Comparison of Sentiment Classification with Naive Bayes and a Pre-trained Transformer Model (BERT)

1st Sasha Denouvilliez-Pech
McGill University
sasha.denouvilliez-pech@mail.mcgill.ca

2nd Shidan Javaheri
McGill University
shidan.javaheri@mail.mcgill.ca

3rd Taz Scott-Talib
McGill University
taz.scott-talib@mail.mcgill.ca

***Abstract*—This paper contrasts the power of simplicity and complexity in machine learning models by comparing a set of simple models - Normal, Bayesian and Multinomial Naive Bayes models - with the complex pre-trained BERT transformer network. All of these models have been either designed or adapted to perform sentiment classification on a benchmark dataset of positive and negative movie reviews from IMDB. For the Naive Bayes models, feature selection was performed on the training dataset, and models were compared with optimal values for smoothing parameters α and β. With these parameters, the best Naive Bayes model was the Bayesian Naive Bayes model with α and β set to 0.2, giving a training accuracy of 86.3% and a test accuracy of 85.3%. For the BERT model, a transfer learning exercise was performed on the same IMDB dataset with a BERT variation, ALBERT, A Lite BERT [1]. The training procedure was inspired from an official TensorFlow tutorial [2] that follows the practices outlined in the original BERT paper [3]. After hyperparameter tuning, the fine-tuned ALBERT model attained a test accuracy of 88.8%. Final Results are summarized in the table below:**

TABLE I: Model Performance Summary

Model	Test Accuracy (%)
Bayesian Naive Bayes	85.3
BERT	88.8

***Index Terms*—BERT, Feature Selection, Naive Bayes, Sentiment Classification, Transformer Networks**

I. Introduction

The question of whether to use a simple or complex model is a crucial one in Machine Learning, one that has bearing on the physical and computational cost of the model, its runtime and its performance [4]. This paper investigates and compares the best of a set of very simple models, namely Normal, Bayesian and Multinomial Naive Bayes, to BERT, a complex transformer network that has been adapted to perform sentiment classification on a textual dataset. Both models are trained and tested on the IMDB dataset of movie reviews, which is pre-processed appropriately for each one.

For the Naive Bayes models, data is pre-processed and feature selection is performed. Firstly, words that occur in above and below two different percentage thresholds of documents are eliminated - removing words that are either specific to a select few reviews, or ubiquitous within the dataset. Then, eliminating words that occur at similar frequencies in both the positive and negative classes of the training dataset is investigated. A combination of both of these techniques proved to be the most effective, with details further analysed in Section II. Subsequently, the values of the smoothing parameters in the Bayesian Naive Bayes Model, α and β, are investigated and optimized. Finally, the Normal and Multinomial Models were compared, with and without smoothing, to determine the Naive Bayes model with the best performance. This ended up being the Bayesian Naive Bayes model with α and β set to 0.2, producing a test accuracy of 85.3%. This performs better than other researched models performing the same task [5] [6], which achieved 81.45% and 84.5% test accuracy respectively.

For the BERT model, a variation called ALBERT is used instead, to provide similar performances with reduced memory consumption and an increase in training speed [1]. The IMDB training reviews are separated into an 80/20 split for training and validation. The architecture of the ALBERT implementation is shown in Figure 1. The reviews are pre-processed into the format expected by the ALBERT model using ***albert_en_preprocess*** from TensorFlow Hub [7] (Preprocessor). Additionally, the ALBERT model, ***albert_en_base***, is also loaded from TensorFlow Hub [8]. The output of the ***albert_en_base*** is concatenated with a dropout layer and a dense layer to perform the classification task. The training procedure replicates the guidelines provided by the original BERT authors [3] as closely as possible. Hyperparameter tuning was performed on the learning and dropout rate to improve test accuracy. Furthermore, to investigate the effect of pre-training on the external text corpus for the ALBERT model, its 11,683,584 pre-trained weights are frozen and only the classification layer is trained, leading to a test accuracy of 50.3% compared to 88.8% for the fined-tuned model, with optimized hyperparameters of 2e-5 for the learning rate and 10% for the dropout rate. Finally, the attention matrix is computed for a few reviews.

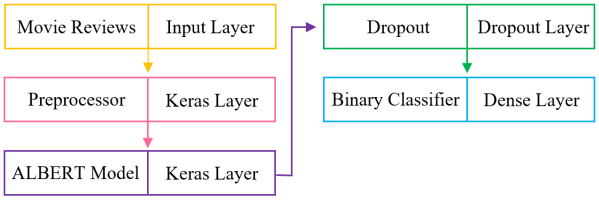

Fig. 1: ALBERT Model Architecture

II. Dataset and Data Pre-processing

The IMDB textual dataset consists of 25,000 training and 25,000 test instances, each with an equal distribution of 12,500 positive and negative reviews. This dataset is pre-processed in different ways for the Naive Bayes model and the BERT transformer network.

A. Naive Bayes

Once the data is downloaded and extracted, the SciKit Learn CountVectorizer function is used to vectorize the data into a sparse matrix. Without any pre-processing, the model's vocabulary from the training data consists of 74,849 words. The majority of these words occur extremely infrequently - including typos, proper nouns and made up words. These additional features give the Naive Bayes model too many parameters, as the presence of a word in even one document forces it to be taken into account in the prediction of all others. This drastically increases computation time and makes the model prone to overfitting, as the parameters it learns are particular to specific instances in the training data. To investigate this effect, we eliminated words that occurred in less than a certain percentage of the reviews. We varied this percentage threshold and evaluated the test accuracy on a normal Naive Bayes model with Laplace smoothing. This experiment was carried out first so that the computation time of future experiments could be dramatically reduced. The results are visible in Figure 2, shown below.

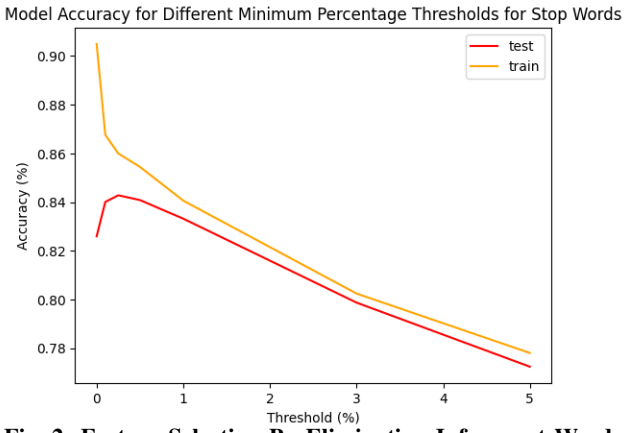

Fig. 2: Feature Selection By Eliminating Infrequent Words

As shown in Figure 2, eliminating the 69,434 words that occur in less than 0.25% of the documents results in the highest test accuracy. The figure also demonstrates that the higher the minimum percentage threshold, the more regularization is added to the model, confirming our hypothesis that with too many features the model overfits. After reaching a maximum at a threshold of 0.25%, test accuracy begins decreasing almost linearly, with training accuracy closely following suit.

Results from this section lead future experiments to be performed on a dataset with a minimum percentage threshold of 0.25%, giving a model with only 5,457 words in its vocabulary. To perform further feature selection, we eliminate words that occur very frequently in the dataset. By analyzing the top ten words as shown in Figure 8 in the appendices, the most frequent words provide little meaning and would not help the model distinguish between classes. To investigate this, we eliminate words that occur in more than a certain percentage of the reviews, varying this percentage threshold. The results are shown in Figure 3 below:

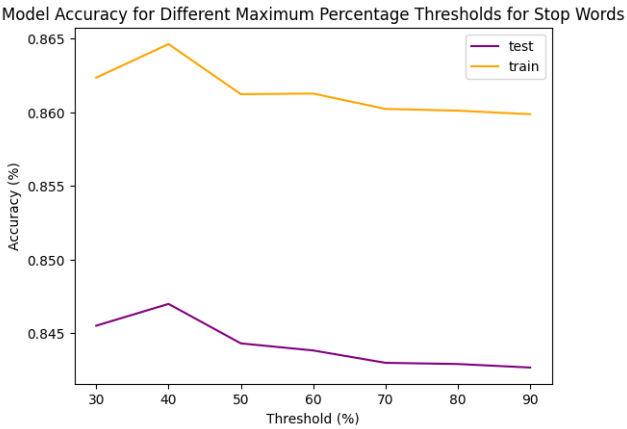

Fig. 3: Feature Selection By Eliminating Frequent Words

As shown in the figure above, this feature selection produces a very slight improvement on model performance, with the highest results occurring at a threshold of 40% (only 0.3% more accurate than a threshold of 90%). This feature selection does not add any regularization to the model, as the training and test curves nearly mirror each other and are always separated by the same distance. This is likely because at this threshold, only 42 more words are removed from the model's vocabulary, lowering the total number to 5,415.

One potential downfall of the previous feature selection methods is their failure to distinguish between positive and negative reviews since they simply measure overall frequency. As such, we performed additional experiments to determine whether eliminating words that occurred at similar frequencies in instances of each review class improved accuracy. We accomplished this by splitting features into frequency 'buckets' for positive and negative data. Any words contained in the same bucket for each class were added to an additional stop list, on top of the set excluded in the previous section. The size of these buckets was a tunable hyper-parameter; the larger the bucket, the more words were eliminated, since larger classes resulted in increased overlap. The most effective bucket size was 0.75%, as visible in Table II.

It is worth noting that due to computational limitations, each of our frequency buckets had a "hard margin", i.e., words were simply classified in the bucket for which they fell within the boundaries. This meant that a word with positive/negative frequencies of 1.1% and 1.9% would have been eliminated with a bucket size of 1%, but not a word with positive/negative frequencies of 0.99% and 1.01%,

TABLE II: Effect of Bucket Size in Common Word Feature Selection

Hyper-parameter (*bucket size*)	Model Information	
	Features	***Test Accuracy (%)***
0.25%	3867	84.9
0.5%	2666	85.2
0.75% (optimal)	1900	85.3
1%	1429	85.1
2%	708	83.5

despite it being more similar between classes. Eliminating based on frequency difference thresholds would likely have further improved our results, but such experiments exceeded the hardware limitations of our environment.

Nevertheless, a noticeable improvement is visible with this type of feature selection implemented. As a result, all experiments in Section III are performed with data with 1900 features, where words occurring in less than 0.25% or more than 40% of instances, as well as within 0.75% frequency 'buckets', are eliminated.

B. BERT Pre-processing

The ALBERT model was pre-trained on BOOKCORPUS and Wikipedia, like the original BERT model [1]. We assumed the pre-training provided a sufficient foundation for the transfer learning exercise on the IMDB dataset. Analysis of this assumption will be investigated in later sections.

Similar to the original BERT model, the ALBERT model expects numerical input. TensorFlow Hub's ***albert_en_preprocess*** preprocessing model allows to transform movie reviews into the three tensors the model expects: ***input_word_ids***, ***input_mask***, and ***input_type_ids*** [7]. Each tensor has the same shape, i.e. the batch size, which is 32, by the maximum token length, which is 128 by default [7]. Each text sequence starts with a 'CLF' token and ends with a 'SEP' token. The ***input_word_ids*** contain token ids that map every word to a unique id. The tensor of token ids is padded with 0s if the length of the text sequence is less than 128. The ***input_mask*** contains 1s for every token of the text sequence and 0s otherwise. The ***input_type_ids*** assigns an id for each sequence in the input, allowing for positional encoding [9].

III. Results

After pre-processing the dataset and training the Naive Bayes model and the BERT transformer network, we can analyse their results. First, smoothing parameters for Bayesian and Multinomial Naive Bayes are investigated, to find an optimal value for α and β. Then, the Normal, Bayesian and Multinomial Naive Bayes methods are compared to determine the best one.

For the ALBERT model, training is performed as outlined in the original BERT paper. The fine-tuning procedure consists of Adam optimizer with L2 weight decay, a warm-up phase, and linear decay of the learning rate. The batch size is 32, the initial learning rate is 3e-5, and 5 epochs are performed, one more than prescribed [3]. Hyperparameter tuning is performed for the learning and dropout rates. Since the training is rather long, cross-validation is not performed, but the best parameters on the validation set are kept, hoping to increase the test accuracy. After analysing each model separately, we can compare the two models and discuss each of their results in relation to each other as well as examining their attention matrix for some movie reviews.

A. Naive Bayes Smoothing Parameters

Both the normal and multinomial Naive Bayes models have the option to introduce smoothing. To find the best values of the smoothing parameters α and β, we assume that they would always be equal like in Laplace smoothing, due to time and computational constraints. We compare how different values of α and β affect the test accuracy of a Bayesian Naive Bayes model. When this is done on data with no feature selection, smoothing drastically increases accuracy. Without it test accuracy is 65%, increasing to 82.6% when α and β are 1. However, no feature selection makes this experiment computationally costly. Its results are shown in Figure 9 in the appendices.

When α and β values are compared over a large range on the feature selected data (see appendices Figure 10), smoothing seems to very slightly decrease model performance. This is because when features are very rare and particular to certain documents, it is important for smoothing to be included so that rare features do not distort the likelihood probabilities. However, when these features are eliminated, smoothing is unnecessary. Despite this, the large range experiment hints that values of α and β between 0 and 1 might give the best results. We can thus perform a more specific experiment in this range, shown in Figure 4 below, to conclude that the value of α and β has next to no effect, but at values of 0.2 the results are imperceptibly better (85.292% vs. 85.288% without smoothing).

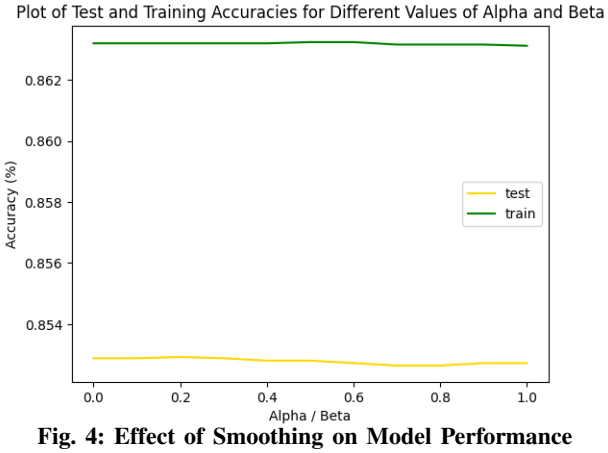

Fig. 4: Effect of Smoothing on Model Performance

B. Comparing Naive Bayes Models

Now that we know the optimal ways to perform feature selection and smoothing, we can compare all our Naive Bayes Models. We implement two main Naive Bayes models - Normal Naive Bayes, based on a Bernoulli probability

distribution, and Multinomial Naive Bayes, based on a multinomial probability distribution [10]. Both of these models have the option to add smoothing, making the Normal Bayes model Bayesian instead. We can compare these two main classes of models with and without smoothing to determine which one has the best performance. All comparisons are done with the optimal feature selected dataset, and α and β values of 0.2 where applicable, and are shown in Table III below:

TABLE III: Comparison of Naive Bayes Models Performance

Naive Bayes Model	Model Accuracy (%)	
	Training	***Test***
Normal	86.32	85.28
Bayesian	86.32	85.29
Multinomial	86.2	84.50
Mutlinomial with Smoothing	86.2	84.50

As can be seen in the table above, all models have extremely similar performance due to the extensive feature selection we performed, but the Bayesian Naive Bayes has the highest test accuracy of 85.3%. This will be the model we use to compare performance with BERT in further analysis.

C. ALBERT Learning Rate

For fine-tuning, BERT authors recommend keeping the best learning between 5e-5, 3e-5, 2e-5 [3]. The three values are used to train the model using the same setup as previously described. Accuracy on the validation set is used to determine the best learning rate for our implementation. As shown in Figure 5, 2e-5 is the most stable value throughout epochs and yields the highest validation accuracy after 5 epochs, 88.8%. For this reason, subsequent tests will use 2e-5 as the learning rate for training.

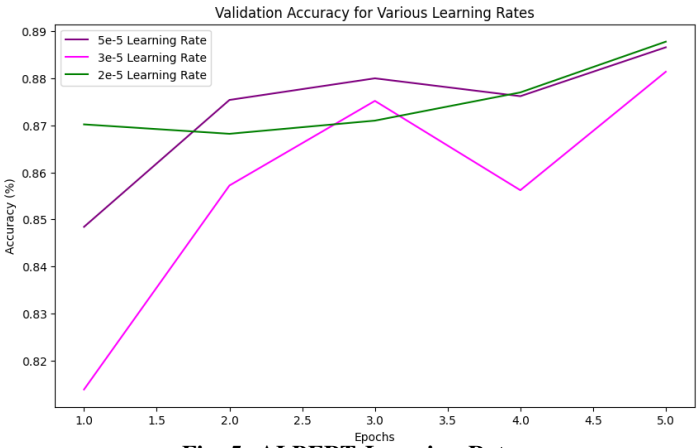

Fig. 5: ALBERT Learning Rate

D. ALBERT Dropout Rate

It is known that "dropout is found to be a good regularizer in text classification" [11]. Since the BERT is pre-trained with a dropout rate of 0.1 on all layers [3], different dropout rates are tested by modifying the dropout layer before the dense classification layer (see Figure 1). Unfortunately, we could not find a way to alter the dropout rate of the ALBERT model since it was loaded and pre-trained.

The ALBERT authors found success in removing dropout [1]. The original BERT model is set with a dropout rate of 0.1 for all layers [3]. Some recommended dropout rates for text classification are between 0.4 and 0.7 [11]. Those references indicate that different dropout rates may improve performances. Following the values mentioned in the previous papers, dropout rates are modified on the classification layer from 0% to 70% in increments of 10%. Figure 6 reports the validation accuracy for the different dropout rates for a training with 5 epochs and 2e-5 as the learning rate. Since we only modify the dropout rate on a single layer after the output of the model, values are clustered together. Nevertheless, we can say that for our implementation, 10% is the best value, leading to the highest accuracy on the validation set.

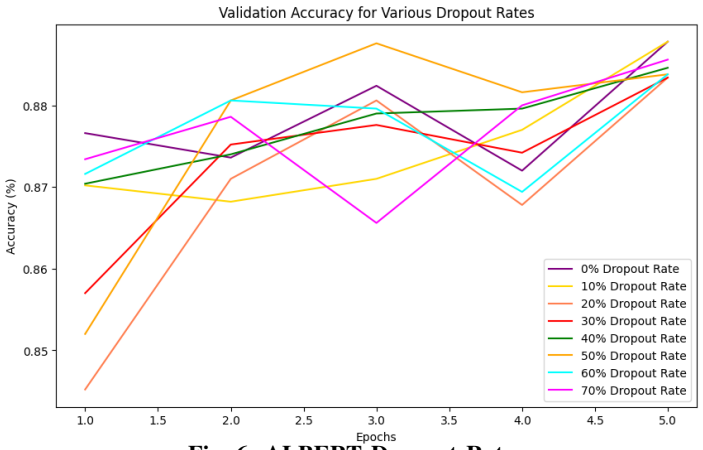

Fig. 6: ALBERT Dropout Rate

E. Examining the Attention Blocks

The attention matrix between tokens can help understand "how much a particular word will be weighted when computing the next representation for the current word", i.e. it can show how the model learns about the English language [12]. In an attempt to see how ALBERT weighs the words of a movie review, the attention matrix for each of the twelve heads are output for the last attention layer for a correctly and an incorrectly classified review. To do this, another model with the ability to output this matrix was trained, with the training environment as similar as possible to our first implementation. Figure 7 shows the attention matrix of the eighth head of last attention layer for a correctly classified negative review. Attention seems to accumulate on the diagonal and on the 'SEP' token column. Furthermore, along the 'CLS' class token row, attention for the word 'poor' is higher, suggesting the model put more weight on this word for its negative classification. Also, along the first row, attention for punctuation tokens is higher, supporting that attention captures "syntactic information" [12]. Figure 11 in the Appendix shows attention matrices for the twelve heads of the last attention layer for

an incorrectly classified negative review. The 'CLS' rows do not have clear patterns unless for punctuation tokens. It is worth noting the word 'hope' has higher attention for few heads, potentially misleading the prediction to be positive. This noisy behavior suggest attention may not be a way to consistently explain predictions, but rather be suited to better understand the model outputs [12].

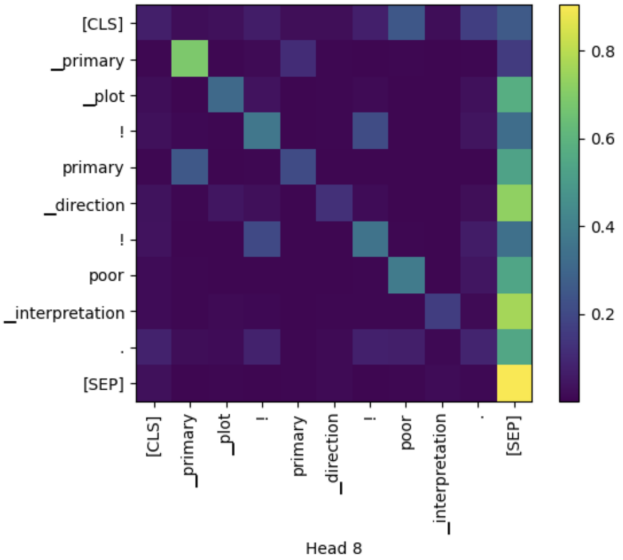

Fig. 7: Attention Matrix for Correctly Classified Review

F. Comparing Naive Bayes and BERT

Table IV compares the accuracies of our best-performing Naive Bayes and pre-trained BERT models for the reviews classification task. As discussed in previous sections, the BERT transformer model performs better overall, despite a strong performance from our Naive Bayes version.

TABLE IV: Comparison of Naive Bayes and BERT Performance

Optimal Sentiment Classification Model	**Accuracy (%)**	
	Training	***Test***
Naive Bayes	86.32	85.29
BERT	98.72	88.78

The small difference in accuracy between the models (compared to the sharp drop-off in models without feature selection and smoothing) is a testament to the importance of choosing the correct parameters for a task with such a large number of parameters. Thanks to its high accuracy (less than 4% below that of the BERT model) and incredible simplicity, Naive Bayes (with appropriate feature selection) may be an ideal choice for sentiment classification when available resources are limited or clarity and understanding of the model are a priority, and accuracy is not critical.

IV. Discussion & Conclusion

After comparing the performance of a Naive Bayes model and a BERT transformer model on a common sentiment classification task, several takeaways can be made.

A. Pretrained Models for Movie Review Classification

ALBERT is pre-trained on BOOKCORPUS and Wikipedia with 2 tasks: Masked Language Model (MLM) and Sentence-Order Prediction (SOP) [1]. MLM consists of randomly masking input tokens and predicting them given only the context [3]. SOP is an improvement from the Next Sentence Prediction (NSP) task originally used on BERT and consists of "modeling inter-sentence coherence" [1]. Pre-training allows ALBERT to learn about English words and possible word sequences therefore improving performances of downstream tasks after fine-tuning. To further analyze the impact of fine-tuning on ALBERT for the movie review classification task, the 11,683,584 ALBERT parameters are frozen and only the top layer is trained. Training, validation, and test accuracy stay around 50%, indicating a random classifier and further supporting the importance of fine-tuning. Pre-training seems to lay the foundation for specific downstream tasks, but only fine-tuning allows to reach adequate performance.

B. Deep Learning vs. Traditional Machine Learning

Various similarities and differences arose upon observation of the juxtaposed models. First, both techniques produced admirable test results with accuracies above 80%, indicating their suitability for sentiment classification of textual data. As may be expected, the deep learning model (BERT) produced a higher accuracy percentage than the traditional probabilistic method (Naive Bayes). However, becasue of the gulf in computation power and understanding required to use a deep learning model like BERT, the increase in accuarcy of around 4% could be considered minimal, raising an important question: how far should we go for marginal performance improvements? Depending on client priorities, the small increase in accuracy may not be worth the vast difference in complexity between the simple and advanced models. This project serves as a valuable reminder that older methods should not be overlooked, as they allow for otherwise necessary resources to be distributed elsewhere in a task.

Moving forward, our team is considering analyzing additional modifications to the work completed. Increased feature selection on new criteria such as general language frequency or repetition within reviews might assist in prioritizing more meaningful words in the Naive Bayes version's likelihood probabilities. Current feature selection techniques could also be improved to eliminate their weaknesses, and data could be pre-processed to correct as many typos as possible. For the BERT model, it would be interesting to further investigate the effects on dropout rate on all layers. Unfortunately, changing the dropout rate for a single layer after the model was not enough to infer on the best value for the classification task, and further research would be required.

C. Statement of Contributions

All group members contributed equally to the project and to the report, and were familiar with all parts of code.

References

[1] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," arXiv preprint arXiv:1909.11942, 2019.

[2] "Classify text with BERT — Text," TensorFlow. https://www.tensorflow.org/text/tutorials/classify_text_with_bert?fbclid=IwAR0FpVibHdeT7yq_GHjdXrXA4F5JJtCTB1lUwIFd6oZ_KNx99aVhTTzv7us (accessed Apr. 14, 2023).

[3] J. Devlin, M. Chang, K. Lee, K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,"

[4] T. A. Dorfer, "Why Simple Models Are Often Better," Medium, Jan. 26, 2023. https://towardsdatascience.com/why-simple-models-are-often-better-e2428964811a (accessed Apr. 08, 2023).

[5] G. Nkhata, "Movie Reviews Sentiment Analysis Using BERT," University of Arkansas, 2022.

[6] M. Mahyarani, A. Adiwijaya, S. Al Faraby, and M. Dwifebri, "Implementation of sentiment analysis movie review based on imdb with naive bayes using information gain on feature selection," in 2021 3rd International Conference on Electronics Representation and Algorithm (ICERA), 2021: IEEE, pp. 99-103.

[7] "TensorFlow Hub," tfhub.dev. https://tfhub.dev/tensorflow/albert/_en/_preprocess/3 (accessed Apr. 14, 2023).

[8] "TensorFlow Hub," tfhub.dev. https://tfhub.dev/tensorflow/albert/_en/_base/3 (accessed Apr. 14, 2023).

[9] P. Shukla, "Pratically Demistifying BERT Language Representation Model," Analytics Vidhya, Aug. 28, 2021. https://www.analyticsvidhya.com/blog/2021/08/pratically-demistifying-bert-language-representation-model/ (accessed Apr. 14, 2023).

[10] R. Rabbany, "Applied Machine Learning," in McGill Computer Science: COMP 551, 09-APR-2023.

[11] T. Semwal, P. Yenigalla, G. Mathur, and S. B. Nair, "A practitioners' guide to transfer learning for text classification using convolutional neural networks," in Proceedings of the 2018 SIAM international conference on data mining, 2018: SIAM, pp. 513-521.

[12] K. Clark, U. Khandelwal, O. Levy, and C. Manning, "What does BERT look at? An Analysis of BERT's Attention." Accessed: Aug. 01, 2022. [Online]. Available: https://aclanthology.org/W19-4828.pdf

Appendix

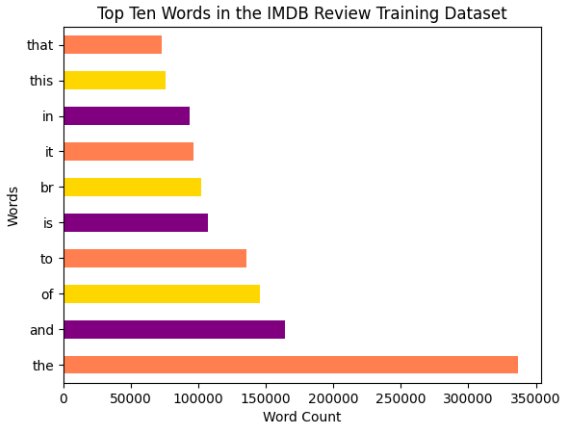

Fig. 8: Top Ten Words in the IMDB Review Training Set

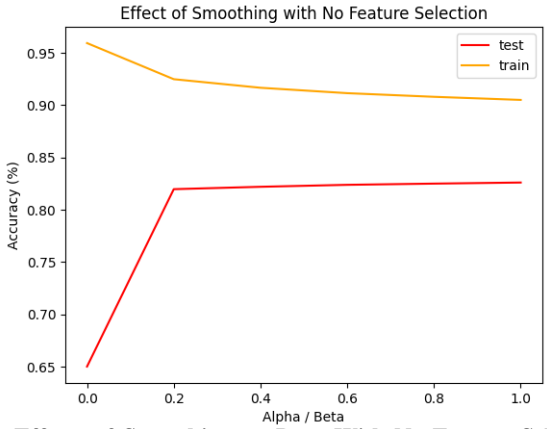

Fig. 9: Effects of Smoothing on Data With No Feature Selection

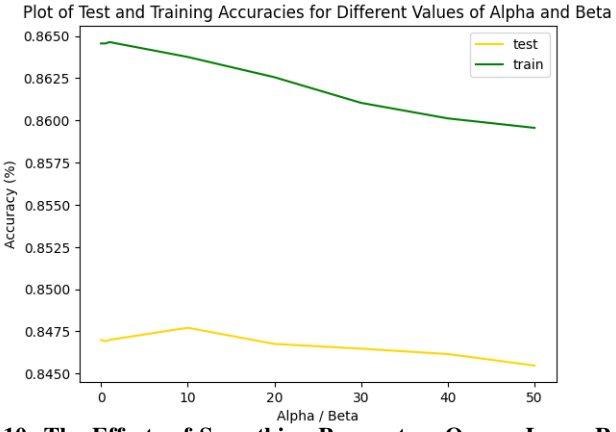

Fig. 10: The Effects of Smoothing Parameters Over a Large Range

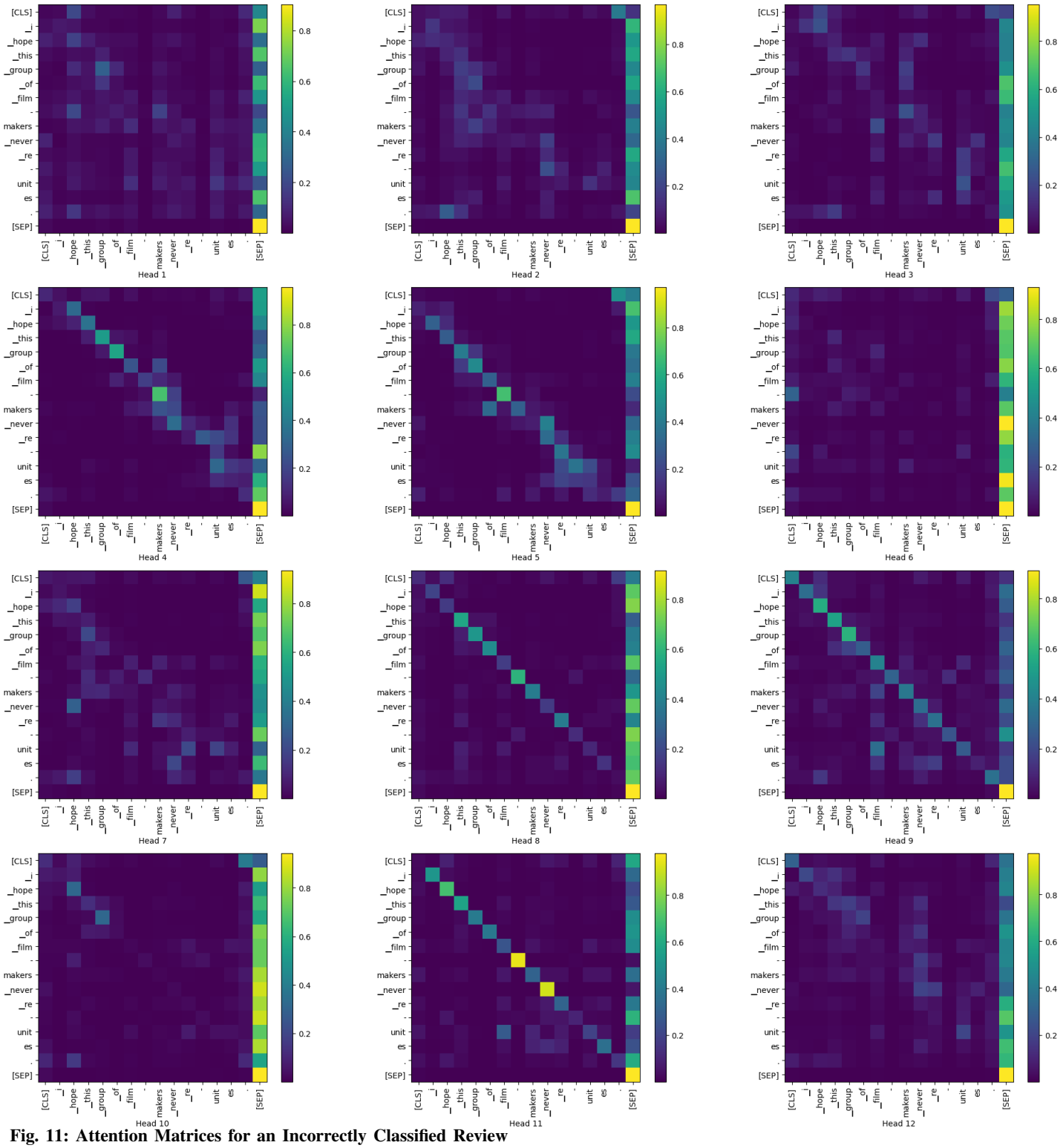

Fig. 11: Attention Matrices for an Incorrectly Classified Review