COMP 551 Assignment 4: Classifying Emotion with LLMs

Ayaz Ciplak, Jacqueline Silver, Christopher Smith

Group 10: Ayaz Ciplak - 261013785, Jacqueline Silver - 261012002, Christopher Smith - 261013926

Abstract

This project involved implementing and analyzing the performance of multiple models for language classification tasks on the GoEmotions dataset. The objective of this project was to gain experience working with both deep learning libraries and traditional methods, while evaluating and comparing their performances. The implemented models include a naive Bayes model, a large language model (LLM), a fine-tuned version of this LLM, and a baseline model. After selecting each model's hyperparameters and preprocessing the dataset respectively, we evaluated their performance in classifying emotions within the GoEmotions dataset. The experiments conducted on these models included evaluating their accuracy on the task, examining the resulting attention matrices, and other experiments for each specific model. These explorations demonstrated how different models behave distinctly within language classification tasks.

Introduction

Machine learning models like LLMs, naive Bayes, and decision trees are all valuable tools for classification in language processing tasks. In particular, emotion classification of text plays an integral role in user interaction across numerous platforms and industries (Stigall et. al, 2024). This project utilizes the following models: a pre-trained BERT base model (uncased), an adjusted version of this BERT model fine-tuned on the GoEmotions dataset, a multinomial naive Bayes model, and sklearn's implementation of the Random Forest classifier as a baseline. Each of these models is trained on existing classifications and subsequently predicts classifications of further inputs using their respective algorithms.

The GoEmotions dataset, available through Hugging Face datasets, consists of Reddit comments, each assigned a corresponding label from a set of 27 distinct emotions. GoEmotions contains 58,000 Reddit comments and is the largest human annotated dataset (Demszky et. al, 2020). They also acknowledge the biases that exist within the dataset such as inherent biases, vulgar word use, and a narrow demographic of annotators (Demszky et. al, 2020). They used a thorough labeling process ensured consistency throughout the labels, which consequently improves the training of our models.

Throughout our explorations, we identified optimal hyperparameters and conducted multiple experiments to fine tune settings and assess model behavior. Overall, this project allowed us to gain valuable insights into the different aspects of language processing and allowed us to gain experience working with LLMs and other classification models.

Methods

Preprocessing

General preprocessing included dropping comments without one label, flattening each label array into a single numerical value, removing any duplicate rows, and dropping emojis comments. Since the proportion of comments using emojis was very low (sub-2% in each subset), and the removal of these comments didn't significantly impact the label distribution, keeping these comments was not essential to the effective training of our models.

Specifically for the baseline model, Random Forest, we used the TfidfVectorizer from sklearn.feature_extraction.text to transform the data into vectors suitable for Random Forest. The naive Bayes model required the use of CountVectorizer from the same library to transform the data into a numerical, sparse representation using a bag-of-words encoding. For the LLM, BERT base (uncased), we utilized its predefined AutoTokenizer to tokenize the input datasets, and the labels were then converted into PyTorch tensors.

Exploratory Analysis

Prior to implementing and training our models, we conducted additional analysis on the underlying data distribution. We observed a disproportionate distribution of class labels, with the Neutral (27) emotion dominating significantly. Classes like grief (16), nervousness (19), pride (21) and relief (23) had very few occurrences. *Note: Results are shown below for the Training Set, but were similar in the validation and test sets.*

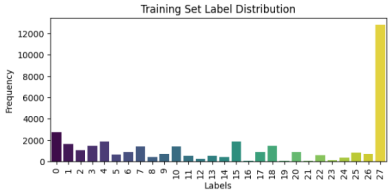


Figure 1: Class Distribution - Training Data

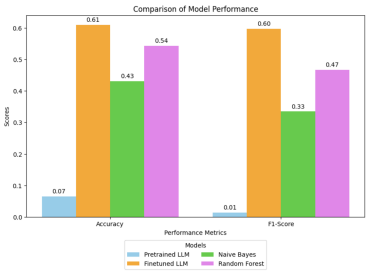


Figure 2: Comparison of Model Performances on Test Set - Accuracy and F1 Scores

Naive Bayes Model

Given the fact that our input data, represented as bag-of-words, contained frequencies for each word class rather than simple boolean values, we wanted to implement a Naive Bayes variation that captured the nature of this data. We therefore decided to implement a Multinomial Naive Bayes model, considering the continuous nature of our data.

Fitting the data: During our fit phase, we compute a categorical class prior $P(\text{class})$, given by the relative frequency of the class among the input data and representing the underlying probability of each class. We also compute the word probability $P(\text{word}_i|\text{class}_c)$, incorporating Laplace Smoothing to address problems with zero probabilities.

Making predictions: Using our prior and conditional probabilities computed in the fit function, we compute posterior probabilities given the current input data. Using the Bayes rule, $p(y \mid x) \propto p(y) \cdot \prod_{i=1}^{n} p(x_i \mid y)$.

Hyperparameter Tuning: Although there are not many tunable hyperparameters in involved in a Naive Bayes model, we incorporated a tunable Laplace Smoothing constant and wanted to explore its effects on generalization. We analysed the performance of models with varying alpha values on the validation set, and concluded that a smoothing constant of 5 was optimal for generalization (Figure A.13). Further analysis on data modification is done in Section 3.

Large Language Model: BERT Base Uncased

For our choice of LLM, we adopted the Bidirectional Encoder Representations from Transformers (BERT) Base Uncased model, first introduced by Devlin et al. (2019), due to its bidirectional architecture and pretraining on general-domain text. This pretraining approach enables the model to capture nuanced sentence-level context, which is particularly advantageous for emotion classification tasks where the meaning of a sentence depends on the subtle interactions between words, influencing the overall sentiment. In contrast, DistilGPT2, as a generative model, lacks this bidirectional understanding and is more naturally suited to text generation rather than classification tasks.

To implement the pretrained BERT model, we utilized the Hugging Face Transformers library, and added a classification-head to output predictions for the 28 emotion classes in our dataset. For the non-finetuned version of our model, we left this classification head with randomly initialized weights, thereby allowing us to leverage BERT's pretrained parameters, while introducing a task-specific output layer.

In the fine-tuning stage, we preserved the learned parameters of all the pretrained layers except for the last two transformer layers (11 and 12) and the classification head, which were allowed to update during training on our dataset. Initially, we employed hyperparameters as recommended in the original BERT paper (Devlin et al., 2019). After iterative experiments on the validation set, we identified optimal hyperparameters for our finetuned model as a learning rate of 7e-5, a batch size of 32, and a training duration of 7 epochs—beyond which the model began to exhibit overfitting.

Random Forest

We implemented the Random Forest Classifier directly from sklearn to use as a baseline model. This model uses a random forest of decision trees in order to make its predictions. Initially, the default parameters were used, but this was followed by testing out different values for n_estimators, criterion, and max_depth. Despite the exploration of these hyperparameters, we found that the default parameters are most optimal in this context anyways.

Results

Classification Performance

To evaluate the classifcation performance of our three models-Naive Bayes, BERT (both non-finetuned and finetuned), and Random Forest-we computed both accuracy and weighted F1-scores. Given the class imbalance present in the dataset, the weighted F1-score provides a more informative picture of performance than accuracy alone. For further analysis, we implemented an enhanced version of the classification report, derived from the scikit-learn library, to gain insight into class-specific metrics,

such as support, precision, recall, f1, and accuracy scores. This analysis facilitated a more comprehensive understanding of each model's strengths and weaknesses across the various emotion categories. See the comparison of results in Figure 2.

Naive Bayes: Our Multinomial Naive Bayes model, with Laplace Smoothing constant optimized for generalization, achieved a train accuracy of 48.29% and a test accuracy of 43.15%. We observe that accuracy score varies significantly by class - Those with high support, like Neutral (27), saw high train and test performances (88.95% and 89.46%, respectively), while those with lower support like Joy (17) had lower train/test performances of 13.79% and 13.19%. Although overall performance is not excellent, our model does generalize relatively well to the test set - There is relatively low discrepancy between any of the train vs. test metrics.

BERT LLM: The pretrained model, with no finetuning, achieved a relatively low accuracy and F1-score on the test set (6.56% and 1.41% respectively), indicating that the model struggles to make meaningful predictions on this dataset. This comes as no surprise given the randomly initialized classification head. The macro-average metrics (Figure A.14) further illustrate poor performance across all classes, with most scores near zero. Interestingly, the model performed somewhat consistently well with the "admiration" class in particular (an accuracy of 80.70%), potentially suggesting an inherent bias in the pretrained embeddings that align with this sentiment. Overall, the results demonstrate that the pretrained BERT embeddings alone, without finetuning, are insufficient for emotion classification tasks, emphasizing the importance of task-specific supervision to adapt the model.

After finetuning, the BERT model showed significant improvement, achieving a training set accuracy of 78.55%, a test accuracy of 60.98%, and a test set F1-score of 59.75%. The macro-average F1-score of 48.15% (Figure A.15) highlights that the model now performs reasonably well across most classes, though there are still challenges amongst the less prevalent classes. Specifically, the model performed well on higher-support classes such as "gratitude" (F1 = 0.9087) and "amusement" (F1 = 0.8189), but struggled with underrepresented emotions like "realization" (F1 = 0.1538) and "grief" (F1 = 0.0). Overall, fine-tuning drastically improved the model's ability to classify emotions, with significant gains in accuracy and F1 scores. The model was better able to leverage BERT's pretrained representations to the emotion classification task. However, while the model showcased significant improvement, there is still room to optimize the performance, specifically for underrepresented emotion classes.

Random Forest The Random Forest Classifier model produced accuracy and f1 scores of approximately 99% on the training set, while obtaining 54.33% and 46.68% respectively on the test set. This could potentially indicate an instance of overfitting; however, as seen in our hyperparameter tuning, the current configurations yielded the best results on the validation set, and therefore were the best choice for this model. The model performed most effectively when classifying the emotions "Gratitude" and "Love" but was significantly weaker for rarer emotion classes like "Disappointment", "Distress", and "Confusion". In addition, it also received a 0.0 accuracy score for the emotions "Embarrassment", "Grief", "Pride", "Relief", and "Nervousness", completely failing to classify them at all. Overall, the Random Forest model did not perform particularly well on the test set, but provided an effective baseline to compare our other models to.

LLM Attention Analysis

To gain deeper insight into how our BERT model attends to different tokens in the input text, we analyzed the attention matrices produced by the last two transformer layers and focused on heads 1 and 6. We chose these layers as they represent the most task-specific learned representations after finetuning, and selected heads 1 and 6 to capture the different attention patterns. To visualize this analysis, we represented token-wise attention from the [CLS] token using bar charts, and showcase attention across the matrix through the use of heatmaps.

The pretrained BERT model, with a randomly initialized classification head, exhibits limited capacity for meaningful attention allocation to tokens indicative of specific emotions. In both correct and incorrect examples, attention weights are broadly distributed across tokens without prioritizing those relevant to the class. For instance, in the correct prediction of admiration, the CLS token allocates significant attention to punctuation and structural tokens such as SEP, rather than words like "strong" or "locations," which are semantically aligned with the emotion. Similarly, for incorrect predictions, the attention appears random, showing no discernible pattern that aligns with either the true or predicted emotion. This lack of focus highlights the model's inability to utilize pretrained contextual embeddings for emotion classification without fine-tuning. Moreover, across layers 10 and 11, and heads 1 and 6, no specific differentiation in attention behavior is observed. The heatmaps reinforce this observation, revealing uniformly distributed attention weights, further validating that the pretrained model lacks task-specific understanding (Figures A.20-A.23).

Finetuning the model introduces significant improvements in the model's ability to allocate attention to semantically important tokens. For instance, in the correct prediction of "gratitude", the CLS token focuses heavily on "thank", assigning almost all of its attention weight to this word in layer 10, head 1. This showcases that the model has correctly associated the words "thank" and "you" with gratitude. Even in the incorrect example where "sadness" has been misclassified as "remorse", we see that the model has correctly identified the word "sorry" as semantically important, suggesting that despite the misclassification, the result most likely stems from class imbalance rather than from attention misalignment. Additionally, layer-specific attention behavior becomes more evident in the finetuned model: layer 10 lays the foundation for token associations, while layer 11 (see figures A.18 & A.19) refines these connections. Similarly, head 1 tends to concentrate on key tokens with greater precision, whereas head 6 exhibits broader attention spans, adding interpretative diversity.

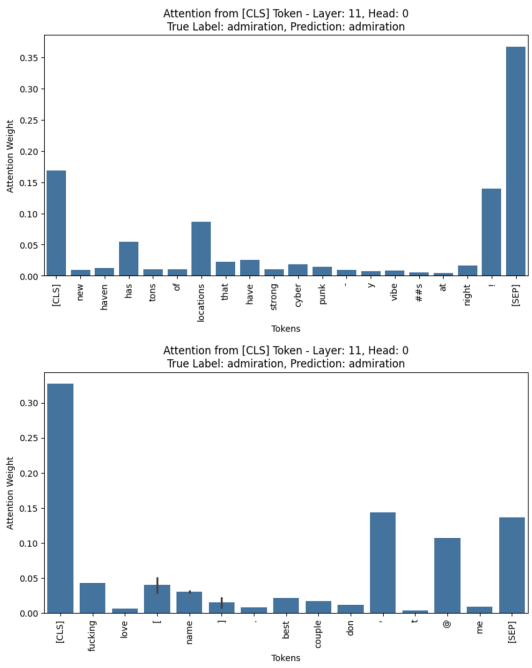


Figure 3: Pretrained Model: Correct Prediction - [CLS] token attending other tokens

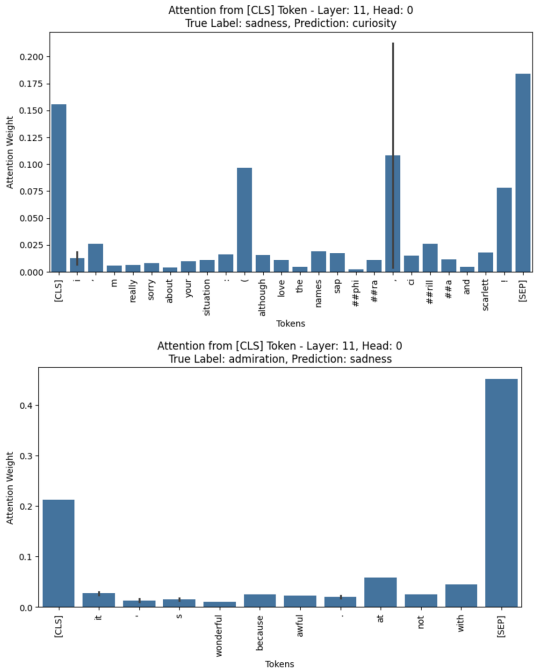


Figure 4: Pretrained Model: Incorrect Prediction - [CLS] token attending other tokens

Creative Additions

Accuracy Comparison - Pre-Implemented vs. Custom Naive Bayes

During our experimentation with a custom Naive Bayes implementation, we noticed an average performance of 45-50% on the train and test sets. Although we attribute this largely to the nature of the model and the inconsistency in class frequencies, making a comparison with a pre-implemented model can confirm our assumptions that the accuracy is model-related and not an issue with implementation. We use sklearn's MultinomialNB model, and observe results in Figure 7. Both models perform very similarly on train and test sets (although ours generalizes slightly more optimally due to hyperparameter tuning), which confirms that the sub-optimal performance of these models is due to their nature.

Naive Bayes - Vocabulary Size vs. Laplace Smoothing Coefficient as Regularization

In the hyperparameter tuning section of our Multinomial Naive Bayes model implementation, we saw how increasing the Laplace Smoothing coefficient could act as a form of "regularization", increasing bias and consequently the performance on the test set. Similarly, by testing our model on smaller vocabularies (e.g. by removing the least frequently used words), we can see whether our model benefits from training/prediction speed and overfitting reduction. We use the preimplemented MultinomialNB model from sklearn, and use dataloaders to retrieve datasets containing only words with more than 2, 5, and 10 occurrences. The train and validation accuracies are displayed in Figure 8. As we can see, vocabulary reduction works very well both as a method of "regularization" and as a method of increasing model accuracy overall! This works for several reasons. First of all, it acts as a means of noise reduction, such that we are able to filter out irrelevant, uncommon words that do not make many contributions to the predictions. This can also include potential typos. Second of all, we generalize better by preventing the model from overfitting to noise or overly specific features, and consequently observe better performance even in less common classes.

DistilGPT2 Implementation

To further expand our understanding of LLMs, we explored DistilGPT2, a lightweight version of GPT2, as an alternative to BERT for our classification task. Similarly to our pretrained BERT model, we added a classification head to predict 28 emotion classes. To finetune the model, we froze all but the last transformer layer and the classification head, preserving generative knowledge while adapting it for classification. The model was trained using a learning rate of 2e-5, a batch size of 32, and 3 epochs, with CrossEntropy loss as the objective function.

DistilGPT2's pretrained model showed poor performance, with a test accuracy of 1.44% and a weighted F1-score of 0.04%. Finetuning improved results to a 49.69% accuracy and a 42.45% F1-score, yet it underperformed compared to finetuned BERT (61.0% accuracy and 0.5976 F1-score). DistilGPT2 struggled with rare classes, frequently misclassifying them into dominant categories like "neutral." These findings highlight the superior contextual understanding of bidirectional models like BERT for classification tasks, especially when capturing nuanced emotions.

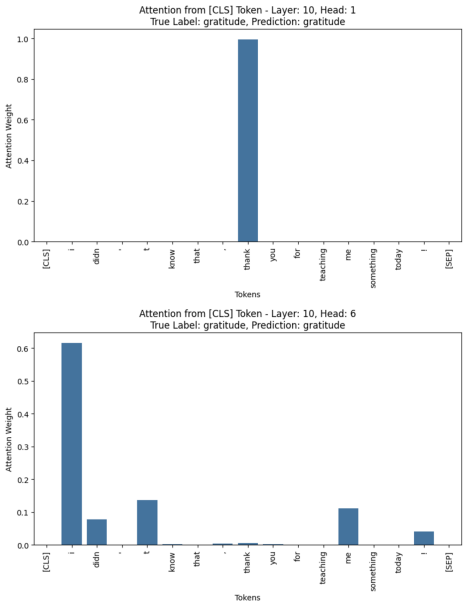


Figure 5: Finetuned Model: Correct Prediction - [CLS] token attending other tokens

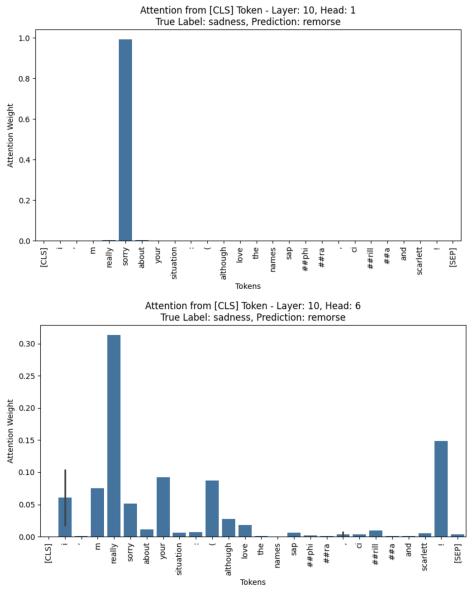


Figure 6: Finetuned Model: Incorrect Prediction - [CLS] token attending other tokens

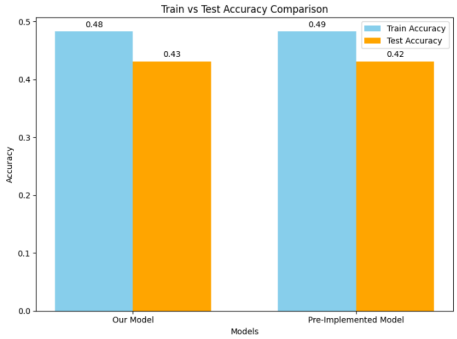


Figure 7: Performance Comparison - Custom vs. Pre-implemented Naive Bayes Model

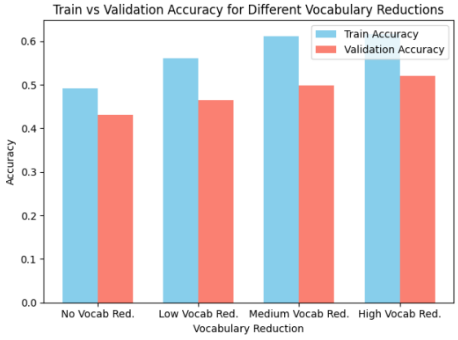


Figure 8: Vocabulary Reduction Performance - Naive Bayes

Class-weighted Loss Function & Sampling

We have seen how the imbalance in class distribution has posed significant challenges to the classification performance of both BERT and DistilGPT2 models, particularly in underrepresented classes. To address this, we wanted to expand our experiments by implementing a class-weighted loss function, as opposed to our standard CrossEntropy loss function, in order to penalize errors more heavily on minority classes, ensuring better representation in the loss gradients during backpropagation. Additionally, weighted sampling was applied during data loading, providing balanced exposure to all classes during training. The class weights were calculated based on the inverse frequency of class occurrences, normalized to ensure proportional penalties. Weighted random sampling ensured each batch included a fair distribution of underrepresented classes, mitigating the skew in training data distribution.

Despite the adjustments, the performance gains were modest (Figure 11). For BERT, the test accuracy was 37.21%, and the F1-score was 21.03%, while for DistilGPT2, the test accuracy reached 35.22%, with a lower F1-score of 18.35%. Both models demonstrated improved recognition of the most frequent class, "neutral," but struggled with the remaining 27 classes, showing marginal precision and recall. This is likely due to the extreme skew in the dataset, where minority classes remain challenging to classify even with enhanced loss and sampling strategies.

The comparison of models reveals that while weighted loss and sampling improved the models' sensitivity to minority classes, the performance remained limited, especially for DistilGPT2. Both models underscore the difficulty of achieving balanced performance in highly imbalanced datasets, highlighting the need for further data augmentation or alternative approaches like oversampling or synthetic data generation.

Implementing the Other Baseline Models

In addition to the Random Forest baseline model, in this section XGBoost and softmax regression are also implemented, using sklearn classes xgboost and LogisticRegression respectively. They were all trained on the same baseline training sets as our Random Forest classifier and evaluated on the test set. Their performances were all relatively similar. We concluded that any of these models would serve as an effective baseline model.

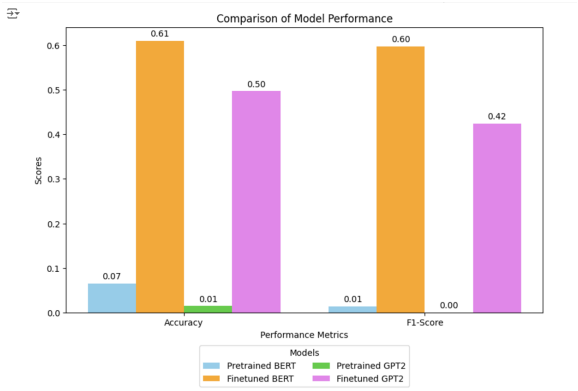


Figure 9: LLM Comparison: BERT vs. DistilGPT2

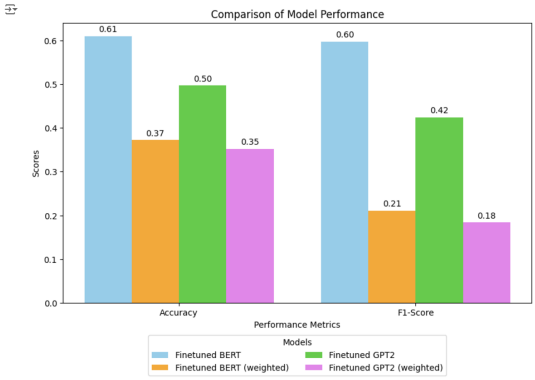


Figure 10: Model Comparison: Class-weighted Loss Function & Sampling vs. Standard CrossEntropy loss & dataloading

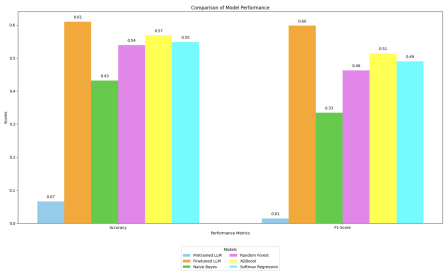


Figure 11: Model Comparison with Other Baselines

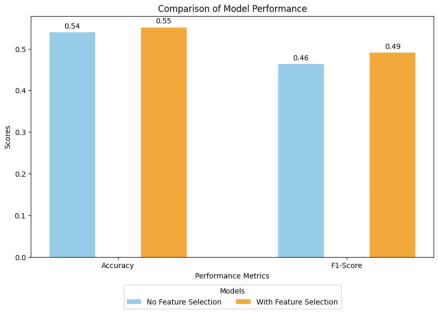


Figure 12: Feature Selection vs No Feature Selection

Feature Selection in Random Forest

To explore further optimization of our Random Forest classifier model, we tried including feature selection within the model. Feature selection allows the model to utilize only the most relevant and consistent subset of features, which can improve model performance and reduce training time.

Sklearn's implementation of the Random Forest classifier includes a method called feature_importances_, which produces the importance of each feature during training. We used this method to assess the importance of each word, and then sorted them and selected the top 1,000. The importance value of top 1,000th word was approximately 0.00012, which we then use as a threshold value for feature selection. This means that only around the top 1,000 important words were used as features within the model.

For this application, we use the SelectFromModel method from sklearn.feature_selection, which takes a threshold as an argument. By setting the threshold to 0.00012 (around the 1000th highest importance), the Random Forest model using feature selection outperformed the model that didn't. It also drastically decreases the training time of the model. Without using feature selection, the training time of the model is about 148 seconds, but this gets cut almost in half to around 77 seconds when feature selection is included. From this, we can conclude that the choice of words being used within a model for emotion classification plays a large role in its behavior and performance.

Discussion and Conclusion

Our project explored the benefits and drawbacks of using a variety of machine learning models to make predictions on data with a high and disproportionate class count. We explored methods of preprocessing word data (e.g. into a bag-of-words representation) to generate consumable input data, as well as preliminary data modification (e.g. vocabulary shrinking) and its effects on model performance. Key challenges related to our dataset were dealing with class imbalance and correctly predicting "rarer" or less-frequent emotion classes. Implementation of attention analysis and other methods provided insight on how fine-tuning models can help focus on semantically meaningful tokens, potentially improving performance on less frequent classes. Ethical concerns include mislabeling online data based on the results of our trained models, which could be a poor reflection of actual sentiment given the inherent biases in the GoEmotions dataset (especially towards more popular classes like Neutrality). Future research should focus more on developing more representative datasets and innovative techniques to address classification challenges across emotional expressions.

Overall, our project demonstrates the complex nature of computational emotional recognition, and various implementation approaches.

Statement of Contribution

This project was a collaborative effort between Ayaz, Christopher, and Jacqueline. Specifically, Task 1 was focused on by Jacqueline with contributions from Ayaz and Christopher, while Task 2 and 3 were worked on by all members, with each of us tackling one of the models and its additional experiments. Finally, the report was written by all team members.

Sources

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. GoEmotions: A Dataset of Fine-Grained Emotions. 2020. (Demszky et. al, 2020) *arXiv preprint arXiv:2005.00547 [cs.CL]*. `https://arxiv.org/pdf/2005.00547`.

William Stigall, Md Abdullah Al Hafiz Khan, Dinesh Attota, Francis Nweke, and Yong Pei. 2024. Large Language Models Performance Comparison of Emotion and Sentiment Classification. (Stigall et. al, 2024) *Proceedings of the 2024 ACM Southeast Conference (ACMSE '24)* Association for Computing Machinery, New York, NY, USA, 60–68. `https://doi.org/10.1145/3603287.3651183`

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (Devlin et al, 2019) *arXiv preprint arXiv:1810.04805 [cs.CL]*. `https://arxiv.org/abs/1810.04805`.

Appendix

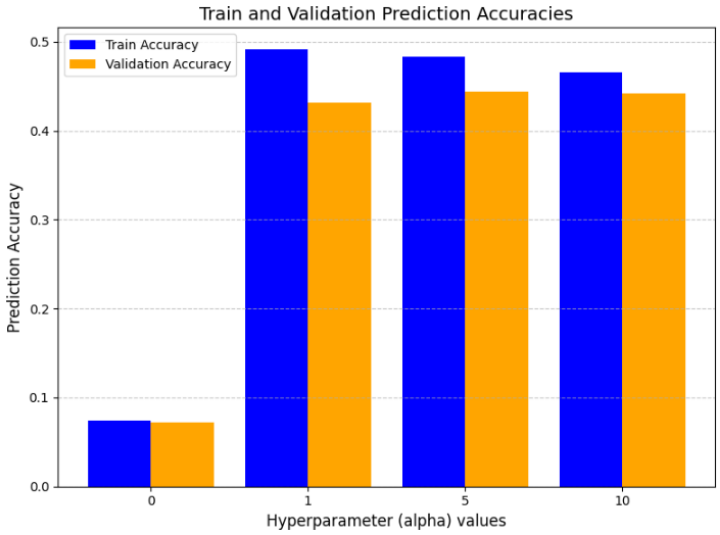


Figure 13: Laplace Smoothing Effect on Naive Bayes Model

Pretrained BERT Model Performance (without finetuning):
Test Accuracy Score: 0.0656
Test F1 Score: 0.0141

	class	precision	recall	f1-score	support	accuracy
0	admiration	0.0786	0.8070	0.1432	342.0000	0.8070
1	amusement	0.0000	0.0000	0.0000	179.0000	0.0000
2	anger	0.0000	0.0000	0.0000	130.0000	0.0000
3	annoyance	0.0000	0.0000	0.0000	192.0000	0.0000
4	approval	0.0000	0.0000	0.0000	230.0000	0.0000
5	caring	0.0000	0.0000	0.0000	86.0000	0.0000
6	confusion	0.0323	0.0208	0.0253	96.0000	0.0208
7	curiosity	0.0320	0.0398	0.0354	176.0000	0.0398
8	desire	0.0000	0.0000	0.0000	55.0000	0.0000
9	disappointment	0.0104	0.0114	0.0109	88.0000	0.0114
10	disapproval	0.0000	0.0000	0.0000	195.0000	0.0000
11	disgust	0.0000	0.0000	0.0000	75.0000	0.0000
12	embarrassment	0.0000	0.0000	0.0000	23.0000	0.0000
13	excitement	0.0000	0.0000	0.0000	55.0000	0.0000
14	fear	0.0082	0.0154	0.0107	65.0000	0.0154
15	gratitude	0.0000	0.0000	0.0000	248.0000	0.0000
16	grief	0.0000	0.0000	0.0000	2.0000	0.0000
17	joy	0.0000	0.0000	0.0000	91.0000	0.0000
18	love	0.0000	0.0000	0.0000	157.0000	0.0000
19	nervousness	0.0000	0.0000	0.0000	12.0000	0.0000
20	optimism	0.0000	0.0000	0.0000	103.0000	0.0000
21	pride	0.0000	0.0000	0.0000	7.0000	0.0000
22	realization	0.0000	0.0000	0.0000	89.0000	0.0000
23	relief	0.0000	0.0000	0.0000	7.0000	0.0000
24	remorse	0.0000	0.0000	0.0000	44.0000	0.0000
25	sadness	0.0298	0.1020	0.0461	98.0000	0.1020
26	surprise	0.0000	0.0000	0.0000	87.0000	0.0000
27	neutral	0.0000	0.0000	0.0000	1594.0000	0.0000
28	accuracy	0.0656	0.0656	0.0656	0.0656	NaN
29	macro avg	0.0068	0.0356	0.0097	4526.0000	NaN
30	weighted avg	0.0088	0.0656	0.0141	4526.0000	NaN

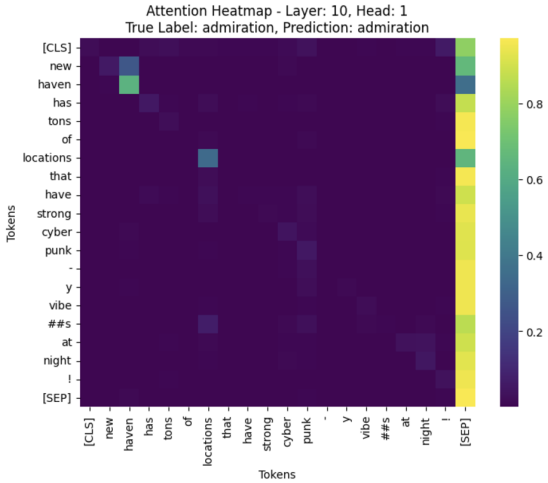
Figure 14: Pretrained BERT Model - Results & Classification Report

Epoch 1/7, Training Loss: 1.7753
Epoch 2/7, Training Loss: 1.3769
Epoch 3/7, Training Loss: 1.2414
Epoch 4/7, Training Loss: 1.1273
Epoch 5/7, Training Loss: 1.0323
Epoch 6/7, Training Loss: 0.9483
Epoch 7/7, Training Loss: 0.8934

Finetuned BERT Model Performance:
Train Accuracy: 0.0656, Train F1 Score: 0.0141
Test Accuracy: 0.6098, Test F1 Score: 0.5976
Hyperparam selection: Batch Size=32, Epochs:7, Adams' Learning Rate: 7e-05
Training Time: 3437.783250s

	class	precision	recall	f1-score	support	accuracy
0	admiration	0.6375	0.7661	0.6959	342.0000	0.7661
1	amusement	0.7366	0.9218	0.8189	179.0000	0.9218
2	anger	0.4470	0.4538	0.4504	130.0000	0.4538
3	annoyance	0.3592	0.2656	0.3054	192.0000	0.2656
4	approval	0.4588	0.3391	0.3900	230.0000	0.3391
5	caring	0.5139	0.4302	0.4684	86.0000	0.4302
6	confusion	0.4419	0.3958	0.4176	96.0000	0.3958
7	curiosity	0.4691	0.5170	0.4919	176.0000	0.5170
8	desire	0.5952	0.4545	0.5155	55.0000	0.4545
9	disappointment	0.3585	0.2159	0.2695	88.0000	0.2159
10	disapproval	0.4044	0.3795	0.3915	195.0000	0.3795
11	disgust	0.4634	0.5067	0.4841	75.0000	0.5067
12	embarrassment	0.6111	0.4783	0.5366	23.0000	0.4783
13	excitement	0.4419	0.3455	0.3878	55.0000	0.3455
14	fear	0.7121	0.7231	0.7176	65.0000	0.7231
15	gratitude	0.9359	0.8831	0.9087	248.0000	0.8831
16	grief	0.0000	0.0000	0.0000	2.0000	0.0000
17	joy	0.5437	0.6154	0.5773	91.0000	0.6154
18	love	0.7861	0.8662	0.8242	157.0000	0.8662
19	nervousness	0.6364	0.5833	0.6087	12.0000	0.5833
20	optimism	0.6117	0.6117	0.6117	103.0000	0.6117
21	pride	1.0000	0.1429	0.2500	7.0000	0.1429
22	realization	0.3214	0.1011	0.1538	89.0000	0.1011
23	relief	0.0000	0.0000	0.0000	7.0000	0.0000
24	remorse	0.6102	0.8182	0.6990	44.0000	0.8182
25	sadness	0.5682	0.5102	0.5376	98.0000	0.5102
26	surprise	0.5000	0.4483	0.4727	87.0000	0.4483
27	neutral	0.6532	0.7089	0.6799	1594.0000	0.7089
28	accuracy	0.6098	0.6098	0.6098	0.6098	NaN
29	macro avg	0.5292	0.4815	0.4880	4526.0000	NaN
30	weighted avg	0.5949	0.6098	0.5976	4526.0000	NaN

Figure 15: Finetuned BERT Model - Results & Classification Report



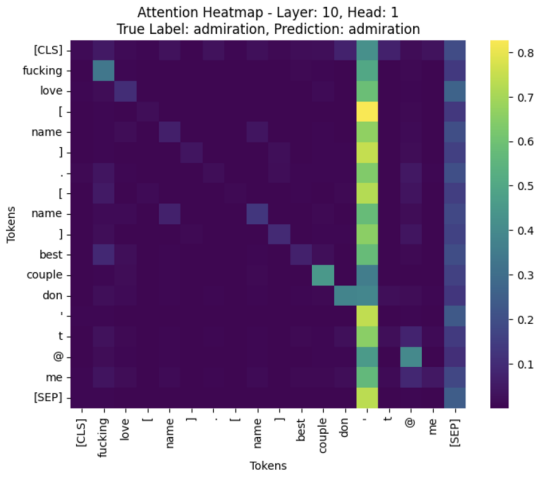
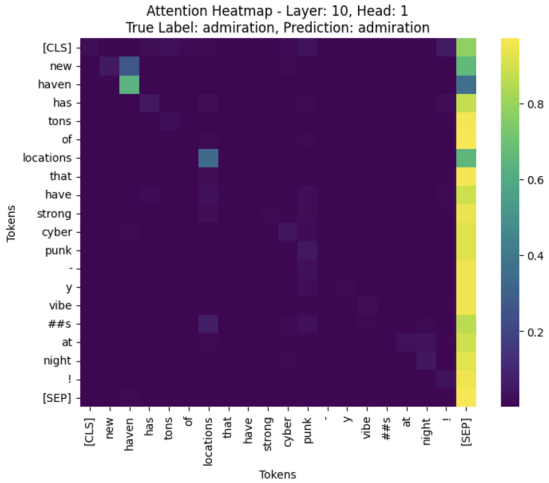


Figure 16: Pretrained Model - Correct Prediction Attention Heatmap



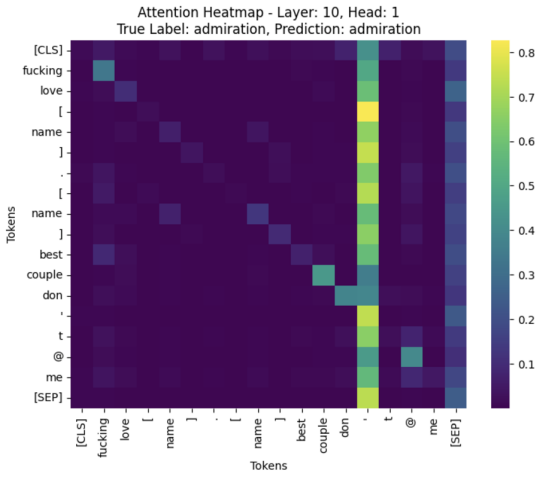


Figure 17: Pretrained Model - Incorrect Prediction Attention Heatmap

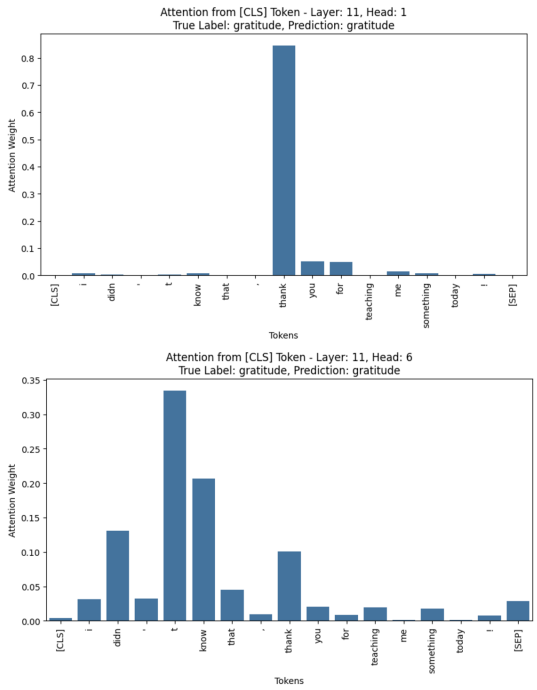


Figure 18: Finetuned Model - Correct Prediction, Layer 11 Bar Charts

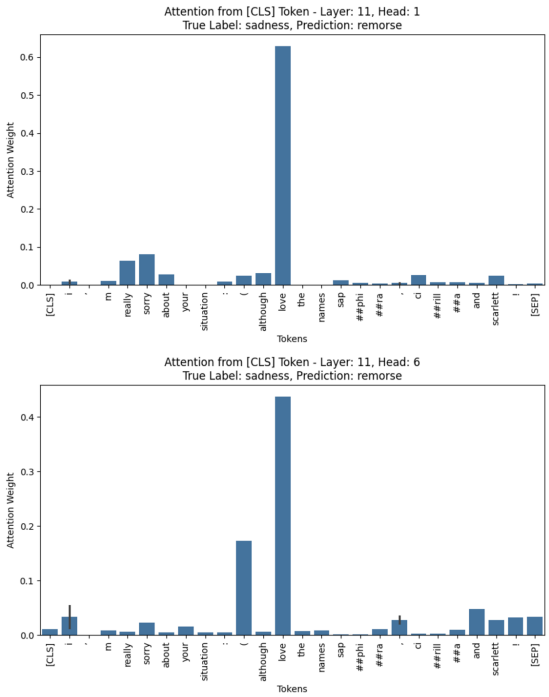


Figure 19: Finetuned Model - Incorrect Prediction, Layer 11 Bar Charts

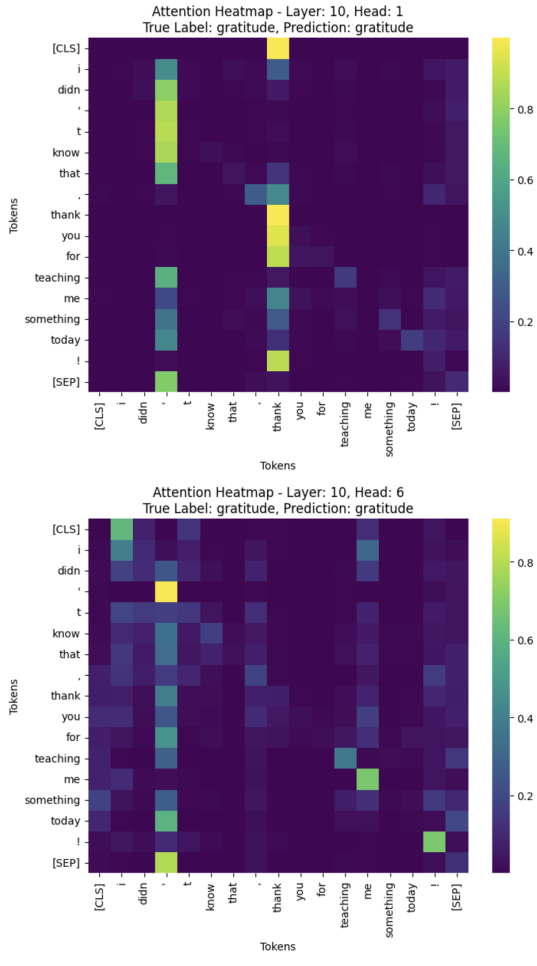


Figure 20: Finetuned Model - Correct Prediction Attention Heatmap, Layer 10

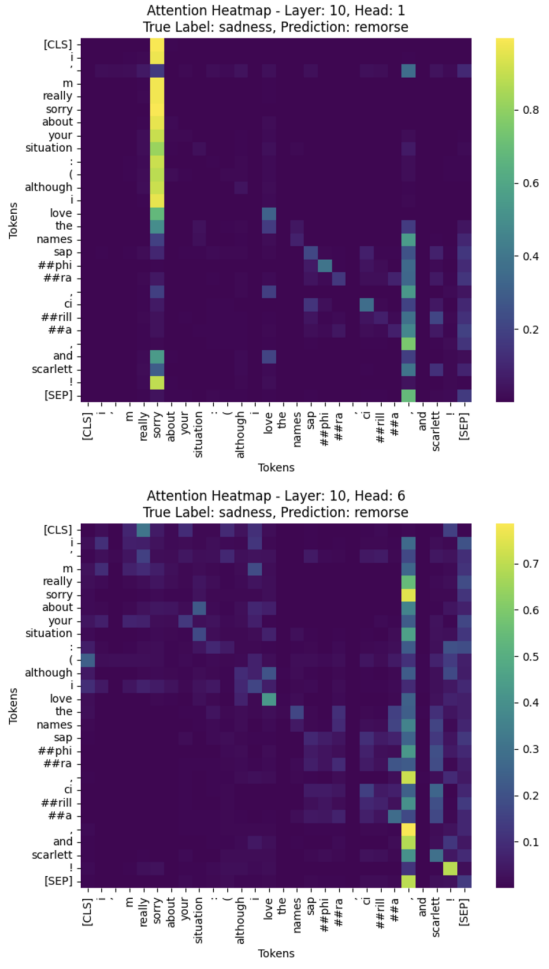


Figure 21: Pretrained Model - Incorrect Prediction Attention Heatmap, Layer 10

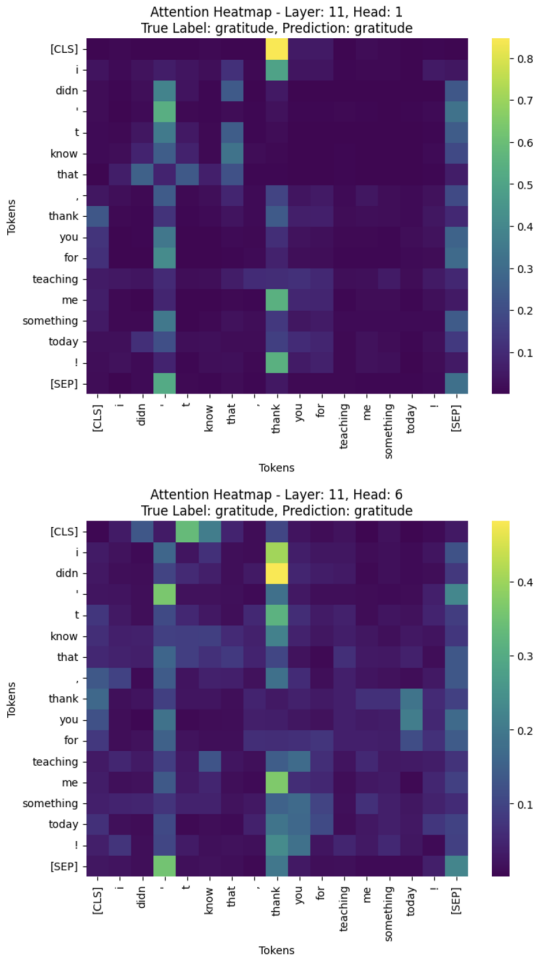


Figure 22: Finetuned Model - Correct Prediction Attention Heatmap, Layer 11

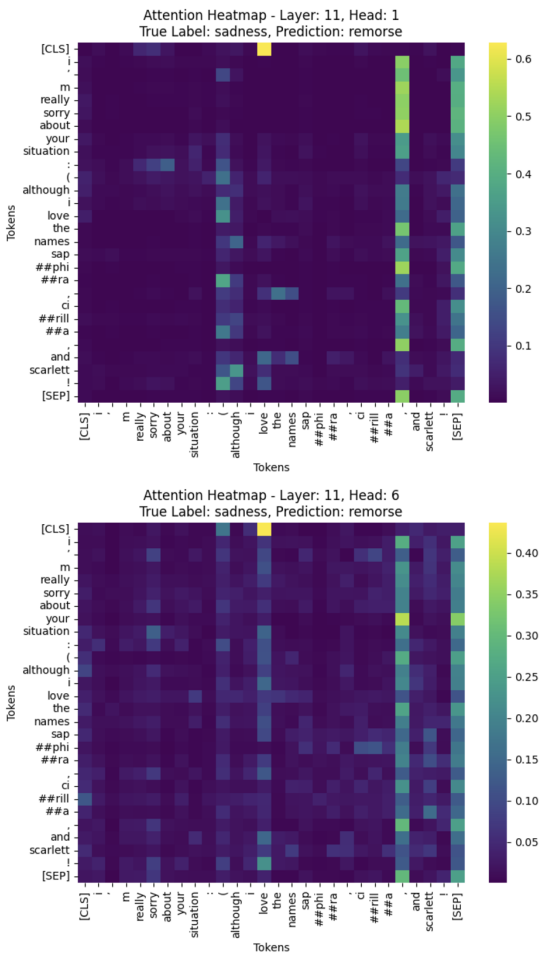


Figure 23: Pretrained Model - Incorrect Prediction Attention Heatmap, Layer 11

Pretrained DistilGPT2 without finetuning:
Test Accuracy Score: 0.0144
Test F1 Score: 0.0004

	precision	recall	f1-score	support
admiration	0.00	0.00	0.00	342
amusement	0.00	0.00	0.00	179
anger	0.00	0.00	0.00	130
annoyance	0.00	0.00	0.00	192
approval	0.00	0.00	0.00	230
caring	0.00	0.00	0.00	86
confusion	0.00	0.00	0.00	96
curiosity	0.00	0.00	0.00	176
desire	0.00	0.00	0.00	55
disappointment	0.00	0.00	0.00	88
disapproval	0.00	0.00	0.00	195
disgust	0.00	0.00	0.00	75
embarrassment	0.00	0.00	0.00	23
excitement	0.00	0.00	0.00	55
fear	0.01	1.00	0.03	65
gratitude	0.00	0.00	0.00	248
grief	0.00	0.00	0.00	2
joy	0.00	0.00	0.00	91
love	0.00	0.00	0.00	157
nervousness	0.00	0.00	0.00	12
optimism	0.00	0.00	0.00	103
pride	0.00	0.00	0.00	7
realization	0.00	0.00	0.00	89
relief	0.00	0.00	0.00	7
remorse	0.00	0.00	0.00	44
sadness	0.00	0.00	0.00	98
surprise	0.00	0.00	0.00	87
neutral	0.00	0.00	0.00	1594
accuracy			0.01	4526
macro avg	0.00	0.04	0.00	4526
weighted avg	0.00	0.01	0.00	4526

Figure 24: Pretrained DistilGPT2 - Classification Report

Epoch 1/3, Training Loss: 2.5582
Epoch 2/3, Training Loss: 1.9833
Epoch 3/3, Training Loss: 1.8732

Finetuned DistilGPT2 Performance:
Train Accuracy: 0.5089, Train F1 Score: 0.4353
Test Accuracy: 0.4969, Test F1 Score: 0.4245
Training Time: 1060.69s

	class	precision	recall	f1-score	support	accuracy
0	admiration	0.4859	0.5526	0.5171	342.0000	0.5526
1	amusement	0.5843	0.5810	0.5826	179.0000	0.5810
2	anger	0.2857	0.2154	0.2456	130.0000	0.2154
3	annoyance	0.2603	0.0990	0.1434	192.0000	0.0990
4	approval	0.4815	0.0565	0.1012	230.0000	0.0565
5	caring	0.0000	0.0000	0.0000	86.0000	0.0000
6	confusion	0.1765	0.0312	0.0531	96.0000	0.0312
7	curiosity	0.4390	0.4091	0.4235	176.0000	0.4091
8	desire	0.6000	0.1091	0.1846	55.0000	0.1091
9	disappointment	0.6364	0.0795	0.1414	88.0000	0.0795
10	disapproval	0.3333	0.0410	0.0731	195.0000	0.0410
11	disgust	0.5789	0.1467	0.2340	75.0000	0.1467
12	embarrassment	0.0000	0.0000	0.0000	23.0000	0.0000
13	excitement	0.8000	0.0727	0.1333	55.0000	0.0727
14	fear	1.0000	0.0462	0.0882	65.0000	0.0462
15	gratitude	0.6729	0.7298	0.7002	248.0000	0.7298
16	grief	0.0000	0.0000	0.0000	2.0000	0.0000
17	joy	0.3103	0.1978	0.2416	91.0000	0.1978
18	love	0.6667	0.8408	0.7437	157.0000	0.8408
19	nervousness	0.0000	0.0000	0.0000	12.0000	0.0000
20	optimism	0.4848	0.3107	0.3787	103.0000	0.3107
21	pride	0.0000	0.0000	0.0000	7.0000	0.0000
22	realization	0.0000	0.0000	0.0000	89.0000	0.0000
23	relief	0.0000	0.0000	0.0000	7.0000	0.0000
24	remorse	0.6129	0.4318	0.5067	44.0000	0.4318
25	sadness	0.3663	0.3776	0.3719	98.0000	0.3776
26	surprise	0.5556	0.0575	0.1042	87.0000	0.0575
27	neutral	0.4895	0.8519	0.6218	1594.0000	0.8519
28	accuracy	0.4969	0.4969	0.4969	0.4969	NaN
29	macro avg	0.3865	0.2228	0.2354	4526.0000	NaN
30	weighted avg	0.4663	0.4969	0.4245	4526.0000	NaN

Figure 25: Finetuned DistilGPT2 - Classification Report

```
labels
27    12719
0      2663
4      1834
15     1809
1      1615
3      1439
10     1387
18     1376
7      1363
2      1016
6       852
17      841
20      841
25      790
26      705
9       702
5       640
22      573
11      495
13      492
14      420
8       375
24      352
12      201
23       88
19       85
21       51
16       37
Name: count, dtype: int64

Before Normalization:
 tensor([  2.8116,  13.4288,  19.4989,  19.7684,  22.1430,  24.8513,  25.7830,
         25.9891,  26.2370,  35.1978,  41.9730,  42.5220,  42.5220,  45.2671,
         50.7248,  50.9416,  55.8766,  62.4101,  72.2444,  72.6850,  85.1452,
         95.3627, 101.5938, 177.9154, 406.3750, 420.7177, 701.1961, 966.5135])

After Normalization:
 tensor([  1.0000,   4.7762,   6.9351,   7.0310,   7.8755,   8.8388,   9.1702,
          9.2435,   9.3316,  12.5187,  14.9284,  15.1237,  15.1237,  16.1000,
         18.0411,  18.1182,  19.8734,  22.1972,  25.6950,  25.8516,  30.2833,
         33.9173,  36.1335,  63.2786, 144.5341, 149.6353, 249.3922, 343.7567])
```

Figure 26: Class-weightings Before & After Normalization

```
Epoch 1/1, Training Loss: 0.1996

Finetuned BERT LLM, weighted loss function and weighted sampling:
        Train Accuracy: 0.9080, Train F1 Score: 0.8785
\Test Accuracy: 0.3721, Test F1 Score: 0.2103
        Training Time: 166.252675s

              precision    recall  f1-score   support

    admiration       0.00      0.00      0.00       342
     amusement       0.00      0.00      0.00       179
         anger       0.00      0.00      0.00       130
     annoyance       0.00      0.00      0.00       192
      approval       0.00      0.00      0.00       230
        caring       0.00      0.00      0.00        86
     confusion       0.00      0.00      0.00        96
     curiosity       0.00      0.00      0.00       176
        desire       0.00      0.00      0.00        55
disappointment       0.00      0.00      0.00        88
   disapproval       0.00      0.00      0.00       195
       disgust       0.00      0.00      0.00        75
 embarrassment       0.00      0.00      0.00        23
    excitement       0.00      0.00      0.00        55
          fear       0.00      0.00      0.00        65
     gratitude       0.00      0.00      0.00       248
         grief       0.00      0.00      0.00         2
           joy       0.00      0.00      0.00        91
          love       0.00      0.00      0.00       157
   nervousness       0.00      0.00      0.00        12
      optimism       0.00      0.00      0.00       103
         pride       0.00      0.00      0.00         7
   realization       0.00      0.00      0.00        89
        relief       0.00      0.00      0.00         7
       remorse       0.31      0.75      0.44        44
       sadness       0.48      0.24      0.32        98
      surprise       0.31      0.54      0.40        87
       neutral       0.37      0.99      0.54      1594

      accuracy                           0.37      4526
     macro avg       0.05      0.09      0.06      4526
  weighted avg       0.15      0.37      0.21      4526
```

Figure 27: Class-weighted Finetuned BERT LLM - Classification Report

```
Epoch 1/1, Training Loss: 0.6437

Finetuned GPT2, weighted loss function and weighted sampling:
        Train Accuracy: 0.8802, Train F1 Score: 0.8241
        Test Accuracy: 0.3522, Test F1 Score: 0.1835
        Training Time: 353.497648s

              precision    recall  f1-score   support

    admiration       0.00      0.00      0.00       342
     amusement       0.00      0.00      0.00       179
         anger       0.00      0.00      0.00       130
     annoyance       0.00      0.00      0.00       192
      approval       0.00      0.00      0.00       230
        caring       0.00      0.00      0.00        86
     confusion       0.00      0.00      0.00        96
     curiosity       0.00      0.00      0.00       176
        desire       0.00      0.00      0.00        55
disappointment       0.00      0.00      0.00        88
   disapproval       0.00      0.00      0.00       195
       disgust       0.00      0.00      0.00        75
 embarrassment       0.00      0.00      0.00        23
    excitement       0.00      0.00      0.00        55
          fear       0.00      0.00      0.00        65
     gratitude       0.00      0.00      0.00       248
         grief       0.00      0.00      0.00         2
           joy       0.00      0.00      0.00        91
          love       0.00      0.00      0.00       157
   nervousness       0.00      0.00      0.00        12
      optimism       0.00      0.00      0.00       103
         pride       0.00      0.00      0.00         7
   realization       0.00      0.00      0.00        89
        relief       0.00      0.00      0.00         7
       remorse       0.00      0.00      0.00        44
       sadness       0.00      0.00      0.00        98
      surprise       0.00      0.00      0.00        87
       neutral       0.35      1.00      0.52      1594

      accuracy                           0.35      4526
     macro avg       0.01      0.04      0.02      4526
  weighted avg       0.12      0.35      0.18      4526
```

Figure 28: Class-weighted Finetuned DistilGPT2 - Classification Report