Sentiment Analysis on COVID Tweets: An Experimental Analysis on the Impact of Count Vectorizer and TF-IDF on Sentiment Predictions using Deep Learning Models

Ghulam Musa Raza, Zainab Saeed Butt, Seemab Latif, Abdul Wahid
School of Electrical Engineering and Computer Science
National University of Sciences and Technology (NUST)
Islamabad, Pakistan
{graza.mscs19seecs, zbutt.mscs19seecs, seemab.latif, abdul.wahid}@seecs.edu.pk

2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2) | 978-1-6654-1285-8/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICoDT252288.2021.9441508

***Abstract*— Due to the higher popularity of social media and its excessive use, COVID-19 has become the topic of the talk since 2019 and it has become a cause of stress, anxiety and depression for people around the world. In this article, we experimented with different classifiers on COVID data to train deep neural networks to enhance the accuracy rate using two popular word embedding techniques: Count Vectorizer and Term Frequency-Inverse Document Frequency. Finally, we compare accuracies and observe that TF-IDF comes out to be more efficient as compared to Count Vectorizer where datasets are of huge volume and in our case i.e., for covid19 tweets, both vectorizers have been approximately similar in performance except on Single Layer Perceptron where Count Vectorizer results in 10% more efficiency in terms of accuracy.**

***Keywords*— COVID, Deep Learning, NLP, Classifiers, Sentiment Analysis**

I. Introduction

COVID-19 has changed patterns of living in the last two years and people have tended to express their sentiments on social media more than ever. In recent years, Machine Learning and Artificial Intelligence have played a remarkable role in the Tech industry and there has been a growing interest in the use of Deep Learning techniques to accomplish sentiment analysis and emotional intelligence.

In this article, we are experimenting with different classifiers on Twitter data to train a deep neural network to optimize accuracy rate using two popular word embedding techniques i.e., Count Vectorizer and Term Frequency-Inverse Document Frequency. Our goal is to analyze significantly how people have been reacting on COVID-19 and its different stages throughout the year by classifying tweets in Negative, Positive and Neutral sentiments. To classify our tweets into number-based vectors, Natural Language Processing techniques for Feature Extraction i.e., Count Vectorizer and TF-IDF have been used to get a bag of words on tweets. TF-IDF uses Count Vectorizer to count term frequencies and also incorporates document inverse frequency which helps in evaluating the measure of a word's relevancy appearing in multiple documents. To retrieve number vectors after the data cleaning process, punctuation marks are removed, and all words are translated to lowercase. The vocabulary of recognizable words is developed, which is subsequently used to encrypt invisible text. In this way, we get features that are participating most significantly in the context. In the methodology section, we elaborate a detailed discussion on word embedding methods Count Vectorizer and TF-IDF. Then we perform classification using Deep Learning classifiers: Support Vector Machine, Logistic Regression, Perceptron, and Bernoulli Naive Bayes Classifier. In the end, we perform comparative analysis after the implementation of both techniques on all four classifiers and share results.

II. Related Work

Chakraborty proposed a study on behaviors of people showing how the popularity of user's data affects information accuracy on social media platforms [1] using English language tweets dataset for COVID-19 [2]. Dataset had a vast capacity and contained around 22 million unique ids of tweets. They have taken two data sets for experimentation. Dataset 1 contains around 2 lac dissimilar tweets from December 2019 to May 2020. Dataset-2 contains tweets that were retweeted a higher number of times from January to March 2020. Then different data cleaning and POS tagging techniques were applied further on tweets.

Fuzzy Logic Model-based Gaussian Membership Function has been proposed to extract positive and negative sentiments. It is a multivalued function whose actuality values range between 0 and 1. For Feature Extraction they have used TF-IDF on dataset 1 and Count Vectorizer (distributed) on dataset 2. They have achieved 79% accuracy through the proposed fuzzy logic model. Paper focused on inference system and use of fuzzy logic-based model i.e., Gaussian membership inference system. More Accuracy could be achieved based on classifiers taken as different hyperparameters.

Kruspe introduced Cross-language based sentiment analysis of European Tweets and messages during COVID-19 using Neural Network Model [3]. The effects of lockdown on the moods of people were analyzed. They have trained their model using training data given in [4]. The dataset contains around 1.5 million tweets later classified into negative, positive and neutral sentiments. They have implemented a neural model based on multilingual sentence embedding. The input layer of the NN model is followed by pre-trained sentence embedding hence resulting in the formation of vectors. These vectors are then fed into a fully connected RELU layer with 128 dimensions. To remove overfitting dropout has been used with a percentage of 50.

[4] have introduced a model to classify the emotion of people through their tweets. What makes this article different is that instead of classifying emotions as simple negative or positive, authors have classified the emotions further (more positive, strong negative etc.) by finding the connection between the words. The authors did not rely on the previous

existing datasets. So, they have used updated Twitter API for data collection.

[5] Offers the global sentiment study of Coronavirus-related tweets and how people's sentiment has shifted over time in various countries. In addition, tweets relating to Work From Home (WFH) and Online Learning were included to assess the effect of Coronavirus on everyday facets of life, and the shift in opinion over time was noted. Three datasets are used in the proposed model as discussed below:

- **COVID tweets:** Using Twitter Scraper, which uses the python box request store to download the content, 165116 tweets were scraped based on the keywords "coronavirus," and Beautifulsoup4 to decode the retrieved content.
- **Work from home tweets:** For the date, 22 Feb 2020 to 3 May, 41349 tweets were scraped using the queries "work from home" and "WFH" with the aid of TwitterScraper.
- **Existing datasets:** This data set was obtained from Kaggle that was related to COVID-19.

The multilayer perceptron is used with backpropagation. Relu activation function is used. Dropout layer for avoiding overfitting. MaxPooling1D for calculation for each path. Finally, the last layer is connected to the Softmax function. Accuracy with ANN: 76% Accuracy with LSTM: 84.5%.

In [6], authors aim to classify in three separate timelines to analyze the sentiments of Filipino users by taking their tweets about COVID-19. A total of 65,396 COVID-19-related tweets were submitted for data processing using R Statistical Tools to accomplish this objective. According to publishers, it was the first article to conduct sentiment analysis on COVID-19 tweets, not just in the Philippines, but also globally. Data is collected from National Capital Region (NCR). R Statistical Software is used to analyzing and processing the data. Only English and Taglish tweets are used. They split the dataset into three milestones: one was before COVID, then during COVID-19 and after the Speech to the Nation by President Duterte.

In [7] Recurrent Neural Network (RNN) has been used to train the model and results have been taken. The first layer is the embedded layer which performs integral encoding. Then random weights are assigned to this layer. Results are compared with existing work (Text Blob) and showed in form of graphs where it is visible that there is a difference in RNN result and Text Blob. The authors have not described specifically that which RNN variant they are using and what accuracy rate they have achieved. User-based input of their proposed model needs more work.

In [8] researcher provide a domain-specific framework for understanding the feelings share among people around the world. COVID-19 related tweets are collected from the Twitter site to do this task. Three data sets are used. All of them are obtained from Tweeter API. The dataset contained tweets with the hashtag #coronavirus, the second dataset includes the hashtag #COVID19, and Dataset3, which is generated manually incorporates previous datasets. There are 10,000 tweets in both Dataset 1 and2, while 20,000 tweets are in the manually generated Dataset 3. They have implemented multiple classifiers like Logistic Regression, Multinomial Naïve Bayes, Decision Tree, Random Forest, Support Vector Machine and XGBoost. Results are obtained in three types of classification.

- **Binary-class Classification:** It deals with Positive and negative. The precision rate reached by all of them is near 91%.
- **Multi-class Classification:** It deals with Positive and negative and neutral. The accuracy rate average is 88-89% by all classifiers. SVM is best for dataset 2 decision tree is good for the dataset.
- **Cross-Dataset Evaluation:** The findings note that with unigram and bigram selected features in binary-class configuration, most classifiers perform better. However, using only the unigram feature collection, stronger results were obtained in multi-class environments. The tests will deduce that the mentioned model was capable enough to predict the emotions behind COVID specified tweets with reasonable precision. However, more accuracy must be achieved.

In [9], the authors introduced a data analyzer and Repository called Covis. With the help of Twitter streaming API and Python Tweepy Package the data was fetched (Module from the stream of tweets containing words like COVID or corona). From 5th March 2020 to 2nd July 2020, they had collected a large number of tweets which were around 1.3 terabytes of raw data. Once every week still they are collecting the data and continuously changing its statistics in the repository. This study lacks focusing on any classifiers. Similar work could be done with more efficiency with the help of DL techniques which are used in Tweet analysis.

III. Proposed Methodology

Our paper deals with the analysis of the sentiments instigated by Tweets shared by users during the COVID-19 pandemic. We aim to get the highest possible accuracy on deep learning models after feeding word vectors to the combination of classifiers.

To get our input into a machine-friendly format; we have converted our cleaned data into vectors that represent words into numbers. By using feature extraction techniques of Natural Language Processing called Count Vectorizer and TF-IDF.

We have defined our experiment into two phases as illustrated in Fig 1. The first part of the sentiment analysis deals with the Dataset gathering and cleaning process and the second comprises word embedding and classifications processes. We have used an open-source dataset from [6] Kaggle and experimented with the models. Dataset had been collected through Twitter API in August 2020.

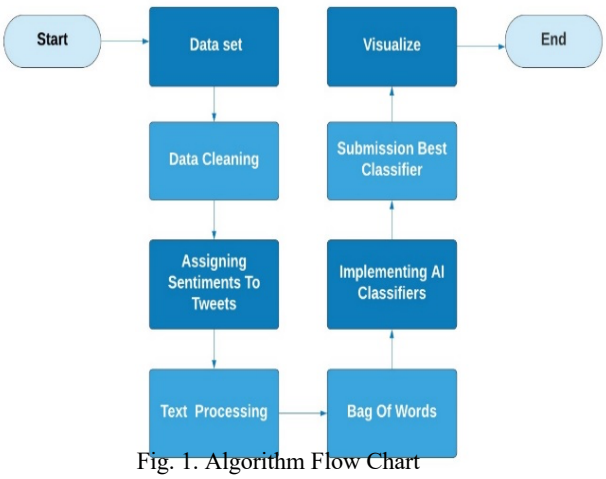

Fig. 1. Algorithm Flow Chart

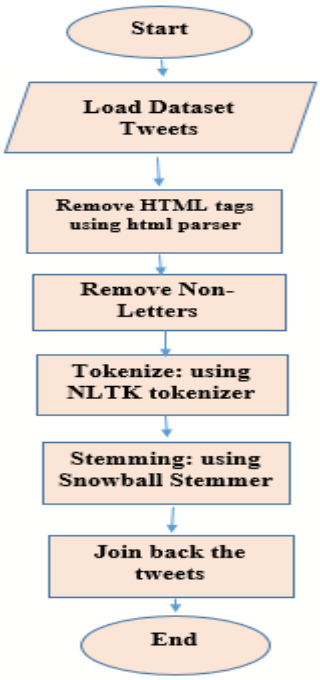

Fig. 2 Data Cleaning Pipeline

A. *Data Cleaning and Sentiment Labeling*

Data originally exists in the form of a cluster of tweets and words that will not be recognized by our learning models. On the first stage of cleaning, see fig. 2, unnecessary columns were removed from the dataset like user_favourites, user_verified, hashtags, source, is_retweet. Then important entities and hashtags were extracted which included user account and tags like #covid_19, #covid19, #covid, #coronavirus, #outbreak, #virus, #pandemic. Then tokenization is done, and special characters are removed.

The next step after data cleaning is to assign sentiments to tweets. Polarity has been measured and used to label each tweet as positive or negative sentiment. We have used an NLP library named *Textblob* that has a method *Sentiment* with the property *Polarity*. Value of polarity is a float type that lies between -1 and 1 [7] where -1 refers to negative and 1 refers to positive sentiment. The subjectivity of the sentiment lies in less and greater than 0 as shown in table I. As neutral sentiments don't have a direct impact on depression or delight, we have discounted 0 polarities. After our tweets have been labelled with sentiments, the dataset is ready to be trained with fields like user_name, user_location, date, text, tags, account, entity_text, sentiment, polarity.

TABLE I
SENTIMENTS AND POLARITY

Text	Sentiment	Polarity
Smelled scent hand sanitisers today someone past think intoxicated.	negative	-0.25
coronavirus covid19 deaths continue rise s almost bad politicians businesses want	negative	-0.7
covid19 will change work general recruiting specifically via recruiting	positive	0.05
let s protect covid19 s real numbers climbing fast continent let s n	positive	0.2
the image doesn't list sourced careful overall risk dying statistics related	negative	-0.033
kolar need blood type b positive jalappa hospital blood component need plasma b ve covid19 recover	positive	0.227273

Since our tweets are labelled with sentiments based on their polarity, we split our data into the Train and Test set. For training data around 60862 random tweets were taken and for the test 40573 tweets were partitioned. From the test set, the labelled sentiments' column was dropped. For training, we perform few more cleaning techniques to get the most frequent words from Bag of Words. First, we again use *BeautifulSoup* library to convert data to lowercase and convert the letters into a vector of words. A sample tweet being tokenized using NLTK *punkt* library looks like *today, s, covid, specifics, oregon, reports, new, cases, two, new, deaths, covid*. By using the NLTK *stem* method, we remove stems whereas the meaning of words remains attained. Our vector after stemming looks like *today, covid, specif, oregon, report, new, case, two, new, death, covid*. In the last step of cleaning, we join back the stemmed letters back to the tweet text. The next step is to convert the most frequent words into integer-based vectors.

B. *Feature Extraction via Count Vectorizer*

We convert text-based word vectors into vectors of tokens to form a Bag of Words based on word's frequency using Count Vectorizer a method of Sklearn Library. In this way, we form our feature representative vectors. Before implementing a count vectorizer it is necessary to dig into an

evaluation of high-frequency words that are the thorough representation of tweets. For this purpose, we plot the words in tweets according to their frequency of occurrence.

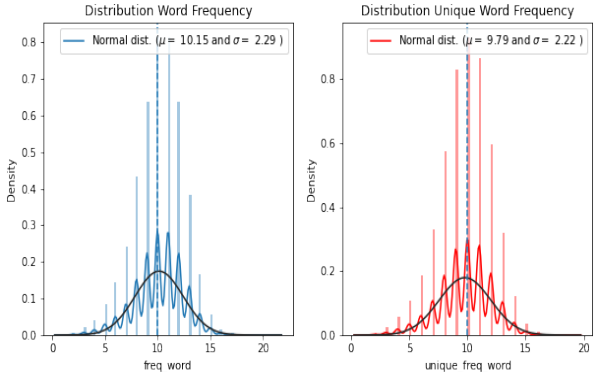

Fig. 3. Frequency Distribution before Bag of Words Count Vectorizer

We use methods to *apply* and *lambda* to filter out frequent words from train data frame [10] We form two frequency distributions of frequently used words *freq_word* from train data and frequently used unique words *unique_freq_word* from the former. The plotted frequency distribution graph is shown in fig 3.

C. Feature Extraction via TF-IDF

Second, we implement Term Frequency-Inverse Document Frequency where we count the TF i.e., frequency of a word in a tweet. We multiply the score of TF and IDF in the end to obtain the TF-IDF score of all tweets. The vocabulary of recognizable words is developed, which is subsequently also used to encrypt invisible text.

We investigate TF-IDF as a measure to count words' frequency of occurrence in multiple tweets. This works by increasing regularly with increasing frequency of a word in a tweet but is counterbalanced by several tweets that contain the word. What makes its vocabulary generation more meaningful is that words like if, what, where, when, how are words that usually appear more often in most tweets but aren't of much weightage. Hence these words must rank lower. However, if the word 'dead' appears in one tweet and also in other tweets, this might mean something relevant when it comes to the COVID-19 situation. To implement TF-IDF, we take our split datasets train and test as we did for Count Vectorizer. We feed our train and test again into the cleaning pipeline.

A random tweet post-cleaning, tokenization and stemming looks like "let offic system enjoy natur covid coronavirus". We use *sklearn.feature_extraction.text* library's method 'TF-IDF Vectorizer' that diminishes the weights of words appearing in most tweets in common and measures them incapable of discriminating the tweets. This is the opposite of simply counting the frequency of words as Count Vectorizer does. The resultant matrix comprises each Tweet represented as a row and each word represented as a column and its significance i.e., weight is finally computed by multiplying TF by IDF. After importing *TfidfVectorizer* from *sklearn.feature_extraction.text,* we set its parametric N-grams as a unigram, bigram and trigram respectively. We specify a maximum of 700 features to avoid computing

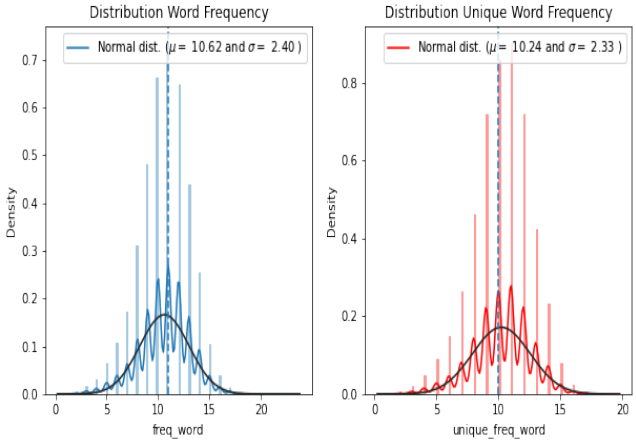

Fig. 4 Frequency Distribution before Bag of Words TF-IDF

complexity for the first experiment. After transforming our words into vectors via TF-IDF first five feature names from built vocabulary are *abl, access, accord, across, act*. Frequencies are represented as 1 x 700 columns as vocabulary is of feature words length. The post multiplication values against feature words are given by table II:

TABLE II
TDF MULTIPLICATION MATRIX

WORDS	TF-IDF
abl	190.8766
access	108.855048
accord	109.660059
across	147.387424

Next, we form a cloud of words using *the wordcloud* library and generate cloud data based on our cleaned trained data word features. The frequently appearing words in tweets are shown more prominently than the low-frequency words as seen in fig. 5 it is helpful to have a quick sight at what tweets look like in terms of words. As done earlier for Count Vectorizer we form two normal distributions of frequently used words 'freq_word' from train data and frequently used unique words 'unique_freq_word' from the former. The plotted frequency distribution graph is shown in fig 4. It can be seen that mean of both frequency distributions of TF-IDF is even closer than that of Count Vectorizer. In unique word frequency, the mean is 10.62 which means that roughly 10 to 11 words of an average tweet are unique. Due to the closure of the tendencies, we assume our graphs to be symmetric which means mean is a good representative of the frequency of occurrence here. Although there is a difference in the parameter setting of both vectorizers, still the frequency distribution graphs tend to be similar. One more thing to be noted here is that the median in the case of TF-IDF is approximately closer to the mean hence, negating skewness and resulting in asymmetric distribution.

As we can see in Count Vectorizer, medians tend to be the same infrequent and unique frequent words graphs but due to

the difference of parameter setting and focusing on Inverse Document Frequencies in TF-IDF median of word frequency is unique word frequency is respectively.

In unique word frequency, the mean is 9.79 which means that roughly 9 to 10 words of an average tweet are unique. Due to the closure of the tendencies, we assume our graphs to be symmetric which means mean is a good representative of the frequency of occurrence here.

The next step is to finally form vectors from bags of words. We use sklearn.feature_extraction.text library whose method 'Count Vectorizer' counts the frequency of words and represent it as an integer. We tune its parameters to achieve optimal accuracy by setting feature words limit up to 700 [11]. When we build a vocabulary, it looks for a specific number of documents where a word has been repeated. We set it to 2 which means a word has to be repeated at least in two tweets for it to be added in Bag of Words. For instance, first five featured words happen to be *abl, access, accord, across, act*.

IV. EVALUATING CLASSIFIERS (COUNT VECTORIZER VS TF - IDF)

We know what the machine needs to be taught, so our job is to create a guideline for teaching or provide the machine to learn from preformatted, applicable, clean tweets. As discussed earlier we experiment with two different architectures of word embedding techniques and produced feature vectors hence we start classification by implementing Support Vector Machine SVM. Sklearn library owns a class SVM which has the method LinearSVC to perform linear support vector classification. For Count Vectorizer, we set Hinge as its standard loss function. By using by default penalty term L2, we set class weight as balanced as it will automatically adjust weights that will be inversely proportional to frequencies against each class from the fed data. We train our data after shuffling on random state = 1800 on 5 k folds and then predict on test data. SVM results in 93.07% accuracy on test data which is a good twitch to experiment with other classifiers and hyperparameters. For TF-IDF, we set Square Hinge to balance its higher and lower frequency inverses. By using by default penalty term L2, we set class weight as balanced as it will automatically adjust weights that will be inversely proportional to frequencies against each class from the fed data. We train our data after shuffling on random state = 1800 on 5 k folds and then predict on test data. SVM results in 93.15% accuracy on test data that is just a minor improvement from Count Vectorizer.

TABLE III
TF-IDF AND COUNT VECTORIZER ACCURACIES

Classifier	Accuracy	
	TF-IDF	Count Vectorizer
Support Vector Machine	93.15	93.07
Bernoulli Naïve Bayes	90.5	90.65
Single Layer Perceptron	48.3	54.1
Multi-Layer Perceptron	92.99	93.73
Logistic Regression	91.84	93.31

TABLE IV
ACCURACIES OF SOME EXISTING MODELS

Authors	Models
(Chakraborty 2020)	fuzzy logic model: 79%
(Mansoor 2020)	Accuracy With ANN: 76% Accuracy With LSTM: 84.5%
(Sethi 2020)	Accuracy rate average SVM is 88-89%
(B. Pang 2002)	with CNN 89%
(al. 2019)	Accuracy LSTM 84.3
(B. Heredia 2016)	Naive Bayes: 75% Support Vector Machine: 78% Multinomial Naive Bayes: 86%

Since we have to classify tweets in negative and positive sentiments it is suitable to use Bernoulli Naïve Bayes classifier as it works well on discrete data. We implement it by using Sklearn library's class *naive_bayes* which has the method BernoulliNB(). After training our data on 0.03 alpha we find its accuracy on the test to be 90.65%. For TF-IDF similar parameters gave 90.50% scored accuracy that doesn't reflect the significant change as well.

Next, we implement Multilayer Perceptron. Starting from the single-layer tuning hidden layers did not improve the accuracy of the sentiment predictions on test data hence, on a single hidden layer, we achieve 54.68% accuracy on Count Vectorizer and 48% on TF-IDF respectively. We tuned different parameter settings to achieve optimal accuracy for MLP and as estimated for TF-IDF, Multilayer perceptron did wonder. On setting its parameters random state as 2018, hidden layers as 5, activation function as RELU, alpha as 0.3 and 1000 iterations, the accuracy achieved is 92.99 which roughly becomes 93.55 % on 9 hidden layers where Count vectorizer holds 93.73% on the similar parameters. We have shown the results in table III below. Finally, we train Logistic Regression as our linear model on 5 folds with penalty term l2 and achieve 93.31% accuracy on test data. We used GridSearchCV [12] to train and test our models using its *fit* and *predict* properties. Using similar parameters for TF-IDF accuracy turns out to be 91.84 % respectively

V. EXPERIMENTAL RESULTS

Out of all four models, SVM performed best by resulting in 93% accuracy on test data. We also use SVM to plot coefficients and analyze what coefficients are significant most. We have seen that a higher frequency of particular words has a significant effect on results. The tenacity of word embedding techniques i.e., Count Vectorizer and TF-IDF is to perform feature extraction by creating word feature vectors. We insert around feature words into an array, sort them index wise and take out few samples from test data to get an overview of words that result in positive or negative sentiments magnitude [13]. We take the thirty smallest and

largest coefficients, extract their coefficients. In fig. 5. Red bars represent the negative scored sentiments and blue bars

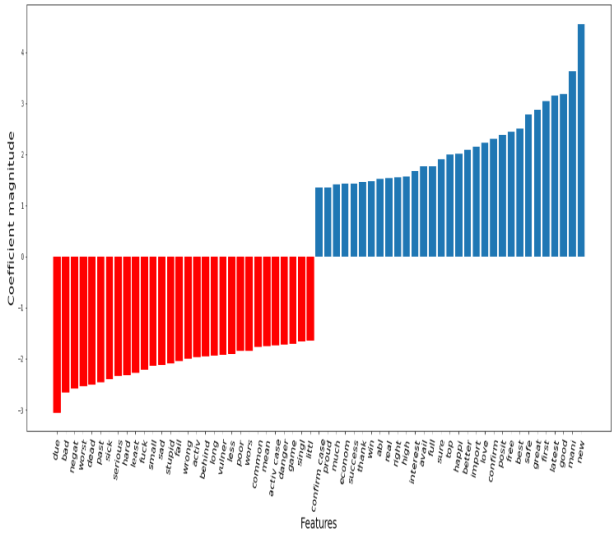

Fig. 5. Most Significant Feature Representation

represent positive ones respectively. We can see that the results are quite native and capable enough to narrate intentions involved in shared tweets. As the highest frequency words are the most relevant words of the tweets, hence, keywords are easier to extract while TF-IDF is implemented. Although Term Frequency – Inverse Document Frequency has a relatively complex method of implementation than Count Vectorizer 3 out 4 classifiers resulted in better accuracy when TF-IDF embedded vectors were fed for training data. TF-IDF has its flaws e.g., Curse of Dimensionality in case of high indexed and proportionally increasing frequencies of features. This happens when we try to classify extremely high dimensional that doesn't occur in low dimension spaces at all. As document frequencies increase proportionally, we must also beware of feature reduction and offsets. Other techniques like Dimensionality Reduction and Clustering can be incorporated in the text analytics phase. One cannot prefer one technique over the other as it is a high dataset and task-dependent hyperparameter. Below are the predicted sentiments against sample tweets taken from predicted outputs table V.

TABLE V
FINAL PREDICTED SENTIMENTS

Tweet	Sentiment
today s covid specifics oregon oregon reports 272 new cases two new deaths covid 19	positive
cool story bro ucla ucregents covid19 onlinelearning hybridclasses wtf paying	negative
unsurprisingly american college football programs appear willing put unpaid student athletes har	negative
let office system enjoy nature covid 19 covid19 coronavirus	positive
remember level quarantine positive covid19 patients back march	positive
don t deceived cases 6 week 4 states hit new record highs 22 states pr red zon	positive

VI. FUTURE WORK

To optimize the curse of dimensionality due to TF-IDF we need to work on dimension reduction mechanisms. We also plan to extend our research to incorporate bifurcation of tweets based on location and country to perform technical analysis on people's seriousness towards COVID-10 before and after lockdown for the year 2020 and onwards. We will work on a combination of deep learning classifiers with LSTM to achieve further accuracy along with exposure to more word bedding and feature extraction techniques. COVID has changed the scope of the overall workforce around the world which has affected economies. Sentiment analysis on pandemic related data can help retrieving useful stats to help produce recommendation systems for betterment.

VII. CONCLUSION

In this article, we applied different classifiers on COVID related tweets to train the Deep Neural network to enhance the accuracy rate on sentiment analysis after implementing word embedding and cleaning techniques using Natural Language Processing. We have made a comparison-based review on our word embedding techniques **Count Vectorizer and TF-IDF**. To make a fair comparison, we use the same datasets and similar classifiers to train and test our embedded data. Further, we compare accuracies and observe that TF-IDF comes out to be more efficient as compared to Count Vectorizer where datasets are of huge volume and in our case i.e., for covid tweets both have been approximately similar in performance. Sentiment Analysis on data related to a global pandemic is of huge importance as we are still living in COVID affected environment and there are lockdowns in many regions of the world. Further, dynamic measures should be taken to provide concentrated comfort to social media users despite negativity being shared every minute.

REFERENCES

[1] K. Chakraborty, "Sentiment Analysis of COVID-19 tweets by Deep Learning Classifiers—A study to show how popularity is affecting accuracy in social media," Elsevier - Applied Soft Computing volume 97, Part A, December 2020, 106754, 2020.

[2] A. Garain, "english-language-tweets-dataset-covid-19," [Online]. Available: https://ieee-dataport.org/open-access/english-language-tweets-dataset-covid-19.

[3] Kruspe, "Cross-language sentiment analysis of European Twitter messages during," German Aerospace Center, 2020.

[4] L. Nemes &. A. Kiss, "Social media sentiment analysis based on COVID-19," Journal of Information and Telecommunication, 2020.

[5] M. Mansoor, "Global Sentiment Analysis Of COVID-19 Tweets Over Time," 2020.

[6] G. Preda, "www.kaggle.com," 2020. [Online]. Available: https://www.kaggle.com/gpreda/covid19-tweets.

[7] S. JAIN, "analyticsvidhya," 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/.

[8] M. Sethi, "Sentiment Identification in COVID-19 Specific Tweets," Ieee- Proceedings of the International Conference on Electronics and Sustainable Communication Systems (ICESC 2020), 2020.

[9] M. Y. Kabir, "CoronaVis: A Real-time COVID-19 Tweets Data Analyzer and Data Repository," arXiv, 2020.

[10] https://towardsdatascience.com/apply-and-lambda-usage-in-pandas-b13a1ea037f7

[11] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html.

[12] "scikit-learn.org," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[13] A. Go, R. Bhayani, and L. Huang. "Twitter sentiment classification using distant supervision." *CS224N project report, Stanford* 1, no. 12 (2009): 2009.