Detecting Sentiment in COVID-19 Tweets Using Transformer Models and Compression Techniques

Group Y | Yarden Cohen, David Pastron | August 21, 2025

Git Repository | Drive Folder

Abstract

This project fine-tuned DistilBERT and RoBERTa on the COVID-19 NLP dataset for sentiment classification using both a manual PyTorch loop and the Hugging Face Trainer API, with Optuna for hyperparameter tuning and W&B for experiment tracking. RoBERTa achieved the best performance, while DistilBERT provided a lighter alternative with competitive results. To enable deployment on resource-limited environments, we applied dynamic quantization, pruning, and knowledge distillation with TinyBERT. Quantization halved model size with minimal accuracy loss, pruning achieved sparsity of 32.09% (DistilBERT) and 34.37% (RoBERTa), and distillation produced a very small but less accurate student model. These results highlight trade-offs between performance and efficiency when compressing transformer models.

1. Introduction

The COVID-19 pandemic has led to a massive increase in online discussions, particularly on social media platforms like Twitter. Analyzing public sentiment during this period provides valuable insights for governments, healthcare organizations, and researchers. Traditional machine learning techniques struggle to capture contextual meaning in short, noisy text such as tweets. Transformer-based models, including BERT, RoBERTa, and DistilBERT, have achieved state-of-the-art performance in various NLP tasks by leveraging self-attention mechanisms and pre-trained contextual embeddings. This study focuses on fine-tuning DistilBERT and RoBERTa on the COVID-19 NLP dataset for sentiment classification. We compare two fine-tuning strategies — manual training loops and the Hugging Face Trainer API - and perform hyperparameter optimization using Optuna. Furthermore, we apply model compression techniques to explore trade-offs between model size and performance for deployment in resource-constrained environments.

2. Related Work

Transformer-based language models have significantly advanced natural language processing tasks, including sentiment analysis. BERT [1] introduced the concept of bidirectional contextual embeddings, while RoBERTa[2] improved upon BERT through optimized training strategies and larger pre-training corpora. DistilBERT[3], a distilled version of BERT, retains around 97% of BERT's performance while being smaller and faster, making it suitable for deployment on devices with limited computational resources. Previous research has demonstrated the importance of hyperparameter tuning, where methods such as Bayesian optimization and tools like Optuna [4] can lead to substantial performance gains. Additionally, model compression methods such as quantization, pruning, and knowledge distillation have proven effective for reducing model size and inference time while maintaining acceptable accuracy. These techniques are critical for deploying transformer-based models in real-world applications where computational efficiency is essential.

3. Exploratory Data Analysis

3.1 The Dataset

The dataset was obtained from Kaggle and consists of over 45k COVID-19-related tweets labeled with sentiment categories: Extremely Negative, Negative, Neutral, Positive & Extremely Positive.

3.2 Exploration of the Data

We began by conducting an exploratory data analysis to understand the dataset and uncover relevant patterns for sentiment classification.

1. Distribution of Sentiments (Figure 1) - The dataset is imbalanced, with positive and negative tweets being the most frequent, followed by neutral tweets, while extreme positive and extreme negative tweets are less common. This imbalance has implications for model training and evaluation.

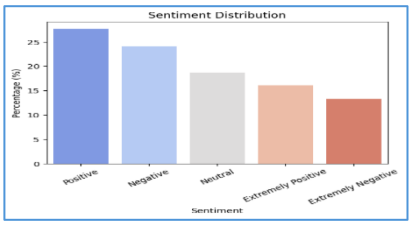

Figure 1. sentiment Distribution

2. Tweet Length by Sentiment (Figures 2) - Tweets with extreme sentiments (both positive and negative) tend to be longer, while neutral tweets are generally shorter. Boxplots further confirm that extreme classes have a higher median and wider variability in length.

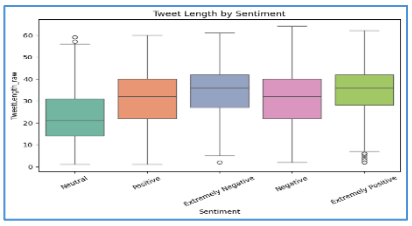

Figure 2. sentiment Length

3. Density Patterns (Figure 3) - The density plot shows that moderately positive and negative tweets have similar length distributions, as do extremely positive and negative tweets. This suggests that the difference between moderate and extreme sentiments is clearer than between positive and negative polarity.

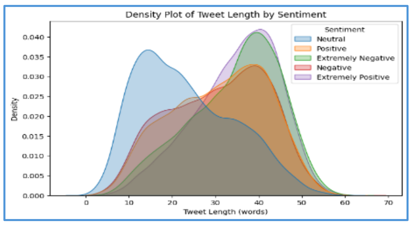

Figure 3. sentiment Density

4. Missing Location Values (Figure 4) - The proportion of missing location values is consistent across all sentiment classes, suggesting a Missing at Random (MAR) mechanism means location information is unlikely to provide predictive value for sentiment classification.

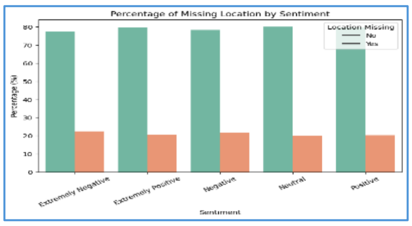

Figure 4. Missing Mapping

5. **The word clouds reveal key terms for each sentiment.** Across all classes, frequent words include "COVID," "coronavirus," "store," and "grocery," reflecting the general pandemic context and consumer behavior. Negative tweets emphasize shortage-related terms such as "panic," "crisis," and "supply." Neutral tweets are mainly factual, mentioning "price," "supermarket," and "stock." Positive tweets highlight support and gratitude with words like "thank," "help," and "safe."

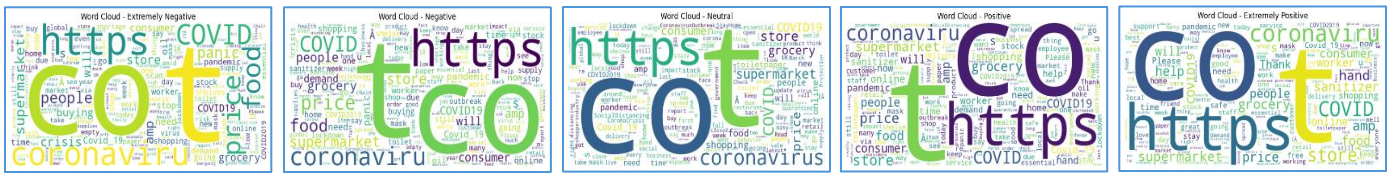

Figure 4. Clouds per Sentiment

These findings highlight that while sentiment intensity (moderate vs. extreme) is reflected in tweet length, polarity (positive vs. negative) is less separable based on this feature alone.

4. Preprocessing

To prepare the data for model training, we applied a structured preprocessing pipeline. This process included cleaning the raw tweets, engineering additional features for exploratory analysis, tokenizing the text for transformer-based models, and splitting the dataset into training, validation, and test subsets. The following subsections describe each step in detail.

4.2 Text Cleaning

The tweets were preprocessed using the following steps:

- Converted all text to lowercase.
- Removed URLs, mentions, and non-alphabetic characters.
- Applied tokenization and lemmatization.
- Removed English stopwords.

4.3 Feature Engineering

Additional features were created:

- TweetLength_raw - number of words in each tweet.
- LocationMissing - boolean indicator for missing location field.
- Stylistic - length, punctuation, capitalization ratio.
- Lexicons - AFINN (valence), NRC (emotions), MPQA (subjectivity).
- Readability - Flesch, SMOG, Dale-Chall.

4.4 Tokenization

For training, we used the model-specific tokenizers:

- DistilBERT tokenizer (distilbert-base-uncased).
- RoBERTa tokenizer (roberta-base).

Both tokenizers were applied with Maximum sequence length of 128 tokens, Padding to the maximum batch length & Truncation for tweets longer than 128 tokens.

4.5 Dataset Splitting

The dataset was stratified-split into training, validation, and test sets (80/10/10) to ensure consistent sentiment distribution across subsets.

4.6 Sentiment Labeling Decision

We kept five sentiment classes to retain the distinction between extreme and moderate responses.

The intensity of sentiment is meaningful for interpretation, and tweet length distributions differ between these subgroups.

Since tweet length is an important feature, merging classes could obscure valuable patterns.

5. Proposed Models

5.1 Model Selection

We selected two transformer-based models for fine-tuning:

- DistilBERT (distilbert-base-uncased) - A lightweight and efficient model, suitable for faster experimentation.
- RoBERTa (roberta-base) - A more robust model trained on larger corpora and optimized hyperparameters, often outperforming BERT.

Both models were fine-tuned using two approaches:

1. Manual PyTorch Training Loop – Provided full control over optimization, scheduling, and logging.
2. Hugging Face Trainer API – Simplified training and integrated seamlessly with W&B for experiment tracking.

5.2 Training Setup

The dataset was tokenized and split into training, validation, and test sets. Hyperparameter tuning was performed with Optuna 4 to optimize learning rate, batch size, and number of epochs. Experiment tracking was done using Weights & Biases (W&B), recording metrics such as accuracy and loss at each training step. For computational feasibility, the number of epochs and Optuna trials was deliberately kept low, as the dataset size made longer schedules prohibitively time-consuming while still allowing fair model comparison.

6. Experiments

6.1 Dataset

We used the preprocessed dataset described in Section 4, which includes cleaned tweets, engineered features, tokenized inputs, and five sentiment classes. The dataset was stratified-split into training, validation, and test sets (80/10/10).

6.2 Hyperparameter Tuning

We performed hyperparameter optimization using Optuna[4], which applies Bayesian optimization to efficiently search the hyperparameter space. For both DistilBERT and RoBERTa, the search included learning rate, batch size, and number of epochs.

We conducted three Optuna tuning experiments, each consisting of five epochs per trial. Weights & Biases (W&B) was used to track all runs, visualize parameter importance, and compare model performance. Across these experiments, we achieved improvements of approximately 3% in key metrics such as accuracy, F1-score, precision, and recall compared to baseline configurations.

6.3 Fine-Tuning

The table summarizes the final test-set performance metrics for DistilBERT and RoBERTa using both a manual PyTorch training loop and the Hugging Face Trainer API.
All metrics – Loss, F1 Score, Accuracy, Precision, Recall, and AUC – were computed on the held-out test set, while *Inference Time* was measured on the same data.

The results show that RoBERTa (Manual) achieved the highest accuracy and F1 score (≈96%), significantly outperforming DistilBERT. However, this came at the cost of a much larger model size and longer inference time.

Both training methods (manual vs. Trainer API) produced comparable results, though the Trainer API simplified experiment tracking and logging, while the manual loop provided more control over the training process and custom metric logging.

6.4 Results

Model	Training Method	Loss	F1 Score	Accuracy	Precision	Recall	AUC	Inference (sec)	Size (MB)
DistilBERT	Manual PyTorch Loop	0.5193	0.83498	0.8351	0.8359	0.8351	0.9702	6.89	254.14
DistilBERT	HF Trainer API	0.8077	0.8564	0.8567	0.8570	0.8567	0.9765	7.30	254.14
RoBERTa	Manual PyTorch Loop	0.1647	0.9609	0.9609	0.9612	0.9609	0.9961	13.57	476.51
RoBERTa	HF Trainer API	0.4761	0.8421	0.8421	0.8581	0.8421	0.9835	13.70	475.58

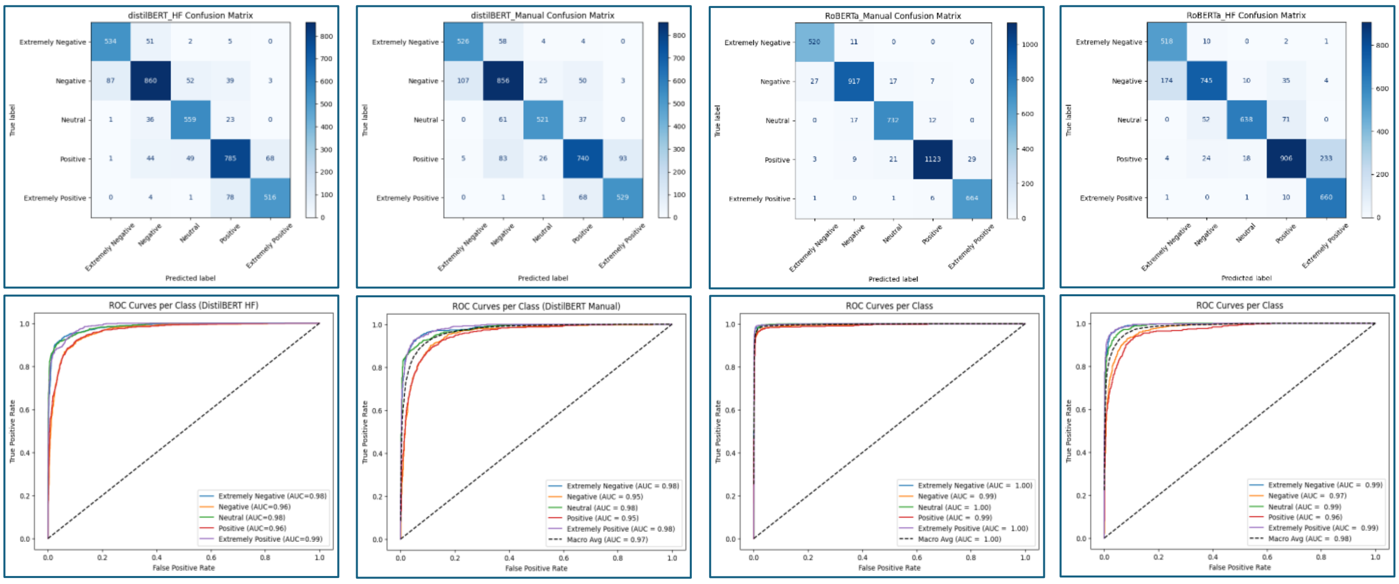

Figure 5. Confusion Matrices and ROC Curves for All Models

7. Model Compression

Following the evaluation of the models, we applied three compression techniques to the models.
Quantization (PTQ vs. QAT) - Post-training dynamic quantization (PTQ) reduced weights from FP32 to INT8. PTQ is fast and simple, though slightly less accurate than quantization-aware training (QAT), which requires fine-tuning.
Pruning - Magnitude-based unstructured pruning removed 50% of the lowest-magnitude weights. Sparsity rose from to 32.09% in DistilBERT and to 34.37% in RoBERTa, showing large weight reduction while keeping architecture intact.
Knowledge Distillation - A smaller TinyBERT-5 student was trained (without soft labels/temperature). The result is a much smaller and faster model, at some cost in accuracy.

Model	Compression Method	Loss	F1 Score	Accuracy	Precision	Recall	AUC	Inference (sec)	Size (MB)
DistilBERT (M)	-	0.5193	0.83498	0.8351	0.8359	0.8351	0.9702	6.89	254.14
DistilBERT (M)	**Quantization**	0.6261	0.8048	0.8051	0.8251	0.8051	0.9596	54.04	132.66
DistilBERT (M)	**Pruning**	0.7576	0.7096	0.7354	0.8032	0.7354	0.9473	14.40	254.14
DistilBERT (M)	**Distillation**	1.3498	0.4100	0.4189	0.4366	0.4189	0.7388	4.25	16.75
DistilBERT (HF)	-	0.8077	0.8564	0.8567	0.8570	0.8567	0.9765	7.30	254.14
DistilBERT (HF)	**Quantization**	0.8168	0.8528	0.8530	0.8538	0.8530	0.9759	54.04	131.96
DistilBERT (HF)	**Pruning**	0.4080	0.8598	0.8601	0.8613	0.8601	0.9787	5.46	254.14
DistilBERT (HF)	**Distillation**	1.2347	0.4888	0.4918	0.4915	0.4918	0.7838	2.56	16.75
RoBERTa (M)	-	0.1647	0.9609	0.9609	0.9612	0.9609	0.9961	13.57	476.51
RoBERTa (M)	**Quantization**	0.4988	0.8428	0.8428	0.8439	0.8428	0.9748	107.59	231.17
RoBERTa (M)	**Pruning**	0.4421	0.8397	0.8393	0.8424	0.8393	0.9722	7.26	476.51
RoBERTa (M)	**Distillation**	1.366	0.3622	0.3991	0.4766	0.3991	0.7250	4.43	16.75
RoBERTa (HF)	-	0.4761	0.8421	0.8421	0.8581	0.8421	0.9835	13.70	475.58
RoBERTa (HF)	**Quantization**	0.6818	0.8417	0.8420	0.8420	0.8420	0.9684	105.77	231.17
RoBERTa (HF)	**Pruning**	0.4800	0.8306	0.8309	0.8377	0.8309	0.9712	8.04	476.51
RoBERTa (HF)	**Distillation**	1.2745	0.4729	0.4765	0.4738	0.4765	0.7668	2.50	16.75

7.4 Comparison of Compression Methods

8. Discussion

Our experiments confirmed that fine-tuning transformer-based models on domain-specific data yields strong performance for sentiment classification tasks. RoBERTa consistently outperformed DistilBERT across accuracy, F1 score, and AUC but came with higher computational demands and a larger model size. The Optuna-based hyperparameter optimization provided notable gains (~3% improvement across metrics), emphasizing the importance of systematic tuning.

Model compression experiments demonstrated distinct trade-offs. Quantization offered the best balance between efficiency and accuracy, reducing model size by nearly half while maintaining strong performance. Pruning successfully removed 50% of the smallest-magnitude weights, resulting in sparsity levels of 32.09% for DistilBERT and 34.37% for RoBERTa, while preserving the architecture. However, pruning caused a moderate drop in accuracy. Knowledge distillation using TinyBERT produced a highly compact model but led to a substantial performance decrease, suggesting that additional techniques such as soft labels and temperature scaling could improve results. Although no data augmentation techniques were applied—given the dataset's size and the focus on baseline model and compression comparisons—future work could explore augmentation strategies to further improve robustness.

These outcomes align with trends in NLP research showing that compression methods can enable real-world deployment of transformer models on resource-constrained devices without excessively sacrificing accuracy. Future improvements could integrate advanced distillation strategies, quantization-aware training, or structured pruning to further optimize performance-efficiency trade-offs.

9. Conclusion

This study demonstrates the effectiveness of transformer-based models for sentiment classification of COVID-19-related tweets, enriched with additional meta features that capture stylistic, lexical, and readability aspects of the text. By combining these meta features with contextual embeddings, the models leveraged complementary information, leading to improved predictive power.

RoBERTa achieved the best performance, while DistilBERT provided a more computationally efficient alternative with competitive results. Optuna-driven hyperparameter tuning and W&B experiment tracking were instrumental in optimizing model performance. We also showcased how Hugging Face's Trainer API can be extended to handle multi-input architectures, integrating both text and meta data seamlessly. Among compression methods, quantization proved most effective for reducing model size without major accuracy loss, whereas pruning provided high sparsity but with a moderate performance decline. Distillation with TinyBERT produced a compact student model but with lower accuracy, highlighting the need for more advanced distillation strategies.

Overall, these findings provide practical insights for deploying transformer-based models in resource-limited environments, offering guidance on the trade-offs between accuracy, efficiency, and model size. Future work could focus on improved distillation techniques and few-shot learning approaches to enhance performance, particularly for rare extreme sentiment classes.

10. References

1. *Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.*
2. *Liu, Y., Ott, M., Goyal, N., et al. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.*
3. *Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.*
4. *Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework.*
5. *Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., & Liu, Q. (2019). TinyBERT: Distilling BERT for Natural Language Understanding.*