Assignment 4 Comp 551

Angad Verma, Angela Moskal, Kai Turanski

December 9, 2024

Abstract

In this assignment, we implement and finetune a Naive Bayes classifier, Bidirectional Encoder Representations from Transformers (BERT), Generative Pre-trained Transformers (GPT2), and baseline models for classifying the GoEmotions dataset. The Naive Bayes classifier achieved a test accuracy of 53.0% while BERT and GPT2 achieved test accuracies of 61.6% and 61.0% respectively. Collectively, our three models outperformed the Random Forest, XGBoost, SGDClassifier, and ComplementNB baselines, which did not manage to exceed 51.0% accuracy. Our results from training the two LLMs illustrate the importance of layer selection for training and the versatility of BERT models. In addition, our analysis of attention heads makes the LLM classifiers more interpretable.

1 Introduction

In this assignment, we implement and finetune a Naive Bayes classifier and two Large Language Models (LLMs)[1, 2] for classifying the dataset GoEmotions - a corpus of 58k comments extracted from Reddit, with human annotations to 28 emotion categories. As baseline models, we implement Random Forest, XGBoost, SGD and ComplementNB, but never exceed an accuracy of 51%. For a Naive Bayes classifier, we use a bag-of-words representation and finetune the word count vectorizer thresholds, achieving an accuracy of 51.4%. As extra experiments, we add Laplace smoothing and normalization, further boosting the accuracy to 53.0%. For the LLM, we use BERT, achieving a test accuracy of 61.6%, outperforming existing BERT-GoEmotions classifiers[3]. As an extra experiment, we also implement GPT2 and achieve a test accuracy of 61.0%.

2 Methods

2.1 Data preprocessing and exploratory analysis

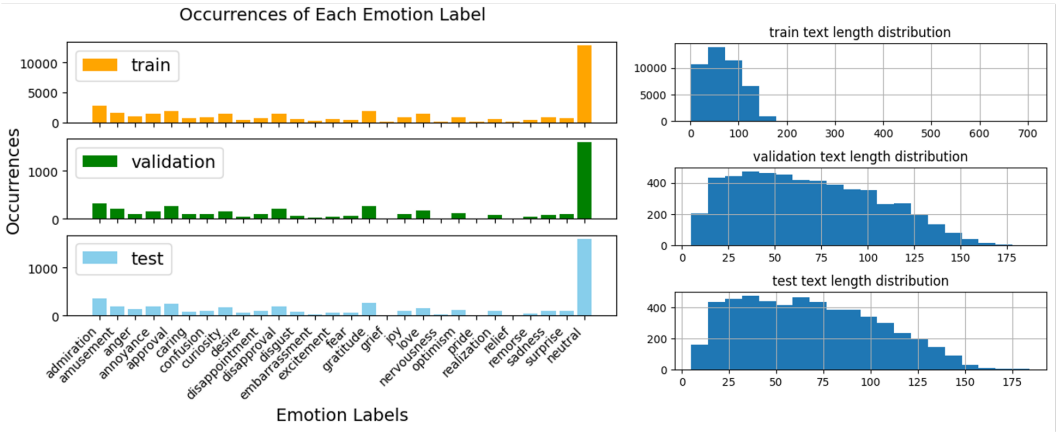


Figure 1: Distribution of emotions and text length in the GoEmotions dataset. There is stark imbalance in emotion representation.

2.2 Implementing the Naive Bayes algorithm

The Naive Bayes classifier first fits to the training data by calculating the prior (equation 1) and the likelihood (equation 2). We include a laplace smoothing parameter α, and a normalization factor, β in equation 2. β normalizes the likelihood to account for the fact that some classes may have more documents than others, preventing overfitting.

$$P(c) = \frac{\text{Number of samples in class } c}{\text{Total number of samples } N} \tag{1}$$

$$P(x_d = 1|c) = \frac{\text{Count of } x_d = 1 \text{ in class } c + \alpha}{\text{Total number of samples in class } c + \beta} \tag{2}$$

Applying Bayes' theorem and assuming conditional independence, we calculate the log-likelihood (equation 3) and predict the label by taking the argument that maximizes the log-likelihood.

$$\log P(c|X) \propto \log P(c) + \sum_{d:x_d=1} \log P(x_d = 1|c) + \sum_{d:x_d=0} \log(1 - P(x_d = 1|c)) \tag{3}$$

2.3 Implementing the LLM

Our Bidirectional Encoder Representations from Transformers (BERT) model was built using the pre-trained *bert-base-uncased* model from HuggingFace library, with a classification head added to the end. The dataset was preprocessed using the associated *tokenizer* model on all instances with only one emotion label. To achieve optimal performance, we experimented with training different subsets of layers and using different hyperparameters 2. In the end, we found that training the last encoder layer, the pooler, and classifier with a learning rate of $1e-3$, weight decay of 1, and batch size 32, over 8 epochs led to the best performance.

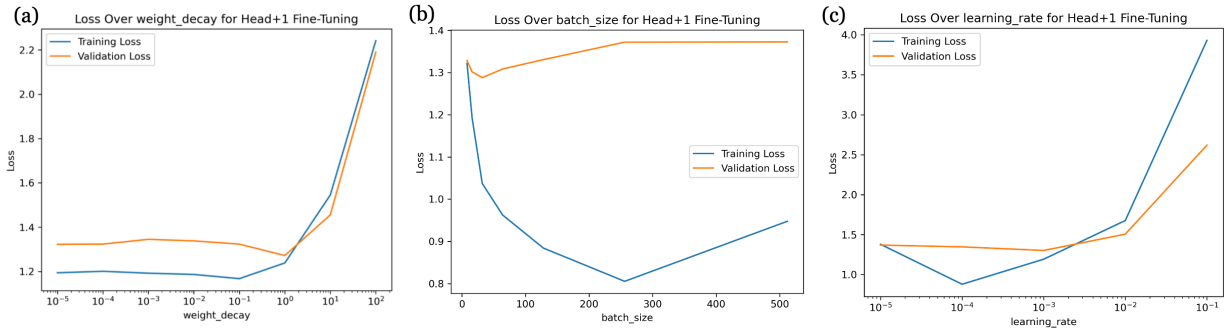


Figure 2: (a,b,c) Shows the parameter searching over weight decay, batch size, and learning rate, respectively.

3 Results

3.1 Baseline methods

First, we test the performance of several baseline models, which we will go on to compare to the Naive Bayes model and LLM in later sections.

Table 1: Performance of baseline models

Model	Settings	Accuracy	WeightedF1
Random Forest	estimators=100, criterion=gini, rest default	51%	0.34
XGBoost	estimators=100, loss=log_loss, rest default	51%	0.35
SGDClassifier	loss=hinge, penalty=l2, rest default	51%	0.34
ComplementNB (better for imbalanced classes)	default	42%	0.31

3.2 Naive Bayes

Filtering out words that are overly rare or overly common:
In our bag-of-words representation, we tune the upper and lower bound for which words are to be encoded. The optimal minimum threshold for word frequency is 0.16%, achieving a validation accuracy of 50.1%. The optimal maximum threshold for word frequency cut is 6.13%, further increasing the validation accuracy to 51.4% (see figure 3). We also see that if we don't apply a sufficiently rigid threshold, the model has a tendency to overfit, as evidenced by the high training accuracy in figure 3.

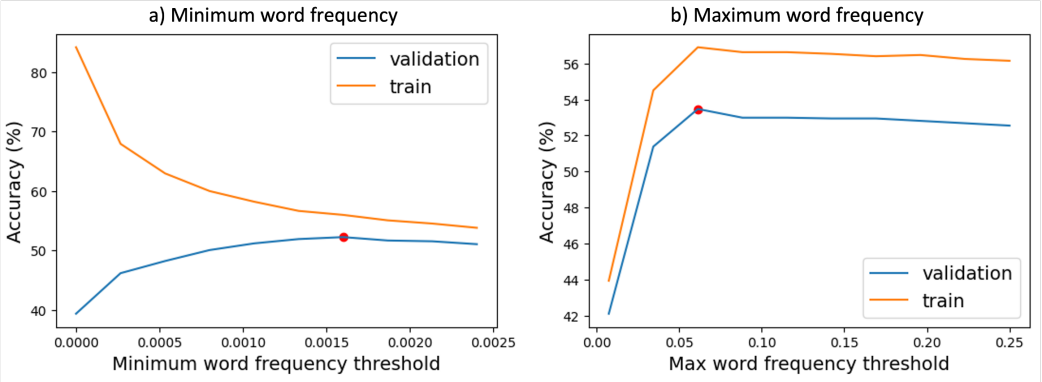


Figure 3: a) Tuning the lower bound for filtering out rare words. b) Tuning the upper bound for filtering out overly common words.

Laplace smoothing (Extra):
Tuning α from equation 2, we get that $\alpha = 1$ is optimal, achieving a validation accuracy of 53.8% (see figure 4).

Normalization (Extra) :
Tuning β from equation 2 in a grid search, we now get that α=2, β=464 are optimal, achieving a validation accuracy of 55.0% (see figure 4).

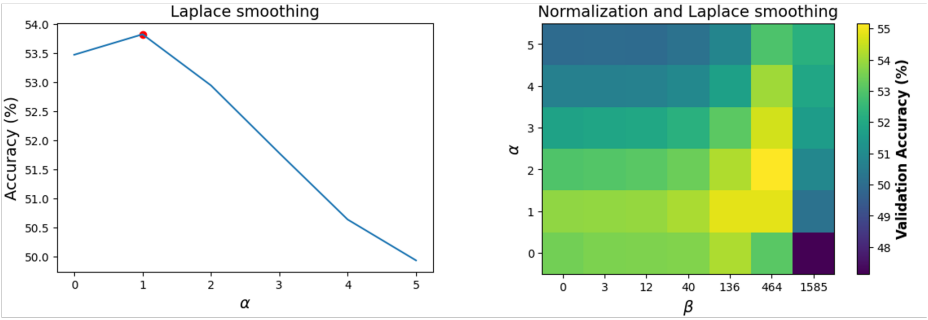


Figure 4: Tuning laplace smoothing and normalization parameters α and β (see equation 2)

Final test accuracy:
The model with the highest validation accuracy has the following hyperparameters: min word frequency = 0.16%, max word frequency = 6.13%, α=2, β=464. The overall test accuracy for this Naive Bayes classifier is 53.0%, correctly predicting as much as 83.0% of instances for common emotion classes such as gratitude, but failing to predict instances of rare classes such as grief (see figure 5).

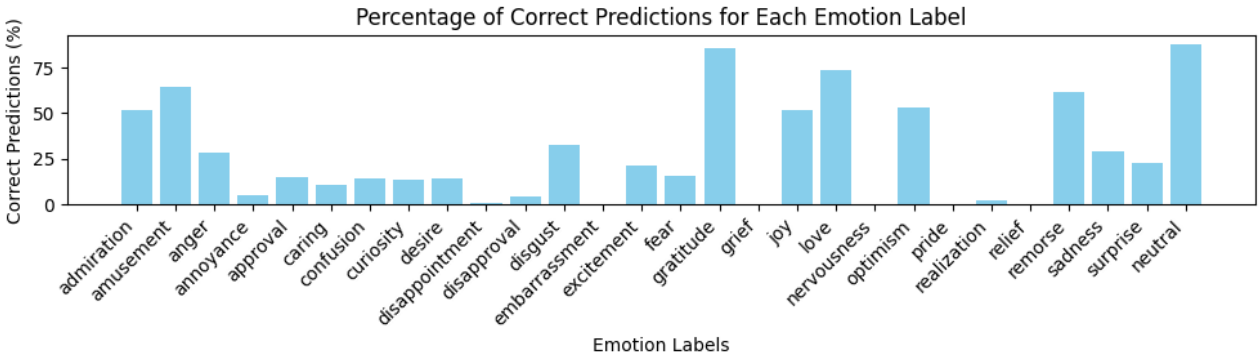
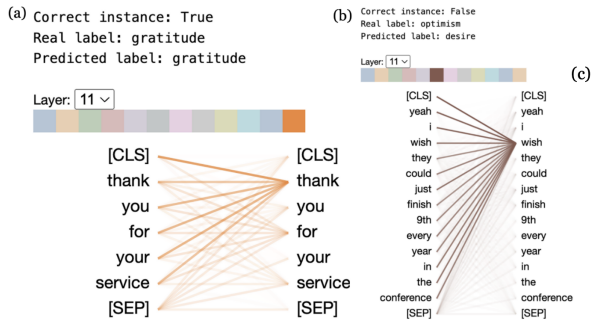


Figure 5: Percentage of correctly labeled test data for each emotion using the Naive Bayes classifier with optimally tuned hyperparameters. The overall test accuracy is 53.0%.

3.3 LLM

BERT:

Using the training methods described in 2, we compiled the accuracy and F1 scores of a variety of settings to compare with the GPT2 model 6. Additionally, we visualized the final attention layer of the best BERT model to gain intuition on its predictions 6 8. Output tokens with high attention scores across all input tokens seem to correlate significantly with final predictions. Despite the model's accuracy, some emotions were significantly under-reported 7. Still, the model achieved a higher macro F1 score than a similar BERT model trained on GoEmotions [3], reaching 0.51 while the other model reached 0.46.



(c)

Model	Layers trained	Accuracy	Macro F1
GPT2	Classification head	46%	0.13
	All layers	61%	0.49
BERT	None	1%	0.01
	Classification head	46%	0.19
	Layer 11, pooler, & classifier	62%	0.51
	All layers	59%	0.51

Figure 6: (a) The attention head shows a high focus on the work "thank" as it correctly classifies the sentence under "gratitude." (b) The head focuses on "wish" as expressing desire, predicting the wrong label. (c) The performance of our two LLM models by trained layers.

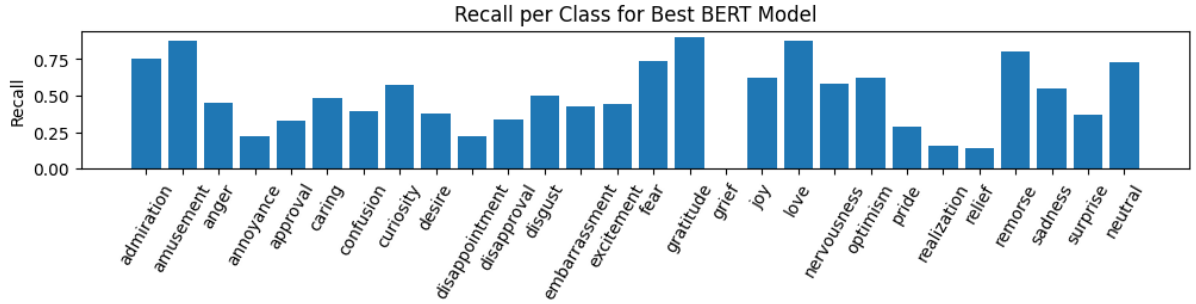


Figure 7: Recall on test data using best BERT classifier. The overall test accuracy is 61.6%.

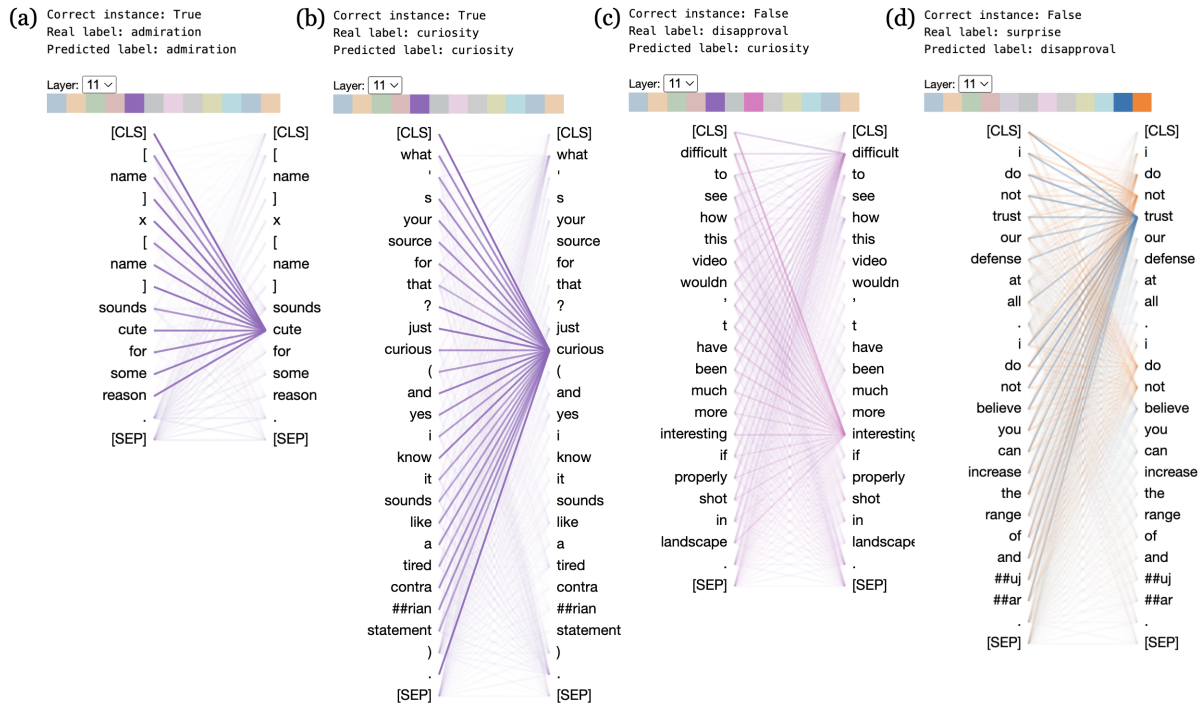


Figure 8: More visualizations of the BERT attention matrix. (a, b) are correctly classified, while (c,d) are not. Tokens on the right side with many connections can be understood as the key words the model used to inform its classification decision for that sentence.

GPT2 (Extra): We implement a GPT2 model and report the validation accuracy after 3 epochs. The pre-trained model freezes all transformer layers and only trains the classifier head, achieving an accuracy of 46% while the fine-tuned model trains 12 transformer layers and the classifier head, achieving an accuracy of 61% (see Table in figure 6).

4 Discussion and Further Improvements

This project evaluated Naive Bayes and Large Language Models (BERT and GPT2) for classifying emotions in the GoEmotions dataset. Naive Bayes was able to outperform our baselines with an accuracy of 53.0%, despite the inherent difficulty in capturing complex, imbalanced emotion categories from bag-of-words representations alone. BERT and GPT2 outperformed Naive Bayes, with accuracies of 61.6% and 61.0%, demonstrating the effectiveness of transformer models for understanding emotional expression. Additionally, the models outperformed a BERT classifier trained on the same data[3], with their F1 scores of 0.51 and 0.49 beating the other model's 0.46. However, rare emotions like grief were often misclassified, while common ones like gratitude were better predicted, emphasizing the need for techniques like class weighting or data augmentation.

Pre-training greatly benefitted the LLMs, as it provided a strong foundation for fine-tuning by leveraging large-scale semantic patterns. Fine-tuning specific layers further optimized training efficiency while maintaining accuracy. Further, visualization of the BERT attention heads improved interpretability by showing how models focused on contextually significant words. Overall, this work shows that attention and encoder mechanisms are key to machine-learning sentiment analysis methods.

In the future, these models could be improved through the use of class-weights, greater experimentation with regularization methods, and exploration of the models' performance on other sentiment analysis datasets. Such research would improve our understanding of the models' generalizability and make them more robust, in order to handle a wide variety of contexts.

Statement of contributions: All team members independently completed the entire assignment, including data preprocessing, model implementation, and experiments.

References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," in *OpenAI*, 2019.

[3] D. Demszky, D. Movshovitz-Attias, J. Ko, A. S. Cowen, G. Nemade, and S. Ravi, "Goemotions: A dataset of fine-grained emotions," *CoRR*, vol. abs/2005.00547, 2020.