Covid Tweets Classification and Sentiment Analysis

Rishabh Chhaparia *Rahul Pandey*

1. Abstract:

In order to better understand the conversation surrounding the virus, tweets related to Covid-19 can be classified and their sentiment can be analyzed. Classification can help to identify the types of information being shared about the virus, while sentiment analysis can provide insights into the emotions being expressed in relation to the pandemic. These techniques can be valuable tools for researchers, policymakers, and others working to combat the virus. We have used machine learning models such as Naive Bayes, RNN, and Transformer based models like BERT AND RoBERTa to predict Sentiments from the tweets.

2. Introduction:

The Covid-19 pandemic has had a profound impact on the world, leading to widespread concern and uncertainty. In response, people have turned to social media, particularly Twitter, to share information and express their thoughts and feelings about the virus. We have more than 41000 tweets that are pulled from Twitter and the name and user handle of the tweets have been removed to preserve the privacy of the individuals along with manual taggings of the sentiments. Data contains features such as tweets along with location and date. We will follow the following approach in this project to classify tweets and extract sentiments from the dataset.

1. Preprocessing
2. Exploratory Data Analysis
3. Vectorization
4. Training Models: Multinomial Naïve Bayes, BERT, RoBERTa, RNN
5. Evaluation of models.

3. Methodology

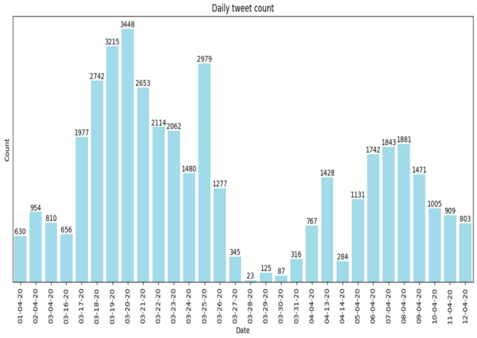
3.1 Preprocessing

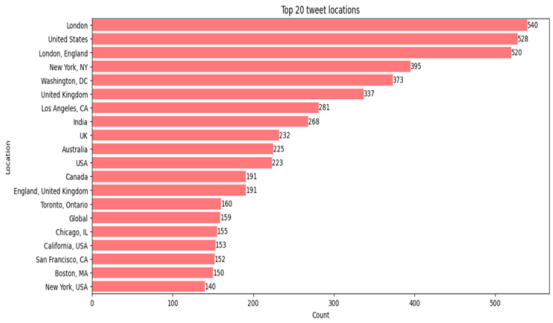
We preprocessed the data to basically clean the tweets in our data as they are unstructured. It is one of the fundamental steps in any NLP project. We removed punctuation, special characters, and hashtags, as these can interfere with natural language processing algorithms. Converted all text to lowercase to have uniformity, removed stop words, which are common words filtered out before natural language processing, and performed Stemming or lemmatizing words, which reduced words to their base form. Tokenizing the text, which involved breaking the text down into individual words or phrases.

The second part of preprocessing was to perform some data manipulation. We changed the “Na” values to the “Unknown location” label in order to get a better understanding of the geographical source of tweets, we changed the sentiment feature which was a categorical variable into a numerical label. From initial exploration, we found that keeping the sentiments in 5 classes led to low accuracy

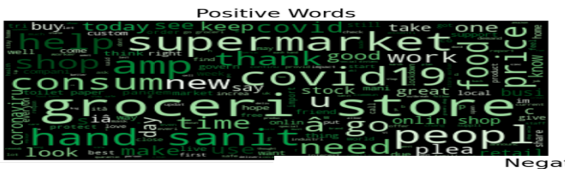
in the predictions. So, we also converted the sentiments to 3 classes - positive, neutral, and negative which were also converted to numerical variables. One-hot encoding of the variables was also done before training the BERT and RoBERTa models as this performed better than numerical encoding. Finally, we changed the date into a recommended format (by pandas) in datetime64 “yyyy-mm-dd” format to make it suitable for grouping and aggregating the data and creating visualizations.

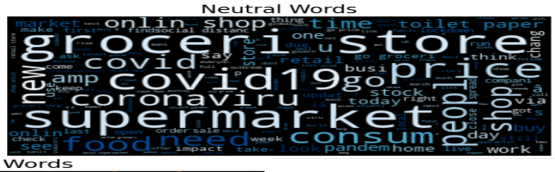
3.2 Exploratory Data Analysis

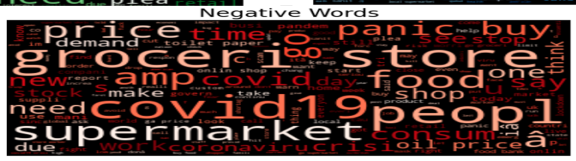




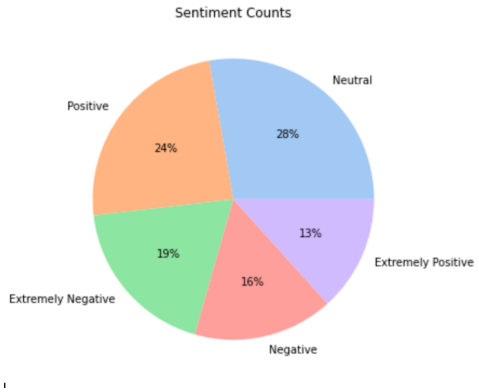
Tweet counts were maximum in March when the first wave of Covid started all over the world. So as a result of that we can see that there is a sudden spike of tweets being tweeted. We can also observe that most of the tweets were published from countries like the USA and UK.

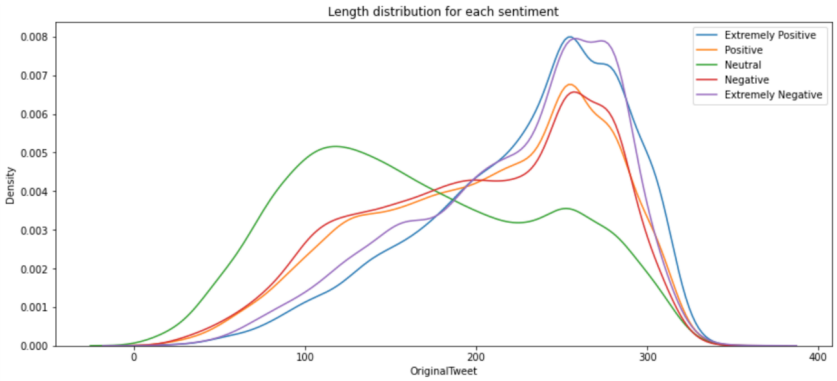






Word clouds are commonly used to quickly and easily understand the most important or frequently used words in a given piece of text. Most of the words in the word cloud are common but there are some unique ones as well like for positive sentiments we have sanitizer and for negative we have panic.





Negative tweets are more than 50% whereas positive tweets are less than 30% which means that people are generally talking negatively about covid in the dataset. An interesting fact about the length of characters of a tweet is that tweets that have neutral sentiments have on average shorter lengths as compared to positive and negative sentiments.

3.3 Model Training

We will be using 4 different algorithms to predict sentiments in our data set. After successfully training our data we will be evaluating the models on the basis of Accuracy score and will gauge the model using confusion matrix and precision scores.
From the EDAs, it was clear that there is a class imbalance among the 5 sentiments. So, we applied random oversampling to balance the sentiment classes for both 5 sentiments and 3 sentiments.

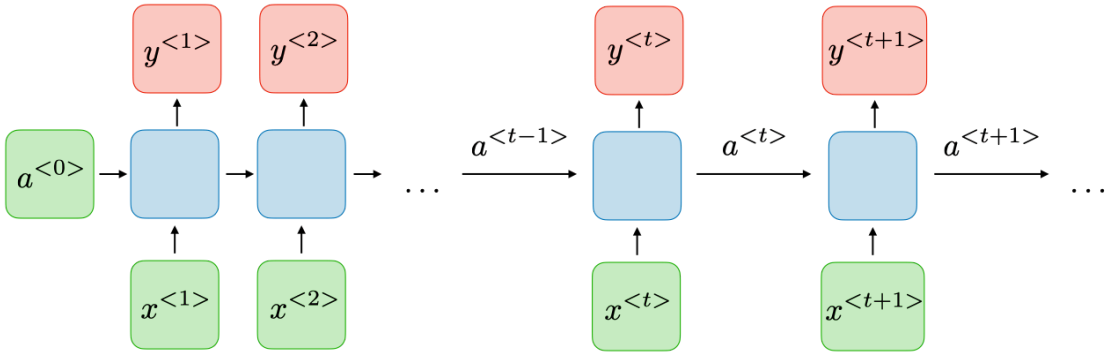
Naive Bayes

It involves using the words in each tweet as features, and the sentiment label (positive, negative, or neutral) as the target. The algorithm used these labeled examples to learn the relationship between the words and the sentiment and to calculate the probability that a given tweet belongs to each sentiment class.
Vectorization:
The first step for training the data on the Naive Bayes model was to vectorize the 'CleanedTweet' data and then apply TF-IDF to the vectorized output. This output was then used to train the model.

We first tested the model with 5 sentiment classes as the output. The performance of our model when having 5 classes was not satisfactory. As a result of that, we reduced the classes from 5 to 3 namely positive, neutral and negative.
We have used Naïve Bayes as our base model and evaluated further models keeping its metrics as the benchmark. Once trained, the algorithm can be used to categorize new tweets as positive, negative, or neutral. The algorithm determines which sentiment class a certain tweet most likely belongs to, then it places the tweet in that class.

As a result, the algorithm can automatically categorize a vast number of tweets, giving significant information about their attitude. The accuracy score for 3 classes came out to be 74%, and the Naive Bayes model was used as a baseline model score for the other models.

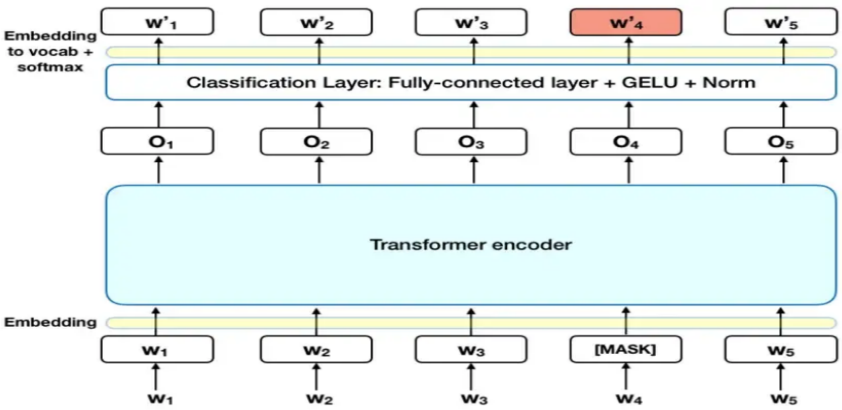
RNN



The Accuracy score of the test dataset is 82% which is a good indicator.
True Positive Rates of Negative and positive sentiments were comparatively higher than neutral sentiments which are reflected in precision and recall scores.
If we increase the number of epochs to more than 2, it caused overfitting in the training set. One of the key advantages of RNNs is that they can take into account the order of the words in the tweet. This allows the RNN to learn the context in which words are used, which can be important for understanding the sentiment of the tweet. For example, the word "not" can negate the sentiment of the words that come after it, so the order of the words can affect the sentiment of the tweet.

BERT

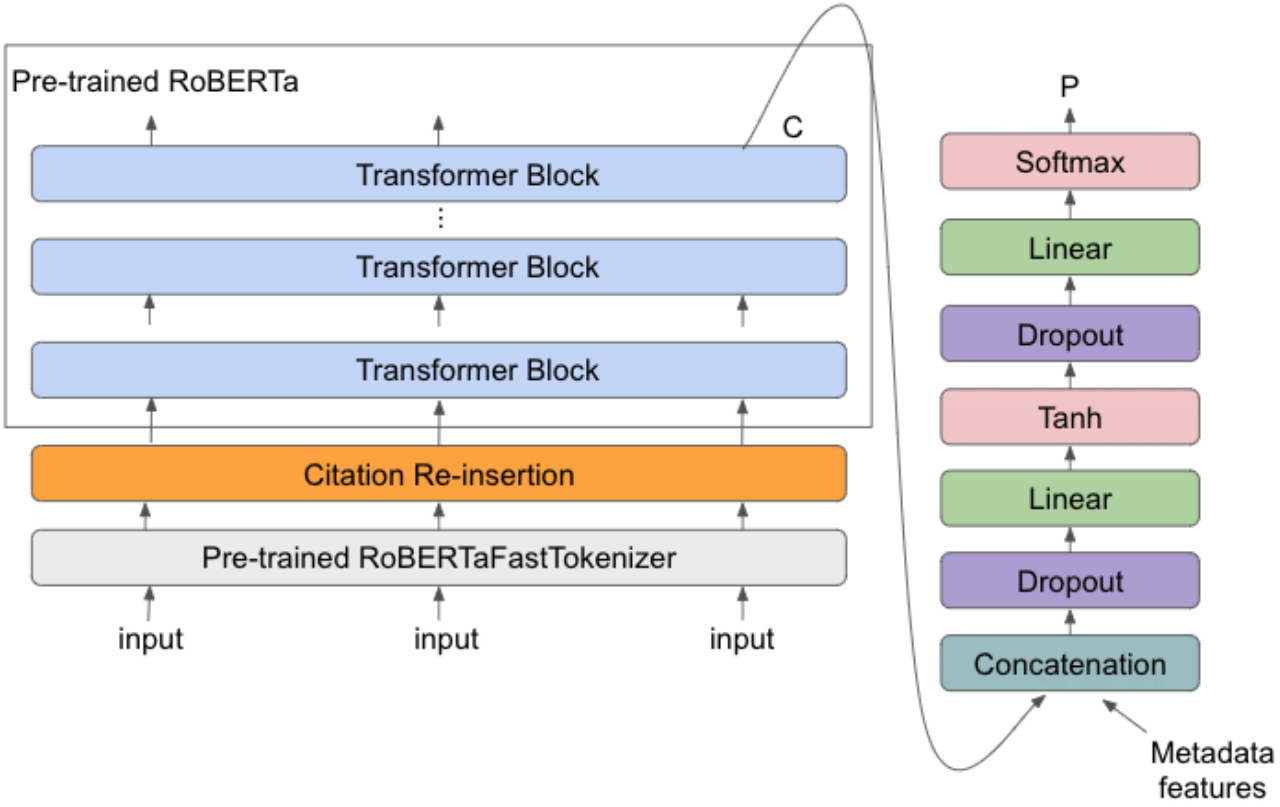
BERT is a variant of the Transformer model. BERT or Bidirectional Encoder Representations is a type of artificial neural network that is designed to perform natural language processing tasks.



For our model when the epochs were set to 4, the BERT model gave us an accuracy of 95% in training data but did not perform well on the test data, which suggests that there wass overfitting, hence we decreased the epochs to 2.

We got an accuracy of 86% in training data with 2 epochs. The true positive rate of positive and negative sentiments is much better than the tweets of neutral sentiment. This could be due to the fact that there is less number of tweets that are present in the neutral sentiment category.

RoBERTa



RoBERTA is a variant of the BERT model that uses dynamic masking, which means that it randomly masks some of the words in the input text during training helping it to predict the masked words based on their context, which can improve its ability to understand the meaning and context of words. On top of that, it uses distributed training", which allows the model to be trained on multiple machines in parallel.

We got an accuracy of 87% in training data with 2 epochs. The true positive rate of positive and negative sentiments is much better than the tweets of neutral sentiment. This could be due to the fact that there is less number of tweets that are present in the neutral sentiment category.

4. Results

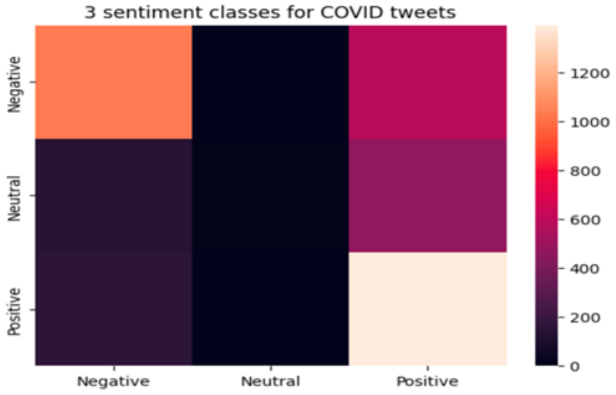
The main metrics used for the comparison of the models were the accuracy score and confusion matrix.

Model	**Accuracy**
Naive Bayes	64%
RNN	82%
BERT	87%
RoBERTa	86%

Naive Bayes

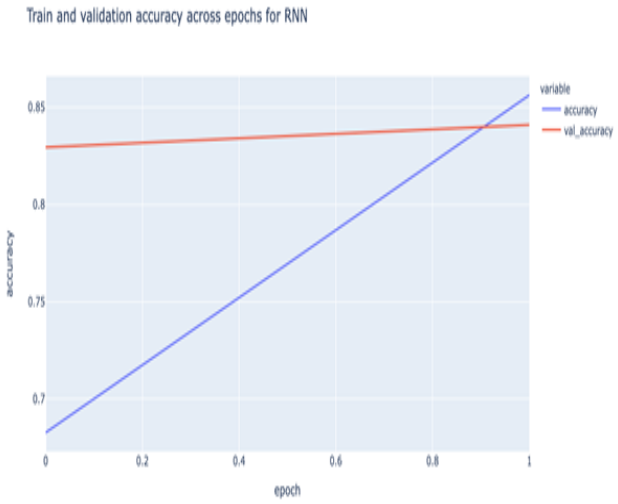
Classification Report for Naive Bayes (3 classes):

	precision	recall	f1-score	support
Negative	0.78	0.64	0.70	1633
Neutral	0.86	0.01	0.02	619
Positive	0.57	0.90	0.70	1546
accuracy			0.64	3798
macro avg	0.73	0.52	0.47	3798
weighted avg	0.71	0.64	0.59	3798

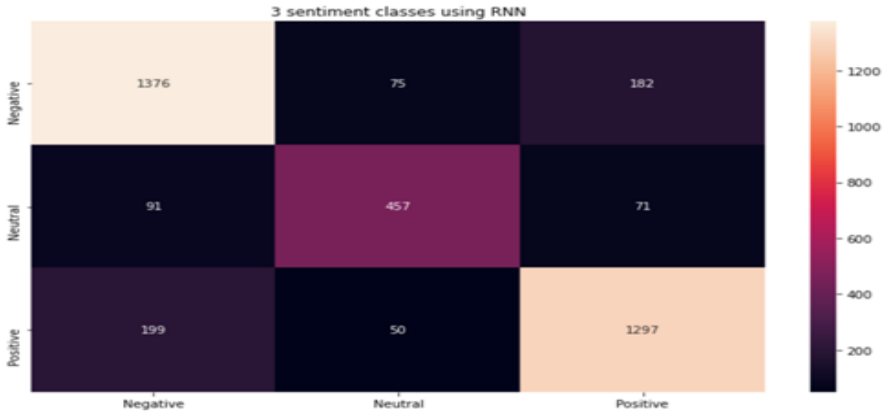


RNN

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 49, 16)            815952
_________________________________________________________________
bidirectional (Bidirectional (None, 49, 512)           559104
_________________________________________________________________
global_max_pooling1d (Global (None, 512)               0
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 32)                16416
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 99
=================================================================
Total params: 1,391,571
Trainable params: 1,391,571
Non-trainable params: 0
_________________________________________________________________
```



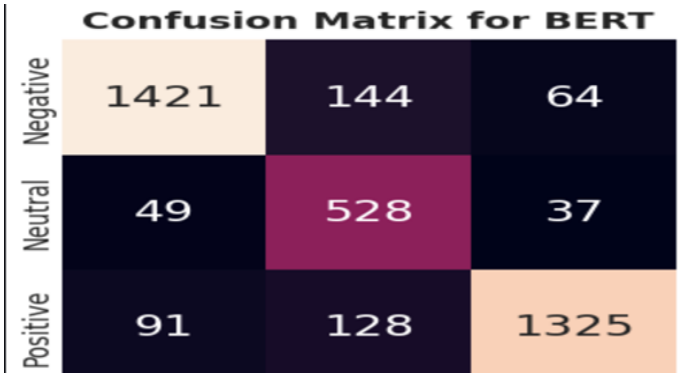

```
Test loss: 0.4782789349555969
Test Accuracy: 0.8241179585456848
```



BERT

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 128)]        0
__________________________________________________________________________________________________
input_2 (InputLayer)            [(None, 128)]        0
__________________________________________________________________________________________________
tf_bert_model (TFBertModel)     TFBaseModelOutputWit 109482240   input_1[0][0]
                                                                 input_2[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 3)            2307        tf_bert_model[0][1]
==================================================================================================
Total params: 109,484,547
Trainable params: 109,484,547
Non-trainable params: 0
__________________________________________________________________________________________________
```

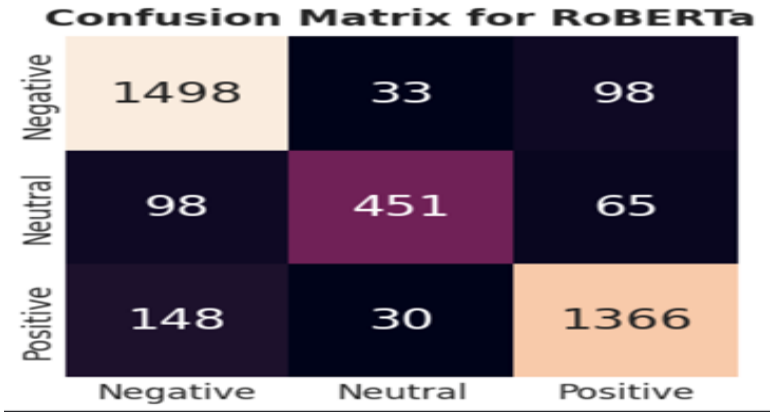
```
Epoch 1/2
1519/1519 [==============================] - 789s 492ms/step - loss: 0.3315 - categorical_accuracy: 0.8833 - val_los
s: 0.2960 - val_categorical_accuracy: 0.8948
Epoch 2/2
1519/1519 [==============================] - 745s 490ms/step - loss: 0.2175 - categorical_accuracy: 0.9231 - val_los
s: 0.2483 - val_categorical_accuracy: 0.9183
```



RoBERTa

```
Model: "model_1"
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
=================================================================================================
input_3 (InputLayer)         [(None, 128)]             0
_________________________________________________________________________________________________
input_4 (InputLayer)         [(None, 128)]             0
_________________________________________________________________________________________________
tf_roberta_model (TFRobertaMode TFBaseModelOutputWit 124645632   input_3[0][0]
                                                                 input_4[0][0]
_________________________________________________________________________________________________
dense_3 (Dense)              (None, 3)                 2307        tf_roberta_model[0][1]
=================================================================================================
Total params: 124,647,939
Trainable params: 124,647,939
Non-trainable params: 0
_________________________________________________________________________________________________
```

```
Epoch 1/2
1620/1620 [==============================] - 802s 487ms/step - loss: 0.5644 - categorical_accuracy: 0.7744 - val_los
s: 0.4044 - val_categorical_accuracy: 0.8613
Epoch 2/2
1620/1620 [==============================] - 772s 477ms/step - loss: 0.3340 - categorical_accuracy: 0.8798 - val_los
s: 0.3068 - val_categorical_accuracy: 0.8876
```



5. Discussion

The best result was obtained from BERT among the 4 models that we had selected for Twitter Sentiment Analysis, closely followed by RoBERTa.This could be attributed to the superior architecture of Transformers which is specially built for Natural Language Processing. It employs "bidirectional" and "attention mechanisms" to process the natural language which gives it an edge over other traditional machine learning models.

The RoBERTa variant's accuracy was slightly less than BERT, which could be due to the training data size which was not very large.

RNN also performed well and had an accuracy of 82%. RNNs take into account the order of the words in the tweet. This allows the RNN to learn the context in which words are used, which can be important for understanding the sentiment of the tweet.

Naive Bayes was used as a benchmark which had an accuracy of 64%.

Another interesting observation was that the True positives Rate of Negative and Positive sentiments was quite high as compared to neutral sentiments for all 4 models which is quite evident from the Confusion Matrix shown in the Result section.

This implies that our model performs better for Positive and Negative sentiments as compared to neutral sentiments. This could be because the tweets that had neutral tagging had most of the words that were present in positive and negative tweets. Secondly, it could also be because the neutral tweets were shorter in length as compared to the other two sentiments present.

6. Conclusion

We implemented sentiment analysis of tweets related to covid-19 extracted from Twitter and were able to identify sentiments of tweets with an accuracy of 87% which is a good score for recognizing the underlying sentiment in any tweet. In the future, we can use other techniques to vectorize tweets instead of tf-idf, for instance, Bag-of-words. We can also try different machine learning algorithms to train for sentiment prediction and increase the size of the dataset by getting tweets from the years 2021 and 2022.

7. Acknowledgement

This project was completed for CS6120 under the instruction of Professor Uzair Ahmad.

Link to Code and Data:

• Data : Coronavirus tweets NLP - Text Classification | Kaggle

• Code: NLP-Sentiment-Analysis

References

[1] https://arxiv.org/pdf/1810.04805.pdf

[2] https://www.frontiersin.org/articles/10.3389/fpubh.2021.812735/full

[3] towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270