Classifying Covid-19 Related Tweets Sentiments Using Deep NLP Models and Compression Methods

Julia Yankovoy[1] and Yonatan Ghefter[2]

[1]Advanced Topics In Machine Learning

Due: August 22, 2025

Abstract

Understanding how people felt and reacted during the Covid-19 pandemic is important for public health messaging and policy. Twitter captures those reactions in real time, but the language is short, noisy, and often emotional. Our goal is simple: **given a tweet about Covid-19, predict one of five sentiment labels** (Extremely Negative, Negative, Neutral, Positive, Extremely Positive). We focus on two strong transformer encoders that are widely used for short text:

- `cardiffnlp/twitter-roberta-base-sentiment`
- `microsoft/deberta-v3-base`

For *each* model we train two baselines: (i) a standard HuggingFace (HF) Trainer setup and (ii) our own custom PyTorch loop with Optuna hyper-parameter search and W&B logging. This yields **four** baselines in total. Then, for *each* baseline, we create three efficient variants using **quantization**, **pruning**, and **knowledge distillation - twelve** compressed models overall. This design lets us compare accuracy and efficiency side by side and pick practical options for real deployments where memory or latency matter.

1 Introduction

The COVID-19 pandemic started in China in December 2019 by 31 of January 2020 it has spread to multiple countries, quickly infecting the population at an alarming rate and by the end of March that same year there were more than 100,000 confirmed cases in the United States alone. This global pandemic was a unique situation for many people as such a pandemic did not occur for the past 50 years before (the hong kong flu), as such the situation amplified people's already widespread use of social media platforms. The pandemic has sharply increased the number of daily posts on Twitter, rising from its November 2018 low point just above 320M tweets a day to an almost all-time high of just shy of 500M tweets per day [1]. People used the social media platform to express many messages, both complain about the day-to-day situation of living during a pandemic and to post helpful instructions about pandemic related conducts, give negative or positive predictions of what they think will happen in the coming days or posted about unrelated topics. In 2020 Twitter had a very big user base, 18.9% of the United States population were registered users, 38.8% in Japan and 22.9% in the United Kingdom [2], as such using posts on social media can provide useful data for analyzing public opinion. Transformer-based machine learning models have proved instrumental in the field of natural language processing, models like BERT [3] RoBERTa [4] and DeBERTa have shown excellent abilities in understanding human language, its complex patterns and more specifically the meaning behind it which makes them a superb tool for analyzing public opinion and large-scale sentiment analysis. These Models use Deep-learning methods that have proven effective

for solving a wide range of issue, allowing them to understand the intricacies of human language and determining the sentiment and ideas behind it to high accuracy. Nonetheless the increasing demand for the use of these models is also computationally taxing, since each model on each computer can only work on one query at a time and some models are only purposed for one task, the need for faster and smaller yet accurate models is on the rise. computational and memory efficiencies are a major area in computer science both for theoretical and practical pursuits, as such various ways have been developed in order to make models more efficient, pruning, quantization, knowledge distillation to name a few, aim to reduce the computational and memory impact of models without harming their performance too much. This research seeks to implement Transformer-based models on a covid-related NLP task to be able to provide insight into public opinion at the beginning of the pandemic and performs small benchmarking on different aspects of the finetuning and model compression process.

2 Related Work

The area of NLP research for social media is too fragmented according to Barbieri F. et al. [7], "Each year, new shared tasks and datasets are proposed, ranging from classics like sentiment analysis to irony detection or emoji prediction. ", since transformers-based models have shown excellent capacity in performing NLP tasks relative to other types of models Barbieri et al. aims to create a unified way of testing social media based NLP tasks and seeks to also build strong benchmarks for future research and development. They compare multiple classic NLP models along with multiple methods of training a RoBERTa-base model, their most effective model was a RoBERTa fine-tuned on Tweeter data, this exemplifies the effectiveness of fine-tunning and of transformers-based models, and gives us a stronger background for the approach we took in our sentiment analysis task. When dealing with a classification tasks common evaluation metrics are accuracy, F1-score, recall, precision, ROC curve, root mean square error (RMSE), BCE Loss and more according to [8]. In their review of social media-based sentiment analysis they point out to the fact that some of these metrics assume a certain cost difference ratio between classes and results that is not necessarily true to the problem in certain tasks, for example accuracy assumes that the cost difference between correctly classifying an instance and incorrectly classifying an instance is the same in each category. Moreover they critique the sole use of these metrics for not considering efficiency, sentiment analysis is a useful tool for governments, businesses and more, hence they claim, there's a need to develop metrics that consider efficiency of models, both in terms of training and inference.

3 Dataset and EDA

We use the Kaggle dataset *Coronavirus Tweets NLP* (link). The combined train+test files include **about 45,000 rows**. Tweets were pulled from Twitter and **manually tagged**. The main columns are: `Location`, `Tweet At`, `Original Tweet`, and `Label`.

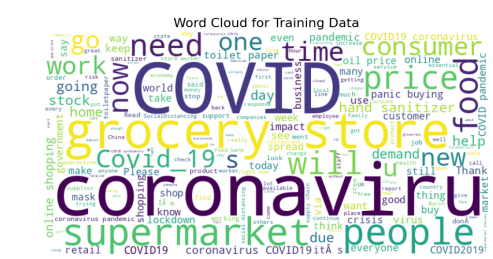
Labels. The task is **five-class** sentiment classification with the following categories:

- *Extremely Negative*, *Negative*, *Neutral*, *Positive*, *Extremely Positive*.

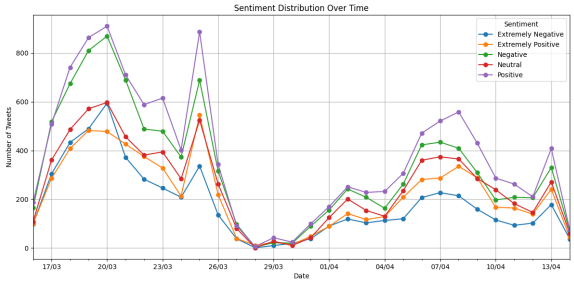
Split & preprocessing. We tokenize with the model-specific tokenizers. Max sequence length and exact split strategy remain as in our training setup. We also clean the dataset as part of our preprocessing, we noticed the letter Â appears frequently with apostrophes in the data. This character can often be found as an artifact of conversion between encodings [6] so we decided to remove it from our data.

Exploration. The word cloud highlights that pandemic-related terms such as "coronavirus", "covid", and "grocery store" dominate the dataset. Sentiment distribution over time reveals peaks corresponding to major pandemic events. Tweet length analysis shows most tweets

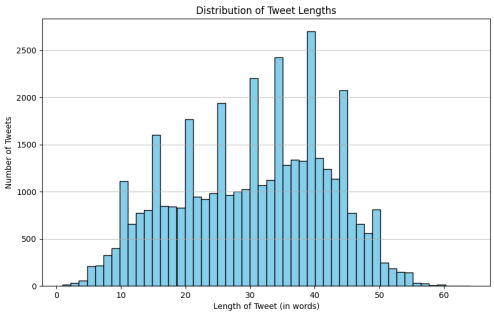
fall between 20–45 words, with neutral and extremely negative tweets generally shorter, while positive and extremely positive tweets are longer, reflecting stronger sentiment expression.



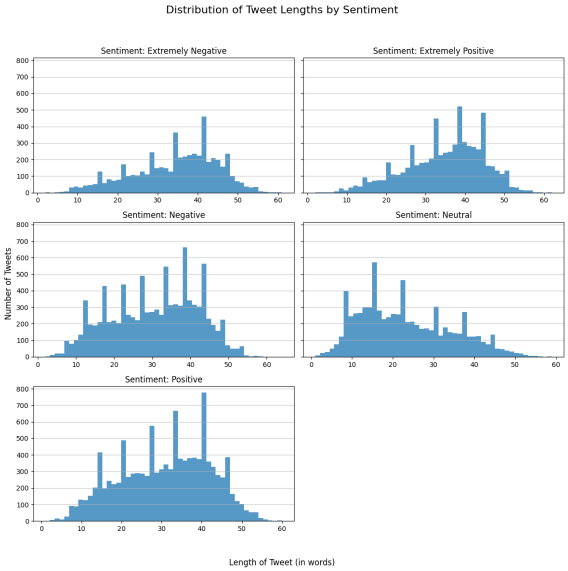
(a) Word cloud



(b) Sentiment over time



(c) Tweet length distribution



(d) Length by sentiment

Figure 1: Exploratory data analysis: vocabulary, sentiment trends, and tweet length characteristics.

4 Proposed Models

To classify Covid-19 tweets into five sentiment labels, we keep the pipeline simple and practical. We work directly with the `Original Tweet` text from the Kaggle dataset and tokenize using the model's fast tokenizer. Emojis and punctuation that carry sentiment are kept. The classifier head is a standard linear layer on top of the final hidden state, trained with cross-entropy for the five classes: *Extremely Negative, Negative, Neutral, Positive, Extremely Positive.*

We produce **four baselines**: for each backbone (RoBERTa and DeBERTa) we train (i) a HuggingFace (HF) Trainer version and (ii) a custom PyTorch version. The custom loop uses Optuna for hyper-parameter search (15 trials, up to 20 epochs, early stopping with `patience=5`) and logs runs to Weights & Biases. The search space covers

$$\text{learning\_rate} \in [1\times10^{-5}, 5\times10^{-5}], \quad \text{weight\_decay} \in [5\times10^{-3}, 3\times10^{-2}],$$

$$\text{batch\_size} \in \{16, 32, 64\}, \quad \text{num\_layers} \in \{2, 3, 4, 5\}.$$

4.1 Model Selection

We focus on two strong, complementary encoders:

RoBERTa (`cardiffnlp/twitter-roberta-base-sentiment`) is pretrained and adapted for Twitter-style text, which is short, informal, and emoji-rich. This domain match often translates into stronger out-of-the-box performance on tweet sentiment.

DeBERTa (`microsoft/deberta-v3-base`) is a powerful general-purpose encoder with disentangled attention and improved training efficiency. While not tweet-specific, it can generalize well after fine-tuning.

Training both gives us a fair comparison between a tweet-tuned encoder and a strong general encoder. For each, we include both HF Trainer and our custom training pipeline to ensure results are not an artifact of one training stack.

4.2 Model Compression Techniques

Our deployment goal is practical: reduce memory and latency without losing much accuracy. Following the evaluation of our four baselines, we applied three well-established compression methods to *each* baseline, yielding twelve compressed models in total. These techniques were chosen to reduce memory footprint and inference latency while maintaining competitive accuracy.

Quantization. We applied post-training quantization (PTQ), converting model parameters from 32-bit floating point (fp32) to 8-bit integer (INT8) representations. This approach reduces model size by roughly a factor of four and can significantly improve inference speed on CPU while keeping accuracy degradation modest.

Pruning. We performed magnitude-based pruning, removing 30% of the lowest-magnitude weights across the network. After pruning, we carried out brief recovery fine-tuning to regain some performance. This method reduces the number of active parameters and can yield gains in efficiency, though with some risk of reduced recall.

Distillation. Lastly, we adopted a knowledge distillation approach, training a smaller student model to replicate the behavior of the larger teacher model. For RoBERTa, we used DistilRoBERTa as the student; for DeBERTa, a compact variant such as DeBERTa-v3-small. The student is trained with a combined objective that blends the original cross-entropy loss with a soft-target KL divergence loss from the teacher. This method results in a lighter model that is faster to deploy while retaining much of the predictive power of the full-sized teacher.

5 Experiments and Results

5.1 Model Zoo

Table 1: Overview of models.

Baselines (4)	RoBERTa (HF), RoBERTa (Custom), DeBERTa (HF), DeBERTa (Custom)
Compressed (12)	For *each* baseline: INT8, Pruned, Distilled

5.2 Custom baselines: best validation metrics

Table 2: Best single-run validation metrics (custom training).

Model	F1	Acc	Prec	Recall	AUC	Loss
RoBERTa (trial_11)	**0.7420**	0.7332	0.7360	0.7538	0.9378	0.7576
DeBERTa (trial_11)	0.7352	0.7252	0.7321	0.7390	0.9282	1.4486

Table 3: Hyperparameters of the best custom runs.

Model	Batch	LR	Weight Decay	num_layers	Epochs	Patience
RoBERTa (trial_11)	64	4.72e-5	0.0296	5	20	5
DeBERTa (trial_11)	32	4.95e-5	0.00543	5	20	5

5.3 HF Trainer baselines

Table 4: Best single-run validation metrics (hf training).

Model	F1	Acc	Recall	AUC	Loss
RoBERTa (trial_15)	0.8684	0.8649	0.8651	0.9688	1.0057
DeBERTa (trial_8)	0.8856	0.8823	0.8873	0.9836	0.4468

Table 5: Hyperparameters of the best hf runs.

Model	Batch	LR	Weight Decay	Epochs	Patience
RoBERTa (trial_15)	8	4.68852e-05	0.03422	20	8
DeBERTa (trial_8)	8	3.4226e-05	0.160900	20	8

5.4 HF vs. custom training

In our custom pipeline we used Optuna to vary the number of unfrozen transformer layers in the range 1–5 because attempts to unfreeze more than five layers repeatedly stalled training. Within that range, "more unfrozen layers → better validation scores," which is expected: deeper adaptation lets the model specialize to our tweet domain. However, the HuggingFace (HF) Trainer consistently outperformed our custom loop across both backbones. The most direct reason is that HF fine-tuning, unless told otherwise, trains all layers end-to-end, while our search capped unfreezing at five layers due to stability issues. In short, HF likely won because It fine-tunes the entire network by default, Its training recipe is very stable for deep fine-tuning (AdamW + linear warmup/decay, gradient clipping, mixed precision) and It sustains a healthy effective batch size (via accumulation and efficient dataloading), which helps convergence.

5.5 Compression results (16 models)

Table 6: Efficiency impact of compression on RoBERTa and DeBERTa across training methods. Model size (MB) is estimated from nonzero parameters and datatype (fp32=4B, fp16=2B, int8=1B).

Model		**RoBERTa**		**DeBERTa**	
Training	**Variant**	**#Params (M)**	**Size (MB)**	**#Params (M)**	**Size (MB)**
HF	Teacher	124.64	475.52	184.41	703.55
HF	Quantized	123.39	120.12	197.43	233.10
HF	Pruned	98.99	475.52	158.75	703.55
HF	Distilled	82.12	313.28	141.89	541.31
Custom	Teacher	124.6	∼498	184.4	∼738
Custom	Quantized*	39.0	∼39	98.8	∼198
Custom	Pruned	99.2	∼397	158.9	∼636
Custom	Distilled	82.1	∼328	141.9	∼568

*Quantization: RoBERTa used INT8, DeBERTa used FP16 in Custom trained models due to severe accuracy loss with INT8.

Table 7: Comparison of RoBERTa and DeBERTa across training methods and compression variants.

Model		**RoBERTa**				**DeBERTa**			
Training	**Variant**	**Acc**	**F1**	**Recall**	**AUC**	**Acc**	**F1**	**Recall**	**AUC**
HF	Teacher	0.8362	0.8408	0.8394	0.9610	0.8512	0.8564	0.8633	0.9784
HF	Quantized	0.8344	0.8391	0.8360	0.9613	0.6074	0.5955	0.5972	0.8915
HF	Pruned	0.7727	0.7632	0.7486	0.9461	0.7638	0.7512	0.7347	0.9623
HF	Distilled	0.8333	0.8403	0.8403	0.9715	0.8641	0.8682	0.8710	0.9805
Training	**Variant**	**Acc**	**prec**	**Recall**	**AUC**	**Acc**	**prec**	**Recall**	**AUC**
Custom	Teacher	0.7227	0.7247	0.7483	0.9354	0.7252	0.7321	0.7390	0.9282
Custom	Quantized*	0.7090	0.7146	0.7248	0.9271	0.7252	0.7321	0.7390	0.9282
Custom	Pruned	0.5807	0.7141	0.5274	0.8963	0.6722	0.7344	0.6625	0.9090
Custom	Distilled	0.7994	0.8010	0.8218	0.9646	0.8548	0.8566	0.8653	0.9767

6 Discussion

When examining the compressed custom models, several insights emerge. The baseline teacher models behaved as expected, with DeBERTa slightly outperforming RoBERTa across most metrics. Quantization showed mixed results: while RoBERTa handled INT8 quantization with only a small drop in accuracy and AUC, applying INT8 to DeBERTa led to severe performance collapse. To overcome this, we switched DeBERTa to FP16 quantization, which preserved results nearly identical to the teacher model. This contrast highlights that quantization robustness is highly architecture-dependent. Pruning, on the other hand, consistently degraded performance in both backbones, reflecting the loss of representational capacity when many weights are removed without extensive retraining. Surprisingly, distillation produced the best results overall, with distilled students outperforming their larger teacher models in both training backbones. This counterintuitive effect likely stems from the regularization benefits of distillation, where students learn from softened teacher outputs that encourage better generalization, a particularly useful property in our noisy Twitter dataset. Using the custom training setup provided more freedom for making custom adjustment to the training process such as freezing layers in

the model or integrating with W&B, the Hugging Face API for model training is a high level version of the model training process as such doing custom alterations using it such as implementing a knowledge distillation process for two models that use different tokenizers proved rather difficult.

Decision Procedure. In our pursuit to make our models better we also suggest a new procedure for label prediction: For each class c we select a threshold τ_c via one-vs-rest accuracy maximization. Given probabilities $\{p_c\}_{c=1}^{K}$ for a new observation:

$$V = \{c \mid p_c \geq \tau_c\}.$$

If $V \neq$, assign

$$\hat{y} = \arg\max_{c \in V} \left(\frac{p_c}{\tau_c} \cdot \frac{f_c}{\sum_{j \in V} f_j} \right),$$

where f_c is the class frequency. Otherwise,

$$\hat{y} = \arg\max_{c} p_c.$$

This procedure showed some improvements over the default argmax prediction method on the test data, since the models' performances were already quite high the effect wasn't significant further experimentation is need but even a small improvments can count toward meaningful results in classification problems.

Git repository: `https://github.com/jigima/deep-tweet-covid.git`

References

[1] The GDELT Project blog `https://blog.gdeltproject.org/visualizing-twitters-evolution-2012-2020-and-how-tweeting-is-changing-in-the-covid-19-e` 11 August 2020.

[2] Yago Martin et al. "Towards real-time population estimates: introducing Twitter daily estimates of residents and non-residents at the county level" - Scientific Figure on ResearchGate. Available from: `https://www.researchgate.net/figure/Countries-with-the-most-Twitter-users-in-2020-and-penetration-rate-Source-Statista_tbl1_346475194` [accessed 21 Aug 2025]

[3] Devlin, Jacob et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." *North American Chapter of the Association for Computational Linguistics* (2019).

[4] Liu, Yinhan et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach." *ArXiv* abs/1907.11692 (2019): n. pag.

[5] He, Pengcheng et al. "DeBERTa: Decoding-enhanced BERT with Disentangled Attention." *ArXiv* abs/2006.03654 (2020): n. pag.

[6] Texin, Tex. "UTF-8 Encoding Debugging Chart" Tex Texin, © 2011, UTF-8 Character Debug Tool . Accessed 22 Aug. 2025.

[7] Barbieri, Francesco, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. "TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification." Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, 2020, pp. 1644–1650. Online.

[8] Xu, Qianwen & Chang, Victor & Jayne, Chrisina. (2022). A systematic review of social media-based sentiment analysis: Emerging trends and challenges. Decision Analytics Journal. 3. 100073. 10.1016/j.dajour.2022.100073.