A Comprehensive Analysis of the GoEmotions Dataset and Emotion Detection Models

Performance Evaluation of DistilGPT-2, BERT, GPT-2, Multinomial Naive Bayes, and Baseline Models

Jessie Kurtz, Marc-Antoine Nadeau, Baicheng Peng

2024-12-04

Abstract

The GoEmotions dataset, created by Google Research, contains over 58,000 Reddit comments labeled with 27 emotions or marked as neutral. This dataset was used to train and evaluate models, including DistilGPT-2, which achieved 61.68% accuracy by fine-tuning hyperparameters, and a Multinomial Naive Bayes model, which achieved 52.77% accuracy without downsampling. Additional experiments with pre-trained embeddings and attention analysis highlighted the challenges of subjective sentiment interpretation, exemplified by mismatches in predicted labels and underscoring the dataset's potential for mapping nuanced emotions to simplified frameworks like Ekman's six basic emotions.

1 Introduction

The GoEmotions dataset, developed by Google Research, is designed for emotion classification and natural language understanding. It features over 58,000 English comments from Reddit, each labeled with one or more of 27 emotion categories or marked as neutral. This dataset is among the most extensive and detailed resources available for studying emotion detection in text [1] This dataset was used to train a large language model (LLM), a Naive Bayes model, a baseline model, and evaluate the accuracy of the predictions they generated. DistilGPT-2 was chosen as the LLM, and its peak performance, yielding 61.68% accuracy on the test set, was found by finetuning the model hyperparamaters on the validation set, and neglecting the 'Neutral' feature.

The Multinomial Naive Bayes model was implemented from scratch and trained with CountVectorizer, achieved 52.77% accuracy without downsampling and 43.74% with downsampling, balancing accuracy and recall for underrepresented classes. Laplace smoothing and rare word filtering improved robustness.

Analyzing the attention heads revealed prioritization of key dependencies in correct predictions, while in incorrect predictions, the focus was scattered across conflicting tokens, such as "love" and "sorry," diluting the intended sentiment.

Additional experiments were done to compare different LLMs (BERT and GPT-2), consider multi-label data, and use pre-trained embeddings instead of dynamic ones.

An analysis done by a Developer Programs Engineer at Google highlighted the trained model's capability to understand the sentiment of the Reddit comment, without achieving the correct classification. For example, the comment "I bet you could get away with this at work just fine" was attributed the label 'Optimism', although the correct answer was 'Approval'[2]. This underscores the subjective nature of sentiment analysis, which cannot be accounted for, unless training data is directly modified. Another study mapped the 28 sentiment labels to Ekman's six basic emotions[3]. This yielded very similar accuracies to our model [4].

[1] S. Bansal, "Go Emotions: Google Emotions Dataset," Kaggle.com, 2021. https://www.kaggle.com/datasets/shivamb/go-emotions-google-emotions-dataset (accessed Dec. 02, 2024).

[2] Google, "GoEmotions Dataset: Training a Classifier," Responsible AI for Developers Blog, Nov. 18, 2022. https://responsible-ai-developers.googleblog.com/2022/11/goemotions-dataset-training-classifier.html (accessed Dec. 02, 2024).

[3] "Figure 2: Ekman's six basic emotions and Plutchik's wheel of emotions...," ResearchGate, 2024. https://www.researchgate.net/figure/Ekmans-six-basic-emotions-and-Plutchiks-wheel-of-emotions-the-middle-circle-contains-8_fig1_346179935.

[4] I. Ameer, N. Bölücü, G. Sidorov, and B. Can, "Emotion Classification in Texts Over Graph Neural Networks: Semantic Representation is Better Than Syntactic," IEEE Access, vol. 11, pp. 56921–56934, Jan. 2023, doi: https://doi.org/10.1109/access.2023.3281544.

2 Methods

2.1 Data Preprocessing & Exploratory Analysis

DistilGPT-2 is a smaller, faster version of OpenAI's GPT-2 language model, developed by Hugging Face. It has 82 million parameters, compared to GPT-2's 124 million, making it more efficient while retaining much of GPT-2's text generation capabilities [5]. DistilGPT-2 was trained on the same data as GPT-2, called the WebText dataset, made up of over 8 million web pages that were collected from links shared on Reddit posts[6]. Since the model was trained on Reddit derived data, we assumed that it would align well with the style and context of the GoEmotions dataset.

For preprocessing, the Hugging Face Transformers package was utilized to tokenize the input text and transform the tokens into numerical features. Additionally, data points with multiple labels were excluded from the dataset. The disproportionate amount of 'Neutral' labels was accounted for when running initial tests, either through oversampling, undersampling, or removing the label completely.

After preprocessing the dataset and discovering the label imbalance, the training process was carried out using the Hugging Face Trainer class. This, combined with the TrainingArguments class, provides a robust API for customizing and executing model training in PyTorch[7]. Table 1 presents the results from different configurations of the dataset using the default arguments from the TrainingArguments class. Configuration ID #7 in Table 1 achieved the highest training and test accuracies and was subsequently selected for hyperparameter tuning.

ID	Features	Sampling Method	Layers Trained	Train Acc. (%)	Test Acc. (%)	Untrained Acc. (%)
1	All 28	None	Last Layer	40.64	39.96	1.80
2	All 28	None	All Layers	72.91	60.89	3.50
3	All 28	Oversampling (27 less-present features)	All Layers	100.00	7.28	0.10
4	All 28	Oversampling (27 less-present features)	Last Layer	100.00	7.19	1.93
5	All 28	Undersampling (feature 28)	All Layers	72.80	49.91	1.98
6	All 28	Undersampling (feature 28)	Last Layer	72.80	25.01	2.06
7	**27 (excluding 'Neutral')**	**None**	**All Layers**	**76.83**	**61.63**	**1.88**
8	27 (excluding 'Neutral')	None	Last Layer	26.20	25.49	3.61

Table 1: Training Configurations, Sampling Methods, and Accuracy Metrics for DistilGPT2 Model

2.2 Implementation of Naive Bayes

We implemented a Multinomial Naive Bayes (MNB) model for emotion classification, leveraging Count Vectorization to transform text data into numerical representations of word frequencies. The model's fit function computes the class priors ($P(C)$) for 28 classes based on their proportions in the dataset and calculates feature likelihoods ($P(X|C)$) for each feature count given the emotion class. To handle unseen features and prevent zero probabilities, we applied Laplace Smoothing, ensuring robust posterior computations even when some features are absent in a class. When predicting, the model calculates posterior probabilities for each class using Bayes' Theorem and assigns the class with the highest posterior probability as the predicted label. Additional

[5] "distilbert/distilgpt2 · Hugging Face," Huggingface.co, May 17, 2024. https://huggingface.co/distilbert/distilgpt2?utm (accessed Dec. 01, 2024).

[6] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," 2019. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

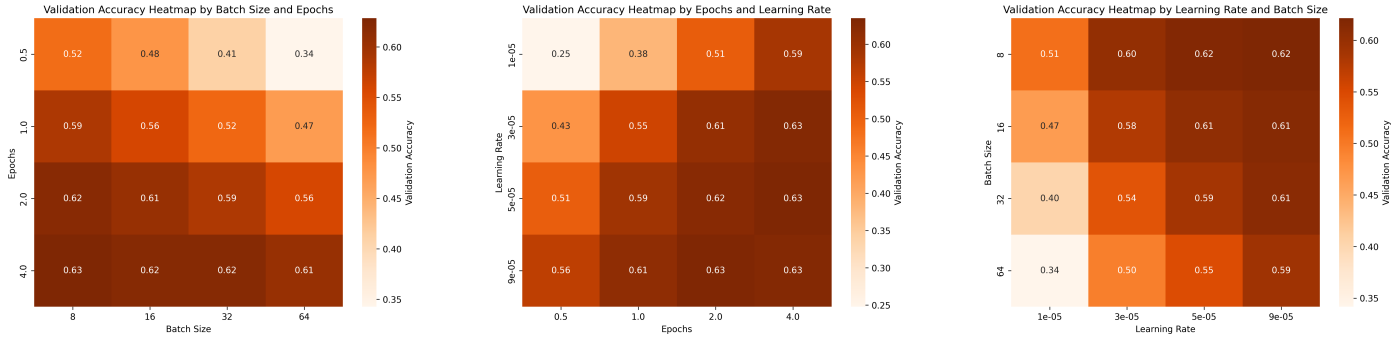
[7] "Trainer," Huggingface.co, 2014. https://huggingface.co/docs/transformers/en/main_classes/trainer#transformers.TrainingArguments (accessed Dec. 01, 2024).

evaluation functions calculate metrics like accuracy, precision, and recall to assess the model's performance comprehensively.

2.3 Large Language Model Finetuning

Taking 4 different values for epoch, batch size, and learning rate, all 64 combinations of these hyper-paramaters were evaluated on the selected training configuration from Table 1. Figure 1 shows that validation accuracy improves with increased epochs but begins to overfit at 4 epochs. Moderate learning rates (5e-05 to 9e-05) yield the best train and validation accuracies, while smaller batch sizes (e.g., 8) outperform larger ones in validation accuracy. The optimal configuration combines smaller batch sizes, moderate learning rates, and sufficient epochs to maximize validation accuracy while avoiding overfitting.

The ideal hyperparameters were found to be a batch size of 8, 2 epochs, and a learning rate of 9e-05, achieving the highest validation accuracy of 64.21% on the DistilGPT2 model, while still maintainig a good training accuracy of 74.18%. Refer to Table 5 in the Appendix for all tests conducted on the DistilGPT2 model.



(a) Validation Accuracy: Batch Size vs. Epochs

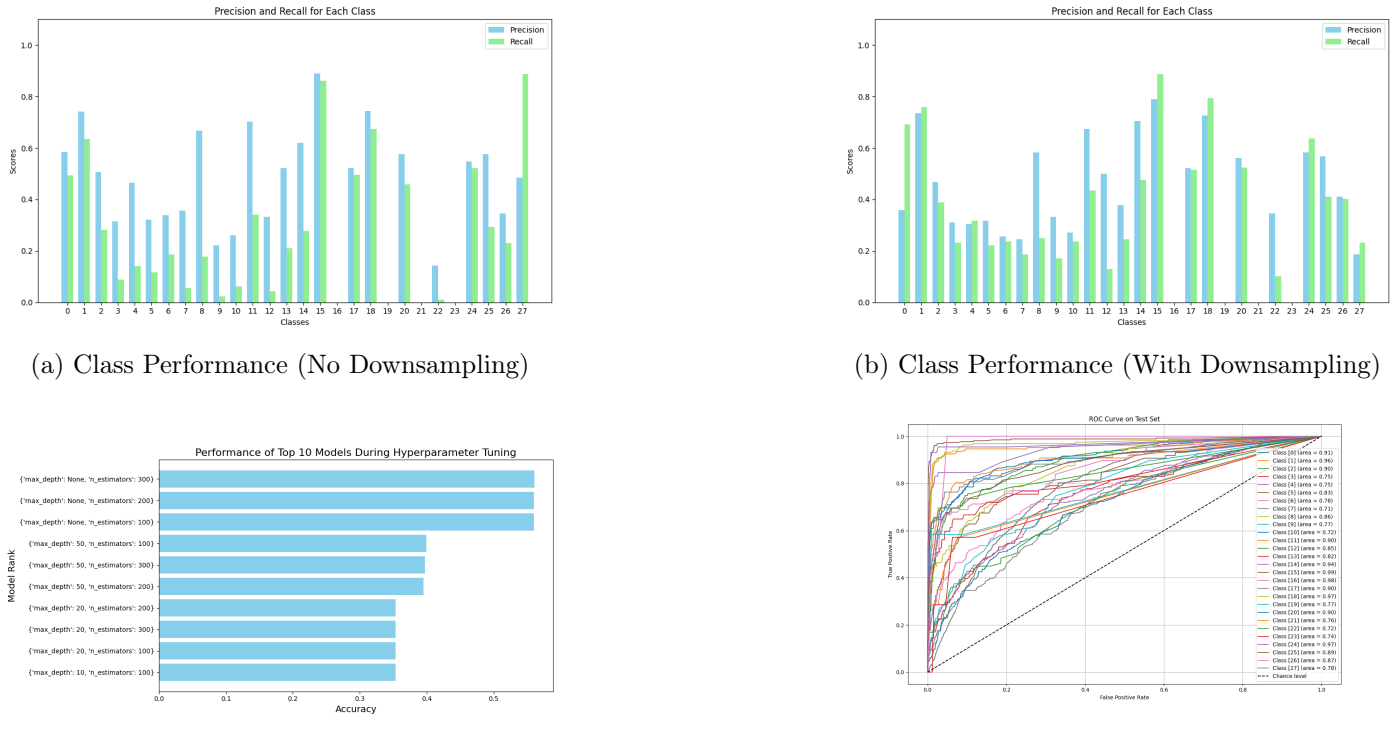
(b) Validation Accuracy: Epochs vs. Learning Rate

(c) Validation Accuracy: Learning Rate vs. Batch Size

Figure 1: Heatmaps Displaying Training and Validation Accuracies Across Hyperparameter Configurations.

2.4 Regularization, Number of Trees, etc

2.4.1 Setting for Naive Bayes and Baseline Model (RF)



(a) Class Performance (No Downsampling)

(b) Class Performance (With Downsampling)

(c) Top 10 RF models during Tuning

(d) ROC curves of best model

Figure 2: Performance graphs for Naive Bayes and Random Forest

Since the Naive Bayes model is trained on the numeric vector produced by the Vectorizer, setting the CountVectorizer(noise deduction, context conservation) directly affects whether the vector represents the underlying emotion of the text. Therefore, common english stopwords, such as 'the','a','and', that don't convey any emotions, were removed, as well as words only showing up in one comment. Furthermore, both unigrams(single word) and bigrams(pairs of consecutive words) were analyzed. This procedure would exclude high-frequency common words that disturbing the classification, provide context information that reduce misclassification(bigram 'not good' provide negative emotion but unigram 'good' imply positive one).

For the baseline model, we did a Grid search on the max-depth and number of trees of the random forest. As shown in Figure 2c, a random forest with unlimited max depth of tree and a total number of 100,200,300 trees perform the best with similar accuracy. Thus we take the random forest with 100 trees which balances time, cost and performance.

3 Results

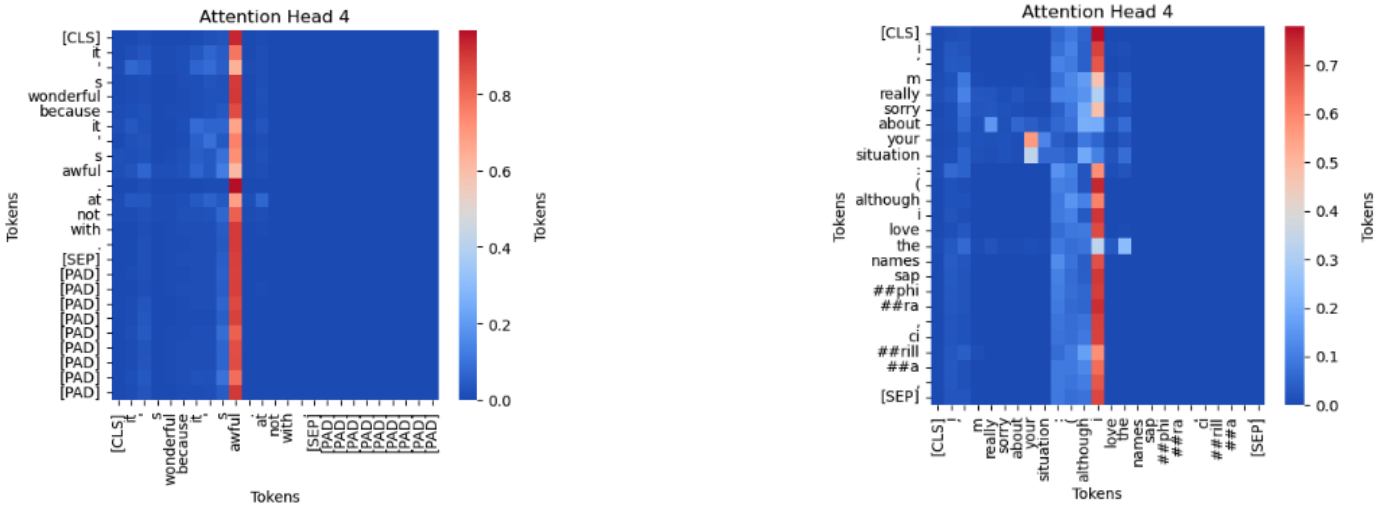
3.1 Classification Performance

As shown in Table 1, the best result for the pre-trained LLM, on the test data, peaks at 3.61%. Furthermore, the finetuned LLM achieved a 64.21% accuracy on the validation data and 61.68% accuracy on the test data. Clearly, the pre-fine-tuned LLM alone cannot be used on the GoEmotions dataset.

The test accuracy of the selected Baseline model (Random Forest with unlimited max-tree-depth and 100 trees) is 56.98%. As shown in Figure 2d the ROC (Receiver Operating Characteristic) of 27 classes varies, with on average the area under curve 0.846, indicating overall ability of classification of Random Forest under such hyperparameter configuration.

As mentioned in 2.1, the dataset is imbalanced with the majority of the text with label 'Netrual'. Then the performance of the Naive Bayes is measured on dataset with and without the downsampling, The test accuracy for imbalanced dataset was 52.77%, as shown in Figure 2a, the recall for the majority 'Neutral' class (27) is high (89%), contributing significantly to the higher overall accuracy, but this is at the cost of extremely low recall of minor classes like 9 and 10. After downsampling Neutral class from over 10000 samples to 2000, as shown in Figure 2b, the overall balance of precision and recall for classes is improved, with recall for minor classes incressed, while the absence of the high recall of Neutral class leading to a decreased overall accuracy 43.74%.

3.2 Attention



(a) Correct Classification (b) Incorrect Classification

Figure 3: BERT Attention Matrix Head 4 and Layer 10 in Incorrect and Correct Classifications

In Figure 3a, for the sentence "It's wonderful because it's awful. At not with," the attention matrix reveals that the token "awful" receives significant attention from many other tokens, emphasizing its role in shaping the sentiment. This indicates that the attention mechanism effectively captures dependencies between key sentiment words to make a correct classification.

In Figure 3b, for the sentence "I'm really sorry about your situation :(Although I love the names Sapphira, Cirilla, and Scarlett!", the model puts too much focus on the word "I", which doesn't carry any real emotional weight in the sentence. At the same time, it doesn't pay enough attention to words like "sorry" or the sad face

":(", which are clearly important for understanding the overall feeling. This mismatch explains why the model gets the sentiment wrong.

3.3 Pre-Trained Word2Vec Embeddings

Word2Vec is a method for creating compact numerical representations of words, called "word embeddings." These embeddings capture semantic relationships between words, such that words with similar meanings have vector representations close to each other in the embedding space. This method generates static embeddings, whereby a word always has the same vector, regardless of the context in which it appears. Static embeddings do not change during downstream task training [8] . This differs from LLMs, which generate contextual embeddings that are dynamically fine-tuned on the dataset to optimize performance. Table 2 depicts the results from using the Wikipedia2Vec (a Word2Vec-like model) on Random Forest, XGBoost and Softmax Regression models [9].

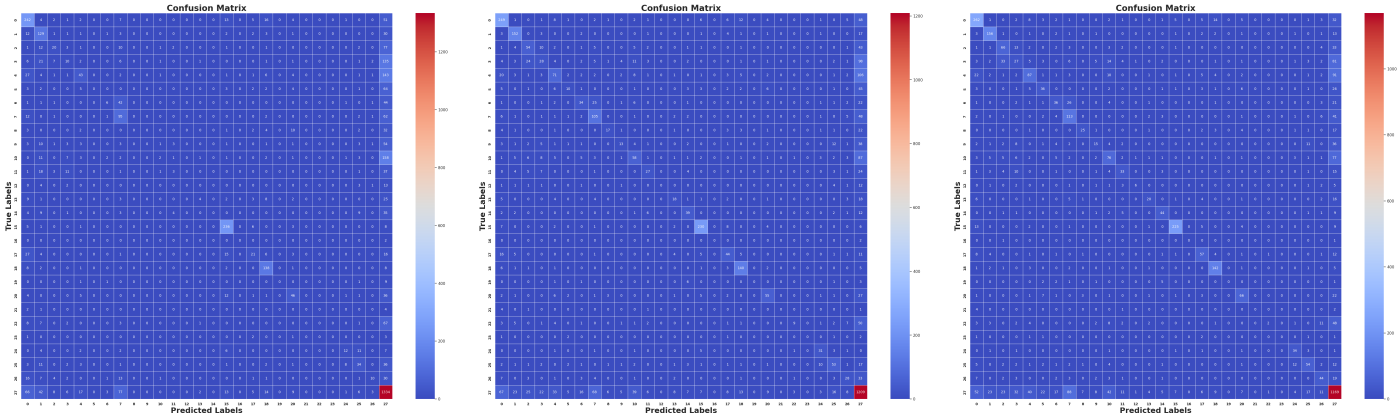
The table highlights that all classifiers exhibit overfitting without regularization. However, regularization, while reducing overfitting, does not consistently improve test accuracy, as seen in Random Forest and XGBoost where test performance slightly declines post-regularization. To improve performance, hyperparameter tuning, particularly for regularization strength, could help optimize the trade-off between overfitting and underfitting. Additionally, experimenting with ensemble methods may enhance test accuracy by combining the strengths of multiple models to improve generalization.

Classifier	Train Accuracy (Regularized) (%)	Test Accuracy (Regularized) (%)	Train Accuracy (%)	Test Accuracy (%)
Random Forest	59.08	46.15	99.48	52.14
XGBoost	60.12	51.98	78.83	54.22
Softmax Regression	61.06	52.95	71.33	51.81

Table 2: Model Training and Test Results Using Pre-Trained Embeddings (Vector Representations of Words)

3.4 Comparing LLMs

As part of our experimentation, we also fine-tuned two other large language models, GPT-2 and BERT, and compared them with our initial chosen model, DistilGPT-2. In Figure 4, you will find the confusion matrices, and in Table 3, the configurations for the three LLMs and the performance of the top three configurations for each model.



(a) Confusion Matrix for DistilGPT2 (b) Confusion Matrix for GPT2 (c) Confusion Matrix for BERT

Figure 4: Confusion Matrices for DistilGPT2, GPT2, and BERT

[8] J. Hockenmaier, Accessed: Dec. 01, 2024. [Online]. Available: https://courses.grainger.illinois.edu/cs546/sp2020/Slides/Lecture04.pdf

[9] Studio Ousia, "Pretrained Embeddings - Wikipedia2Vec," *Github.io*, 2024. https://wikipedia2vec.github.io/wikipedia2vec/pretrained/ (accessed Dec. 01, 2024).

LLM	Batch Size	Epochs	Learning Rate	Training Accuracy	Testing Accuracy
DistilGPT2	16	2	9e-05	0.6975	0.6181
GPT2	8	1	9e-05	0.6639	0.6253
BERT	8	1	5e-05	0.6751	0.6245

Table 3: Comparing the best configurations for DistilGPT2, GPT2, and BERT

As one can observe in Table 3, the three large language models—DistilGPT-2, GPT-2, and BERT—perform very similarly, with their best performing accuracies ranging between 0.61 and 0.63. However, while GPT-2 and BERT achieve this in just one epoch, it takes DistilGPT-2 two epochs, likely due to its smaller model size You can find the rest of 64 hyperparameter combinations tested for DistilGPT-2, GPT-2, and BERT respectively in Tables. 6, 8, and 7.

3.5 Multilabel Dataset

LLM	Accuracy	Recall	Precision	F1-Score
DistilGPT2	0.43	0.47	0.68	0.53
GPT2	0.44	0.49	0.68	0.55
BERT	0.45	0.47	0.68	0.53

Table 4: Performance Metrics for DistilGPT2, GPT2, and BERT on Multi-Label Data

We also experimented with the models' ability to classify multi-label data and observed a significant drop in classification performance compared to single-label data. However, we noticed that the three large language models (DistilGPT-2, GPT-2, and BERT) performed similarly across Accuracy, Recall, Precision, and F1-Score, as listed in Table 4. The marginally best-performing model was GPT-2, with both a slightly higher Recall (0.49) and F1-Score (0.55).

An interesting aspect was that Precision remained consistent at 0.68 across all models and was notably high compared to the other metrics. This high Precision indicates that when the models did predict a label, they were generally correct,but it suggests that due to the conservativeness in classification, they missed some relevant labels, which explaining the lower Recall values.

4 Discussion & Conclusion

This study evaluated DistilGPT-2, Multinomial Naive Bayes, and Random Forest models for classifying the GoEmotions dataset, focusing on accuracy, recall, and efficiency. Multinomial Naive Bayes achieved moderate performance (accuracy: 0.5277), with Count Vectorization and Laplace smoothing handling text data well, though class imbalance reduced recall for minority labels.

DistilGPT-2 performed best, with a test accuracy of 0.6168, benefiting from fine-tuning and excluding the "Neutral" label, though misclassifications showed scattered attention. Random Forest offered competitive accuracy (0.5698) with balanced precision and recall, supported by high ROC-AUC scores.

While large language models like DistilGPT-2 improved accuracy, their computational demands highlight the value of simpler models for efficiency. Future research should explore hybrid methods and ensemble strategies to enhance performance and address dataset imbalances.

5 Statement of Contributions

All team members contributed equally by independently implementing all parts of the report. Once all models were completed, we analyzed each one and merged them by selecting the best aspects of each. From these merged models, we collaboratively compiled our findings into this report.

6 Appendix

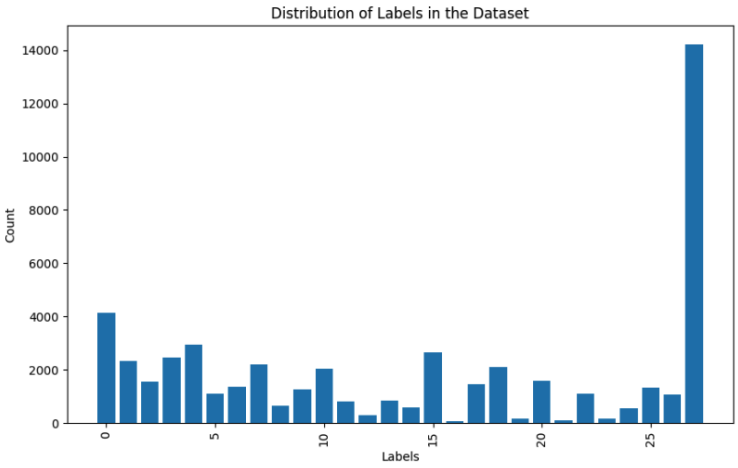


Figure 5: Label distribution visualization.

Batch Size	Epochs	Learning Rate	Train Accuracy	Val Accuracy
8	0.5	1e-05	0.3307	0.3488
8	0.5	3e-05	0.5376	0.5406
8	0.5	5e-05	0.5852	0.5829
8	0.5	9e-05	0.6155	0.5961
8	1	1e-05	0.4843	0.4915
8	1	3e-05	0.6153	0.6086
8	1	5e-05	0.6444	0.6221
8	1	9e-05	0.6674	0.6265
8	2	1e-05	0.5850	0.5832
8	2	3e-05	0.6720	0.6262
8	2	5e-05	0.7018	0.6286
8	2	9e-05	0.7418	0.6421
8	4	1e-05	0.6456	0.6198
8	4	3e-05	0.7534	0.6363
8	4	5e-05	0.8219	0.6367
8	4	9e-05	0.8912	0.6208
16	0.5	1e-05	0.2944	0.2953
16	0.5	3e-05	0.4779	0.4848
16	0.5	5e-05	0.5503	0.5467
16	0.5	9e-05	0.5937	0.5802
16	1	1e-05	0.4086	0.4232
16	1	3e-05	0.5866	0.5778
16	1	5e-05	0.6250	0.6143
16	1	9e-05	0.6551	0.6225
16	2	1e-05	0.5392	0.5460
16	2	3e-05	0.6475	0.6211
16	2	5e-05	0.6832	0.6286
16	2	9e-05	0.7198	0.6286
16	4	1e-05	0.6175	0.6001
16	4	3e-05	0.7166	0.6275
16	4	5e-05	0.7777	0.6326
16	4	9e-05	0.8564	0.6248
32	0.5	1e-05	0.2078	0.2026
32	0.5	3e-05	0.3819	0.3870
32	0.5	5e-05	0.4898	0.4942
32	0.5	9e-05	0.5631	0.5609
32	1	1e-05	0.3382	0.3447
32	1	3e-05	0.5339	0.5365
32	1	5e-05	0.5951	0.5951
32	1	9e-05	0.6351	0.6137
32	2	1e-05	0.4770	0.4855
32	2	3e-05	0.6166	0.6035
32	2	5e-05	0.6591	0.6225
32	2	9e-05	0.6958	0.6326
32	4	1e-05	0.5827	0.5805
32	4	3e-05	0.6824	0.6289
32	4	5e-05	0.7397	0.6414
32	4	9e-05	0.8189	0.6357
64	0.5	1e-05	0.1484	0.1441
64	0.5	3e-05	0.2999	0.3112
64	0.5	5e-05	0.3923	0.3995
64	0.5	9e-05	0.5039	0.5156
64	1	1e-05	0.2609	0.2673
64	1	3e-05	0.4630	0.4685
64	1	5e-05	0.5394	0.5440
64	1	9e-05	0.6092	0.5964
64	2	1e-05	0.4003	0.4147
64	2	3e-05	0.5813	0.5819
64	2	5e-05	0.6293	0.6137
64	2	9e-05	0.6660	0.6248
64	4	1e-05	0.5398	0.5403
64	4	3e-05	0.6541	0.6208
64	4	5e-05	0.6998	0.6292
64	4	9e-05	0.7644	0.6316

Table 5: Training and Validation Accuracies for Hyperparameter Configurations of DistilGPT2 with 27 labels

Batch Size	Epochs	Learning Rate	Train Accuracy	Val Accuracy
8	0.5	1e-05	0.4798	0.4919
8	0.5	3e-05	0.5684	0.5677
8	0.5	5e-05	0.5962	0.5880
8	0.5	9e-05	0.6127	0.6007
8	1	1e-05	0.5392	0.5405
8	1	3e-05	0.6139	0.5976
8	1	5e-05	0.6295	0.6051
8	1	9e-05	0.6455	0.6130
8	2	1e-05	0.5937	0.5855
8	2	3e-05	0.6517	0.6115
8	2	5e-05	0.6783	0.6128
8	2	9e-05	0.7101	0.6146
8	4	1e-05	0.6358	0.6049
8	4	3e-05	0.7282	0.6091
8	4	5e-05	0.7929	0.6000
8	4	9e-05	0.8658	0.5974
16	0.5	1e-05	0.4582	0.4510
16	0.5	3e-05	0.5419	0.5444
16	0.5	5e-05	0.5778	0.5772
16	0.5	9e-05	0.6034	0.5934
16	1	1e-05	0.5101	0.5136
16	1	3e-05	0.5988	0.5954
16	1	5e-05	0.6209	0.6049
16	1	9e-05	0.6382	0.6124
16	2	1e-05	0.5700	0.5677
16	2	3e-05	0.6355	0.6075
16	2	5e-05	0.6617	0.6148
16	2	9e-05	0.6975	0.6181
16	4	1e-05	0.6155	0.5974
16	4	3e-05	0.6921	0.6146
16	4	5e-05	0.7508	0.6058
16	4	9e-05	0.8329	0.5965
32	0.5	1e-05	0.4155	0.4147
32	0.5	3e-05	0.5048	0.5108
32	0.5	5e-05	0.5514	0.5569
32	0.5	9e-05	0.5840	0.5792
32	1	1e-05	0.4832	0.4890
32	1	3e-05	0.5745	0.5701
32	1	5e-05	0.6050	0.5959
32	1	9e-05	0.6269	0.6082
32	2	1e-05	0.5434	0.5451
32	2	3e-05	0.6183	0.6053
32	2	5e-05	0.6437	0.6110
32	2	9e-05	0.6726	0.6124
32	4	1e-05	0.5975	0.5899
32	4	3e-05	0.6621	0.6139
32	4	5e-05	0.7123	0.6121
32	4	9e-05	0.7881	0.6093
64	0.5	1e-05	0.3695	0.3652
64	0.5	3e-05	0.4733	0.4782
64	0.5	5e-05	0.5072	0.5117
64	0.5	9e-05	0.5587	0.5591
64	1	1e-05	0.4560	0.4628
64	1	3e-05	0.5411	0.5422
64	1	5e-05	0.5824	0.5741
64	1	9e-05	0.6112	0.6009
64	2	1e-05	0.5114	0.5150
64	2	3e-05	0.6007	0.5939
64	2	5e-05	0.6266	0.6040
64	2	9e-05	0.6533	0.6172
64	4	1e-05	0.5790	0.5792
64	4	3e-05	0.6396	0.6062
64	4	5e-05	0.6768	0.6128
64	4	9e-05	0.7405	0.6139

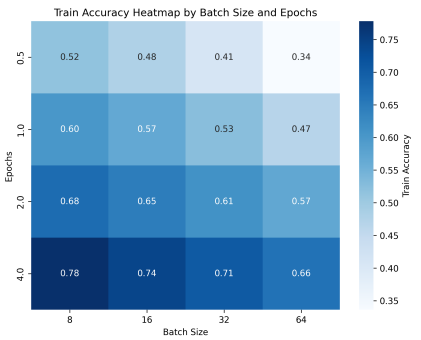
Table 6: Training and Validation Accuracies for Hyperparameter Configurations for Distilled GPT2 with 28 labels

Batch Size	Epochs	Learning Rate	Train Accuracy	Val Accuracy
8	0.5	1e-05	0.5569	0.5453
8	0.5	3e-05	0.6191	0.5939
8	0.5	5e-05	0.6301	0.6016
8	0.5	9e-05	0.6287	0.6018
8	1	1e-05	0.6109	0.5844
8	1	3e-05	0.6639	0.6168
8	1	5e-05	0.6751	0.6245
8	1	9e-05	0.6709	0.6141
8	2	1e-05	0.6699	0.6108
8	2	3e-05	0.7509	0.6159
8	2	5e-05	0.7790	0.6095
8	2	9e-05	0.7548	0.6084
8	4	1e-05	0.7628	0.6077
8	4	3e-05	0.9175	0.5926
8	4	5e-05	0.9415	0.5886
8	4	9e-05	0.9241	0.5803
16	0.5	1e-05	0.5201	0.5163
16	0.5	3e-05	0.5995	0.5871
16	0.5	5e-05	0.6236	0.5992
16	0.5	9e-05	0.6286	0.6020
16	1	1e-05	0.5883	0.5787
16	1	3e-05	0.6496	0.6119
16	1	5e-05	0.6689	0.6236
16	1	9e-05	0.6761	0.6198
16	2	1e-05	0.6407	0.5981
16	2	3e-05	0.7267	0.6119
16	2	5e-05	0.7648	0.6181
16	2	9e-05	0.7795	0.6104
16	4	1e-05	0.7188	0.6115
16	4	3e-05	0.8813	0.5912
16	4	5e-05	0.9338	0.5888
16	4	9e-05	0.9499	0.5853
32	0.5	1e-05	0.4415	0.4415
32	0.5	3e-05	0.5831	0.5697
32	0.5	5e-05	0.6013	0.5836
32	0.5	9e-05	0.6218	0.5970
32	1	1e-05	0.5621	0.5543
32	1	3e-05	0.6307	0.5992
32	1	5e-05	0.6563	0.6192
32	1	9e-05	0.6700	0.6168
32	2	1e-05	0.6138	0.5825
32	2	3e-05	0.6963	0.6113
32	2	5e-05	0.7381	0.6132
32	2	9e-05	0.7675	0.6124
32	4	1e-05	0.6818	0.6110
32	4	3e-05	0.8315	0.6003
32	4	5e-05	0.9010	0.5904
32	4	9e-05	0.9415	0.5756
64	0.5	1e-05	0.4271	0.4305
64	0.5	3e-05	0.5494	0.5396
64	0.5	5e-05	0.5838	0.5686
64	0.5	9e-05	0.5996	0.5794
64	1	1e-05	0.5063	0.5033
64	1	3e-05	0.6030	0.5838
64	1	5e-05	0.6376	0.6033
64	1	9e-05	0.6566	0.6170
64	2	1e-05	0.5871	0.5699
64	2	3e-05	0.6722	0.6139
64	2	5e-05	0.7097	0.6148
64	2	9e-05	0.7482	0.6170
64	4	1e-05	0.5789	0.5792
64	4	3e-05	0.6396	0.6062
64	4	5e-05	0.6768	0.6128
64	4	9e-05	0.7405	0.6139

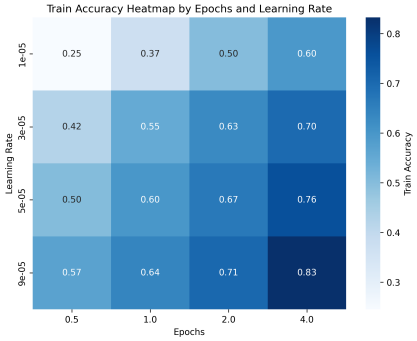
Table 7: Training and Validation Accuracies for Hyperparameter Configurations for BERT with 28 labels

Batch Size	Epochs	Learning Rate	Train Accuracy	Val Accuracy
8	0.5	1e-05	0.4935	0.4923
8	0.5	3e-05	0.5927	0.5851
8	0.5	5e-05	0.6151	0.5996
8	0.5	9e-05	0.6281	0.6161
8	1	1e-05	0.5673	0.5723
8	1	3e-05	0.6319	0.6152
8	1	5e-05	0.6497	0.6201
8	1	9e-05	0.6639	0.6253
8	2	1e-05	0.6174	0.6042
8	2	3e-05	0.6774	0.6146
8	2	5e-05	0.7093	0.6172
8	2	9e-05	0.7425	0.6172
8	4	1e-05	0.6588	0.6077
8	4	3e-05	0.7803	0.6062
8	4	5e-05	0.8610	0.5996
8	4	9e-05	0.9188	0.5932
16	0.5	1e-05	0.4557	0.4613
16	0.5	3e-05	0.5609	0.5657
16	0.5	5e-05	0.5971	0.5921
16	0.5	9e-05	0.6214	0.6055
16	1	1e-05	0.5332	0.5405
16	1	3e-05	0.6193	0.6088
16	1	5e-05	0.6391	0.6220
16	1	9e-05	0.6550	0.6216
16	2	1e-05	0.5931	0.5904
16	2	3e-05	0.6595	0.6139
16	2	5e-05	0.6900	0.6198
16	2	9e-05	0.7272	0.6181
16	4	1e-05	0.6394	0.6088
16	4	3e-05	0.7339	0.6080
16	4	5e-05	0.8146	0.6036
16	4	9e-05	0.8994	0.5932
32	0.5	1e-05	0.4167	0.4178
32	0.5	3e-05	0.5080	0.5114
32	0.5	5e-05	0.5655	0.5624
32	0.5	9e-05	0.6076	0.5952
32	1	1e-05	0.4860	0.4912
32	1	3e-05	0.5946	0.5860
32	1	5e-05	0.6236	0.6095
32	1	9e-05	0.6453	0.6161
32	2	1e-05	0.5654	0.5697
32	2	3e-05	0.6401	0.6069
32	2	5e-05	0.6672	0.6146
32	2	9e-05	0.7003	0.6143
32	4	1e-05	0.6187	0.5992
32	4	3e-05	0.6949	0.6185
32	4	5e-05	0.7616	0.6119
32	4	9e-05	0.8591	0.5982
64	0.5	1e-05	0.3638	0.3582
64	0.5	3e-05	0.4748	0.4734
64	0.5	5e-05	0.5220	0.5288
64	0.5	9e-05	0.5811	0.5778
64	1	1e-05	0.4462	0.4446
64	1	3e-05	0.5607	0.5635
64	1	5e-05	0.6024	0.5981
64	1	9e-05	0.6302	0.6093
64	2	1e-05	0.5299	0.5365
64	2	3e-05	0.6192	0.6018
64	2	5e-05	0.6445	0.6082
64	2	9e-05	0.6821	0.6148
64	4	1e-05	0.5962	0.5871
64	4	3e-05	0.6645	0.6182
64	4	5e-05	0.7152	0.6119
64	4	9e-05	0.8030	0.6014

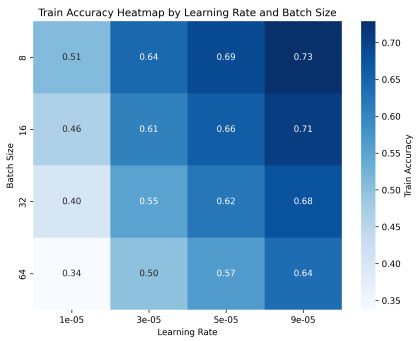
Table 8: Training and Validation Accuracies for Hyperparameter Configurations for GPT2 with 28 labels



(a) Train Accuracy: Batch Size vs. Epochs

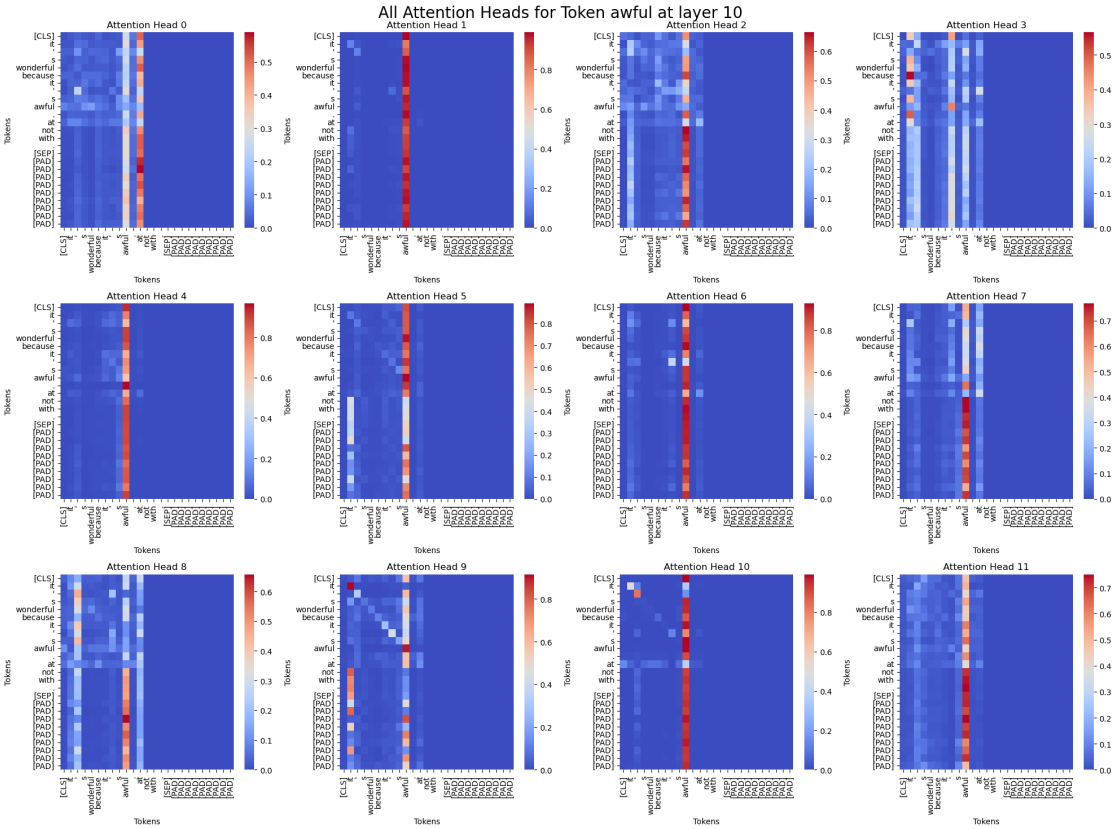


(b) Train Accuracy: Epochs vs. Learning Rate

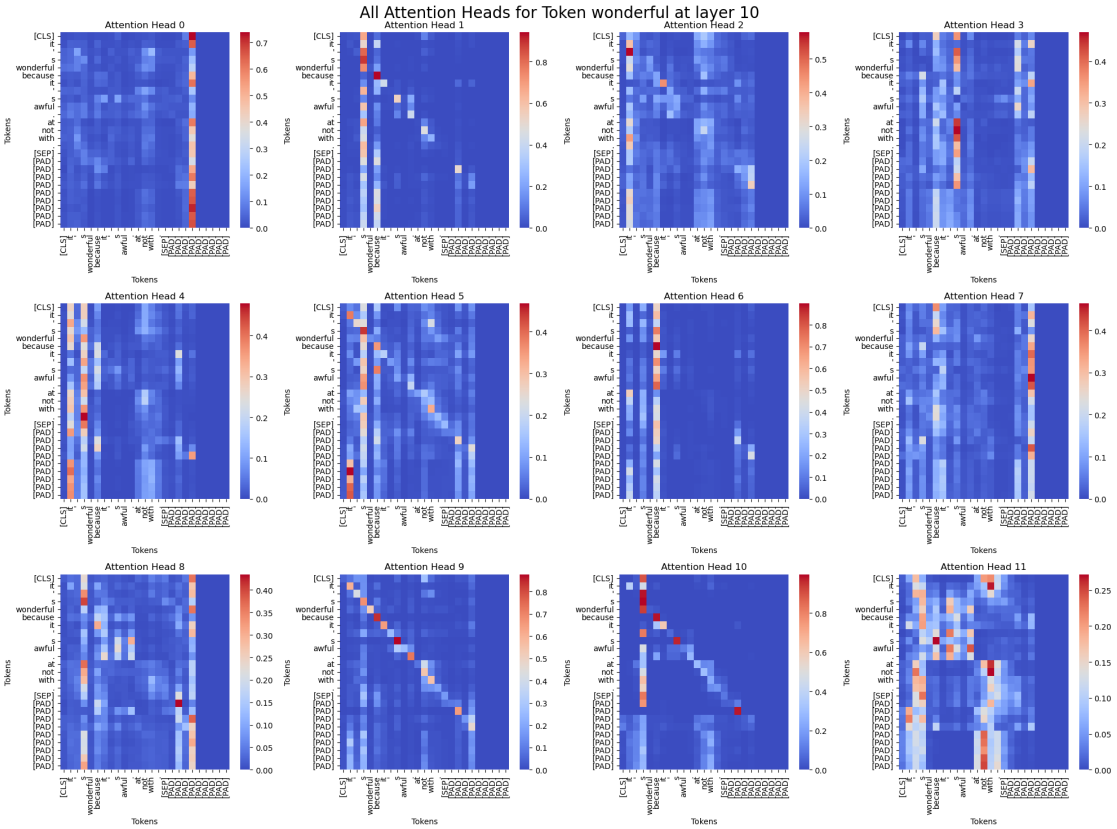


(c) Train Accuracy: Learning Rate vs. Batch Size

Figure 6: Heatmaps Displaying Training Accuracy Across Hyperparameter Configurations for DistilGPT-2.

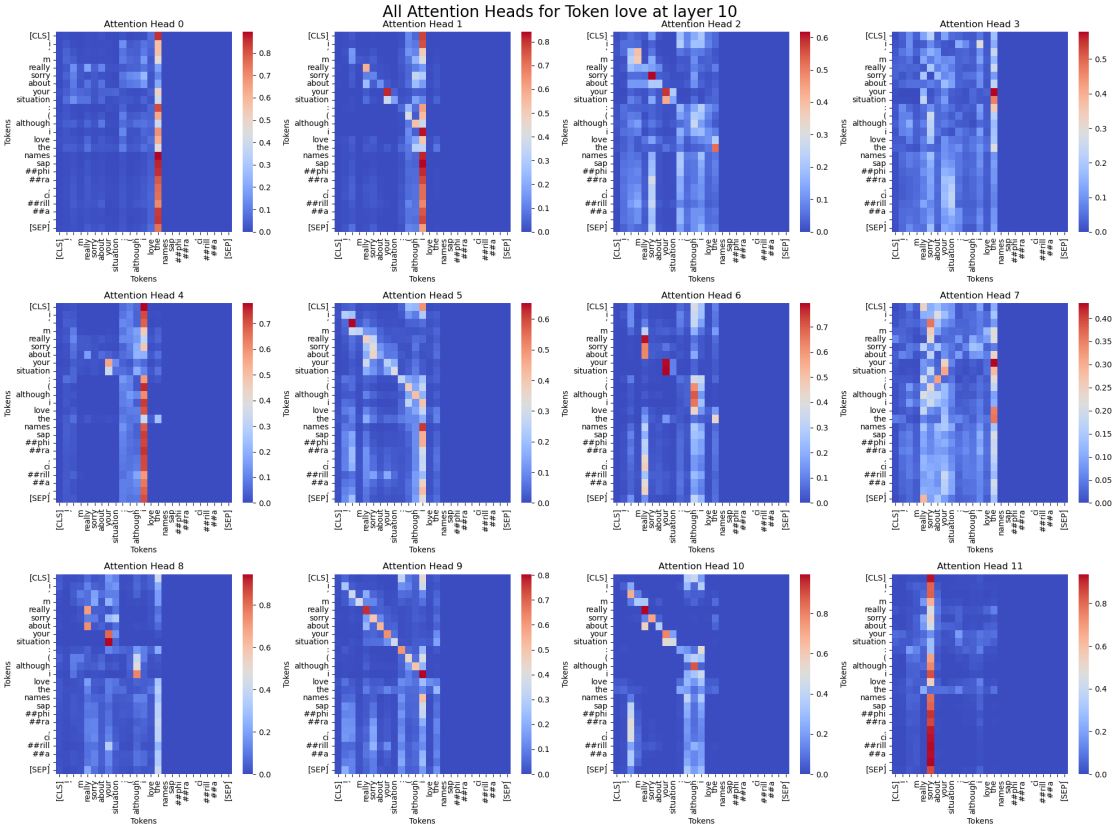


(a) Awful

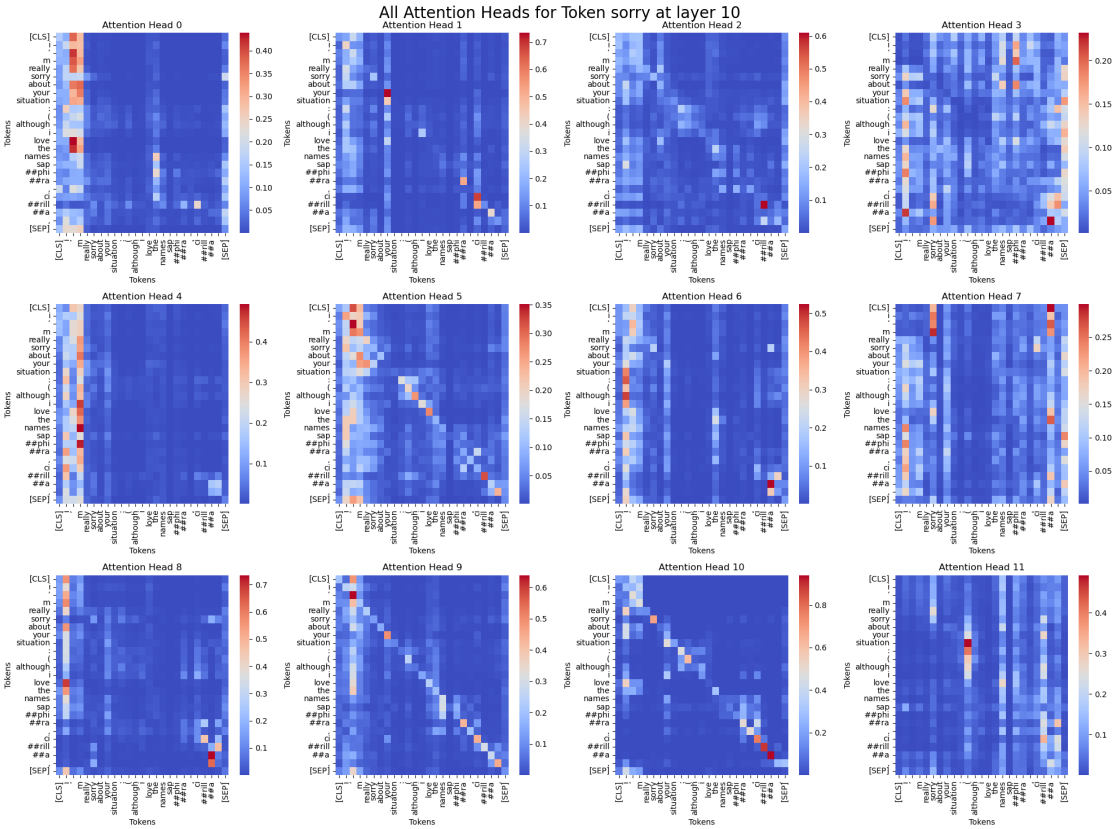


(b) Wonderful

Figure 7: BERT Attention Matrices in Correct Classification



(a) Love



(b) Sorry

Figure 8: BERT Attention Matrices in Incorrect Classification