Term Paper: Twitter Sentiment Analysis with RoBERTa Model

by Tatiana Bakwenye, Sebastien Boxho, Mikel Gallo and Joaquin Ossa

INTRODUCTION

Objective: Develop a smaller neural network capable of competing with a Large transfer-learning Model.

Context: Transfer learning typically involves training a model on a vast dataset to address a more generic range of tasks, followed by fine-tuning it on a smaller dataset for a more specific task. In our project, this would translate to loading a pre-trained Large Language Model (LLM) such as RoBERTa and fine-tuning it for sentiment analysis on a corpus of Tweets, classifying them as positive or negative.

Main problem: BERT and similar models usually require a lot of computing power, work relatively slowly, and have a large number of parameters. For some NLP tasks, fine-tuning BERT may seem like using a tool that's far too powerful or complex for the relatively simple job at hand.

Relevant methodologies: Hinton et al. proposed to improve machine learning performance by training multiple models on the same data and then averaging their predictions. However, deploying an ensemble of models can be computationally expensive. Caruana et al. demonstrated a technique to compress the knowledge from an ensemble into a single model, making it easier to deploy.

Proposed solution: We'll use Distillation to train a smaller model to mimic the behavior of a larger one. The aim is to achieve comparable performance while significantly reducing computational overheads and improving efficiency.

Data: Twitter Sentiment Analysis Training Corpus (TSATC)

Original DATASET contains 1,578,627 classified tweets (1 for positive and 0 for negative sentiment). Our dataset has already been randomly sampled, cleaned, and split into training and testing sets. Both positive and negative classes are well balanced within each subset, and the training subset has been further divided into an 80% training set and a 20% validation set. We will use 5% of the original dataset due to computational constraints.

Business Case: Twitter Sentiment Analysis for Brand Monitoring

Problem: Businesses need to understand public perception of their brand on Twitter to manage reputation effectively.

Solution: Implement Twitter sentiment analysis to monitor brand mentions and interactions in real-time.

State of the Art: A Transformer Based Model for Twitter Sentiment Analysis using RoBERTa

In their research Nazmul Abdal et al. present a novel approach to sentiment analysis on Twitter using RoBERTa. By fine-tuning on annotated Twitter data, the system achieves exceptional accuracy of 96.78%, surpassing traditional machine learning models.

We based our experiment on the same dataset used in this research, implementing a RoBERTa model for sentiment classification on Twitter data. However, it's essential to note that in SOA research, authors were classifying tweets as either sexist/racist or not. In our case, we focused on sentiment classification to determine whether a tweet is positive or negative. As a result, it's reasonable to expect that we may not achieve the same level of performance as the state of the art.

SUMMARY OF THE TASKS

The code is split into two notebooks with a repeated preprocessing section. This segmentation is necessary to utilize the maximum GPU capacity available in Premium Google Colab.

Section 1: Baseline implementation

Our first task was to establish a baseline for our project. To accomplish this, we developed two straightforward classification models. Firstly, we constructed a **random baseline model** achieving an average accuracy of 50%. Secondly, we created a **rule-based classification model** utilizing positive and negative patterns, extracted from text mining techniques like tf-idf and topic modeling with LDA, achieving an average accuracy of ~64%.

Twitter data presents certain challenges:

- Frequently, we encounter expressions and or questions that lack clear sentiment.
- At times, when the number of positive and negative instances is equal, determining the true sentiment becomes challenging. In such cases, we classify them as negative.
- Given the intricate variety of positive patterns that exist in social media (e.g emojis, media slang, etc) it would be very difficult to cover all possibilities.
- Positive and negative sarcasm further complicates accurate classification, as seen in examples like: "I live in a basement. That's why I am so mysterious. (class: 0)"

Proposal to reduce False negatives

To address the issue of false negatives, we suggest a strategy suited to the complexity of tweets, characterized by social media slang. Our plan involves conducting regular error analyses and augmenting our matcher patterns with new positive patterns. This iterative approach aims to develop a robust classifier capable of capturing all positive patterns accurately over time. However, we must also devise methods to handle neutral tweets and positive/negative sarcasm. By implementing this strategy, we were able to improve accuracy (60% → 65%) reducing false negatives by 15%.

Comparison with Rule-Based Vader for Sentiment Analysis

Furthermore, we integrated another rule-based model known as Vader, specifically designed for calculating sentiment in social media text. Our baseline implementation using Spacy yielded results consistent with this established rule-based benchmark model.

Section 2: Enhancing model performance with limited labeled data

BERT Model with Limited Data

In many real-world cases, labeled data may be scarce. In this exercise, we aim to assess the model's performance using a very limited dataset. To accomplish this, we will fine-tune our model on a random subset of 32 observations from our training dataset.

We chose a RoBERTa-base model trained on approximately 124 million tweets spanning from January 2018 to December 2021. It has been fine-tuned for sentiment analysis using the TweetEval benchmark. Note that our initial attempt involved using a distilled version of the BERT base model. However, distilled BERT showed significantly poorer performance, likely because it was trained on more generalized data.

We then implemented several techniques to increase our training set size:

Dataset Augmentation

We considered the following data augmentation approaches: synonym replacement, random insertion, random deletion, random swap, text masking and so on. Given the small size of our dataset, we chose not to employ deletion or masking techniques, but instead focused on using synonyms. We implemented two approaches:

Our first approach involved manually gathering synonyms by analyzing the most common words associated with positive and negative sentiments. We then created a dictionary of synonyms and augmented the data with 50 additional observations. We experimented with augmenting more

data, but found that performance was worse compared to 50 additional observations, likely due to the limited variety of changes and the risk of misclassifying observations multiple times.

As manually assigning synonyms to many tokens proved inefficient, we decided to explore WordNet package. However, WordNet may not accurately reflect the meaning of the word, especially considering its multiple meanings and lack of contextual consideration. In the second approach, we randomly changed 30% of the tokens in the comments (after experimenting with different shares, 30% yielded better results) and added 32*20 new rows (again, experimenting with different numbers).

Zero-Shot Learning with LLM

Doing some research on Hugging Face, we found multiple LLM models that we could try using to help us generate more data for our model. Ultimately, we decided to use the model 'facebook/bart-large-mnli', which had an approximate accuracy of 80%. So, using a small sample from our data validation set, we obtained new labeled data with Zero-Shot, which we later joined with the 32 labeled data to improve our initial Roberta model.

Data Generation with LLM

For this task, we used ChatGPT from OpenAI to generate a sample of 1000 tweets, both negative and positive. After multiple attempts, we managed to get the data we needed, but we realized that the text generated by it was all very simple examples and very easy to predict. To capture more complex patterns, we prompted sarcastic and ironic tweets as well.

Optimal Technique Application

In the following table we presented the results of our model using various techniques. Among them, the model with Augmented data WordNet and Zero-Shot performed were usually the ones that performed the best. We believe this result is reasonable because the other techniques had limitations.

Augmented data, generated by adding a limited number of synonyms, is highly similar to the existing data and provides minimal new information.

WordNet synonyms augmentation provides a higher variation in generated data and allows for more observations, though it is also not infinite.

Additionally, data obtained from OpenAI proved to be too simplistic and easily labeled, rendering it ineffective for improving model performance.

Finally, Zero-Shot learning technique is limited only by the validation set, and the classification itself may be biased at the level of the performance metrics. Thus, for the Zero-Shot we saw that,

by itself, it already had a good performance (accuracy of 76%) when predicting tweets, so the data added to our model, would be more extensive and although it might have some miss labeled cases, the diversity and amount of data it provides makes it the best option for our model.

	accuracy	precision	recall	f1
32 labels	69.47	74.94	69.35	67.62
synonyms	70.60	74.53	70.50	69.31
synonyms (wordnet)	76.90	77.33	76.87	76.79
zero-shot	76.09	77.87	76.03	75.67
LLM generated data	74.64	75.44	74.68	74.46

Considering all these limitations, we will not focus on a particular model but rather aim to implement further analysis by incorporating all techniques.

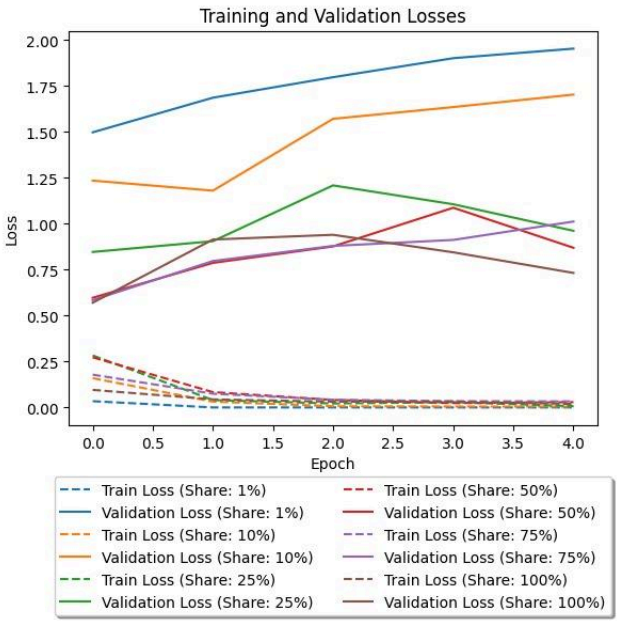
Section 3: State of the Art Comparison

Full Dataset Training

We fine-tuned the RoBERTa-base model on an increasing percentage of the dataset, which led to performance enhancements. However, we observe diminishing returns in performance gains as more data is incorporated. This suggests that beyond a certain point, additional training data may offer diminishing benefits, particularly if the pre-trained model already adequately represents the relevant linguistic features.

Learning Curve

To gain insight into the learning process of the models, we visualized the loss functions for both the training and validation sets across the epochs. It's evident from the graph that for all the shares of data convergence begins as early as the second epoch, which serves as the baseline for subsequent models. Besides, we observe that while the loss for 1% and 10% of the data is significantly higher, the difference is not as pronounced between shares of 25% to 100%.



Note that the shape of the loss curves we obtained is quite similar to SOA (check figure 7).

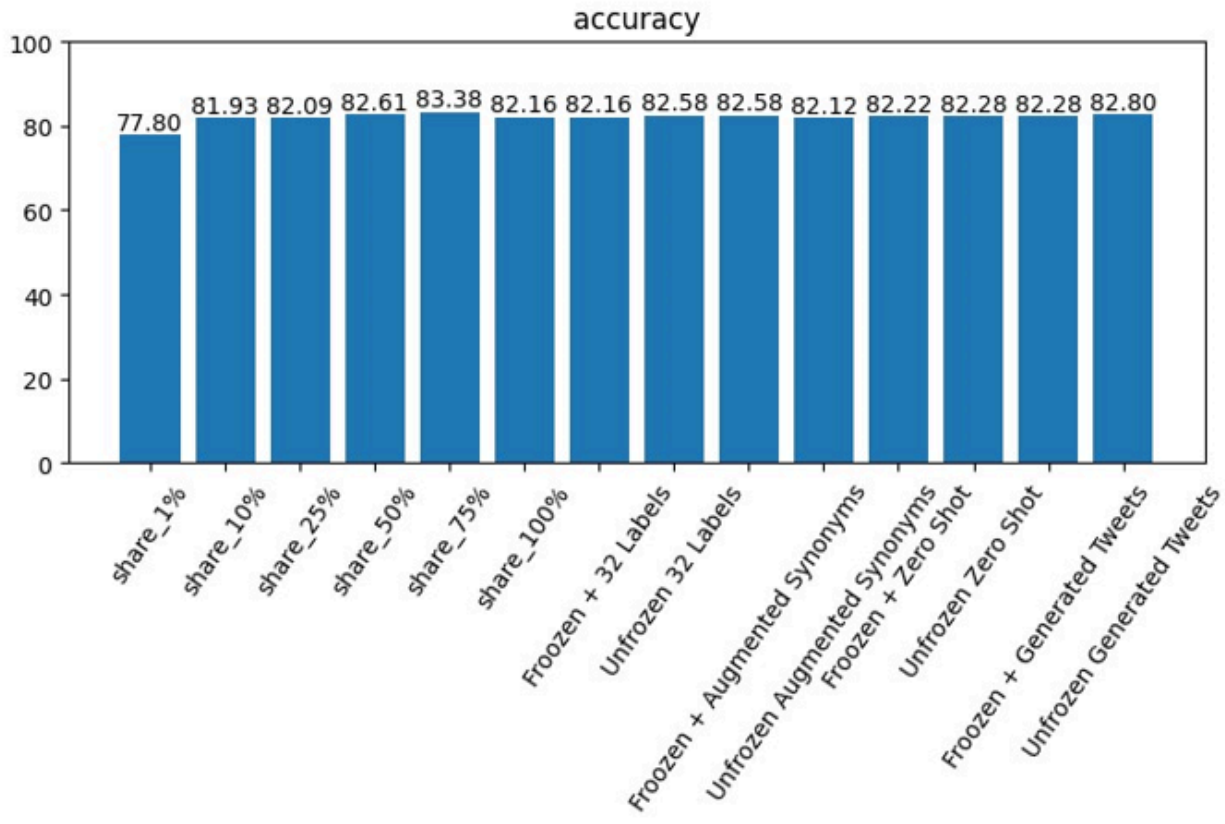
Technique Comparison

To compare the various data augmentation techniques, we opted to implement a neural network model for each of them. Instead of solely appending the generated data to the training data (as we assumed this would only have a minimal impact), we decided to freeze the BERT model and employ one of the data generation techniques to train the frozen model. Subsequently, we utilized the original training data to train the unfrozen model.

This approach enables us to compare the impact of each data generation method on the final model. Additionally, within this loop, we retained the best-performing model (measured by loss) to be used as a teacher model in the subsequent stages of the project.

Methodology Analysis

The results obtained from implementing different techniques indicate that the model's performance generally slightly improves as the dataset size increases. Interestingly, the augmentation techniques used in part II show a similar increase in accuracy compared to simply increasing the size of the actual training dataset. This observation could support the methodology we used to train the frozen model. We see from the graph , that expanding the dataset size again did not result in a substantial improvement in the model's performance.



Part 4: Model Distillation

Model Distillation

For our distilled model we trained our bert model using "distilbert/distilroberta-base" to improve the computational efficiency.

The Distiller class in Keras facilitates knowledge distillation by training a smaller model (the "student") to mimic a larger, more accurate model (the "teacher"). Note, that the architecture of the student model is the same as teacher model. It achieves this by combining two loss functions: the standard student loss and a distillation loss, which measures the discrepancy between the softened predictions of the teacher and student models. The distillation loss encourages the student to mimic the behavior of the teacher, aiding in model compression and speedup during inference on resource-constrained devices. The class's methods handle model initialization, compilation, loss computation, and forward pass, making it a convenient tool for implementing knowledge distillation techniques.

Performance and Speed Comparison

Comparing our previous model, with the distilbert we can see an improvement both in time and in memory of around 30%. Having at the beginning around 124 million parameters on the model, we managed to reduce it to 82 million (Almost 500 MB to 310 MB) which also helped with the speed time of the model.

The student model lost some accuracy but still performs quite well. With a teacher model accuracy of 81.22, the student model achieved 79.77.

Analysis and Improvements

Our student model may face some deficiencies in learning process due to several issues:

Biased Training Data: if the training data is biased, the student model may inherit and perpetuate these biases. We assume that the bias in our case could be partially transferred from the augmented data. Employing debiasing algorithms could help mitigate this issue. Augmentation of the training data with various transformations could expose the student model to a diverse range of examples as well.

Hyperparameter Tuning: suboptimal choices of hyperparameters such as learning rate, batch size, or optimization algorithm could hinder the learning process. Conducting systematic hyperparameter tuning experiments could optimize model performance. In our case, we were limited by computational power, as, for instance, we couldn't use a batch size higher than 8.

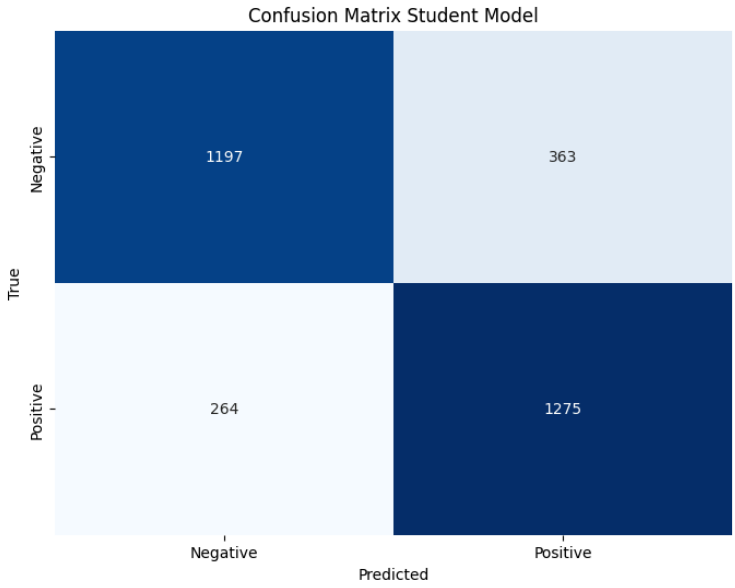
Limited Representational Power: the student model may struggle to represent the nuanced features and relationships present in the data. Pre-training the student model on a larger dataset or utilizing transfer learning from a pre-trained model could enhance its representational power. But again we were limited by computational power.

FINAL REMARKS

Comparing the results of our model with the SOA, we observe that our performance is approximately 10 percentage points worse than the SOA. Since we used the same RoBERTa model with similar hyperparameter tuning, the difference could be mainly attributed to:

- Differences in the actual target variable (in SOA, the authors classify whether the comment is sexist/racist or not).
- The data in the SOA model is highly unbalanced, with 91% of the data classified as true positive, whereas our data is balanced.
- Furthermore, when we tested the RoBERTa model from Hugging Face, it achieved an accuracy of around 80%, slightly lower than our performance.

The entire exercise turned out to be computationally expensive. Although we contemplated enhancing our model by augmenting the training data and diversifying it further, our analysis revealed that increasing the dataset size did not substantially improve the model performance.



For future improvement we considered the potential of enhancing data augmentation techniques with specialized methods tailored for Twitter sentiment analysis, which take into account unique features such as hashtags, mentions, emojis, and informal language. It would also be potentially interesting to explore multi-class sentiment analysis, where two approaches could be implemented: either considering different strengths of positive and negative sentiments, or including different emotions as multiple classes.