Engineering Systems & Design

TAE Kaggle Competition Report

Perform Sentiment Analysis on Tweets

Team 17

Yong Zhe Rui Gabriel 1005154

Lim Azib Bin Adam 1005209

Ong Kai Xin Chloe 1005517

1 Introduction

The goal of the Kaggle Competition was to perform sentiment analysis on tweets, a Natural Language Processing (NLP) task.

Our group approached this problem utilizing a transformer neural network and was able to predict the sentiments on the test set with an accuracy of 0.805 on the public leaderboard and 0.794 on the private.

2 Approach

Our group aimed to maximise our model's accuracy. Hence, to identify the best model with the highest accuracy, we did the following:

1. Use the basic models taught in class
2. Tune our best model
3. Explore more complicated models to use

2.1 Understanding the data

Upon getting hold of the data, our group sought to understand if there was any topical tendency in the dataset. However, from the word cloud (Figure 1) generated using our training data, we see that there was no central theme to the text and there was no need to make our pre-processing specific.

We also took a look at the proportion of each sentiment shown in Table 1, and noted that the neutral sentiment had the most with 0.404. Hence, our goal was to make a model that had at least 0.404 accuracy.

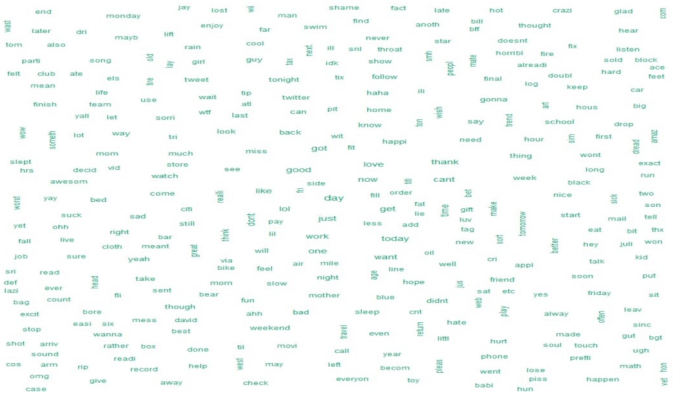

Figure 1: Word Cloud of higher frequency words

2.2 Initial Attempt with Basic Models

Out of the models taught in class, we used Naïve Bayes classifier, CART, Multinomial Logistic and Random Forest as they were the most applicable to the context of the competition. After our initial test, setting the sparsity of the document term matrix (DTM) to 0.9995 ($\sim$ 1600 features), the accuracies when we did a train-test split on the training data of the Naïve Bayes Classifier, CART, Multinomial Logistic and Random Forest were 0.38, 0.66, 0.66 and 0.70 respectively. Hence, we decided to focus on the Random Forest model as it has the highest accuracy.

We then identified 2 main ways we could improve our accuracy: 1) Pre-processing/feature engineering and 2) Utilising models not covered in class (such as neural networks). We used a divide and conquer approach to explore these methods and implemented whichever looked more promising and yielded results first.

Negative	Neutral	Positive
0.283	0.404	0.312

Table 1: Proportion of Sentiments

2.3 Improving Random Forest

From our understanding in class, we tuned our Random Forest model using these 2 hyperparameters: the number of trees and the number of predictors used in each tree.

We also varied the sparsity of the DTM, limiting the number of features to prevent overfitting and to find the optimal number.

Furthermore, we explored other approaches such as improving our pre-processing methods and feature engineering.

2.3.1 Improved Pre-processing

To better the accuracy of our model, we used the textclean package to clean the tweets more effectively. The package would allow us to remove more specific parts of the tweets such as URLs, hashtags, emoticons, abbreviations, etc.

2.3.2 Feature Engineering

Moreover, as certain characteristics of the data might not have been captured in the cleaning. We created our own features for our model to train on, as listed below:

- Percentage of upper case in a sentence
- Explicit (****, replace with explicit)
- ???..., replace with qstn (used a shortened version of the word so there is a distinction between the word and the punctuation)
- !!!..., replace with exclm

Additionally, we tried using N-grams to better predict the tweet's sentiments. However, before completing it, the more complex models were already implemented and had better accuracies than the N-grams and feature engineering. Thus, we decided to focus on the complex models instead.

2.4 Using More Complex Models

Since sentiment analysis is a NLP task, we knew that there were models better suited for analysis, outside of the module's scope. Models such as Recurrent Neural Networks (RNN) were used for sequential analysis of sentences.

However, RNNs are slow to train due to their sequential manner and may miss out on key information as it is unidirectional. Thus, we implemented the Transformer Neural Network model, as it is the better model as proposed in the article, "Attention is all you need" (Vaswani et al., 2017).

2.4.1 What is a Transformer? (A Brief Overview)

A transformer is a model that is built around the concept of attention, specifically self-attention.

The model is able to evaluate a string of words for its input. It does not only take in the 1 or 2 words adjacent to the focus word, but takes in the whole sentence containing the word. It then evaluates how much "attention" it should pay to the other words in the sentence. Thus, this allows the model to better understand the context of the sentence.

In addition, when taking in a sentence, it also adds positional encodings to each word which allows the model to understand how different positions of words may have different or similar meanings.

2.4.2 Pre Training and Transfer Learning

Due to the non-topical nature of the data, we opted to use pre-trained models as it would be simpler for us to incorporate our desired model. Since they would have already been tuned to a particular task, minimal fine-tuning is required on our part to achieve our aim. We only had to add our own layers to the model and fine-tune it further based on our classification task.

This concept is illustrated in Figure 2 taken from a Tensorflow video as shown below.

The pre-trained model used were accessed from "Hugging Face" and its corresponding package transformers.

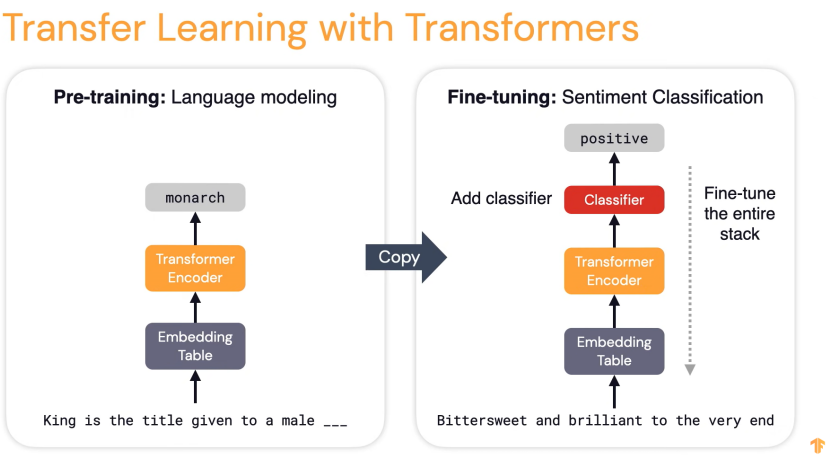

Figure 2: How we utilize pre-trained models to perform classification tasks (Turc, 2021)

2.4.3 Hardware

During our first attempt at implementing the model, we noted that the model is computationally intensive, and a single epoch of training took up to 3 hours or more. Thus, we moved to using GPUs, specifically Nvidia GTX2060i at the iDia Lab, as they are good for parallel processing and will shorten our training time.

2.4.4 Testing different Pre-trained Models

Utilizing the AI community, "Hugging Face", we explored various pre-trained models such as BERT, Electra and RoBERTa

We also did not need to perform any pre-processing on our data since Transformer Models have their own tokenizers.

Ultimately, we settled on a RoBERTa model as it had a specific pre-trained model built to perform sentiment analysis, as shown in the paper 'TweetEval' (Barbieri et al., 2020).

2.4.5 Hyperparameter Tuning

Next, we proceeded to tune the hyperparameters of our model, focusing on the number of epochs, the batch size and the learning rate[1]. For the number of epochs, we ended up settling on a single epoch. As for learning rate, we used a relatively low value of '1+e5' and lastly we chose a batch size of 8.

All these efforts were to avoid our model from overfitting on the training dataset. The values that we chose are relatively low since the pre-trained model is already very similar to our desired task and simply needs fine-tuning.

2.4.6 Layers

Lastly, for the layers, we started with a single 64-node layer with the ReLU activation function and a 3-node output layer with a soft-max activation function.

We eventually modified this to become 2 hidden layers of 128 nodes and another with 64, both with ReLU activation. Dropout layers[2] were also added in between with a rate of 0.1 and 0.4 respectively. The additional dropout layers were to aid in our model becoming more robust and better at performing on unseen input.

[1]For more information on the terminology please refer to Google's Machine Learning glossary

[2]Refer to TensorFlow Documentation for more information

3 Results

Overall, we submitted a few models to Kaggle, namely the multinomial logit, random forest and the transformer neural network.

From Table 2, we see that the final model, after performing the tuning and the addition of layers, was the best possible iteration that we had achieved on both the private and public leaderboard.

	Accuracy	
Model	**Public**	**Private**
Multinomial Logistic	0.678	0.665
Random Forest	0.719	0.712
Random Forest (More Processing)	0.722	0.714
RoBERTa (Base model)	0.799	0.786
RoBERTa (TweetEval + Dropout)	0.806	0.794

Table 2: Results of models submitted to Kaggle

Examining the results from the other models, we see that there did not seem to be a definite prediction on the difference between private and public accuracy.

4 Interpretability

Due to the complexity of the model implemented, it is difficult to understand what occurs while the model is training the dataset. It is also hard to understand how the model comes to its final decision.

Nonetheless, when using the concept of attention, more research can be done in the future to interpret how the weights of the model help in predicting sentiments. This is especially since people have attempted to interpret how attention works in performing translation tasks.

From this competition, we learned that there is an accuracy-interpretability trade-off when it comes to machine learning models. Less complex models are not capable of modelling complex relationships between sentences and sentiments but the relationship can be easily interpreted.

However, due to the goal of the Kaggle competition, which was model accuracy, interpretability is forgone and the accuracy and complexity of neural networks are instead desired.

5 Limitations

5.1 Hardware

Due to the limited hardware available, training time for the model was quite significant, ranging from 7 to 20 minutes on a GPU and anywhere from 1 to 3 hours per epoch on a CPU. This meant that with better hardware, we could have achieved more in less time.

5.2 Time Limit

Given more time or better hardware, our group believes that our results could be better predicted if we were to perform k-folds cross-validation on our transformer models, and use the average accuracy to allow us to estimate what our private score would be. We would also want to perform further tuning on the model's hyperparameters.

5.3 Data, data, data...

Finally, with a task as complex as sentiment analysis, a model is almost as good as the amount of data it sees. Although we are already using a pre-trained model tuned to our specifications, the model could deal with more exposure to labelled tweets for better training. Due to the relatively small dataset, training the model to be very accurate proves to be a tall task.

Bibliography

Barbieri, F., Camacho-Collados, J., Neves, L., & Espinosa-Anke, L. (2020). Tweeteval: Unified benchmark and comparative evaluation for tweet classification. *arXiv preprint arXiv:2010.12421*.

Turc, I. (2021). *Transfer learning and transformer models (ml tech talks)*. Retrieved August 13, 2022, from https://www.youtube.com/watch?v=LE3NfEULV6k

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.