COVID-19 Tweet Scanner
CS6140 Project Report - Northeastern University

Vidhi Parekh [1], **Visheshank Mishra** [1] , **Sourabh Natesh** [1]
[1] Khoury College of Computer Science, Northeastern University
parekh.vi@northeastern.edu, mishra.vis@northeastern.edu, natesh.s@northeastern.edu

Abstract

The coronavirus pandemic has severely hampered the everyday life of people around the globe. Although it has its cons, the social media environment has lately become a platform for people to vent their emotions and concerns towards COVID-19. To counteract the risks of misleading information gathered by analytics, we present preliminary vectorization techniques to transform the textual data into vectors for further text analysis. The findings show that our baseline models provide promising results and set the direction and motivation for future improvements. We were able to (1) produce various visualizations to help explore and understand the data; (2) test and compare six machine learning models (Naïve Bayes, Logistic Regression, Random Forest, SVM, CatBoost, MLP) along with their performance results (3) implement a neural network model using Bidirectional LSTM and BERT transformer that successfully classify tweets based on user's sentiments and found the optimal one based on our analysis. Morever, we also implemented the prediction of user sentiment using spaCy which outputs the sentiment of a given tweet. Furthermore, the goal of this project is to provide logical strategies to comprehend data and provide high performance models that would accurately classify user's tweets based on sentiments.

1 Introduction

Coronavirus disease 2019 (COVID-19) has severely impacted the daily lives of individuals across the globe. People worldwide use online media to state their viewpoints and general feelings concerning this phenomenon that has assumed control over the world by storm. Social media platforms like Twitter have experienced exponential growth in tweets related to the pandemic in a short period. The social networking site Twitter is a commonly used online media platform. It provides real-time information related to ongoing events concisely and captures the emotions and thoughts of the people. However, data from social media can be misleading at times. Sufferings get multiplied if they find misleading and depressing information on social media. Online Social media platforms such as Twitter and Facebook contain a lot of noisy data, so identifying informative content from large and noisy data is a challenging task, but after cleaning it, this noisy data captures human feelings and emotions, expression, and thoughts. When analyzed carefully, it conveys a lot about the present mood, attitude, and nature of a large human community. In this project, we use Twitter data for sentiment analysis to identify public sentiments and investigate the increased fear associated with coronavirus. This study focused on the tweets analysis and identified global people sentiments from February 2020 to March 2020. The tweets from February 2020 to March 2020 are collected using the Twitter API. The tweets' sentiments are categorized into five classes (extremely positive, positive, neutral, negative, and extremely negative). This project focused on creating a new dataset by combining extremely positive to positive and extremely negative to negative to produce efficient categorization of users' sentiment and avoid misclassification of tweets. We use machine learning (ML) and deep learning (DL)-based classification methods primarily used in text classification; however, the discussion and the comparison of tweets' sentiment classification mechanisms is one of the significant elements of this project. The main contributions of this project are as follows:

- Design a Transformation-based BERT BASE model for sentiment analysis of tweets to identify sentiments concerning coronavirus from tweets.
- Extract sentiment-related concise information from tweets to automatically learn features without human intervention and predict the sentiment of the tweet using advanced spaCy NLP model.
- Present a broad comparison between existing ML and DL text classification methods and discuss the given baseline results. The proposed model outperformed on Bidirectional LSTM compared to all previously used methods.

The remainder of this article is organized as follows. Existing work related to COVID-19 sentiment analysis is presented in section 2. Section 3 provides a detailed explanation of the selected dataset. The modeling methodology used in the project is described in section 4. Experimental analysis and results are explained in section 5, and finally, the conclusion of this project is presented in section 6.

2 Related work

Since the outbreak of the coronavirus in 2020, people have taken to the world of Twitter to express their opinions and

the views about the various facets of the pandemic. Naturally, this has led to multiple reports trying to model the public sentiment about the pandemic. We have studied a few reports and have presented our observations in this section.

In the paper "Sentiment Analysis of Twitter Data: A Survey of Techniques"[1], the authors use machine learning techniques such as Naïve Bayes and Support Vector machines to classify twitter data. They use different twitter datasets such as Sarcasm Detection, EMOT and ISIEVE. They also discuss different methods of feature extraction such as POS Tagging, N-gram counts and using Negation for sentiments. They were able to achieve a 74% accuracy for Naïve Bayes and 76% accuracy for Support Vector Machine.

In the paper "Text based Sentiment Analysis using LSTM" [2] , the authors talk about using an LSTM based approach to perform sentiment analysis. Their model architecture consisted of an embedding layer, an LSTM layer of 128 units and a Dense layer as the output layer. They performed this sentiment analysis on the IMBD movie review dataset and the Amazon reviews dataset. They were able to get a 99% accuracy on the above two datasets.

In the paper "Sentiment Analysis Based on Deep Learning: A Comparative Study" [3], the authors compare various deep learning approaches including a Deep Neural Network, a Convolutional Neural Network and a Recurrent Neural Network. They have also used different vectorizing techniques such as TF-IDF and Word2Vec. They have performed sentiment analysis on different datasets such as 'Sentiment140', 'Airline Tweets', 'Book Reviews' and 'Movie Reviews'. It was found that CNN and RNN based networks performed best and Word2Vec outperformed TF-IDF based vectors.

The paper "COVID-19 Related Sentiment Analysis Using State-of-the-Art Machine Learning and Deep Learning Techniques" [4] uses many state of the art models such as XGBoost, a Multi-depth DistilBERT transformation model and a hybrid 6 layer CNN based model. It also uses current state of the vectorization techniques like Fasttext and Glove. This paper uses a dataset similar to ours, containing 90,000 unique tweets distributed into positive, negative and neutral classes. It was found that the Transformer based model gave best results with an accuracy of 96%.

3 Dataset Description / Analysis

3.1 Dataset Selection

We used dataset gathered from Twitter API for experimental analysis. The dataset is obtained through the Kaggle competition using the link provided (dataset link). The dataset features 44,955 tweets collected from March to April 2020. Figure 1 provides an overview of the sample dataset utilized in this study. Figure 2 describes the preliminary visualization of how the dataset is distributed based on sentiments.

3.2 Exploratory Data Analysis

Exploratory Data Analysis is one of the major steps to fine-tune the given data set in a different form of analysis to understand the insights of the key characteristics of various

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
4	3803	48755	NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative
5	3804	48756	Â□T: 36.319708,-82.363649	16-03-2020	As news of the regionÂ□s first confirmed COVID...	Positive
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative

Figure 1: Sample Dataset used in the project

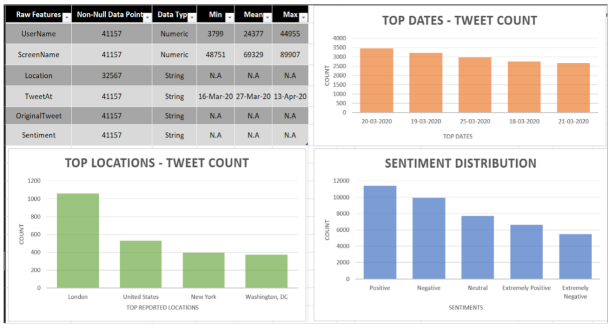

Figure 2: Sample Dataset used in the project

entities of the data set like column(s), row(s) by applying Pandas, NumPy, Statistical Methods, and Data visualization packages. EDA is important in any project as it performs initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of statistics and graphical representations. In this project, the outcome of EDA process (1) gives us the understanding the given dataset and helps clean up the given dataset (2) indicates a clear picture of the features and the relationships between them (3) provides guidelines for essential variables and removing non-essential variables (4) handling missing values and identify outliers and lastly, (5) would be maximize insights of our dataset. EDA performed in this project has been described in the figures below.

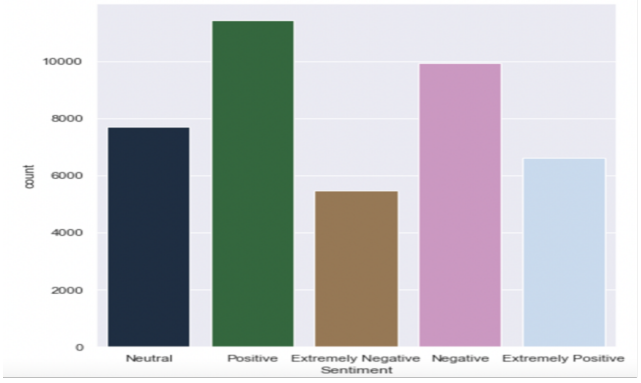

Figure 3: Bar plot containing count of different sentiments

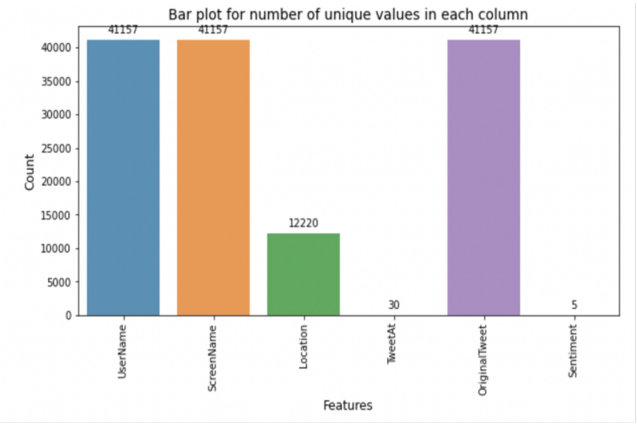

Figure 4: Bar plot for number of unique values in each column

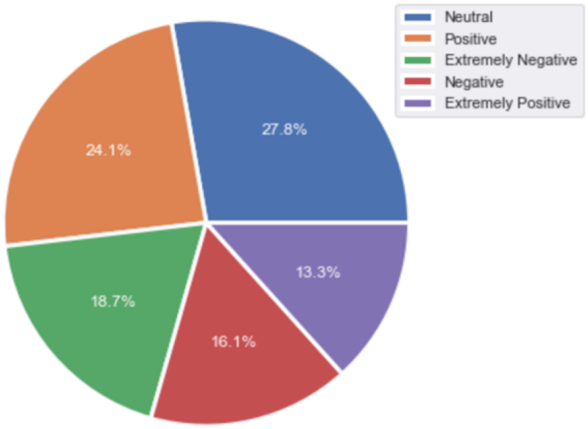

Figure 5: Pie chart representing label distribution

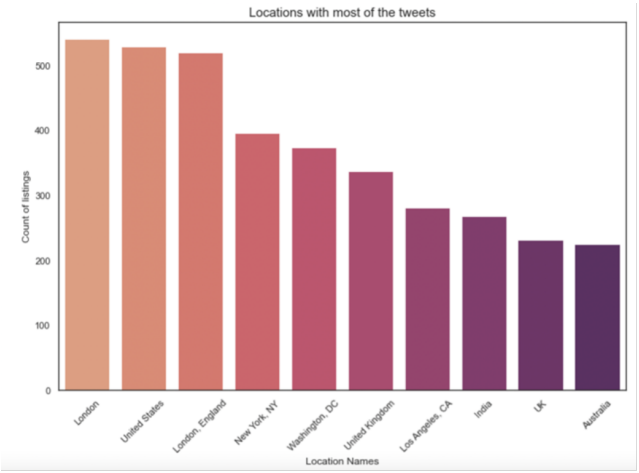

Figure 6: Bar plot containing locations with most number of tweets

	Location	Location_Count
0	London	540
1	United States	528
2	London, England	520
3	New York, NY	395
4	Washington, DC	373
5	United Kingdom	337
6	Los Angeles, CA	281
7	India	268
8	UK	232
9	Australia	225

Figure 7: Dataframe representing top 10 locations where the tweet came from

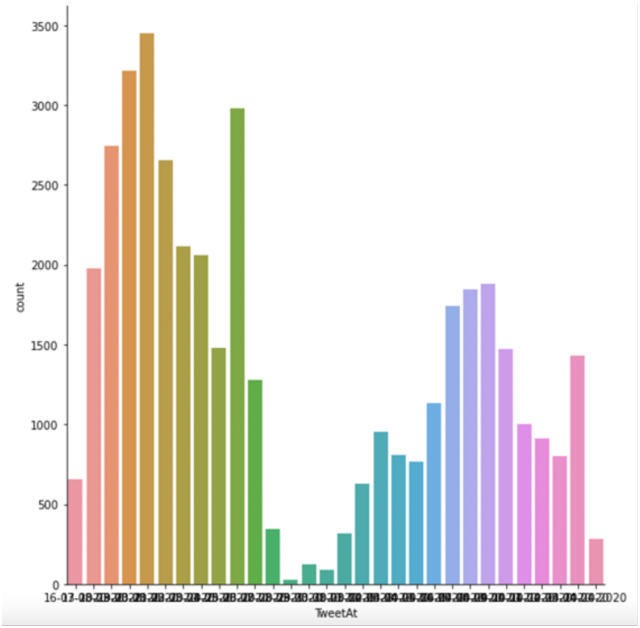

Figure 8: Graph containing the most frequent locations of tweet

3.3 Keyword Trend Analysis

We performed keyword trend analysis using Wordclouds on our pre-processed corpus to find the most frequent words. Wordclouds are visual representations of words that give greater prominence to words that appear more frequently. In this project, it helped us to generate and visualize most common words in the entire dataset and, most common words in the dataset for negative and positive tweets, respectively. To

get an idea of what are the words that contribute the most to different labels, we developed an automated function to generate word clouds. The function takes in a parameter label and will produce the word cloud for that sentiment in our dataset. The top 10 most used keywords are “Coronavirus”, “Corona”, “COVID-19”, “Virus”, “People”, “Sanitizer”, “Social distance”, “Food”, “Consumer”, and “Price” which depict the viewpoints and general feelings concerning this phenomenon that has assumed control over the world by storm. Wordclouds for positive and negative sentiments has been described in Fig. 9 and Fig. 10, respectively.

Figure 9: Keyword trend analysis for positive sentiment

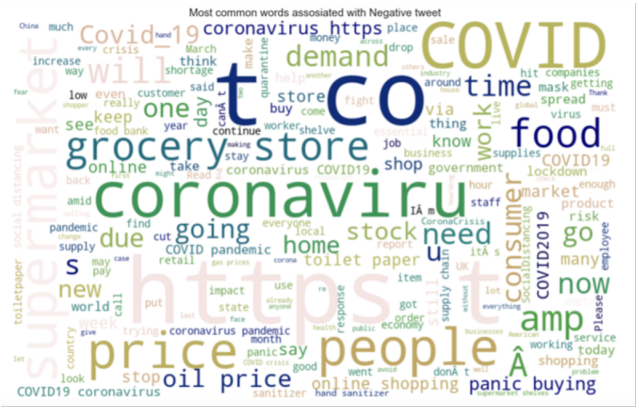

Figure 10: Keyword trend analysis for negative sentiment

3.4 Data Preprocessing

To make the data suitable for usage in machine learning models, it must be cleaned and brought to a more usable form. We undertake the following steps to preprocess the data:

- First, we select the only two columns that we need for predicting the sentiment - ’Original Tweet’ and ’Sentiment’. We drop the other columns.
- Next, text related pre-processing is done to clean up the the tweets. All the words are converted to lower case and the stop words are then removed. Next, we also remove all the URL’s and tags in the tweet and follow it up with stemming - which is the process of converting a word to its root form.
- Upon close inspection, we observed that the ’Extremely Negative’ and ’Negative’ sentiments had a lot of similarities. The same thing was also observed in the ’Extremely Positive’ and ’Positive’ sentiments. To avoid misclassification, we change the label of ’Extremely Negative’ to ’Negative’ and ’Extremely Positive’ to ’Positive’. Now, our dataset has 3 labels instead of 5.

3.5 Vectorizing

After the data is pre-processed, the textual data must be converted to feature vectors for it to used in machine learning models. We primarily use three techniques to vectorize the data - Count Vectorizer, TF-IDF Vectorizer and Word2Vec.

Count Vectorizer Count Vectorizer transforms a given text into a vector on the basis of the frequency (count) of each word. It gives us a vector of the size of the vocabulary where each index in the vector represents a word. We are using Scikit-learn’s implementation of CountVectorizer in our project.

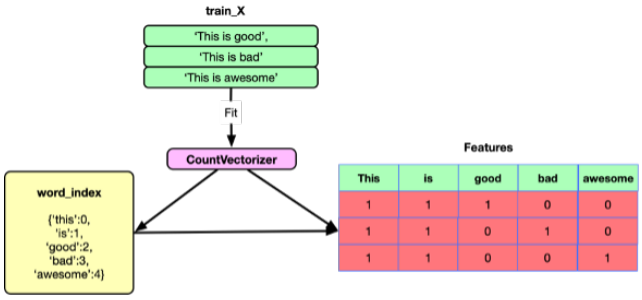

Figure 11: Basics of Count Vectorizer

TF-IDF Vectorizer A TF-IDF vector is very similar to a count vector except that it uses a TF-IDF score instead of the raw frequency counts. Count Vectorizer gives equal importance to all words while TF-IDF penalizes terms which occur very frequently or very rarely. We are using Scikit-learn’s implementation of TfIdfVectorizer in our project.

$$TF(t,d) = \frac{\textit{Number of times t appears in d}}{\textit{Number of terms in d}}$$

$$IDF(t) = \log \frac{N}{1+df}$$

where df (document frequency) is the number of documents containing the term ’t’.

$$TF - IDF(t,d) = TF(t,d) * IDF(t)$$

Word2Vec Word2Vec is an embedding technique developed by Google which uses a neural network to generate dense word vectors. It can figure out the context of a word in a document and also identify similar words. It uses the concept of cosine similarity to calculate the similarity between different words. Given two vectors A,B, their similarity can be captured by calculating the cosine of the angle between them. Word2Vec embeddings can be built using two methods: Continuous Bag of Words(CBOW) and SkipGrams.

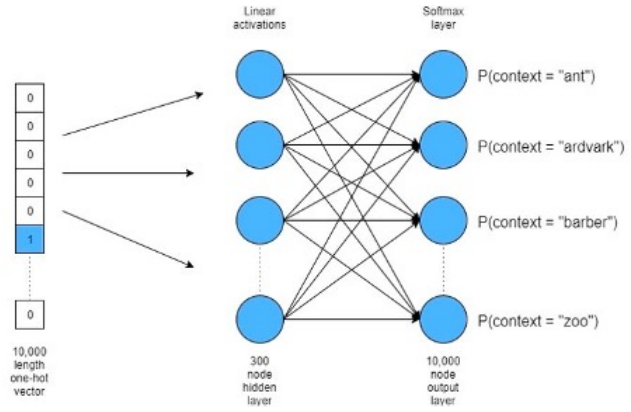

Figure 12: Neural network model to train Word2Vec

We are using Gensim's Word2Vec implementation in our project. We train the model on our training dataset to create dense vectors for every word in the sentence. The model maintains a dictionary for all the words in its vocabulary and its vector representation.

4 Modelling Methodology

To classify the sentiment of the tweet, we are primarily using three modelling methodologies - A machine learning approach, a neural network based approach and a BERT based approach

4.1 Machine Learning Approach

Naive Bayes Naive Bayes is a simple classification technique that works on the principle of Bayes Theorem. Bayes Theorem says that:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Given a set of words, our task is to predict its sentiment. Naive Bayes works by assuming that all the words contribute independently to determine the sentiment of the tweet. It also does not assign any weights to any of the words and assumes that each word has an equal weightage in determining the sentiment of the tweet. This can be represented by:

$$P(S|w_1, w_2....w_n) = \frac{P(w_1|S)...P(w_n|S)P(S)}{P(w_1)P(w_2)....P(w_n)}$$

where S stands for the Sentiment and w_i is each word in that tweet. The sentiment having the highest probability is chosen as the correct sentiment to represent the tweet. We have implemented Naive Bayes from scratch for predicting the sentiments of the tweets.

Multinomial Logistic Regression Logistic Regression is a technique used to classify data points into multiple classes. Logistic Regression primarily uses a Softmax function to classify data. A softmax function takes in a input vector of K real numbers and normalizes it into a probability distribution consisting of K probabilities. A softmax function is defined as:

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

The training process of a logistic regression model is as follows. A weight vector having the same dimensions as the input vector is initialized and their dot product is then calculated. This dot product is then sent to a softmax function, which gives the probability of each class to be predicted. Then, the log loss is calculated as the logarithm of the difference between the the actual output and the predicted output. We then use gradient descent to minimize this loss. This process is continued for a set number of epochs or until the loss is minimal. This model has been implemented from scratch in our code.

Random Forest Random Forest classifier comes in the category of ensemble learning where the results are generated from training multiple classifiers. Random Forest builds on top of Decision Tree Classifiers. Decision Tree classifiers learn simple decision rules from the features and predicts the incoming data based on that. The major drawback of Decision Tree Classifiers is that they tend to overfit to the data.

To deal with the overfitting (high variance) of the decision trees, bagging (Bootstrap Aggregation) is used. In bagging, n decision trees are trained on randomly sampled training data. Predictions are decided using the majority vote from the n trees. However bagging causes another problem of tree correlation where the trees generated can be highly correlated if a specific feature is heavily contributing to the classification.

To deal with the problem of high correlation, Random Forest includes the original bagging algorithm along with feature bagging. In every tree at each split, instead of including all the samples a subset of random features are included. This helps in tackling the issue of correlation of trees.

Support Vector Machine SVM is a supervised ML algorithm which can be used for both regression and classification tasks. The main idea behind SVM is to find a hyperplane which can separate the data points into different classes.To avoid misclassification during testing, SVM tries to find a hyperplane with a large margin between the data points and the hyperplane.

We are utilizing Scikitlearn's implementation of SVM for this task. A linear kernel is chosen and the 'C' value is set to 1.

CatBoost Catboost is an algorithm for gradient boosting on decision trees. During training, a set of decision trees

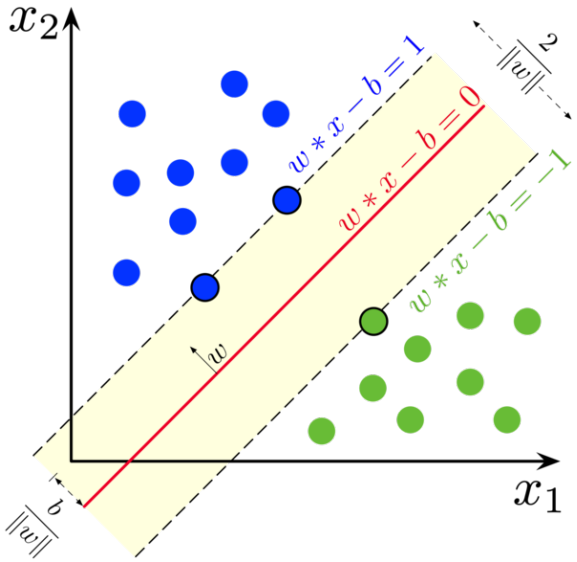

Figure 13: Example of SVM in action

is built consecutively. Each successive tree is built with reduced loss compared to the previous trees. A process called quantization is performed for each numerical feature to determine the possible ways to split data into buckets. Next, a "feature-split" pair is chosen for a leaf. This is done by selecting all possible "feature-split" options and then choosing the one with the minimum penalty. Then, regularization is performed to prevent overfitting. In the last step, "unbiased boosting" is performed to give the resultant tree.

We are utilizing the CatBoostClassifier available in the Catboost python library.

Multi Layer Perceptron A multi layer perceptron is a supplement of a feed forward neural network. It consists of 3 layers - an input layer, a hidden layer and an output layer. The perceptron uses gradient descent and back propagation to train the model. The perceptron could also use different activation functions in different layers such as Tanh and RELU. We also experiment with different hidden layer sizes and learning rates and choose the best model.

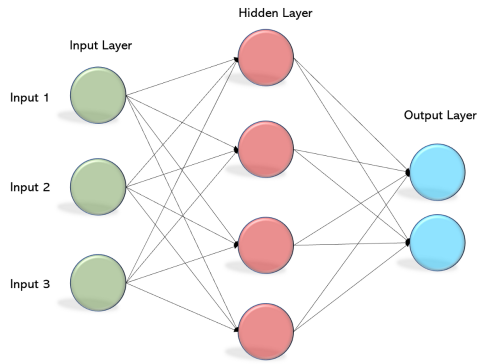

Figure 14: Example of a multi layer perceptron

4.2 Neural Network Approach

Recurrent neural networks (RNN) in general feature a context memory layer to incorporate previous state and current inputs for propagating information to the next state, and eventually the output layer for decision making. Canonical RNNs (simple RNNs) feature several different architectures, besides the Elman RNN for modelling temporal sequences and dynamical systems. One of the major challenges in training simple RNNs is due to the architecture's properties of unfolding in time for long-term dependency problems. Backpropagation through time (BPTT), which is an extension of the backpropagation algorithm, has been prominent for training simple RNNs. Due to problem of learning long-term dependencies given vanishing and exploding gradients with simple RNNs, long short-term memory (LSTM) recurrent neural networks have been developed. LSTM networks have better capabilities for learning long-term dependencies in temporal data using memory cells and gates.

In the last decade, with the deep learning revolution, several LSTM architectures have been developed. Bidirectional LSTM models process information in two directions, rather than making use of only previous context state for determining the next states which are based on bidirectional RNNs. In this way, two independent LSTM models allow both backward and forward information about the sequence at every time step. This enables better access to long range state information which have been useful for word embedding and several other sequence processing problems.

We present an experimental study that compares multi-label classification using LSTM and Bidirectional-LSTM as shown in the framework Fig. 11. We use approximately 90% of the dataset for training and 10% for testing and use the respective models and train them using the Kaggle COVID-19 twitter dataset which is publicly available.

In the LSTM model, we determine the model hyperparameters based on how the model performs in trial experiments. We used Keras preprocessing text Tokenizer to vectorize the text corpus by turning each text into either a sequence of integers (each integer being the index of a token in a dictionary) or into a vector where the coefficient for each token could be binary. We use a dropout regularization probability of 0.2 and 0.4 for LSTM and BD-LSTM models respectively, which feature: 256 input units, 32 batch size, adam optimizer and running the model for multiple epoch size found optimal results when running the model for 2 epochs for classifying the sentiments. The model summary for both the models is shown in Fig. 15 and Fig. 16, respectively.

4.3 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a Transformer-based model, where the training is bi-directional. This approach of BERT gives a better understanding of complex natural language and its context. The Transformers is an attention mechanism that learns the contextual relations between text entities in a text. Generally, a Transformer includes two different components, an Encoder and a Decoder, the responsibility of the Encoder is to convert the text information into mathematical vector form and

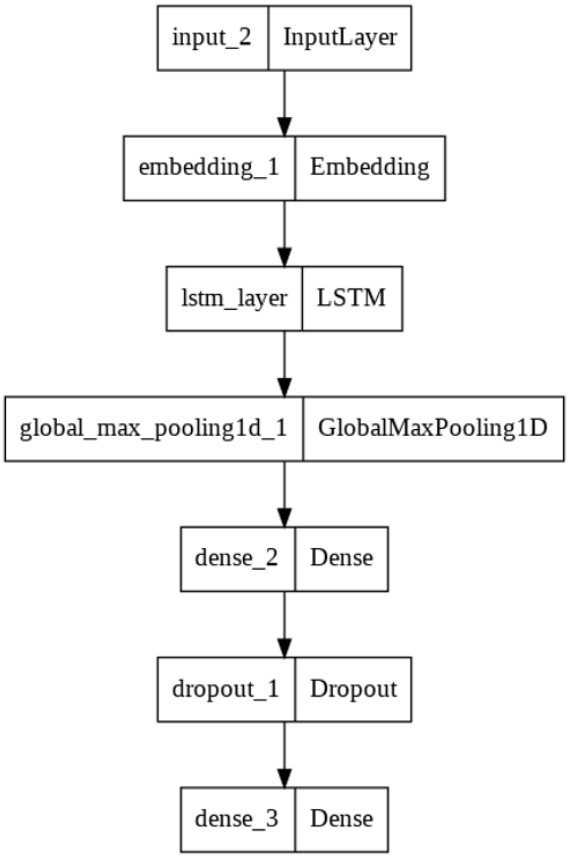

Figure 15: Model summary using LSTM

using the mathematical forms. The Decoder is used to make predictions based on the requirements. BERT does not include the decoders in its standard form because originally, it was created to generate a language model. Moreover, BERT is very well suited for various tasks because of its transfer learning mechanism as it gives us flexibility of incremental training in our model, which in turn helps us to fine tune our model. The model is trained on over 2,500 million words from text passages of Wikipedia (English) and "BookCorpus", which is a dataset that comprises more than 11,000 of unpublished books that are from 15+ different genres. This statistically increases the chance of the model having a near human sense of language model. Out of a plethora of pre-trained BERT layers, the two most popular models are "BERT-BASE" and "BERT-LARGE". The major difference is the number of encoders and bidirectional self-attention heads. BERT-BASE has 12 encoders, 12 attention heads whereas BERT-LARGE comprises 24 encoders, 16 attention heads. For our project, we implemented and developed the BERT-BASE model, which would aid us in reducing the turnaround time for training of the model with different hyper parameters. For our model, we used approximately 90We performed various preprocessing steps such as Label Re-Mapping, Data Sanitization, Data Balancing, Data Splitting, One-Hot Encoding, Data Re-formatting. The model has 4 layers in the network along with their contributing neu-

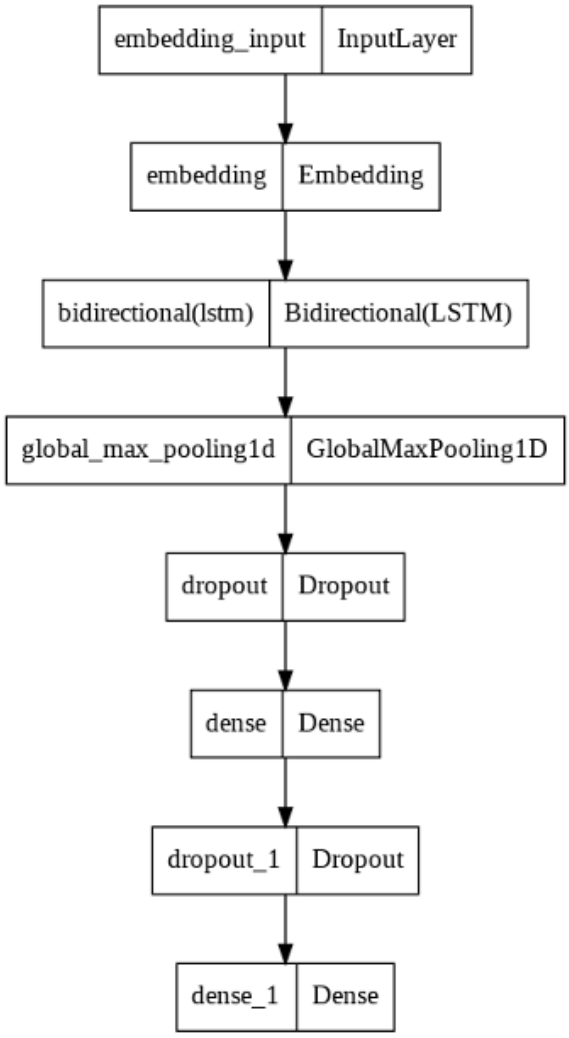

Figure 16: Model summary using bidirectional LSTM

ron counts, an input id layer (180), an attention masks layer (180), a pre-trained BERT-base layer (768), and a dense output layer (3). The activation function for the output layer was set to softmax. The BERT model yielded highest performance results with 5 epochs and batch size 16.

5 Ablation Study

An ablation study is when you eliminate sections of the input systematically to see which elements of the input are significant to the output. The importance of the ablation study is to understand causality in the system which is the goal of any research. An ablation is a low-effort approach to investigate causation where you systematically remove parts of the input to see which parts of the input are relevant to the networks output. If we take any sophisticated deep learning experimental configuration, there is a good possibility that we may delete a few modules without affecting performance. In deep neural network model, we have removed layers in a rational manner and examined how this affects the network's

performance. Removing the noise from the research process by conducting ablation studies helped us achieve good results. To that end, we conducted a series of experiments to tune the models on hyperparameters related to many aspects of the modeling process. For instance, we tuned some parameters related to the dataset pre-processing (like the token level, lower text), embedding parameters (such as the embedding type, dimensions) and others related to training and the architecture of the model (such as learning rate, number of epochs, filter size, drop out etc.).

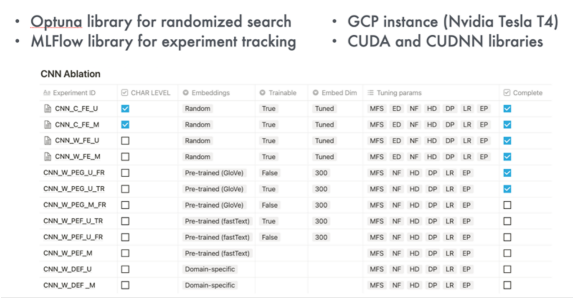

Figure 17: Ablation Study for RNN

input : A time series $\mathbf{T}_{l,f}$ // length l and features f
: An RNN `model` $(\cdot)$ $l-(w+h)$ predictions
output: An error matrix $\mathbf{E}_{Avg}$ of size $w \times f$

$Error_{Avg}$ =[ ]
$\hat{\mathbf{y}} \leftarrow$ `model`$(\mathbf{X})$
for *feature in* f **do**
  $E_{Feature}$ = [ ]
  /* Iterate over region A */
  for i *in* 1:`len` $(\mathbf{A})$ **do**
    $\mathbf{X}_{Ablated} \leftarrow$ `ablate`$(\mathbf{X}, i+91, feature)$
    $\hat{\mathbf{y}}_{Ablated} \leftarrow$ `model`$(\mathbf{X}_{Ablated})$
    $\mathbf{e} \leftarrow \frac{|\hat{\mathbf{y}}_{Ablated} - \hat{\mathbf{y}}|}{\hat{\mathbf{y}}}$
    /* Concatenate nonzero errors from RNN sliding over ablated value */
    $\mathbf{E}_{Feature} \leftarrow$ concat$(\mathbf{E}_{Feature}, \mathbf{e}\,[i{:}i+91])$
  end
  $\mathbf{E}_{Avg}[feature] \leftarrow \frac{1}{\mathtt{len}(\mathbf{A})} \sum_{1}^{j=\mathtt{len}(\mathbf{A})} \mathbf{E}_{feature}[j]$
end
return $\mathbf{E}_{Avg}$

Figure 18: Ablation Algorithm for RNN

6 Results and Discussion

The main aim of this study is to evaluate the classification performance on COVID-19 datasets and provide benchmarked results. This project applies ML, DL, and transformer-based methods to COVID-19 datasets and examines their performance using several evaluation measures. The proposed approach is used to gauge the performance in the sentiment classification task. The performance assessment metrics in this study include accuracy, precision, recall, and F1-score. These standard performance indicators have been carefully chosen to attest to the model's capacity to generate the best categorization performance. Following the experimentation procedure, the experimental outcomes are compared to state-of-art-methodologies.

6.1 Evaluation Metrics

Before proceeding with the results, we need to first define the metrics that we are using to evaluate our models.

Confusion Matrix A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm.

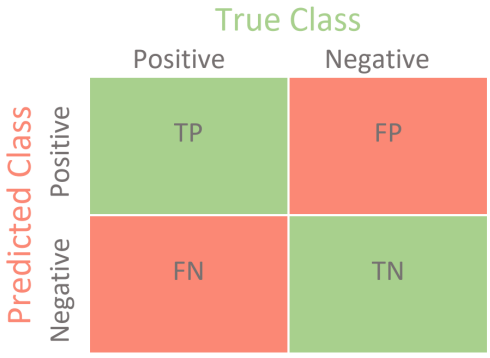

Figure 19: A sample confusion matrix

Accuracy Accuracy is the fraction of predictions the model correctly classified.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision Precision is the fraction of the relevant instances among the retrieved instances

$$Precision = \frac{TP}{TP + FP}$$

Recall Recall is the fraction of the relevant instances that were retrieved

$$Recall = \frac{TP}{TP + FN}$$

F-1 Score F-1 score is the harmonic mean of precision and recall.

$$F\text{-}1\ Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

6.2 Machine Learning Model Performance

Baseline model - Naive Bayes

A variety of models ranging from Naive Bayes to Multi Layer Perceptron have been explored in this section. We started out with Naive Bayes as the baseline model and experimented with more complicated models later.

As it can be seen in Figure 20, Naive Bayes has the lowest score with an F-1 score of 0.63. Next, Logistic Regression, Support Vector Machine and Random Forest have comparable scores of 0.78, 0.76 and 0.75 respectively. Even the

multi layer perceptron performs similar to the aforementioned three methods. However, Catboost stands out among the rest with an F-1 Score of 0.80.

Classifier	Accuracy	Precision	Recall	F-1 Score
Naïve Bayes	0.67	0.60	0.68	0.6375
Logistic Regression	0.79	0.78	0.78	0.78
SVM	0.78	0.76	0.77	0.76
Random Forest	0.76	0.74	0.76	0.75
CatBoost	0.81	0.80	0.80	0.80
MLP	0.79	0.78	0.77	0.78

Figure 20: Machine Learning Model Performance

6.3 Neural Network Model Performance

The experiments show that it is very important to the knowledge of the RNN model with the accuracy of the scores. We achieved an accuracy of 91.71% along with 94% recall, 91% precision and 92% f1-score which is great performance for our model. This model performed exceedingly well on our corpus and amongst all the research we have done. To that end, LSTM model proved to achieve the highest accuracy in this project.

	precision	recall	f1-score	support
0	0.91	0.94	0.92	654
1	0.87	0.91	0.89	263
2	0.96	0.89	0.92	603
micro avg	0.92	0.92	0.92	1520
macro avg	0.91	0.91	0.91	1520
weighted avg	0.92	0.92	0.92	1520
samples avg	0.92	0.92	0.92	1520

Figure 21: Classification report for RNN using LSTM

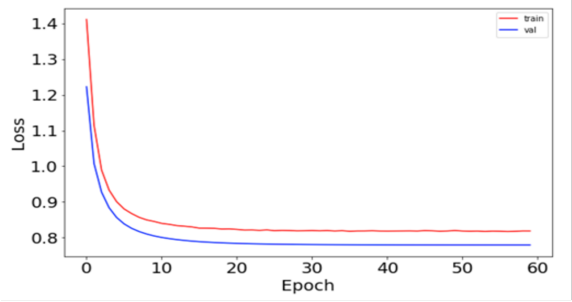

Figure 22: Loss curve

```
48/48 [==============================] - 2s 45ms/step - loss: 0.4124 - accuracy: 0.9171
[0.4124337136745453, 0.9171052575111389]
```

Figure 23: Loss and Test accuracy for RNN using LSTM

6.4 BERT Model Performance

The BERT model achieved an accuracy of 98.46%, along with recall 98.33%, precision 98% and f1-score 98.33%. The context driven encoded data from the BERT embedding layer helps in achieving such good results as shown in Fig 24.

```
_____________Classification Report for BERT_____________

              precision    recall  f1-score   support

    Negative       0.99      0.99      0.99     15398
     Neutral       0.96      0.98      0.97      7713
    Positive       0.99      0.98      0.99     18046

   micro avg       0.98      0.98      0.98     41157
   macro avg       0.98      0.98      0.98     41157
weighted avg       0.98      0.98      0.98     41157
 samples avg       0.98      0.98      0.98     41157


Test Accuracy -----> 0.9846684646597177
```

Figure 24: BERT Classification report

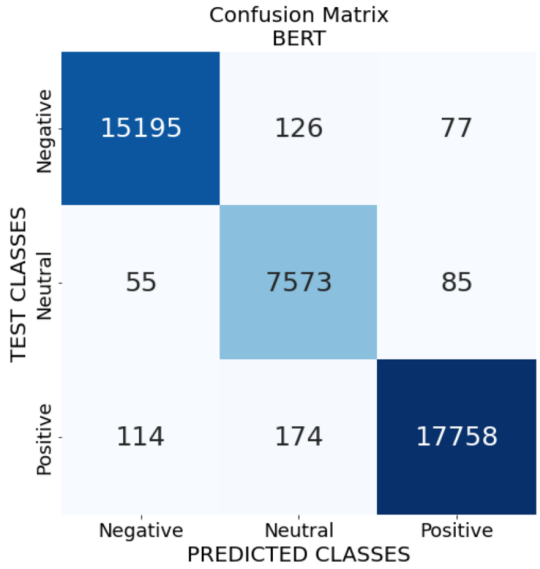

Figure 25: Confusion Matrix for BERT

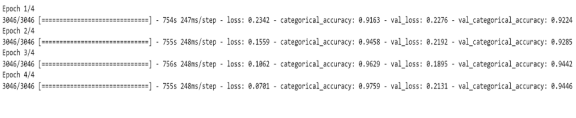

Figure 26: BERT Training

Classifier	Accuracy	Precision	Recall	F-1 Score
CatBoost	0.81	0.80	0.80	0.80
RNN with LSTM	0.91	0.91	0.91	0.91
BERT	0.98	0.98	0.98	0.98

Figure 27: Comparison of all the three methods

7 Conclusion

Since the outbreak of the COVID-19 pandemic and with a new normal of staying at home, working from home, and "isolation time," social networking media has been extensively used to share news, opinions, emotions, advice; however, most of the data on social media are irrelevant and do not belong to the actual scenario. This project deals with classifying user tweets based on sentiments using the COVID-19 dataset. We evaluate ML and DL classifiers using several vectorization techniques that automatically learn features without human interference. There is much misinformation on social media; therefore, health organizations need to develop a stable system for detecting coronavirus precisely to preclude the spread of fake news. In this project, we presented the effectiveness of underlying machine learning models by classifying tweets based on sentiments and achieving good accuracy scores. The neural network approach based on LSTM performed very well on the given dataset and showed higher accuracy of 91.7% when compared to similar state-of-the-art studies along BERT transformer model which obtained an accuracy of 91.07%. Moreover, we explored textual data using spaCy library and built a text classification model which will predict the sentiment of the tweet provided by a user. We were able to get a deep dive into the world of Machine Learning and the timeline of doing a project within the field. It was a great cumulative experience for us taking the theory we learn in class and using it as a real-world application. To that end, this project adds to a growing corpus of research obtaining the best test accuracy of around 92% using Neural Network model (Bidirectional LSTM) amongst all the acquired research.

8 Future Directions

In future work, we plan to analyze public sentiments toward other essential topics, such as government response to the pandemic situation, healthcare facilities by government, offline examination, and mental health by using DL algorithms to increase their performance on the dataset. Furthermore, future studies could dive deeper into advanced models and advanced ensemble models which might prove beneficial along different feature engineering techniques.

9 Statement of Contribution

Vidhi Parekh conceived and planned the experiments to perform data pre-processing, exploratory data analysis to understand the insights of key characteristics and implemented CountVectorizer. Sourabh Natesh worked on Tf-idf Vectorizer, Word2Vec and also developed Multinomial Logistic Regression and Multinomial Naïve Bayes models from scratch. Sourabh also worked on producing the evaluations for the same and developed an RNN based approach using LSTM. Vidhi Parekh developed state of the art machine learning models (RF, SVM, multinomialNB, LR, CatBoost, MLP classifier) and provided comparative analysis of the results with baseline approach. Visheshank Mishra developer BERT model using different hyperparameters and produced the results for the same. Vidhi Parekh contributed to the interpretation of the results, designed the model and the computational framework for RNN using LSTM along with an ablation study. Vidhi Parekh performed sentiment prediction using spaCy and took the lead in writing the report along with Sourabh Natesh and Visheshank Mishra.

10 Github

The code used for analysis, experiments and discussion is provided in a repository at https://github.com/Vidhi00/CS6140-COVID-19-Tweet-Scanner.

References

[1] Kharde, V., Sonawane, S. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications, 139(11), 975–8887. https://arxiv.org/pdf/1601.06971.pdf

[2] Murthy, Dr Allu, Shanmukha Andhavarapu, Bhargavi Bagadi, Mounika. (2020). Text based Sentiment Analysis using LSTM. International Journal of Engineering Research and. V9. 10.17577/IJERTV9IS050290.

[3] Dang, N. C., Moreno-García, M. N., De la Prieta, F. (2020). Sentiment Analysis Based on Deep Learning: A Comparative Study. Electronics, 9(3), 483. https://doi.org/10.3390/electronics9030483

[4] Zunera Jalil1, Ahmed Abbasi1, Abdul Rehman Javed, Muhammad Badruddin Khan2, Mozaherul Hoque Abul Hasanat2, Khalid Mahmood Malik3 and Abdul Khader Jilani Saudagar2, COVID-19 Related Sentiment Analysis Using State-of-the-Art Machine Learning and Deep Learning Techniques, https://www.frontiersin.org/articles/10.3389/fpubh.2021.812735/full

[5] Twitter Sentiment Analysis LSTM. (n.d.). Kaggle.com. Retrieved April 25, 2022, from https://www.kaggle.com/code/kritanjalijain/twitter-sentiment-analysis-lstm/notebook

[6] Sentiment analysis: Predicting sentiment of covid-19 tweets. Analytics Vidhya. (2021, February 22). Retrieved March 8, 2022,https://www.analyticsvidhya.com/blog/2021/02/sentiment-analysis-predicting-sentiment-of-covid-19-tweets

[7] Coronavirus tweets NLP - Text Classification. (2020, September 8). Kaggle.https://www.kaggle.com/datatattle/covid-19-nlp-text-classification

[8] R. (2022, February 4). Covid-19 Text Classification Using BERT (TPU). Kaggle. https://www.kaggle.com/ravikumarmn/covid-19-text-classification-using-bert-tpu/data

[9] Ludovico Cuoghi. (2021, December 22). Twitter sentiment analysis with Bert + Roberta . Kaggle. https://www.kaggle.com/code/ludovicocuoghi/twitter-sentiment-analysis-with-bert-roberta

[10] Multilayer Perceptron - an overview — ScienceDirect Topics. (n.d.). Www.sciencedirect.com. https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron

[11] Alammar, J. (n.d.). The Illustrated Word2vec. Jalammar.github.io. https://jalammar.github.io/illustrated-word2vec/

[12] Overview. (n.d.). Catboost.ai. Retrieved April 25, 2022, from https://catboost.ai/en/docs/references/training-parameters/

[13] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. ArXiv.org. https://arxiv.org/abs/1301.3781

[14] Pennington, J., Socher, R., Manning, C. (2014). GloVe: Global Vectors for Word Representation. https://nlp.stanford.edu/pubs/glove.pdf

[15] Chandra, R., amp; Krishna, A. (2019, August 19). Covid-19 sentiment analysis via deep learning during the rise of novel cases. PLOS ONE. Retrieved from https://journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0255615

[16] RAZ, M. (2021, September 18). LSTM 89% coronavirus tweets sentiment analysis. Kaggle. Retrieved from https://www.kaggle.com/code/meisamraz/lstm-89-coronavirus-tweets-sentiment-analysis

[17] Ching, D. (2021, December 2). How do neural networks really work? Analytics Vidhya. Retrieved from https://www.analyticsvidhya.com/blog/2021/12/how-do-neural-networks-really-work/

[18] Mishra, A. (2020, May 28). Metrics to evaluate your machine learning algorithm. Medium. Retrieved from https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[19] Hosamwajeeh. (2022, March 28). Corona virus tweets classification NLP/NLTK /81. Kaggle. Retrieved from https://www.kaggle.com/code/hosamwajeeh/corona-virus-tweets-classification-nlp-nltk-81

[20] Gauravsahani. (2021, April 29). Covid-19 sentiment analysis using spacy. Kaggle. Retrieved from https://www.kaggle.com/code/gauravsahani/covid-19-sentiment-analysis-using-spacy

[21] Sentiment analysis: Predicting sentiment of covid-19 tweets. Analytics Vidhya. (2021, February 22). Retrieved from https://www.analyticsvidhya.com/blog/2021/02/sentiment-analysis-predicting-sentiment-of-covid-19-tweets/

[22] Horev, R. (2018, November 17). Bert explained: State of the art language model for NLP. Medium. Retrieved from https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

[23] WOLFRAM. (2019, April 12). BERT Trained on BookCorpus and English Wikipedia Data. Bert - Wolfram Neural Net Repository. Retrieved from https://resources.wolframcloud.com/NeuralNetRepository/resources/BERT-Trained-on-BookCorpus-and-English-Wikipedia-Data

[24] Ludovico Cuoghi. (2021, December 22). Twitter sentiment analysis with Bert + Roberta . Kaggle. https://www.kaggle.com/code/ludovicocuoghi/twitter-sentiment-analysis-with-bert-roberta

[25] Alammar, J. (2018, December 3). The illustrated Bert, Elmo, and Co. (how NLP cracked transfer learning). The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning) – Jay Alammar – Visualizing machine learning one concept at a time. Retrieved from http://jalammar.github.io/illustrated-bert/