Our multinomial naive Bayes model implemented from scratch obtained an accuracy of 0.82228 without hyper-parameter tuning. The Laplace smoothing parameter α was simply set to 1. Figure 7 shows the ROC curve of the model and Figure 8 is the confusion matrix of the model output. The entries in the confusion matrix represent the number of instances that fall into each of these categories.

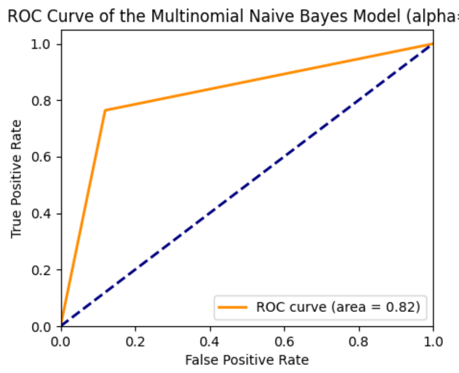


Figure 7: The ROC curve of the Multinomial Naive Bayes Model

3.1.2 Improvements

Hyper-parameter Tuning

To improve the performance of the model, we also conducted hyper-parameter tuning for the Multinomial Naive Bayes model on the α for Laplace smoothing using *GridSearchCV* in Python and the maximum feature size during the vectorizing process of the data using the `optuna` package. We set α equal to 1 to 100 and compared the accuracy of the model. It was discovered that the maximum accuracy was reached when the smoothing parameter value approached zero. This means that only a small amount of smoothing was required to achieve the best accuracy. The model reached to the highest average accuracy of 0.8596 when $\alpha = 2$. Figure 9 illustrates the training and testing scores for α equals 1 to 100.

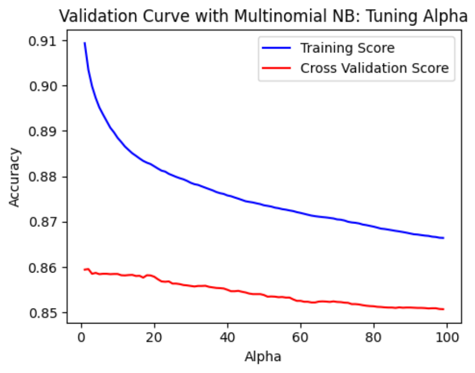


Figure 9: Hyper-Parameter Tuning on the Laplace Smoothing Parameter From 1 to 100

Then we manually set $\alpha = 2$ for tuning the maximum feature size using the `optuna` package. We set the maximum features equal to 3000 to 50000 and compared the accuracy of the model. The trial that achieved the best accuracy among 100 study trial iterations was chosen. Figure 10 is the result of maximum feature size tuning for the Multinomial Naive Bayes model. It turns out that trial 38 with max features equalling 4000 reached the highest accuracy of 0.83832 among 100 trials.

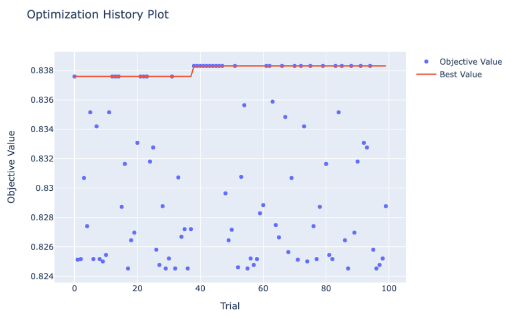


Figure 10: Hyper-Parameter Tuning on Maximum Feature Size From 3000 to 50000

We discovered that the maximum accuracy was reached when the maximum feature size is small (i.e., less than 10000). Thus, we set the max features from 3000 to 13000 and tuned this parameter more precisely again. It turns out that trial 13 with max features equalling 4000 reached the highest accuracy of 0.83832 among 30 trials, which agrees with our first tuning result. Figure 11 shows the results of our second tuning.

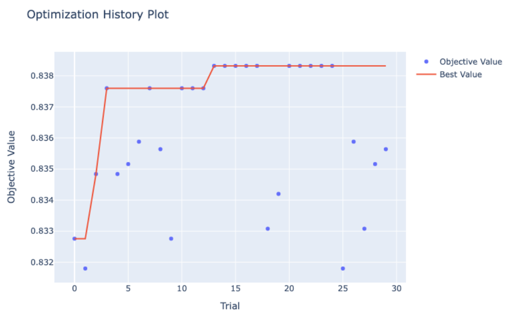


Figure 11: Hyper-Parameter Tuning on Maximum Feature Size From 3000 to 13000

Note that we tuned the hyper-parameters using the built-in MultinomialNB model from the Scikit-Learn package for much less running time.

Cross Validation

We also try to improve the model performance using the five-fold cross-validation while using the tuned hyper-parameters for the model. Each fold was used as the test set once, and the remaining four folds were used as the training set. Our model obtained an average training accuracy of 85.938 and an average testing accuracy of 84.664, which is much higher than our initial try. Figure 12 shows the training and validation accuracy curves of the multinomial naive Bayes model.

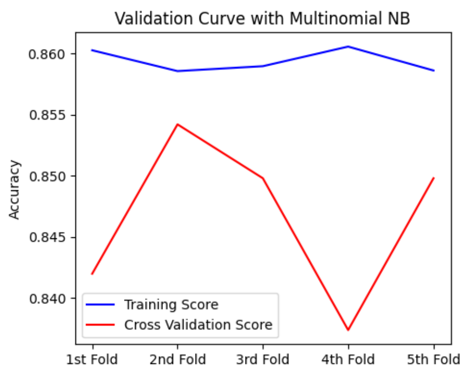


Figure 12: Multinomial Naive Bayes Models: Training and Validation Curves

Gaussian Naive Bayes

As mentioned at the beginning of section 3.1.1, the Gaussian Naive Bayes model obtained an accuracy of 0.72952. Figure 13 shows the ROC curve of the model and Figure 14 is the confusion matrix of the model output.

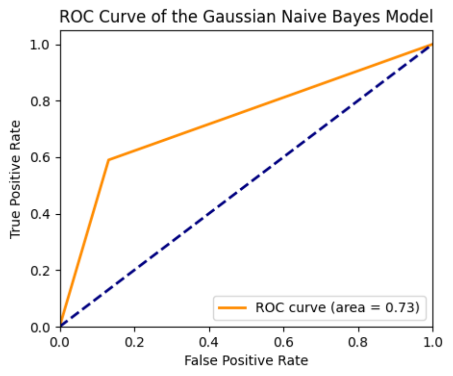


Figure 13: The ROC curve of the Gaussian Naive Bayes Model

After hyper-parameter tuning on the smoothing parameter equals to $1.519911082952933e-09$ and cross-validation, the model achieved the highest training accuracy of 77.891% and testing accuracy of 75.452%. Figure 15 compares the validation curves of the Gaussian model and the Multinomial model. The results showed that the Multinomial Naive Bayes model had greater overall accuracy than the Gaussian Naive Bayes model, with an average accuracy of roughly 0.85. This implies that the Multinomial Naive Bayes model might be a better fit for our task.

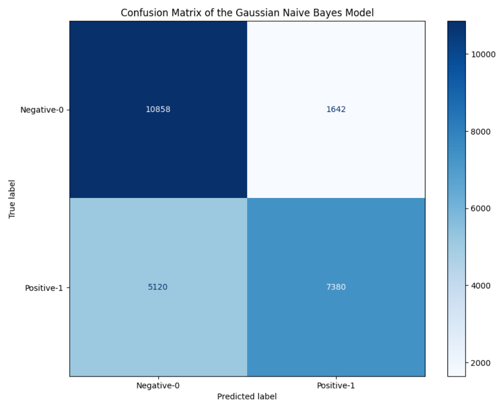


Figure 14: The Confusion Matrix of the Gaussian Naive Bayes Model

Also, with a higher ROC curve than the Gaussian Naive Bayes classifier, the Multinomial Naive Bayes classifier performs better overall in terms of correctly categorizing positive and negative cases. The type of the data, for example, can be a contributing factor to the difference in ROC curves.

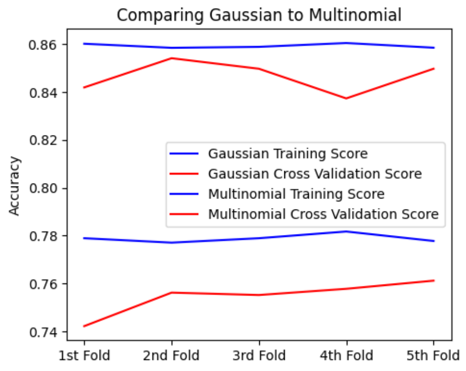


Figure 15: Comparison between Gaussian and Multinomial Naive Bayes Models: Training and Validation Curves

3.2 BERT

BERT is a language model that was pre-trained on a vast dataset comprised of 3.3 billion words from various sources, including Wikipedia and Google's BooksCorpus.[2] During pre-training, BERT was trained to predict missing words in sentences by analyzing the context in which they appeared. This allowed BERT to develop a deep understanding of the meaning of words and phrases in context, making it useful for a range of natural language processing tasks like sentiment analysis.

When it comes to sentiment analysis of movie reviews, pretraining with BERT can help the model recognize the complexities of language and context that are crucial for the understanding sentiment. For instance, pretraining can enable the model to detect negation, sarcasm, and other forms of figurative language that can impact sentiment. It can also teach the model to distinguish between words and phrases that are frequently used in positive or negative contexts, such as "amazing," "terrible," "good," and "bad." Pretraining with BERT can improve the performance of models by providing them with a deeper understanding of the structure and meaning of language, which would not have been possible with smaller datasets and without pretraining. We

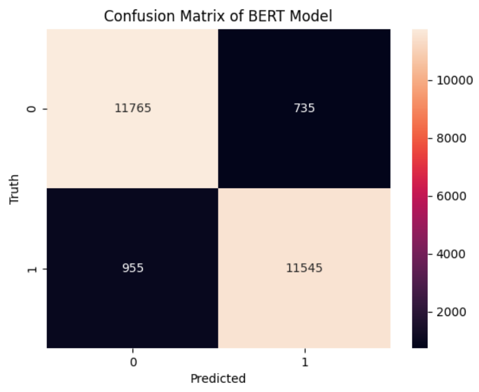


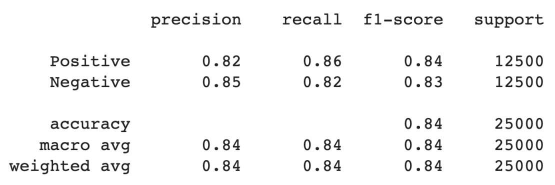
Figure 16: The Confusion Matrix of the BERT model

ran the BERT model on the IMDB dataset and the confusion matrix is shown in Figure 16. The true precision and recall rates are 0.94 and 0.92 and the false precision and recall rates are 0.92 and 0.94. The accuracy of the BERT model is 0.93.

3.3 Task3.1

According to the findings of an analytical paper, deep learning algorithms outperform the Naive Bayes model. These findings are congruent with those obtained in our study, indicating that deep learning approaches outperform classic machine learning methods such as Naive Bayes for a variety of tasks. Compared to the Naive Bayes model, the BERT model has an overall better performance than it. This is because BERT is pre-trained on a large corpus of text, which enables it to learn general language patterns that can be applied to a wide range of downstream tasks, including sentiment analysis while the Naive Bayes Model only learns from the given training dataset. [1]

The two tables in Figure 17 compare the performance of the Multinomial Naive Bayes Model and the BERT model on the IMDB Reviews classification task.

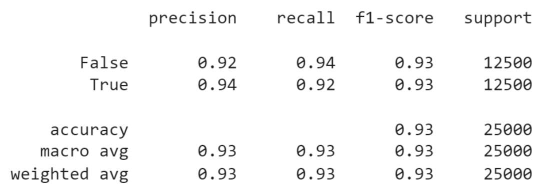


```
              precision    recall  f1-score   support

    Positive       0.82      0.86      0.84     12500
    Negative       0.85      0.82      0.83     12500

    accuracy                           0.84     25000
   macro avg       0.84      0.84      0.84     25000
weighted avg       0.84      0.84      0.84     25000
```

(a) Classification Report of Multinomial Naive Bayes Model



```
              precision    recall  f1-score   support

       False       0.92      0.94      0.93     12500
        True       0.94      0.92      0.93     12500

    accuracy                           0.93     25000
   macro avg       0.93      0.93      0.93     25000
weighted avg       0.93      0.93      0.93     25000
```

(b) Classification Report of the BERT Model

Figure 17: Classification Report of Naive Bayes Model and BERT Model

Conclusions can be made about the performance difference between deep learning and traditional machine learning methods. Naive Bayes is a straightforward and effective model for sentiment analysis, but it has drawbacks that could affect how well it performs, one of which is the assumption of word independence, which makes it potentially incapable of capturing the intricate links between words and phrases. This can be particularly difficult when the sentiment of a sentence is affected by the context in which certain words appear or when the sentence contains contrasting sentiments. Take the phrase "The food was terrible, but the service was great," for instance. Here, the term "terrible" conveys a bad feeling, whereas the word "great" conveys a good one. Naive Bayes would treat these words independently and might not be able to correctly predict the sentence's overall tone.

Naive Bayes may also have trouble generalizing to new or unexplored data because it needs a labeled dataset for training. On the contrary, deep learning models like BERT can make use of unsupervised pretraining to learn the grammatical structure and semantics of a language, which can improve their ability to recognize the intricate links between words and phrases that are crucial for comprehending sentiment. The BERT model can better capture the complex relationships between words and phrases. This allows it to better understand the overall sentiment of a sentence, even in cases where the sentiment of individual words might be ambiguous.

3.4 Task3.2

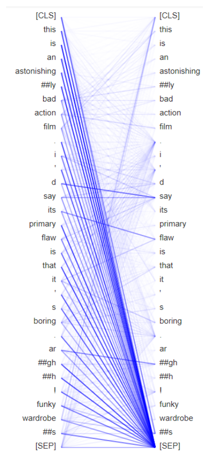


Figure 18: The distribution of attention across words in an incorrectly predicted sentence

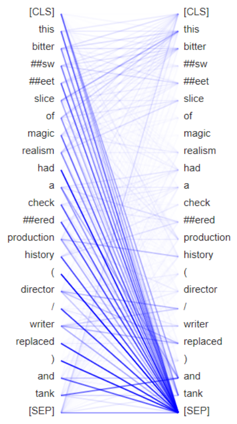


Figure 19: The distribution of attention across words in a correctly predicted sentence

We picked two sentences (Figure 18, Figure 19), one from the correctly predicted results and one from the incorrectly predicted results, to examine the attention matrix between the words and the class tokens. We chose to observe the 9th layer and the third head and eventually found out that for the sentence that was correctly predicted, the BERT model pays attention to words like "bad" that are straightforward and representative of the tone of the sentence. However, for the one that was incorrectly predicted, we concluded that the sentence was too complicated for the model to understand tone such as sarcasm, and therefore, even though the model also pays attention to the words that are adjectives and critical, it was difficult for it to classify the text.

4 Discussion and Conclusion

This project examined the ability of the Naive Bayes Model and BERT model to perform text classification. The results show that in comparison, BERT model with an accuracy of 0.93 outperformed the Naive Bayes Model with an accuracy of 0.84. We concluded that this is largely due to the fact that BERT is pre-trained on a large corpus that enables it to understand some words ahead while the Naive Bayes Model can only learn from the given training data which is limited. Therefore, we claim that deep learning methods perform better than traditional machine learning methods. Although with higher accuracy, we found out that it is still challenging for BERT to understand sentences with rhetorical devices such as sarcasm which often mislead the attention mechanism and cause the model to output wrong results. Overall, the study highlights the effectiveness of BERT over Naive Bayes Model for text classification tasks, and suggests that further exploration of deep learning methods in natural language processing is warranted, especially in developing models that can better handle rhetorical devices like sarcasm.

5 Statement of Contributions

Every group member cooperated and contributed to each part of this project.

References

[1] K. Amulya et al. "Sentiment Analysis on IMDB Movie Reviews using Machine Learning and Deep Learning Algorithms". In: *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*. 2022, pp. 814–819. DOI: `10.1109/ICSSIT53264.2022.9716550`.

[2] Britney Muller. *BERT 101 - State Of The Art NLP Model Explained*. `https://huggingface.co/blog/bert-101#5-enviornmental-impact-of-deep-learning`.