Sentiment Analysis of Tweets

Shiva Hari Gundeti
u1460836@utah.edu

Abstract

Along with the Coronavirus pandemic, another crisis has manifested itself in the form of mass 2 fear and panic phenomena, fueled by incomplete and often inaccurate information. The paper investigates the impact of the COVID-19 pandemic on social media discourse, particularly on Twitter, with a focus on sentiment analysis. This study delves into sentiment analysis methodologies, specifically concentrating on the classification of users' sentiments derived from Twitter posts related to COVID-19. The temporal scope of the analysis spans from March to mid-April 2020, capturing a period when the pandemic significantly affected the entire world.

The primary focus lies in sentiment analysis, and the study employs distinct deep learning models like RNN, LSTM, and Distil-BERT. A comparative analysis is conducted, contrasting the performance of these deep learning models with traditional machine learning classifiers like Naive Bayes as a baseline. The objective is to train the models to effectively categorize tweets into three sentiment classes: negative, neutral, and positive. The investigation contributes insights into the sentiments expressed on Twitter during a critical period of the COVID-19 pandemic using advanced computational approaches in natural language processing. [1]

1. Introduction

The expansion of the internet is advancing at a breakneck pace, exerting a profound influence on every facet of our daily existence. This exponential growth persists as an outcome of the ever-increasing deluge of data and information. A significant portion of this data originates from human engagement within social networks, such as Twitter, Facebook, and Linkedin, which render remote communication possible. Among these social media platforms, Twitter reigns as one of the most widely embraced applications, offering a multifaceted reservoir of information and empowering its users to disseminate concise textual updates known as 'tweets.'

[1] Code for the project

The SARS-CoV-2 virus (COVID-19) pandemic started in December 2019. The virus was first detected in the Wuhan region of China and is affecting 221 countries and territories around the globe [2]. It is a new strain of coronavirus that until then had not been identified in humans[3] . This virus mainly affects the respiratory system, although other organ systems are involved.

Researchers and practitioners mine massive textual and unstructured datasets to generate insights about mass behavior, thoughts and emotions on a wide variety of issues such as product reviews, political opinions and trends, motivational principles and stock market sentiment. Textual data visualization is also used to identify the critical trend of change in fear-sentiment, using the "Fear Curve" in Fig. 1, with the dotted Lowess line demonstrating the trend, and the bars indicating the day to day increase in fear Tweets count. The source data for all Tweets data analysis, tables. Tweets were first classified using sentiment analysis, and then the progression of the fear-sentiment was studied, as it was the most dominant emotion across the entire Tweets data. This exploratory analysis revealed the significant daily increase in fear-sentiment towards the end of March 2020, as shown in Fig[1]

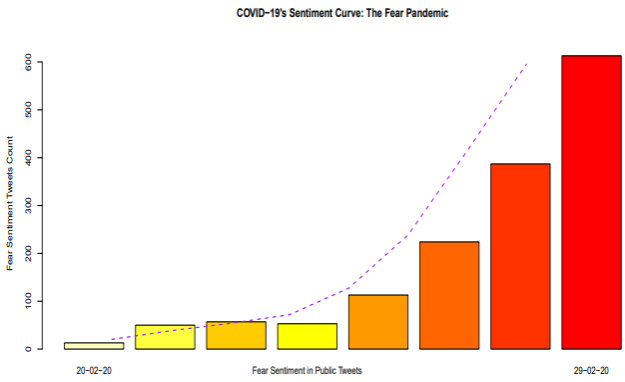

Figure 1. Fear curve.

This study presents several deep-learning models that

[2] https://www.worldometers.info/coronavirus/
[3] https://www.who.int/health-topics/coronavirus

aim at categorizing the sentiment found in the posts of Twitter users. The sentiment that classifiers are utilized to distinguish is either negative, neutral, or positive, and the topic of these tweets concerns the COVID-19 pandemic. More to the point, the dataset used in the paper concerns the period between March and April 2020, when the disease had already spread around the world, while the cases were constantly increasing and new measures to limit further spread of the disease were announced. The number of tweets was 41,157 and this was mainly because the sentiment that prevails in each recording has been manually categorized. For the sentiment identification that prevails in the tweets, different deep learning models were implemented consisting of long short-term memory (LSTM/BiLSTM) recurrent neural networks and 6 models based on traditional machine learning algorithms. We highlight that our paper introduces a novel framework that makes use of information from social media for understanding public behavior during the most popular topic of our days, the COVID-19 pandemic. Our proposed framework compares the different types of sentiments expressed about the rise of the number of cases, which impacted the economy and had different levels of lockdowns. We use LSTM and bidirectional LSTM (BiLSTM) [2] model with a bag of words (BoW) and term frequency-inverse document frequency (Tf-Idf) for word representation for building a language model. Moreover, the neural network model using the DistillBERT [3] as an embedding layer followed by max pooling, dense, dropout layer followed by the affine layer as a final layer. I'll try to compare the DistillBERT model with the Naive Bayes in terms of accuracy and the reasons for it. This framework is focused on multi-label sentiment classification, consisting of three different classes, namely negative, neutral and positive.

2. Related Work/Lit Survey/Background

The exploration of COVID-19 tweets through social network analysis, utilizing sophisticated machine learning techniques, is regarded as an eminent domain within the realm of data mining. This prominence is attributed to the copious and continually expanding body of available literature on the subject.

The work [4] considers classifier ensembles formed by diversified components that are promising for tweet sentiment analysis. The authors compared bag of words and feature hashing-based strategies for the representation of tweets and depicted their advantages and drawbacks, where classifier ensembles were obtained from the combination of lexicons, a bag of words, emotions and feature hashing.

LSTM neural networks have been widely used for forecasting COVID-19 infection in multiple countries, especially in the period of lockdown[2, 5]. Additionally, a sentiment analysis research paper based on posts from Sina Weibo, a popular Chinese social media platform, was presented in [6]. The posts were classified into 3 categories (negative, neutral and positive) with the use of a fine-tuned unsupervised BERT model and a Tf-Idf model for the topic post identification. Sentiment classification was, also, implemented by the authors of [1]. Specifically, negative and positive sentiment classification was implemented with the use of naive Bayes and logistic regression machine learning techniques.

3. Approach

The approach is mainly present various types of classifiers in Machine Learning and compares them with sequence models like LSTM, RNN, and DistillBERT. A detailed description of the various categorizers is presented, and the the way the input data is utilized is explained. The data mainly consists of the tweets and their labels. After training the models, they are able to categorize the sentiment into three distinct categories, which are negative, neutral and positive.

3.1. Data Preprocessing

This is the first step mainly to prepare the data so that we can train on different models on the same data. As a part of the preprocessing step I have removed the columns which are not important. For example, for this dataset, I have only maintained two columns [Sentiment, Original Tweet].

Earlier, in the milestone, I tried having five attributes (Extremely Positive, Positive, Extremely Negative, Negative, and Neutral) and it resulted in an accuracy of 34 percent on Naive Bayes. Results have been demonstrated in the Project Milestone paper. The authors of the the paper considered combining the target classes[4]. The aim is to classify the data into three categories led to merging the extremely negative and negative tweets in the same class, while the same happened for the data belonging to the extremely positive and positive classes, respectively.

I have observed the tweet lengths and normal distribution is skewed towards the right. Also, another observation is that it has outliers and the maximum value is of 127 as shown in Fig 2. I have performed some preprocessing steps so that the box plot can have a maximum value up to 80 and doesn't have any outliers. Some of the steps are listed below

For every tweet, all characters are converted to lowercase, and the hyperlinks are removed as they do not add any useful linguistic information [7]. Moreover, references to and hashtags commonly employed in Twitter messages to garner the attention of other users have been expunged. Additionally, regular expressions were deployed to substi-

[4] https://rdcu.be/dsLjy

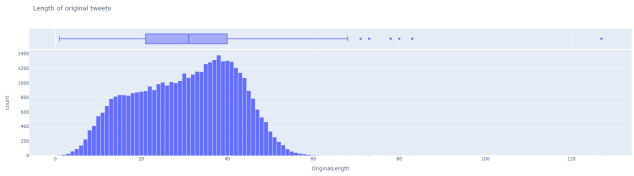

Figure 1. Tweet Length Box plot

tute certain significant words, such as the username, as discerned in numerous scholarly articles.

Tokenization and lemmatization are integral components of the analysis. The initial phase, tokenization, involves the meticulous disintegration of the text into smaller units referred to as "tokens." Subsequently, each term within every tweet is meticulously cataloged in a token list, and the arrangement of the text's tokens aligns with their inherent sequential order.

Concerning lemmatization, an intricate process unfolds utilizing previously generated Part-of-Speech (POS) tags and the application of the WordNet Lemmatizer. Every word extracted from the tweets undergoes a transformative journey, assuming its fundamental or dictionary form. This intricate operation entails the reduction of inflected words to their root forms present in the lexical repository, fostering a refined analysis of linguistic structures.

3.2. Multinomial Naive Bayes Classifier

In multinomial NB classifier[8], every word w_i gets a say in determining which label $c \in 1, ...C$ should be assigned to an unseen document $w = (w_1, ..., w_N)$. In order to choose a label c for w, multinomial NB classifier begins by calculating the prior probability Pr(c) of each label c, which is determined by assuming equiprobable classes, or checking the frequency of each label in the training set. The contribution from each word is then combined with prior probability to arrive at a likelihood estimate for each label. This is known as the maximum a posteriori (MAP) decision rule. It can be formally defined as

$$c = argmax_c Pr(c) \prod_{n=1}^{N} Pr(w_n|c) = argmax_c v_c \prod_{n=1}^{N} \phi_{c,w_A}$$

Given a training document set $D = {w_m, c_m}_{m=1}^{M}$, $v_c and \phi_{c,v}$ are usually estimated by a smoothed version of the maximum likelihood (ML) as follows. In fact, these are both MAP estimates given the uniform Dirichlet priors

$$v_c = \frac{n^{(c)} + \alpha}{M + C\alpha}$$

$$\phi_{c.v} = \frac{n_c^{(v)} + \beta}{n_c + V\beta}$$

where V is the number of unique words, $n^{(c)}$ is the number of documents with the class c in the document set D, $n_c^{(v)}$ is the number of times word v appears in the document of class c in D. and $n_c = \sum_{v=1}^{V} n_c^{(v)}$

3.3. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a type of artificial neural network designed to process sequences of data. They work especially well for jobs requiring sequences, such as time series data, voice, natural language, and other activities.

RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer. "x" is the input layer, "h" is the hidden layer, and "y" is the output layer. A, B, and C are the network parameters used to improve the output of the model. At any given time t, the current input is a combination of input at x(t) and x(t-1). The output at any given time is fetched back to the network to improve on the output.

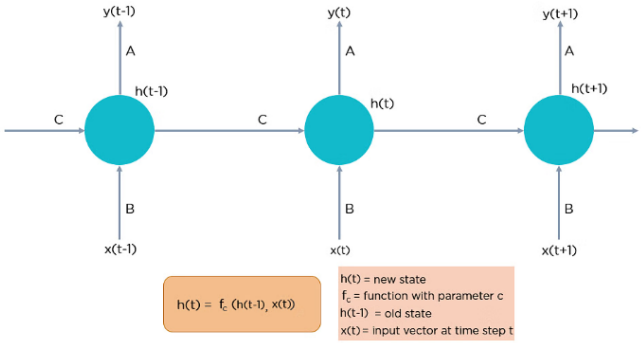

Figure 2. Fully connected Recurrent Neural Network

3.4. LSTM architecture

This category uses a special gateway mechanism that decides which pieces of information to remember, which to update and which to pay attention to. These abilities are based on the cell and do not take into consideration the update as well as the output gate that LSTMs are composed of. The architecture described above is illustrated in Fig 3 In Eq. 1, the update gate consists of a sigmoid function that decides which of the new information should be updated or ignored by merging the $x^{(t)}$ and $\alpha^{(t-1)}$ in the memory cell $c^{(t)}$

$$\chi_u^{(t)} = \sigma(W_u[\alpha^{(t-1)}, x^{(t)}] + b_u) \quad (1)$$

The forget gate presented in Eq. 2 consists of a sigmoid function, which takes the current input of the cell $x^{(t)}$ and the output of the previous one $\alpha^{(t-1)}$, deciding which parts of the old output should be removed to free a substantial part of memory.

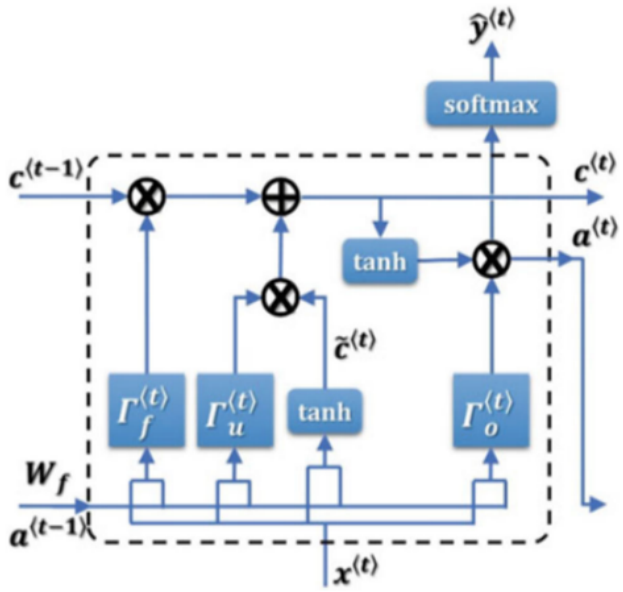

Figure 3. LSTM architecture [9]

$$\chi_f^{(t)} = \sigma(W_f[\alpha^{(t-1)}, x^{(t)}] + b_f) \quad (2)$$

Equation 3 depicts the output gate which consists of a sigmoid function that decides which memory cell information will be extracted.

$$\chi_o^{(t)} = \sigma(W_o[\alpha^{(t-1)}, x^{(t)}] + b_o) \quad (3)$$

Equation 4 presents $\bar{c}$, which is a layer consisting of the hyperfunction tanh that takes the same inputs as before and creates a vector of all possible values from the new input.

$$\bar{c}^{(t)} = tanh(W_c[\alpha^{(t-1)}, x^{(t)}] + b_c) \quad (4)$$

The new cell state is presented in Eq. 5, where the outputs of the Eqs. 2 and 4 are multiplied to update the new memory cell. This is added to the old memory c^{t-1} multiplied by the forget gate, so that c^t occurs.

$$c^t = c^{(t-1)} * \chi_f^t + (\bar{c})^t * \chi_u^t \quad (5)$$

The memory cell goes through a layer composed of a tanh function creating a vector of all possible values and multiplied with the output gate the hidden state is obtained; this information is forwarded to the next unit of the LSTM.

$$\alpha^{(t)} = \chi_a^{(t)} * tanhc^{(t)} \quad (6)$$

3.5. Distil BERT model

DistilBERT is a transformers model, smaller and faster than BERT, which was pretrained on the same corpus in a self-supervised fashion, using the BERT base model as a teacher. This means it was pretrained on the raw texts only, with no humans labelling them in any way (which is why it can use lots of publicly available data) with an automatic process to generate inputs and labels from those texts using the BERT base model.

4. Experiments

4.1. Dataset

For the scope of this project, the dataset used must contain tweets related to the Covid pandemic with a category of sentiment being added.

The dataset used is Coronavirus Tweets NLP from kaggle datasets[5]. The dataset consisted of tweets classified into extremely negative, negative, neutral, positive and extremely positive sentiment. The data was collected in the period from 2/3/2020 to 14/4/2020 and the tweets included are exclusively in the English language. The period taken into consideration is when the coronavirus had already spread throughout the world and the pandemic created unprecedented situations with long-term quarantines to reduce the spread of the virus, traveling restrictions, etc

The training dataset consists of 41157 rows and the test set consists of 3798 rows with 4 columns (Location, Tweet At, Original Tweet, Label). I might use up to one GPU mainly to accelerate the process of training the models as I am working with a large neural net model distilBERT.

To analyse the dataset, I have plotted the training and testing datasets to check whether all the target classes are balanced or not. The plot has been shown below.

Figure 4. Train/Test target value count

[5] https://www.kaggle.com/datasets/datatattle/covid-19-nlp-text-classification

4.2. Experiment and Evaluations

After preprocessing the data, training on the Multinomial Naive Bayes Classifier is performed. We can observe that the training and validation accuracy for Multinomial NB classifier is **70.82%** and **66.57%**. The test accuracy is **39.59%**. I have performed hyperparameter tuning and by cross-validation observed that the best test accuracy I could reach is **41%**. As it is multinomial NB classifier, alpha is the only parameter that we can perform hyperparameter tuning on and it is giving best value at **alpha = 0.1**. I have performed training on Gradient Boost Classifier to compare it with the baseline models and observed that it has the test accuracy of **41.99%** which is slightly better than Multinomial NB classifier.

Sentiment classification using Random Forest Classifier has been performed and it gave the test accuracy of **40.15%**. Using the RandomizedSearchCV library, the hyperparameter tuning of Random Forest Classifier model has been done and gave the best it gave best accuracy when the parameters are $criterion = entropy, max\_depth = 10, min\_samples\_leaf = 5, min\_samples\_split = 10$. The training accuracy of the model is **56.72%** and the best test accuracy is **41.36%**. We can observe that the model didn't overfit the data as we performed cross-validation and chose the best parameters. The accuracy of Random Forest Classifier is slightly better than other models.

In general, we observed that the test accuracy of these models is way low around 42%. We need a better algorithm to get a higher test accuracy. Sequence models plays a key role in identifying the structure of the sentence in a tweet. As we need to look at each word and keep track of it, sequence models like RNN, LSTM and BERT perform better on this type of data.

While performing the training of the RNN model we tokenize the training data using the Tokeniser library. After tokenising the data we pad the sequences so that all the training examples have the same size of the array. We can initially test it using the baseline model. After performing the training, I have observed that the model has overfit the data based on the train/validation splits as shown in figure 5.

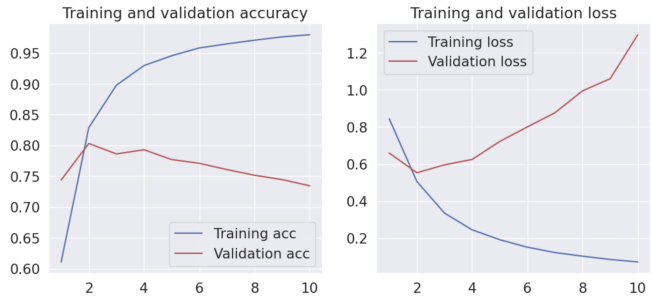

Figure 5. RNN Baseline Train/Val Curves

To reduce the overfitting of the model, I have added dropout layers after an embedding layer and in between the dense layers. The resulting model now has the total parameters of 6401059 and the model architecture has been shown in the Figure 5. RNN model has attained the accuracy of **77.84%** which is significantly better than the baseline models. It is mainly because the sequence nature of the text is preserved in the RNN models.

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 50, 128)	6400000
dropout (Dropout)	(None, 50, 128)	0
global_average_pooling1d_1 (GlobalAveragePooling1D)	(None, 128)	0
dense_2 (Dense)	(None, 8)	1032
dropout_1 (Dropout)	(None, 8)	0
dense_3 (Dense)	(None, 3)	27

Total params: 6401059 (24.42 MB)
Trainable params: 6401059 (24.42 MB)
Non-trainable params: 0 (0.00 Byte)

Figure 6. RNN Model with Dropout

I have trained the model using LSTM architecture, where initially I have added the embedding layer followed by the dropout and then by the LSTM layer followed by the dense layers. The model architecture and the parameters has been listed below. The LSTM model has attained the test accuracy of **78.69%** which is slightly better than the RNN model. The accuracy is higher mainly due to the fact that for simple RNN layer, input from previous timestamps gradually decrease as we go further through the input. With LSTM we have a long-term memory data structure storing

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, 50, 128)	6400000
dropout_2 (Dropout)	(None, 50, 128)	0
lstm (LSTM)	(None, 32)	20608
dense_4 (Dense)	(None, 32)	1056
dropout_3 (Dropout)	(None, 32)	0
dense_5 (Dense)	(None, 3)	99

Total params: 6421763 (24.50 MB)
Trainable params: 6421763 (24.50 MB)
Non-trainable params: 0 (0.00 Byte)

Figure 7. LSTM model architecture

all the previously seen inputs as well as when we saw them. This allows for us to access any previous value we want at any point in time. This adds to the complexity of our network and allows it to discover more useful relationships between inputs and when they appear.

Another model which gave significant results in the test accuracy is Bi-Directional LSTM. It combines the power of LSTM with bidirectional processing, allowing the model to capture both past and future context of the input sequence. Using this model I have achieved an accuracy of **81.41%**. The accuracy of Bidirectional LSTM is better than LSTM models. The classification report of the Bidirectional models has been shown below.

	precision	recall	f1-score	support
Negative	0.87	0.76	0.81	1633
Neutral	0.78	0.72	0.75	619
Positive	0.78	0.90	0.84	1546
accuracy			0.81	3798
macro avg	0.81	0.80	0.80	3798
weighted avg	0.82	0.81	0.81	3798

Figure 8. Bidirectional LSTM classification report

The training of the DistilBERT model has been performed and initially, I have tokenized the inputs using the DistilBertTokenizer class and have observed the maximum amount of token that the tweet contains is 103. I have set the max length and input for the deep learning model to 105 as I considered two special character tokens [SEP] and [CLS]. After performing the tokenization I trained the model containing masked, maxpool, dense, and dropout layers. The architecture of the model is shown in Fig 9.

The model has **66126083** parameters and I have a sparse categorical cross-entropy loss on the model using the Adam optimizer. I have trained the model on 10 epochs, batch size of 16, and performed the cross-validation split of 0.2. The train and validation splits of the DistilBERT has shown in Fig 10. The model has attained an accuracy of **76.5%**. I need to investigate further the low accuracy of the model compared to Bi-Directional LSTM. I have plotted the Confusion Matrix of the predictions and the classification report has shown in Figure 11 and 12 respectively.

5. Conclusions

This paper introduces a series of implemented models designed to classify the sentiment expressed in tweets by users on the Twitter platform. The classifiers are tasked with predicting whether the sentiment of the tweets falls into the categories of negative, neutral, or positive. The tweets under consideration revolve around the topic of the COVID-19 pandemic, which emerged in December 2019.

Layer (type)	Output Shape	Param #	Connected to
input_token (InputLayer)	[(None, 105)]	0	[]
masked_token (InputLayer)	[(None, 105)]	0	[]
tf_distil_bert_model (TFDistilBertModel)	TFBaseModelOutput(last_hidden_state=(None, 105, 768), hidden_states=((None, 105, 768), (None, 105, 768), (None, 105, 768), (None, 105, 768), (None, 105, 768), (None, 105, 768)), attentions=((None, 12, None, 105), (None, 12, None, 105), (None, 12, None, 105), (None, 12, None, 105), (None, 12, None, 105)))	65190912	['input_token[0][0]', 'masked_token[0][0]']
bidirectional_2 (Bidirectional)	(None, 105, 256)	918528	['tf_distil_bert_model[0][13]']
global_max_pooling1d (GlobalMaxPooling1D)	(None, 256)	0	['bidirectional_2[0][0]']
dense_8 (Dense)	(None, 64)	16448	['global_max_pooling1d[0][0]']
dropout_24 (Dropout)	(None, 64)	0	['dense_8[0][0]']
dense_9 (Dense)	(None, 3)	195	['dropout_24[0][0]']

Total params: 66126083 (252.25 MB)
Trainable params: 935171 (3.57 MB)
Non-trainable params: 65190912 (248.68 MB)

Figure 9. DistilBERT model architecture

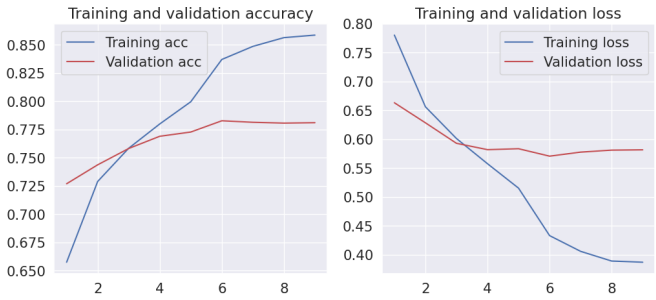

Figure 10. DistilBERT Train/Validation curves

The performance of Deep Learning models has been measured in comparison with the machine learning models. The sequence nature of the text plays a crucial role in understanding the sentiment of the user and we have observed that the sequence nature models like RNN, LSTM, and Bi-Directional LSTM perform better than the baseline models of the Naive Bayes classifier. The training of the DistilBERT model is performed and have observed the accuracy is slightly lower than the LSTM. Further investigation on this side has to be done. Overall, from this course project, I have gained practical exposure to training large models and since I am relatively new in this field, I have gained experience of training large models on a relatively simple task. Now I can use this experience for future prospects and handle complicated tasks in this field.

Looking ahead, it would be beneficial to try out different combinations of the models we discussed in this study to see if we can make them even more accurate. We should

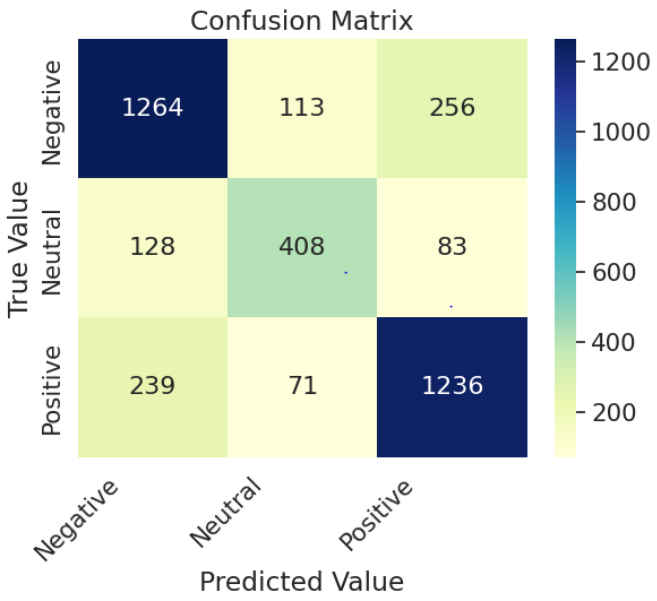

Figure 11. DistilBERT Confusion Matrix

	precision	recall	f1-score	support
Negative	0.77	0.77	0.77	1633
Neutral	0.69	0.66	0.67	619
Positive	0.78	0.80	0.79	1546
accuracy			0.77	3798
macro avg	0.75	0.74	0.75	3798
weighted avg	0.76	0.77	0.77	3798

Figure 12. DistilBERT Classification Report

also test these models on larger sets of data to confirm their accuracy in spotting sentiments.

Expanding the dataset not only in terms of volume but also by incorporating a richer set of numerical features beyond those utilized in this study is crucial. This augmentation can contribute to a more comprehensive understanding of sentiment patterns and potentially boost overall model performance.

Instead of relying on one model alone, we can use a mix of techniques to combine them, making our results more reliable. This approach, mentioned in studies [10, 11], involves taking the strengths of different models to offset their weaknesses.

Looking forward, it's worth exploring "explainable machine learning," which means making models that not only predict well but also explain why they made a certain prediction. This can help us understand where our models are strong and where they might struggle.

To enhance user interaction and understanding, a prospective avenue involves combining these explainable models with cutting-edge human–computer interface techniques. These interfaces are adept at translating complex model outputs into coherent and useful explanation dialogues for end-users. This comprehensive approach ensures not only superior predictive capabilities but also a user-friendly experience with insightful model explanations.

References

[1] Jim Samuel, G. G. Md. Nawaz Ali, Md. Mokhlesur Rahman, Ek Esawi, and Yana Samuel. Covid-19 public sentiment insights and machine learning for tweets classification. *Information*, 11(6), 2020. 1, 2

[2] Singh Chauhan D. Chandra R, Jain A. Deep learning via lstm models for covid-19 infection forecasting in india. *PLoS One. 2022;17(1):e0262708*, 2022. 2

[3] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. 2

[4] Nádia Félix, Eduardo Hruschka, and Estevam Hruschka. Tweet sentiment analysis with classifier ensembles. *Decision Support Systems*, 66, 07 2014. 2

[5] Abdelhafid Zeroual, Fouzi Harrou, Abdelkader Dairi, and Ying Sun. Deep learning methods for forecasting covid-19 time-series data: A comparative study. *Chaos, Solitons Fractals*, 140:110121, 07 2020. 2

[6] Tianyi Wang, Ke Lu, K.P. Chow, and Qing Zhu. Covid-19 sensing: Negative sentiment analysis on social media in china via bert model. *IEEE Access*, 8:1–1, 07 2020. 2

[7] Jashanjot Kaur and Preetpal Buttar. Stopwords removal and its algorithms based on different methods. *International Journal of Advanced Research in Computer Science*, 9:81–88, 10 2018. 2

[8] Shuo Xu, Yan Li, and Wang Zheng. Bayesian multinomial naïve bayes classifier to text classification. pages 347–352, 05 2017. 3

[9] Basant Agarwal and Namita Mittal. *Prominent Feature Extraction for Sentiment Analysis*. 01 2016. 4

[10] Georgios Drakopoulos, Andreas Kanavos, and Athanasios Tsakalidis. Evaluating twitter influence ranking with system theory. 04 2016. 7

[11] Ioanna Kyriazidou, Georgios Drakopoulos, Andreas Kanavos, Christos Makris, and Phivos Mylonas. Towards predicting mentions to verified twitter accounts: Building prediction models over mongodb with keras. 07 2019. 7