Sentiment Analysis on Large-Scale Covid-19 Tweets using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling

By

Arunava Kumar Chakraborty

UNDER THE GUIDANCE OF

Dr. Anup Kumar Kolya

THESIS REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE POST GRADUATE DEGREE

OF

MASTER OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

RCC INSTITUTE OF INFORMATION TECHNOLOGY

Session: 2019 – 2021

श्रमम बिना न किमपि साध्यम

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
RCC INSTITUTE OF INFORMATION TECHNOLOGY
A Unit of RCC Institute of Technology an autonomous Society of Department of Higher Education, Govt. of West Bengal
[Affiliated to Maulana Abul Kalam Azad University of Technology]
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA -700015

July, 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
RCC INSTITUTE OF INFORMATION TECHNOLOGY
A Unit of RCC Institute of Technology an autonomous Society of Department of Higher Education, Govt. of West Bengal
[Affiliated to Maulana Abul Kalam Azad University of Technology]
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA-700015

TO WHOM IT MAY CONCERN

I hereby recommend that the Thesis entitled "**Sentiment Analysis on Large-Scale Covid-19 Tweets using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling**" prepared under my supervision by **Arunava Kumar Chakraborty** (Reg. No. 026712 of 2019-20 and Roll No. 11711219006) of M.Tech (Final year), may be accepted in partial fulfillment for the post graduate degree of **Master of Technology in Computer Science & Engineering** under Maulana Abul Kalam Azad University of Technology (Formerly known as West Bengal University of Technology).

……………………………………….
Thesis Supervisor

Countersigned:

……………………………………….
Head
Department of Computer Sc. & Engg,
RCC Institute of Information Technology
Kolkata – 700015.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RCC INSTITUTE OF INFORMATION TECHNOLOGY

A Unit of RCC Institute of Technology an autonomous Society of Department of Higher Education, Govt. of West Bengal

[Affiliated to Maulana Abul Kalam Azad University of Technology]

CANAL SOUTH ROAD, BELIAGHATA, KOLKATA-700015

श्रमम बिना न किमपि साध्यम

CERTIFICATE OF APPROVAL

The foregoing Thesis is hereby accepted as a credible study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the thesis only for the purpose for which it is submitted.

FINAL EXAMINATION FOR EVALUATION OF THESIS

1. ____________________________

2. ____________________________

(Signature of Examiners)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
RCC INSTITUTE OF INFORMATION TECHNOLOGY
A Unit of RCC Institute of Technology an autonomous Society of Department of Higher Education, Govt. of West Bengal
[Affiliated to Maulana Abul Kalam Azad University of Technology]
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA-700015

ACKNOWLEDGEMENT

I, Arunava Kumar Chakraborty, take this opportunity to express my profound gratitude and deep regards to my Thesis Supervisor, Dr. Anup Kumar Kolya (Assistant Professor, Department of Computer Science & Engineering, RCC Institute of Information Technology), for his guidance, constant monitoring, and unmatched corroboration throughout the course of my work. Besides responding to all my questions and queries so spontaneously, he also encouraged me to communicate several journals and conference proceedings to publish my work as research paper(s). The blessing, help and guidance given by him time to time shall carry me in my journey to explore the future life, which I am about to embark.

I would also like to express my deepest appreciation and obeisance to Mr. Rajib Saha (Professor and HOD, Department of Computer Science & Engineering, RCC Institute of Information Technology), for his extended support, cooperation and endearment towards me.

In accordance with that, I would most humbly like to express my absolute amenity and reverence to respected Mr. Sourav Das (Assistant Professor, Dept. of Computer Science & Engineering, Future Institute of Technology), for guiding and mentoring me on and off throughout my thesis work. Sir shared some of his immensely valuable time and knowledge with me to influence me more about my research.

Finally, I would like to thank my parents, shri Ashis Kumar Chakraborty and smt. Ashima Chakraborty, for their continuous guidance through all the ups and downs of life.

Ma and Baba, Thank You.

Arunava Kumar Chakraborty
(MCS2019/001)

"It is possible to invent a single machine which can be used to compute any computable sequence."

- Alan Turing

Contents

Abstract

In this advanced period of sophisticated technology, most of the people in the world have been able to connect with each other through various social media platforms. The progressively expanding development of instructive sources on the web is continually keeping us up-to-date on what's going on around the world. Today on social media, almost everyone is seen expressing their point of views on various issues starting from commercial items, movies, sports, social and international issues, government strategies to even on global pandemic. The Natural Language Processing is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data. Being one of the most important fields of NLP, Sentiment Analysis is the process of unearthing or mining meaningful patterns from text data. Sentiment analysis can help us attain the attitude and mood of the wider public which can then help us gather insightful information about the context. This can then help us to predict and make accurate calculated decisions that are based on large sample sets. Sentiment analysis may be performed on social media data to extract human sensations and determine overall opinion on any trending topic.

Based on social connections and interactions, there are lots of online platforms and among them Facebook and Twitter are the two leading platforms. These social platforms regularly witness a lot of opinion waves regarding some of the most trending and discussed topics and nowadays one of them clearly is the Novel Coronavirus Pandemic. Natural Language Processing research domain has seen an enormous surge in large-scale event analysis since the last year, due to the rise of Covid-19 or Coronavirus situation around the world. WHO declared it as a pandemic on 11th March 2020[1]. Quite naturally, an event of such a high impact has attracted a majority of the researchers to cultivate the experimentations, and report any breakthrough on the same, as we continue to explore the mass psychological states and emotional viewpoints about it from the social networks and micro-blogs.

With this work, we present an approach to descriptively study and summarize several recent and high-quality research works regarding sentiment analysis, knowledge discovery, predictive analysis, topic classification, learning models, etc. We compare and contrast the methodology of these works, the data and models that were worked with, and the respective contributions. The proposed deep learning-based hybrid models predict the sentiment of collected large-scale tweets on Coronavirus or Covid-19. Through this work, we define a novel direction on developing corpora with Covid-19 tweets in two different phases. We also introduce two distinct features, namely word trend detection using N-gram model, and relevant word

[1] https://www.who.int/westernpacific/emergencies/covid-19/news-covid-19

identification. To analyze the public sentiment trend on the topics related to Covid-19 we use Naïve Bayes classifier followed by the evolutionary sentiment classification. Then we assign the refined ratings on collected tweets based on their sentiment class. Furthermore, we train both Convo-Sequential and Convo-Bidirectional long-short term network using two types of rated tweets to predict sentiment on Covid-19 data. Finally, our proposed Convo-Sequential and Convo-Bidirectional LSTM models achieve 95.61% and 95.81% of validation accuracy respectively for first phase dataset whereas on the second phase dataset the models obtained the validation accuracy as 95.53% and 95.75% respectively. To ensure the consistency of these models, we train these models with different public corpora to compare the results with previous benchmark experiments based on their prediction accuracy. With this work, we aim to bridge our contribution with that of some concerning works to gain a better sentiment scenario from the web. The organization of this thesis paper is as follows:

- In **Chapter One**, the introduction is presented along with the research motivation and objectives, which elaborates the facts that why we felt to work on this specific topic, and also the focus of this thesis where we have thoroughly concentrated to mine up the details and showcase through this work.

- In **Chapter Two**, a detailed literature review is presented, which not only shows the significance and course of such experiments over the past decade but also their close relevance with our work justifies the approach and perspective towards this project.

- **Chapter Three** depicts the application and scope of our work. This chapter elaborates the area that we tried to cover with our work, the problems that we encountered and addressed.

- In **Chapter Four**, the design methodology of the proposed system is elaborated. We have shown the workflow in the light of an ideal project management perspective along with a segmental breakdown for each module in this chapter.

- In **Chapter Five**, the data corpus collection, build and deployment techniques are discussed.

- In **Chapter Six**, we have discussed the implementation details of the project. With that, we also make a detailed results analysis obtained from the implications. Not only that, but we also made an organized error analysis of each segment of the project modules to show what could have made the analysis and results better, and what are the shortcomings.

- **Chapter Seven** concludes the comparative performance analysis with the previous NLP-based experiments on large-scale Covid-19 data as well as on different public corpora.

- In **Chapter Eight**, we concluded the findings of the work, and also described our future plans for an extension of this work.

- **Chapter Nine** mentions all the references and bibliography, which has been accounted as citations in the course of this thesis work.

- ***Appendix I*** refers to a brief discussion about our published work, i.e. Sentiment Analysis of Covid-19 Tweets using Evolutionary Classification-Based LSTM Model, under the supervision of my respected thesis guide during the course.

- ***Appendix II*** contains all the Application, Package and Framework details with respective versions along with the significant Source Codes of the main thesis work.

List of Publications

List of Publication on Thesis Topic:

1. **Chakraborty, A.K.**, Das, S. and Kolya, A.K., 2021. Sentiment Analysis of Covid-19 Tweets Using Evolutionary Classification-Based LSTM Model. In *Proceedings of Research and Applications in Artificial Intelligence* (pp. 75-86). Springer, Singapore. https://doi.org/10.1007/978-981-16-1543-6_7

    - This paper has received "**Best Paper**" Award for the respective track in the International Conference on Research and Applications in Artificial Intelligence (RAAI 2020).

List of Communicated Book Chapter on Thesis Topic:

1. **Arunava Kumar Chakraborty**, Sourav Das, "A Comparative Study of Natural Language Researches and Novel Approach with Baseline Attributes Leading to Sentiment Analysis of Web-Based Covid-19 Corpora", In Computational Intelligence Applications for Sentiment Data Analysis (CIASDA), To be Published in Book Series Hybrid Computational Intelligence for Pattern Analysis and Understanding, ELSEVIER [Submitted].

List of Figures

List of Tables

Chapter 1:

Introduction

The Covid-19 outbreak was first reported in the Wuhan, Hubei Province, China on 31st December, 2019 when it was spreading rapidly all over the world. On 11th March, 2020, WHO finally announced Covid-19 outbreak as pandemic observing its continuous intensity to spread [1]. Starting from China, this virus infected and killed thousands of people from Italy, Spain, USA, UK, Brazil, Russia and other many more countries as well. On 21st August 2020, more than 22.5 million cases of Covid-19 were reported in more than 188 countries and territories, yielding more than 7,92,000 deaths; although 14.4 million people have reported to be recovered[2]. While this pandemic has continued to affect the lives of millions, many countries had enforced a strict lockdown for different periods to break the chain of this pandemic [1]. Since the Covid-19 vaccines are still yet to be discovered, therefore maintaining social distancing is the one and only one solution to check the spreading rate of SARS-CoV-2, the virus that causes COVID-19 [2]. To share the regular information about this pandemic all over the world, the social media platforms perform a leading role throughout the entire lockdown session. During the lockdown period a lot of people have chosen the Twitter to share their expression about this disease so we have been inspired to measure the human sensations about this epidemic by analyzing this huge Twitter data [3].

As most of the peoples of different countries have used their native languages rather than English to express their feelings on social media, initially we have to face many challenges at the time of streaming the English tweets from the multilingual tweets all over the world [3]. However, we collected English tweets on Covid-19 in two phases over a period of seven months for April - June, 2020 and August - October, 2020 respectively. We exclusively developed the first and second phase datasets with almost 235k and 320k Covid-19 tweets. The trend of tweets was analyzed using n-gram model so that we can extract the most relevant grams for further computation. We identified the most popular Covid-19 related words from the word corpus using Bag-of-Words model. The sentiment polarities of the pre-processed tweets were found to classify them into *positive*, *negative* and *neutral* classes. For fine-grained classification of these datasets, we used Naïve Bayes classifier for assigning the refined sentiment scores to the classified tweets based on their sentiment polarity. Now, instead of using conventional Machine Learning algorithms, we introduced Hybrid Deep Learning model, since the deep hybrid neural networks are very beneficial in text generation, vector representation, word representation estimation, sentence classification, sentence modeling and feature presentation. Finally, both Hybrid Convolutional LSTM networks were trained using the vectorized tweets along with their updated sentiment scores to achieve certain accuracy on sentiment prediction.

[2] https://gisanddata.maps.arcgis.com

1.1 Motivation & Objective

The motivation behind this work is to study the recent flow of works on Covid-19 from a dedicated natural language research perspective and also to analyze the behavioral direction and nature of people throughout the world about such a pandemic. As a social media platform, Twitter is one of the trendiest microblogging sites and can be considered as a repository of meaningful information [41]. In 2020, out of 320 million active Twitter users from all over the world, India has 18.9 million active Twitter users i.e., currently the third highest in the world[3]. The users tweet their expressions within allotted 280 characters. During the lockdown phase due to this pandemic situation since many people have shared their feelings about this epidemic on Twitter, we have been inspired to analyze the emotional tendencies of the public from time to time.

This particular epidemic has not only claimed millions of lives but has plunged the entire world economy into recession. Many people have lost their jobs in these difficult times and the unemployment rate has been rising day by day all over the world. The main objective of this research work is to analyze the public sentiment for finding the global sentiment trend about this pandemic and finally train both Convo-Sequential and Convo-Bidirectional LSTM models based on the Naïve Bayes sentiment classification to predict the sentiment by analyzing correlations between tweet words.

1.2 Contribution

The novel contributions of this research are:

- Lexical N-gram Model for extraction of most popular unigrams, bigrams and trigrams to find the trend of tweet words.
- Covid-19 Specific Word Identification based on the Bag-of-Words Model to find the impact of those words over the tweets.
- Sentiment Analysis of the unlabeled tweets to evaluate the polarity ratings of each of the tweets.
- Naïve Bayes sentiment model for evaluation of refined sentiment ratings for each tweet based on the previous extracted features.
- Fine-grained Classification of tweets based on their sentiment polarity ratings using Naïve Bayes Model for finding the global sentiment trend.
- Sentiment Modelling using Hybrid Convolutional LSTM based on the classified tweets and their refined sentiment ratings for sentiment prediction.

[3] www.statista.com

Chapter 2:

Literature Review

The sentiment analysis is a process of computationally analyzing and identifying opinions and judgments from a piece of text. Although the sentiment analysis can be implemented for different purposes like – Aspect-based, Document-level, and Sentence-level sentiment analysis with all possible approach methods such as Lexicon-based, Syntactical and Semantic approach with various learning algorithms that are available. The literacy survey regarding sentiment analysis on social media data demonstrates that a decent amount of research has been carried out by different researchers in all three levels. Numerous research works on this domain concentrated on recognizing the polarity of a sentence (e.g. positive, neutral, negative), according to semantic information learned from the textual content of sentences.

With this dissertation, we present a concise and chronological description of the research works carried out on this domain till now. However, the entire literature survey has divided into four sections. The first section represents some previous NLP based research works on sentiment analysis. The second section contains some research works on sentiment analysis from Web and Social Media. In third section, we demonstrated some research works regarding sentiment analysis in Healthcare industry. Finally, we presented some recent research works on Covid-19 Sentiment & Trend Analysis.

2.1 Groundwork of Sentiment Analysis

In the year 2015, some researchers found that the Document-level sentiment analysis can be improved by using Rhetorical Structure Theory which describes the structure of a document in terms of text spans that form discourse units and the relations between them. From this research, it is evident that reweighting the discourse units based on their position in a dependency representation can improve lexical-based sentiment analysis. Finally, they proposed a recurrent neural network over their evaluated data to further improvements in sentiment classification [4].

A research work demonstrates the machine translation of multilingual sentence-level sentiment analysis. In their experiment, they evaluated the sentiment prediction accuracy of 21 English sentiment-level methods on different datasets in 9 different languages. As the result, they found that SO-CAL, Umigon, and Vader methods have performed well on different datasets based on Macro-F1 score and Coverage [5].

A group of researchers developed a three-stage statistical hybrid model to detect explicit and implicit polarity shifts from the reviews. After elimination of the identified polarity shifts using the antonym reversion method, the piece of text divided into four subsets as polarity-unshifted text, eliminated negations, explicit contrasts, and sentiment inconsistency. Finally, they trained different base classifiers on these textual subsets and computed a weighted ensemble as the final layer of sentiment classifier to decrease the polarity shifts [7].

Another research work presents an aspect-based sentiment analysis on the different product reviews. After the pre-processing of data using part-of-speech (POS) tagging the researchers used Chi-Square probability distribution to extract highly relevant words from each opinion. At last, they have used Naïve Bayes classifier for sentiment classification of the polarity ratings and as the result, their model achieved 78.21% of the highest F1-score [8].

The comparative study on document-based sentiment analysis using the clustering method presents the experimental result of several clustering techniques. In this study, researchers found that considering the clustering accuracy K-means algorithms performed better on balanced datasets compare to unbalanced datasets. They also found some clustering models like – Agglo-WSlink, Slink, UPGMA, Spect-Sy, Spect-RW, PCA-Kmeans, and Spect-Un achieved better accuracy on unbalanced datasets rather than balanced datasets. The BM25, DPH_DFR, and H_LM model performed better than traditional weighting models like – Binary, TF, and TF_IDF [9].

For polarity classification, the sentiment analysis using radical-level and character-level processing achieved a better result. Research work on Chinese sentiment analysis claims that using Chinese radical-based hierarchical embeddings their system achieved better accuracy over state-of-art textual and embedding features [10].

Few researchers proposed a deep Sentic-LSTM model for targeted aspect-based sentiment analysis to incorporate implicit and explicit pieces of knowledge. The two-step attention model based on the sentiment aspects and polarity sequentially encodes the target expression. The target-level attention model achieved more accurate target representation as it attends only the sentimental part of the sentence [11].

Another research work presents the extraction of sentence-level features (SLF) and domain-sensitive features (DSF) from textual data. The researchers trained different machine learning algorithms based on extracted features. They also compared the feature selection method WEKA with baseline features. Finally, the supervised ML algorithm improves 7.1%, 7.2%, and 7.4% performance for precision, recall, and F1-score with the combined SLF-DSF features [12].

2.2 Sentiment Analysis from Web and Social Media

In traditional media different form of text is available from which microblogging texts are noisy, short, and embedded with social relations. Some researchers proposed a novel sociological approach (SANT) to handle networked texts in microblogging. They extracted sentiment relations between tweets based on social theories, and model the relations using Laplacian graph, which is employed as a regularization to a sparse formulation to facilitate sentiment classification and effectively handle noisy Twitter data. They developed an optimization algorithm for SANT to achieve consistent performance for different sizes of training data, a useful feature for sentiment classification [13].

Semi-automated sentiment analysis was developed based on the online social network using the probability model. The authors proposed a model that reads sample text messages in a train set and builds a sentiment lexicon that contains the list of words that appeared in the text messages and the probability that a text message is a positive opinion if it includes those words. Then, it computes the positivity score of text messages in a test set using the list of words in a message and sentiment lexicon. Each message is categorized as either positive or negative, depending on the threshold value calculated using a train set [14].

Some researchers demonstrated a system for a possible combined approach between Social Network Analysis and Sentiment Analysis, which can operate on Twitter data. They collected three types of data and used a classifier based on the Multinomial Naive Bayes algorithm to identify the tweets from different sentiment classes. They experimented with their approach on a couple of Twitter channels like the SamSmith channel during the Grammy Awards in 2015, and the #Ukraine channel during the crisis of 2014. They also mentioned a methodology and some guidelines for the automatic classification of Twitter content [15].

In 2016, few researchers found that using Bidirectional LSTM with two-dimensional max-pooling improves text classification. They introduced two combinational models one is BLSTM-2D Pooling and the other is BLSTM-2DCNN, which can be seen as an extension of BLSTM-2D Pooling. They achieved the best accuracy as 52.6% with 2D filter size (5,5) and 2D max-pooling size (5,5) [16].

Another research work describes the use of a multimodal feature learning approach, using neural network models such as Skip-gram and Denoising Autoencoders, to address sentiment analysis of micro-blogging content like tweets, that is composed by a short text and, possibly, an image. The authors developed a semi-supervised model CBOW-LR for learning concurrently vector representation. By using a sentiment polarity classifier, the model achieved better accuracy over CBOW representation on the same quantity of tweets. For learning text and image representation their another unified model (CBOW-DA-LR) works in an unsupervised and semi-supervised manner, obtained a higher classification accuracy compared to SentiBank, a state-of-the-art approach on a publicly available Twitter dataset [17].

Some researchers described a deep LSTM architecture for Message-level and Topic-based sentiment analysis. The authors used LSTM networks augmented with two kinds of attention mechanisms, on top of word embeddings pre-trained on a big collection of Twitter messages [18].

In 2017, two researchers proposed a new method for learning to overcome language variation, leveraging the tendency of socially proximate individuals to use language similarly. By learning basis models that focus on different local regions of the social network, their method was able to capture subtle shifts in meaning across the network. They have formulated this model by employing a social attention mechanism to predict the weighted combination of the outputs of the basic models where each author has a unique weighting, depending on their position in the social network. Their model significantly improves the accuracies of sentiment analysis over the standard CNN model on Twitter and on review data [19].

A group of researchers developed LSTM hyperparameter optimization for neural network-based Emotion Recognition framework. In their experiment, they found that optimizing LSTM hyperparameters significantly improves the recognition rate of four-quadrant dimensional emotions with a 14% increase in accuracy, and the model based on an optimized LSTM classifier achieved 77.68% accuracy by using the Differential Evolution algorithm [20].

To analyze the problem of user behavior modeling by utilizing multiple types of user-generated data a generative model HUB was developed to integrate two companion learning tasks of opinionated content modeling and social network structure modeling so that the learning tasks are paired and clustered to reflect the homogeneity among users while each user is modeled as a mixture over the instances of paired tasks. The learned user behavior models are interpretable and predictive to achieve more accuracy in sentiment classification and item/friend recommendations than the corresponding baselines on two large collections of review datasets from Amazon and Yelp with the social network structures [21].

Topic modeling was introduced to automatically extract the different subjects of discussion on climate change and sentiment analysis was used to classify tweets as positive, neutral, or negative. The pooled Latent Dirichlet Allocation (LDA) model was used by the researchers to train a topic model by looking at the word co-occurrences within collected tweets. Finally, their proposed pooled LDA model outperforms both LDA (unpooled) and Biterm Topic Model (BTM) based on average UMass coherence score [22].

Another research work describes the comparative study on different deep learning models for sentiment analysis. During the survey, the authors found that it is better to combine deep CNN with word embedding than with TF-IDF when performing a sentiment analysis [23].

There have different research opportunities of sentiment analysis on public views from social media. A research paper demonstrates some sentiment analysis techniques based on deep learning and word embedding methods from different state-of-the-art studies. The researchers tried to explore some critical research gaps by analyzing the past researches on the election result prediction in different countries and states respectively [24].

2.3 Sentiment Analysis in Healthcare

Sentiment Analysis for textual classification can be used to measure the public concern about a disease outbreak. The authors present an Epidemic Sentiment Monitoring System (ESMOS) to identify the progression and peaks of concern for disease. In their experiments, Multinomial Naïve Bayes achieved overall the best results on classifying negative sentiment tweets versus neutral tweets and took significantly less time to build the classifier than other methods [25].

In 2014, few researchers from Johns Hopkins University analyzed the mental health phenomena in publicly available Twitter data by using natural language processing. After gathering data for mental illnesses, they analyzed the data for finding symptoms of post-traumatic stress disorder (PTSD), depression, bipolar disorder, and seasonal affective disorder (SAD). They provide some insight about classified the quantifiable linguistic information by examining correlations between the various analytics [26].

There are some commercial and non-commercial tools for analyzing the sentiment polarities (positive, negative, or neutral) on textual data. Some popular commercial tools are Semantria and TheySay and the non-commercial tools are WEKA and Google Prediction API. Another research work demonstrates the difficulty of analyzing sentiment by using different tools in the healthcare domain and the comparative study of the ability of different tools over a large dataset. As the result of their experiment, they found that WEKA, performed best on their data [27].

Another research work describes the sentiment tracking methodology to detect the evidence of depression from the tweets. Then the proposed system further checks that whether the detected tweets contained a depressive symptom or not. The sentiment classifier classified the tweets having depressive symptoms into three sub-classes: depressed mood, disturbed sleep, or fatigue or loss of energy. Researchers found that different machine learning approaches improved the precision of simple keywords for precisely detecting depressive symptoms and all their subtypes from the tweets [28].

In 2017, based on hybrid matrix factorization methods, a group of researchers proposed a healthcare recommendation system named as iDoctor. They used the LDA model for topic modeling to extract the user's latent priority and doctor features can be extracted from user review comments on doctors, which are involved in matrix factorization integrated with two feature distributions for providing more accurate counseling [29].

A psychometric analyzer can be used for sentiment analysis & emotion recognition of a patient's health based on their previous medical history and recorded voice. The proposed method helps to analyze the intensity, emotion, polarity, and subjectivity of the tweets. The researchers implemented the Support Vector Machine (SVM) and Support Vector Regression (SVR) for sentiment analysis and Decision Tree for detecting the emotions [30].

A group of researchers proposed a methodology for finding the trends of health information from social media to examine the public's opinions about health technology and identify their needs. They used SentiWordNet for the sentiment classification of the collected tweets based on the most frequent words from the corpus [31].

Sentiment analysis can be used in healthcare as their large-scale information about healthcare is available online. To increase healthcare quality, this provides many benefits such as using medical information to achieve the best result. In a comparative study on different research works regarding sentiment analysis in healthcare the authors found that SVM with Naïve Bayes classifier outperforms other similar supervised or corpus-based approaches [32].

2.4 Sentiment Analysis on Covid-19

A Covid-19 trend prediction model was introduced for predicting the number of Covid-19 positive cases in different states of India. The authors mainly focused on LSTM based prediction model as the LSTM model performs better for time-series predictions. They tested different LSTM variants such as stacked, convolutional, and Bidirectional LSTM on the historical data, and based on the absolute error they found that Bi-LSTM gives more accurate results over other LSTM models for short-term prediction [33].

The evolutionary K-means clustering on Twitter data related to Covid-19 has been done by some researchers. They analyzed the tweet patterns using the n-gram model. As the result, they observed the difference between the occurrences of n-grams from the dataset [34].

A deep CNN-based LSTM model was developed by some researchers for sentiment accuracy evaluation on large-scale tweets on novel Coronavirus. Then using Logistic regression, they build a data model for visualization and finally achieved 90.67% of overall accuracy. They performed Bayesian regression on the evaluated polarity tweets for constant point-to-point predictability based on the prior and posterior conditional distribution on a common plane [35].

Some researchers developed an interactive web application for real-time tweets tracking on Covid-19 to analyze public behaviors during the pandemic situation. The popular words, as well as the bigrams, were extracted from the tweets. They extracted trending topics using the LDA model. Finally, they used the Sentiment Intensity Analyzer from the NLTK library for finding the sentiment polarities of each tweet [36].

A group of researchers demonstrated the comparative performance analysis of four different machine learning classifiers on Coronavirus Tweets data. They observed that the Naïve Bayes classifier achieved 91% classification accuracy for short Tweets whether using the Logistic Regression classifier provides classification accuracy of 74% on the same number of tweets [37].

Another research work describes the application of NLP on sentiment evaluation using the deep learning model. The researchers used the LDA model to detect the topics on Coronavirus-related issues from online healthcare forums. During the experiment, their proposed deep LSTM model achieved 81.15% of classification accuracy on Covid-19 comments for classifying them into positive, negative, and neutral classes [38].

In 2020, two researchers deployed a system for forecasting the trend of the Covid-19 pandemic in Canada using the Long short-term memory model. Using the public datasets provided by John Hopkins University and the Canadian health authority they trained the deep LSTM network and obtained 93.40% & 92.67% accuracy for short-term & long-term predictions respectively [39].

A hybrid artificial-intelligence (AI) model was proposed by few researchers for predicting the worldwide Covid-19 trend. RoBERTa, the pre-trained model of the BERT model is used for textual feature selection. They developed an improved susceptible–infected (ISI) model with multiple parameters to identify the different infection rates for analyzing the transmission pattern of Coronavirus. The NLP module and the LSTM network are embedded into the ISI model to build the hybrid AI model so that the prediction results of the model will be highly consistent with actual epidemic cases [40].

2.4.1 Comparisons of Methodology on Recent Covid-19 based NLP Research

The Covid-19 pandemic has not only claimed the lives of millions of people but also caused the global market to face a global recession today. There has been a lot of research on Covid-19 in various fields as well as in the NLP domain over the last year and a half. In the literature survey, we found several approaches and techniques to analyze the global sentiment trend of different events. In this section, we present a comparative survey on recent Covid-19 based research experiments. Table 1 represents the proposed contributions of those research works concerning their methodologies.

Table 1. Comparative experimental study on Covid-19 based NLP research.

Year	Paper Title	Model	Proposed Contribution
2020	Prediction and analysis of COVID-19 positive cases using deep learning models: A descriptive case study of India [33].	RNN based Long Short-Term Network • Deep LSTM • Convolutional LSTM • Bidirectional LSTM	• Dataset- Covid-19 Positive case from 32 states of India. • Covid-19 Trend prediction. • Bi-LSTM model achieved better prediction accuracy.
2020	Analysis of Twitter Data Using Evolutionary Clustering during the COVID-19 Pandemic [34].	K-means clustering based on n-gram Model	• Dataset- 43 million collected tweets. • Analyze tweet patterns using clustering on frequently occurred grams.
2021	Predicting the pandemic: sentiment evaluation and predictive analysis from large-scale tweets on Covid-19 by deep convolutional neural Network [35].	• Convolutional Neural Network with *GloVe* Embedding • Bayesian Regression	• Dataset- 600k collected tweets. • Gradient scale comparison contrasting between the major affected countries. • Sentiment evaluation using deep CNN model with *GloVe* embedding layer. • Predictive analysis of Covid-19 Trend using Bayesian Regression.

Year	Paper Title	Model	Proposed Contribution
2020	CoronaVis: A Real-time COVID-19 Tweets Data Analyzer and Data Repository [36].	Latent Dirichlet Allocation (LDA)	• Dataset- CoronaVis Twitter dataset. • Extraction of frequent words and bigrams. • Topic modeling based on extracted features using the LDA model. • Sentiment analysis using NLTK based sentiment analyzer.
2020	COVID-19 Public Sentiment Insights and Machine Learning for Tweets Classification [37].	• Naïve Bayes Classifier • Logistic Regression	• Dataset- 900k collected tweets. • Comparisons of different textual classification for sentiment prediction. • Naïve Bayes classifier outperforms the Logistic Regression model based on classification accuracy.
2020	Deep Sentiment Classification and Topic Discovery on Novel Coronavirus or COVID-19 Online Discussions: NLP Using LSTM Recurrent Neural Network Approach [38].	• Latent Dirichlet Allocation (LDA) • Deep LSTM Model	• Dataset- 563k collected comments from Reddit. • LDA for topic modeling and Gibbs sampling • Sentiment prediction using deep LSTM model.
2020	Time series forecasting of COVID-19 transmission in Canada using LSTM networks [39].	Deep LSTM Network	• Dataset- Covid-19 dataset from John Hopkins University. • Covid-19 trend prediction in Canada.
2020	Predicting COVID-19 in China Using Hybrid AI Model [40].	Hybrid AI Model • Improved Susceptible-Infected (ISI) Model • LSTM Network	• Dataset- News data on Covid-19 from different provinces in China. • RoBERTa, pre-trained BERT model for textual feature selection. • ISI model detects the infection rate. • LSTM is used to update the weights for infection rate.

Year	Paper Title	Model	Proposed Contribution
2020	Sentiment Analysis of COVID-19 tweets by Deep Learning Classifiers—A study to show how popularity is affecting accuracy in social media [41].	Different Deep Learning Classifiers with - • Bag-of-Words Model • Doc2Vec Model • Gaussian Fuzzy logic	• Dataset- DATA_SET 1 & DATA_SET 2 consist of 227k & 315 million collected tweets respectively. • Sentiment identification using Logistic Regression with trigrams under the Tf-Idf Vectorizer outperforms other classifiers on DATA_SET 1. • Sentiment prediction using SVM with Gaussian membership function based Fuzzy logic on DATA_SET 2.
2020	A hybrid deep learning and NLP based system to predict the spread of Covid-19 and unexpected side effects on people [42].	Deep RNN Model	• Dataset- COVID-19 Open Research Dataset Challenge (CORD-19). • Sentient classification on real-time data with several grammatical errors. • THE deep RNN model achieved better accuracy than the CNN model.
2020	COVID-19 open source datasets: a comprehensive survey [43].	NA	• Dataset- Open-source datasets on Covid-19 based on medical images, textual data, and speech data. • Comparative study on several datasets to identify features from them.
2020	Design and analysis of a large-scale COVID-19 tweets dataset [44].	Deep LSTM Network	• Datasets- COV19Tweets Dataset (Lamsal 2020a) consists of 310 million tweets & GeoCOV19Tweets Dataset (Lamsal 2020b). • Topic modeling using frequent grams • Sentiment analysis on tweets.
2020	NLP-based Feature Extraction for the Detection of COVID-19 Misinformation Videos on YouTube [45].	Bayesian Regression with Tf-Idf features	• Dataset- 32k misinformative & 119k factual comments collected from 113 and 67 YouTube videos. • The proposed model achieved better accuracy on sentiment classification instead of other ML models.

Year	Paper Title	Model	Proposed Contribution
2020	Senwave: Monitoring the global sentiments under the covid-19 pandemic [46].	Pre-Trained Models for different language • XLNet for English • AraBert for Arabic • ERNIE for Chinees	• Dataset- 105 million collected multilingual tweets and Chinese Weibo messages. • Fine-grained sentiment classification. • ERNIE model outperforms other models and performs better on Chinees tweets.
2020	What are We Depressed about When We Talk about COVID19: Mental Health Analysis on Tweets Using Natural Language Processing [47].	Multilingual BERT Model	• Dataset- EmoCT (Emotion-COVID19-Tweet) dataset consists of 8.1 million collected tweets. • Pretrained model used for single-level and multi-level classification and achieved significant accuracy.

Chapter 3:

Scope of Research Work

Sentiment Analysis is a procedure used to determine if a chunk of text is positive, negative or neutral. In text analytics, natural language processing (NLP) and machine learning (ML) techniques are combined to assign sentiment scores to the topics, categories or entities within a phrase.

3.1 Sentiment Analysis using NLP & Machine Learning

Natural Language Processing (NLP) is the area of machine learning that focuses on the generation and understanding of language. Its main objective is to enable machines to understand, communicate and interact with humans in a natural way. NLP has many tasks such as Text Generation, Text Classification, Machine Translation, Speech Recognition, Sentiment Analysis, etc. For a beginner to NLP, looking at these tasks and all the techniques involved in handling such tasks can be quite daunting. And in fact, it is very difficult for a newbie to know exactly where and how to start.

The main role of Machine Learning techniques in sentiment analysis is to automate the text analytics functions that sentiment analysis relies on segmentation, POS-tagging, entity extraction and many more. For example, when data scientists train a Machine Learning model by feeding it with a great number of text documents containing pre-tagged examples, it will automatically detect sentiment analysis in future documents. This is possible due to the supervised and unsupervised machine learning techniques, such as neural networks and deep learning. Machine learning also helps data analysts solve context-dependent problems caused by the evolution of natural language. For example, the adjective 'burned-out' may bear different meanings. However, considering training methods as feeding machine learning models with thousand pre-tagged examples, the ML system can learn to understand what 'burned-out' means in the context of fire, versus in the context of work life.

The proposed hybrid sentiment analysis system combines machine learning with natural language processing techniques to reach higher accuracy. At this point, it is important to make a difference between natural language processing and machine learning. On the one hand, an NLP-based sentiment analysis becomes an effective tool to build a foundation for POS-tagging and sentiment analysis. On the other hand, machine learning techniques can help solve complex natural language processing tasks, such as understanding double-meanings through automated training.

A combination of ML and NLP techniques will, therefore, cover the entire text analytics procedure for sentiment analysis, from low-level segmentation and syntax analysis up to semantic differentiation depending on the context in which a word appears.

3.1.1 Major Challenges in Sentiment Analysis

Language is the most wonderful, dynamic and mysterious phenomenon in the universe. Language and its structure are the primary challenge. There are several open-ended issues and challenges in the field of sentiment analysis. Few of them are:

- ***Word Sense Disambiguation:*** In Word Sense Disambiguation (WSD), correct meaning of word based on the context needs to be extracted as word can have different meanings for different domains. For example, "*small size*" can be positive opinion for mobile phones but negative for hotels.

- ***Comparisons:*** To determine the polarity for comparative sentences can be a challenge. For example, "*Battery life of phone X is better than phone Y*". This review has positive word "*better*" but the author's preferred object is not easy to determine which is the key piece of information in a comparative review.

- ***Negations:*** Negations if not handled properly can give completely wrong results. For example, "*There is a good chance that this phone will not break easily*". This review shows positive polarity but presence of negation changes the effect completely.

- ***Intensity:*** Depending upon the intensity of opinion (mild or strong), to obtain result as highly positive or highly negative can also be challenging. It is called as degree of polarity.

- ***Sarcasm:*** Another interesting challenge can be of identification of sarcasm and to analyze emotions expressed in text at a more fine-grained level.

3.2 Different Levels of Sentiment Analysis

Sentiment analysis can occur at different levels: document level, sentence level or aspect/feature level. Three different levels on which sentiment analysis can be performed depending upon the granularities required are:

3.2.1 Document level Sentiment Analysis

This is the simplest form of classification. The whole document of opinionated text is considered as basic unit of information. It is assumed that document is having opinion about single object only (film, book or hotel). This approach is not suitable if document contains opinions about different objects as in forums and blogs. Classification for full document is done as positive or negative. Irrelevant sentences need to be eliminated before processing. There are two approaches to do classification.

- ***Supervised ML Approach:*** In supervised machine learning approach there is finite set of classes for classification. Training dataset is also available. Given the training data, the system classifies the document by using one of the common classification algorithms such as Support Vector Machine, Naïve Bayes, K Nearest Neighbours and Maximum Entropy etc.

- ***Unsupervised ML Approach:*** In unsupervised approach, Sentiment Orientation (SO) of opinion words in document is determined. If the SO of these words is positive then the document is classified as positive otherwise negative.

3.2.2 Sentence level Sentiment Analysis

Sentence level sentiment analysis is the most fine-grained analysis of the document. In this, polarity is calculated for each sentence as each sentence is considered as separate unit and each sentence can have different opinion. Sentence level sentiment analysis has two tasks:

- ***Subjectivity Classification:*** A sentence can be either subjective sentence or objective sentence. Objective sentence contains the facts. It has no judgement or opinion about the object or entity while subjective sentence has opinions.

- ***Sentiment Classification:*** Sentence can be classified as positive, negative or neutral depending upon the opinion words present in it.

3.2.3 Aspect-based or Feature level Sentiment Analysis

The Aspect-based sentiment analysis is a text analysis technique that categorizes data by aspect and identifies the sentiment attributed to each one. Feature engineering is an extremely basic and essential task for Sentiment Analysis. The basic step in Feature Level sentiment analysis is to identify the piece of text as a feature of some product. The basic steps for feature-based sentiment analysis are:

- Preparing Tweet Database
- POS Tagging
- Feature Extraction
- Opinion Word Extraction
- Opinion Word Polarity Identification
- Opinion Sentence Polarity Identification
- Summary Generation

3.3 Relevance of this Research Work

Through the detailed literature survey, we have found limited number of publications regarding the sentiment analysis on such a large-scale Covid-19 textual data collected from any social media. In this research work we present a novel approach of preparing the Covid-19 datasets consisting large-scale tweets. After pre-processing of tweets, the popular N-grams and Covid-19 exclusive words have been extracted from the tweets during April - June and August - October, 2020. As initially the collected raw tweets from twitter had no sentiment labels, we need to analyze the sentiment ratings of the tweets to understand the overall sentiment distribution of the tweets over both of the datasets. In next phase the Naïve Bayes classifier was used to classify the sentiment rated tweets on the basis of the extracted features during the experiment. After the classification of the tweets into *positive* and *negative* sentiment classes the classified tweets are used to train the proposed hybrid deep neural networks for sentiment prediction.

3.4 Challenges Faced

Initially, the main focus of this research work is collecting large-scale global tweets on Covid-19 or Coronavirus pandemic for deep learning influenced sentiment analysis. As many Twitter users from different foreign countries used their native language for their tweets, our first challenge was to gather only English tweets for developing the datasets. With such vision we started streaming tweets exclusively related to this pandemic and collected almost 235k tweets during April - June, 2020 for the first phase dataset (Covid-19 Dataset I). A relatively small problem that we have faced while collecting the tweets was (and many of the other researchers often face this problem too), Twitter API mostly allows users to live-stream only 1-2% of the total tweets on any keywords, we observed that we were able to collect tweets at a rate of almost 10 thousand tweets per day. After pre-processing of tweets from the first phase dataset we started further experiments on the same. After one month we again started collecting tweets from the twitter as on that time the pandemic was spreading with its fatal intensity. We collected almost 320K tweets during August - October, 2020 for the second phase dataset (Covid-19 Dataset II). To handle these large datasets, we need good computing power which is a challenge often faced by experts dealing with Big Data.

Initially, we have developed deep hybrid Convolutional LSTM network for sentiment prediction task so we need to import several deep learning packages from popular deep learning framework distribution platforms like *TensorFlow*[4] and *Keras*[5]. With a little experience with these frameworks anyone can understand that the GPU (Graphics Processing Unit) enabled distributions[6] of these frameworks process much faster than the normal CPU enabled versions because CPUs are designed for more general computing workloads. A GPU is a specialized processor with dedicated memory that conventionally

[4] https://www.tensorflow.org/
[5] https://keras.io/
[6] https://developer.nvidia.com/deep-learning

perform floating point operations required for rendering graphics. In other words, it is a single-chip processor used for extensive Graphical and Mathematical computations which frees up CPU cycles for other jobs. The main difference between GPUs and CPUs is that GPUs devote proportionally more transistors to arithmetic logic units and fewer to caches and flow control as compared to CPUs. GPUs are optimized for training artificial intelligence and deep learning models as they can process multiple computations simultaneously. Mostly, for normal researchers without the facility of any workstation or GPU-based servers, the Nvidia GPUs with Pascal architecture and CUDA [7](Compute Unified Device Architecture) are sufficient for deep learning, but they are not very economical to buy in Indian market anyway. However, with the help of the new machine, we overcame this major obstacle of our work.

3.5 Applications of this Research Work

3.5.1 Application of Sentiment Analysis in Business

The sentiment analysis is mostly used as a tool for Voice of Customer and Voice of Employee with different purposes. Companies use sentiment analysis to understand how customers and employees feel about some topics, and to learn the main reason for those opinions. These insights are then used to enhance customer/employee experience which contributes to higher incomes and stronger productivity for the company:

- ***Social Media Monitoring:*** In today's day and age, brands of all shapes and sizes have meaningful interactions with customers, leads, and even competition on social networks like Facebook, Twitter, and Instagram. By using sentiment analysis on social media, we can get incredible insights into the quality of conversation that's happening around a brand.

- ***Brand Monitoring:*** Analyze news articles, blog posts, forum discussions, and other texts on the internet over a period of time to see sentiment of a particular audience. Automatically alert designated team members of online mentions that concern their area of work.

- ***Customer Feedback:*** Target individuals to improve their service. By automatically running sentiment analysis on incoming surveys, we can detect customers who are 'strongly negative' towards our product or service, so we can respond to them right away.

- ***Customer Service:*** Sentiment Analysis can be used to automate text classification all incoming customer support queries, route queries to specific team members best suited to respond.

[7] https://developer.nvidia.com/cuda-zone

- ***Market Research:*** Sentiment Analysis empowers all kinds of market research and competitive analysis. It can be used to analyze formal market reports or business journals for long-term, broader trends.

3.5.2 Application of Sentiment Analysis in Other Aspects

Sentiment analysis is used to extract such opinion and remarks of users by classifying them as positive, negative and natural sentiment. Although there are a number of definitions about sentiment analysis in the literature, but in simple terms sentiment analysis is a technique used to extract intelligent information based on the person's opinion from raw data available on the internet. The opinion Mining and sentiment analysis cover a wide range of applications. That implies for this research work as well.

- ***Election Result Prediction:*** Twitter sentiment analysis is quick and inexpensive way for real-time election monitoring and modern-day election predictions. Recent research relies on explicit mining of public sentiment using lexical and syntactic features in tweets. The text mining and sentiment analysis framework can be used for finding most popular topics about contesting political parties discussed on Twitter.

Chapter 4:

Design Methodology of Proposed Work

In this chapter we present the Design Methodology of our proposed work through both the visual and documented representation of complete phase by phase planning of this thesis work over the duration of last one year (April 2020 - May 2021). This includes a cohesiveness to gain a transparent visual idea about all the components of the experiment for past months. We have done our planning and estimation using Software Project Management (SPM) concept as all project needs a calculated planning and estimation. This concept is a summarization of Work Breakdown Structure (WBS), Time Line Chart or Task Sheet and Gantt Chart to show the complete experiment workflow with the time, and the Flowchart of the proposed methodology.

4.1 Work Breakdown Structure (WBS)

In project management and systems engineering, a Work Breakdown Structure is a deliverable-oriented breakdown of a project into smaller components. It's a way to divide and conquer large projects to get things done faster and more efficiently. Hence, we have merged all the activities into major categories as shown in the following WBS diagram. In Figure 1, we present the Work Breakdown Structure of our overall project work.

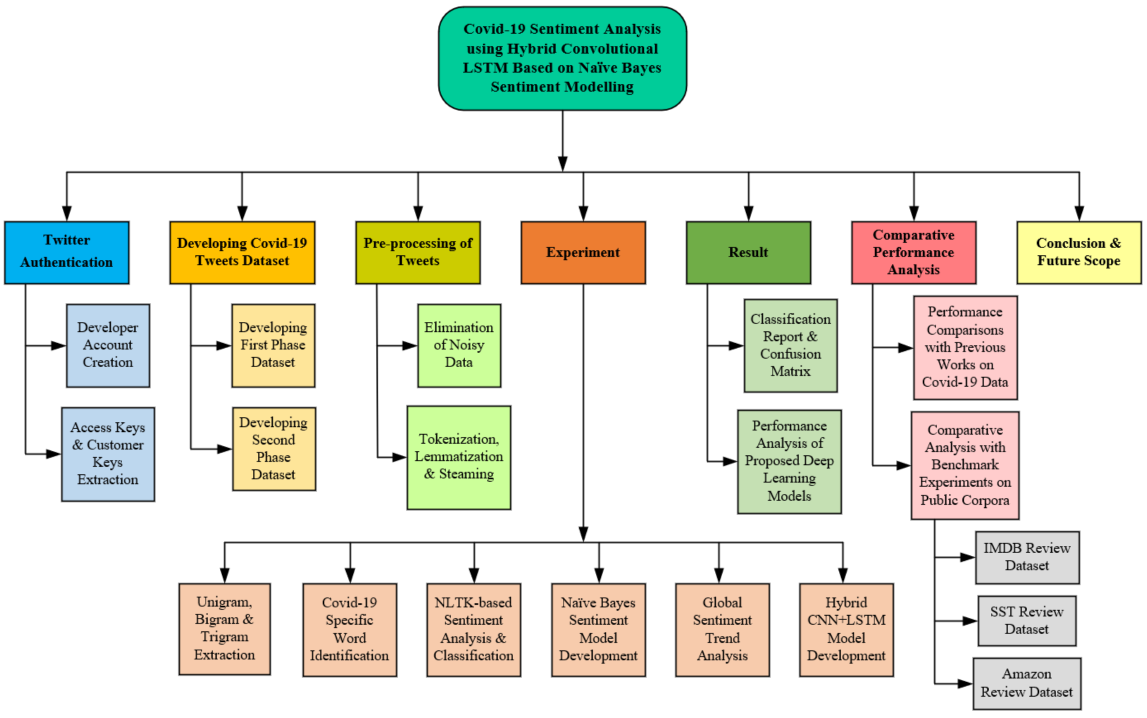

Covid-19 Sentiment Analysis using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling
Twitter Authentication
Developer Account Creation
Access Keys & Customer Keys Extraction
Developing Covid-19 Tweets Dataset
Developing First Phase Dataset
Developing Second Phase Dataset
Pre-processing of Tweets
Elimination of Noisy Data
Tokenization, Lemmatization & Steaming
Experiment
Unigram, Bigram & Trigram Extraction
Covid-19 Specific Word Identification
NLTK-based Sentiment Analysis & Classification
Naïve Bayes Sentiment Model Development
Global Sentiment Trend Analysis
Hybrid CNN+LSTM Model Development
Result
Classification Report & Confusion Matrix
Performance Analysis of Proposed Deep Learning Models
Comparative Performance Analysis
Performance Comparisons with Previous Works on Covid-19 Data
Comparative Analysis with Benchmark Experiments on Public Corpora
IMDB Review Dataset
SST Review Dataset
Amazon Review Dataset
Conclusion & Future Scope

Figure 1. Work Breakdown Structure of Covid-19 Sentiment Analysis using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling.

4.2 Timeline Task Sheet

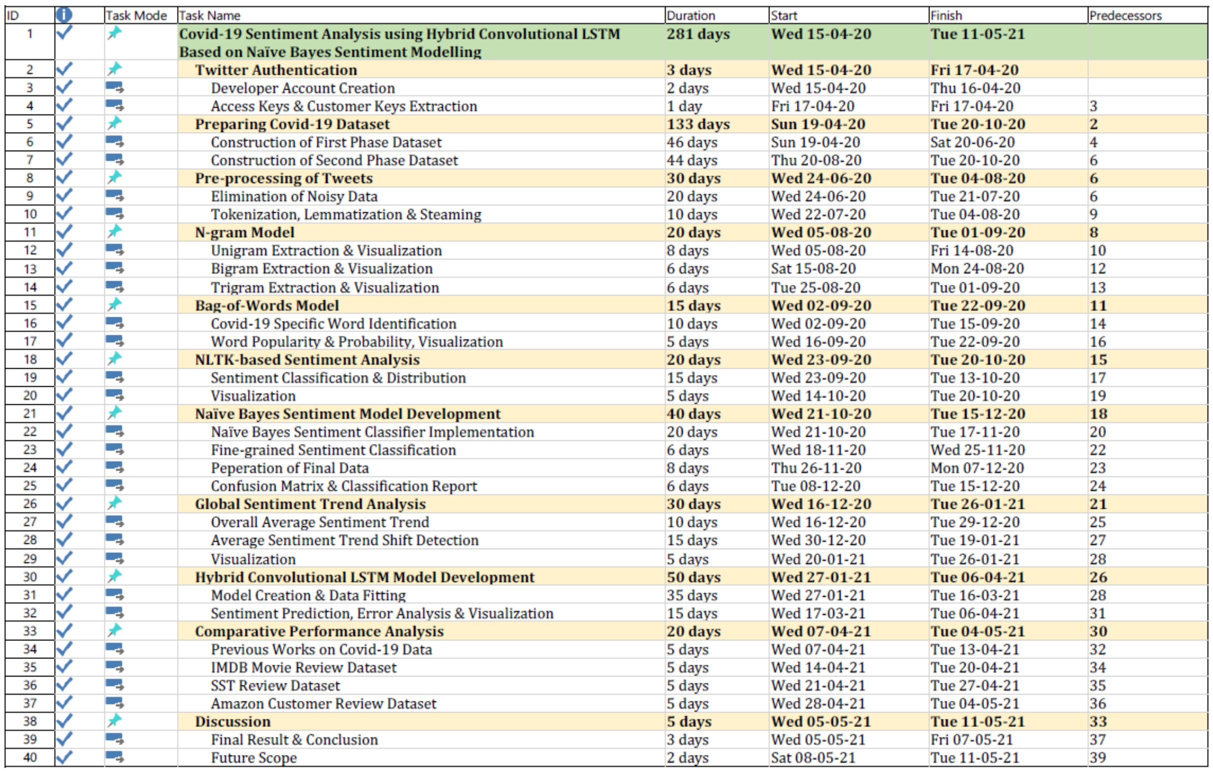
A project is a set of multiple tasks. Project teams execute and complete tasks to deliver projects. So that a project task list is the foundation of any project. The Timeline Task Sheet functionality allows us to view information about each task (such as task duration, start dates, finish dates etc.) in a sheet format. Many people use the task sheet to generate the following visualizations:

- List of tasks along with the relevant information about the task helps us to understand the detailed breakdown of the project. Assign personnel and other resources to tasks.

- Review progress by comparing planned (baseline) dates and actual start and finish dates, and by checking the progress of each task.

- Establish sequential task dependencies by linking tasks. When we link tasks, we can see how a change in the duration of one task affects the start and finish dates of other tasks, as well as the project finish date.

Since we did a long-term experimental research work on large Covid-19 Twitter data for about one year, we needed to keep track of our work timelines according to the schedule, for both completing this work within the time, and to meet the submission deadlines for our publications. Figure 2 represents the Timeline Task Sheet of our overall project work.



ID	ⓘ	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1	✓	📌	**Covid-19 Sentiment Analysis using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling**	**281 days**	**Wed 15-04-20**	**Tue 11-05-21**	
2	✓	📌	**Twitter Authentication**	**3 days**	**Wed 15-04-20**	**Fri 17-04-20**	
3	✓	➡	Developer Account Creation	2 days	Wed 15-04-20	Thu 16-04-20	
4	✓	➡	Access Keys & Customer Keys Extraction	1 day	Fri 17-04-20	Fri 17-04-20	3
5	✓	📌	**Preparing Covid-19 Dataset**	**133 days**	**Sun 19-04-20**	**Tue 20-10-20**	**2**
6	✓	➡	Construction of First Phase Dataset	46 days	Sun 19-04-20	Sat 20-06-20	4
7	✓	➡	Construction of Second Phase Dataset	44 days	Thu 20-08-20	Tue 20-10-20	6
8	✓	📌	**Pre-processing of Tweets**	**30 days**	**Wed 24-06-20**	**Tue 04-08-20**	**6**
9	✓	➡	Elimination of Noisy Data	20 days	Wed 24-06-20	Tue 21-07-20	6
10	✓	➡	Tokenization, Lemmatization & Steaming	10 days	Wed 22-07-20	Tue 04-08-20	9
11	✓	📌	**N-gram Model**	**20 days**	**Wed 05-08-20**	**Tue 01-09-20**	**8**
12	✓	➡	Unigram Extraction & Visualization	8 days	Wed 05-08-20	Fri 14-08-20	10
13	✓	➡	Bigram Extraction & Visualization	6 days	Sat 15-08-20	Mon 24-08-20	12
14	✓	➡	Trigram Extraction & Visualization	6 days	Tue 25-08-20	Tue 01-09-20	13
15	✓	📌	**Bag-of-Words Model**	**15 days**	**Wed 02-09-20**	**Tue 22-09-20**	**11**
16	✓	➡	Covid-19 Specific Word Identification	10 days	Wed 02-09-20	Tue 15-09-20	14
17	✓	➡	Word Popularity & Probability, Visualization	5 days	Wed 16-09-20	Tue 22-09-20	16
18	✓	📌	**NLTK-based Sentiment Analysis**	**20 days**	**Wed 23-09-20**	**Tue 20-10-20**	**15**
19	✓	➡	Sentiment Classification & Distribution	15 days	Wed 23-09-20	Tue 13-10-20	17
20	✓	➡	Visualization	5 days	Wed 14-10-20	Tue 20-10-20	19
21	✓	📌	**Naïve Bayes Sentiment Model Development**	**40 days**	**Wed 21-10-20**	**Tue 15-12-20**	**18**
22	✓	➡	Naïve Bayes Sentiment Classifier Implementation	20 days	Wed 21-10-20	Tue 17-11-20	20
23	✓	➡	Fine-grained Sentiment Classification	6 days	Wed 18-11-20	Wed 25-11-20	22
24	✓	➡	Peperation of Final Data	8 days	Thu 26-11-20	Mon 07-12-20	23
25	✓	➡	Confusion Matrix & Classification Report	6 days	Tue 08-12-20	Tue 15-12-20	24
26	✓	📌	**Global Sentiment Trend Analysis**	**30 days**	**Wed 16-12-20**	**Tue 26-01-21**	**21**
27	✓	➡	Overall Average Sentiment Trend	10 days	Wed 16-12-20	Tue 29-12-20	25
28	✓	➡	Average Sentiment Trend Shift Detection	15 days	Wed 30-12-20	Tue 19-01-21	27
29	✓	➡	Visualization	5 days	Wed 20-01-21	Tue 26-01-21	28
30	✓	📌	**Hybrid Convolutional LSTM Model Development**	**50 days**	**Wed 27-01-21**	**Tue 06-04-21**	**26**
31	✓	➡	Model Creation & Data Fitting	35 days	Wed 27-01-21	Tue 16-03-21	28
32	✓	➡	Sentiment Prediction, Error Analysis & Visualization	15 days	Wed 17-03-21	Tue 06-04-21	31
33	✓	📌	**Comparative Performance Analysis**	**20 days**	**Wed 07-04-21**	**Tue 04-05-21**	**30**
34	✓	➡	Previous Works on Covid-19 Data	5 days	Wed 07-04-21	Tue 13-04-21	32
35	✓	➡	IMDB Movie Review Dataset	5 days	Wed 14-04-21	Tue 20-04-21	34
36	✓	➡	SST Review Dataset	5 days	Wed 21-04-21	Tue 27-04-21	35
37	✓	➡	Amazon Customer Review Dataset	5 days	Wed 28-04-21	Tue 04-05-21	36
38	✓	📌	**Discussion**	**5 days**	**Wed 05-05-21**	**Tue 11-05-21**	**33**
39	✓	➡	Final Result & Conclusion	3 days	Wed 05-05-21	Fri 07-05-21	37
40	✓	➡	Future Scope	2 days	Sat 08-05-21	Tue 11-05-21	39

Figure 2. Timeline Task Sheet of Covid-19 Sentiment Analysis using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling.

Timelines are designed to provide a broad overview of a sequence of events in time, and it also helped us to monitor and manage our work progress over the time in an organized manner. While this chart may not go into detail, but it is helpful for links to events, information and keeping the scheduling updated as needed. In project management, timelines are most useful for showing important milestones and deadlines.

The Timeline Task Sheet diagram consists of information about stages of our experiments with time progressing from left to right. We have listed the major experimental modules along with all the sub modules in the chart. The Task Name tab is filled up with events or steps to indicate when they should or did happen.

4.3 Gantt Chart

A Gantt Chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. A Gantt Chart can be generated from a Time Line Task Sheet, but not the visa versa. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. Gantt Charts are used for planning projects of all sizes and they are a useful way of showing what work is scheduled to be done on a specific day. They also help us to view the start and end dates of a project in one simple view. In Figure 3, we present the Gantt Chart of the entire project work.

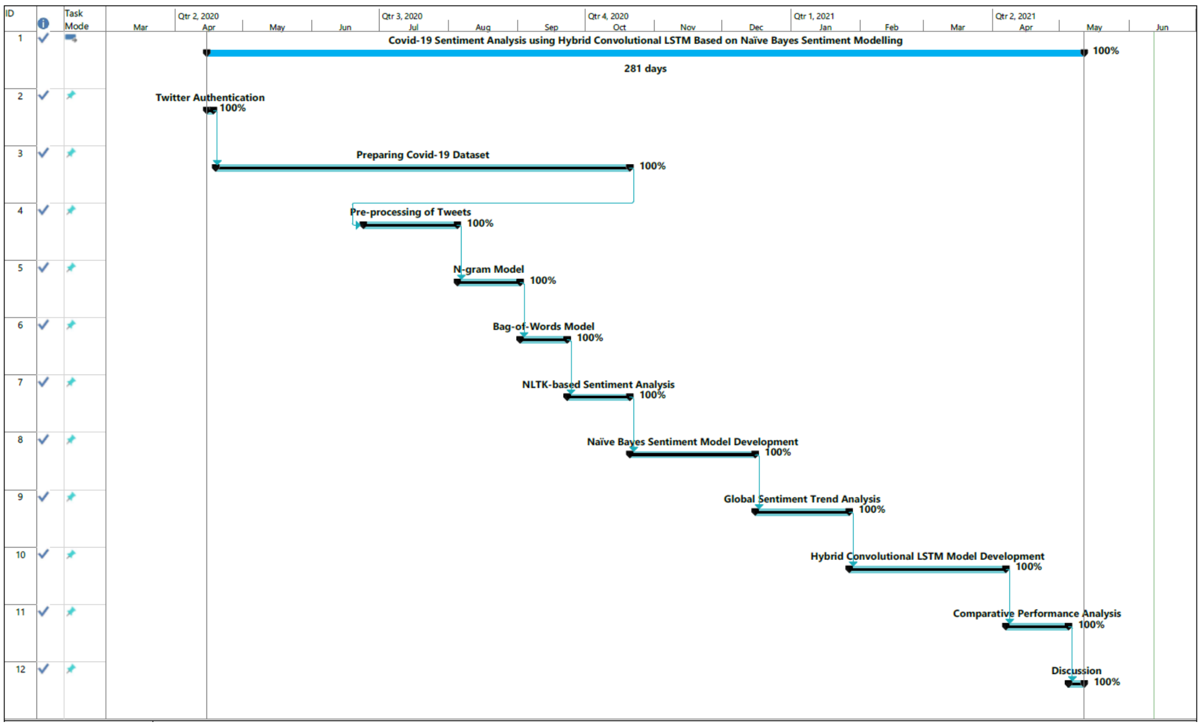


Figure 3. Gantt Chart of Covid-19 Sentiment Analysis using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling.

With the documented major project modules list as Time Line Task Sheet, we also needed to visualize our work progress, as it reflects a clear idea about the timeliness of the work. Hence, we developed the Gantt Chart time to time as our research work progressed, to maintain the transparency and dependency among all the segments of our experiment, as well as to maintain the dates accordingly and accurately.

4.4 Proposed System Architecture

Large scale tweet collection is the foundation of this research as well as the first phase of our proposed method. Initially we collected 235k tweets related to Covid-19 during the phase April - June, 2020. The second dataset consists of 320k Covid-19 tweets which have collected during the phase August - October, 2020. Then using the *NLTK*[8] (Natural Language Toolkit) package, the collected tweets are cleaned and pre-processed from both of the corpora by eliminating noisy data from the tweets. Then the tweets are tokenized to extract the smaller tokens from the tweets and stemming is used to convert the tokens into their original word stem. After cleaning and pre-processing the most frequent unigrams, bigrams and trigrams are extracted from the dataset. Then we identify the Covid-19 exclusive words and their frequency of recurrence in the corpus to understand the impact of those words over the tweets.

On the other hand, the cleaned and pre-processed tweets are analyzed using *NLTK* based sentiment analyzer to evaluate the sentiment polarity of each tweet. Then using evolutionary classification, the tweets are classified into *positive*, *negative* and *neutral* classes on the basis of their *compound* sentiment ratings. Then Naïve Bayes Sentiment model is developed based on the popular grams and classified *positive* & *negative* tweets for assigning the Refined Sentiment rating to each of the tweet. The model assigns sentiment scores between 0.0 - 1.0 to each of the tweets. Finally, we further classify the tweets into five sentiment classes for fine-grained classification. The tweets are classified into Most Negative (-1.0), Negative (-0.5), Neutral (0.0), Positive (0.5) and Most Positive (1.0) classes respectively. Then we find date wise Global Sentiment Trend for Covid-19 using refined sentiment ratings.

Considering the polar tweets from the corpora we transform them into word vectors for further processing. The datasets are divided into 80:20 ratio for training and validation of the hybrid models. Finally, both Convo-Sequential and Convo-Bidirectional deep LSTM networks are trained using the vectorized tweets along with their updated sentiment scores to predict the sentiment from the tweets. After completion of training the proposed models achieve certain accuracy on predicted and evaluated data from Covid-19 datasets. We also train these models on different widely available public corpora to identify the noticeable performance efficiency of our system by performing the comparative study with other previous benchmark experiments. In Figure 4, we present the Proposed System Architecture of this project work.

[8] https://www.nltk.org/

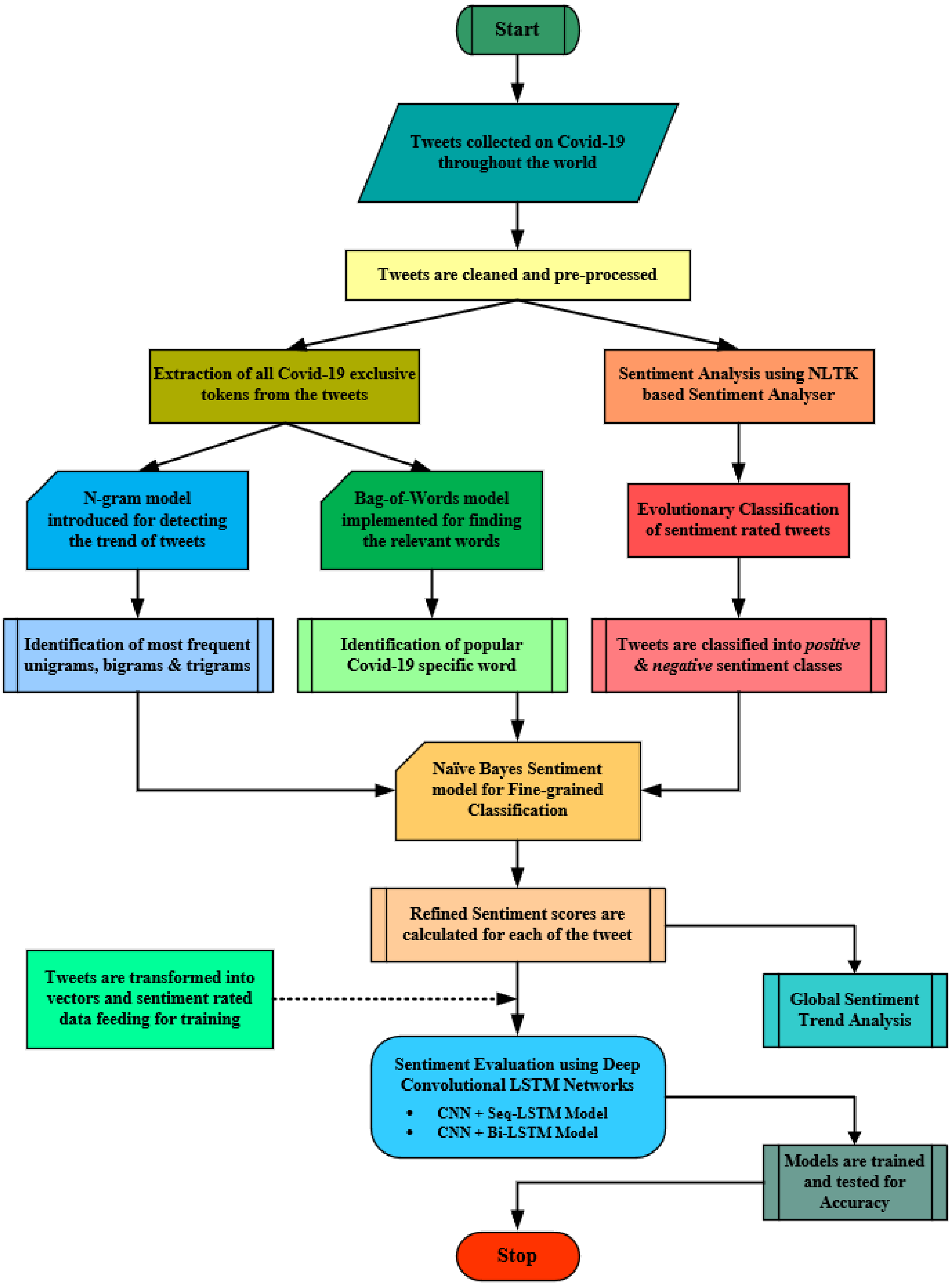

Start
Tweets collected on Covid-19 throughout the world
Tweets are cleaned and pre-processed
Extraction of all Covid-19 exclusive tokens from the tweets
Sentiment Analysis using NLTK based Sentiment Analyser
N-gram model introduced for detecting the trend of tweets
Bag-of-Words model implemented for finding the relevant words
Evolutionary Classification of sentiment rated tweets
Identification of most frequent unigrams, bigrams & trigrams
Identification of popular Covid-19 specific word
Tweets are classified into positive & negative sentiment classes
Naïve Bayes Sentiment model for Fine-grained Classification
Refined Sentiment scores are calculated for each of the tweet
Tweets are transformed into vectors and sentiment rated data feeding for training
Global Sentiment Trend Analysis
Sentiment Evaluation using Deep Convolutional LSTM Networks
CNN + Seq-LSTM Model
CNN + Bi-LSTM Model
Models are trained and tested for Accuracy
Stop

Figure 4. Proposed System Architecture of Covid-19 Sentiment Analysis using Hybrid Convolutional LSTM Based on Naïve Bayes Sentiment Modelling.

Chapter 5:

Preparing Covid-19 Dataset

The coronavirus pandemic has reached almost every country in the world. The World Health Organization declared the outbreak a Public Health Emergency of International Concern on 30 January 2020, and a pandemic on 11 March 2020. According to official reports, the largest numbers of confirmed cases were in the United States, Italy, Spain, and France. However, even the countries that the new coronavirus has hit less aggressively are still under considerable strain. Many countries have declared restrictive measures, such as lockdown, shelter in place, or stay at home orders, to contain the pandemic at a local level. However, the wildly differing responses and response timelines have left people wondering if authorities failed to take the situation seriously early on when they could have done more to slow down the spread of the coronavirus. It has now been a year since the start of the Covid-19 pandemic, which has claimed millions of lives and changed the ways in which each of us relates to and navigates the world.

5.1 Collection of Tweets

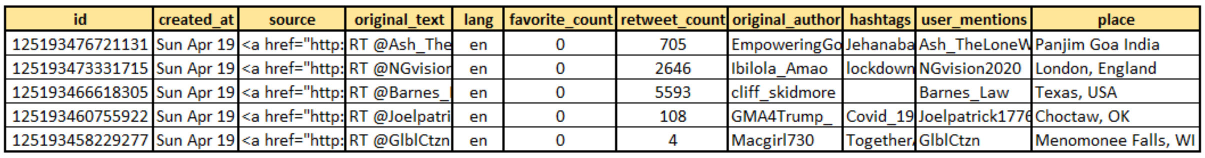
We have started streaming live tweets from the twitter after WHO declared Covid-19 as a pandemic. Since this Covid-19 epidemic has affected the entire world, worldwide Covid-19 related English tweets were collected at a rate of 10k per day in two phases starting from April - June, 2020 to August - October, 2020. We prepared the first phase dataset of about 235k tweets collected from 19th April to 20th June, 2020. After one month we again start collecting tweets from the twitter as on that time the pandemic was spreading with its fatal intensity. Almost 320K tweets were collected in the period 20th August to 20th October, 2020 for the second phase dataset.

Unsurprisingly, these tweets were significantly noisy with grammatical errors, misspelled words, extra white spaces, numbers, URLs, misspelled punctuations marks, stop words, misused emojis and emoticons, similar tweets, tweets without any information and so on. The datasets we developed contains the important information about most of the tweets as their attributes. The attributes of both of these datasets are:

- id [Number]
- created_at [DateTime]
- source [Text]
- original_text [Text]
- favorite_count [Number]
- retweet_count [Number]
- original_author [Text]
- hashtags [Text]
- user_mentions [Text]
- place [Text]

Twitter API mostly allows the users to live stream only 1-2% of the total tweets on a particular keyword, we observed that we were able to collect tweets at a rate of almost 10 thousand tweets per day. Until October 2017, Twitter used to support a highest of 140 characters per tweet, including emojis. But on 7th November 2017, Twitter expanded its character limits to 280 characters per tweet.

Finally, we collected 2,35,240 tweets for first phase dataset and 3,20,316 tweets for second phase dataset containing the hash-tagged keywords like - **#covid-19**, **#coronavirus**, **#covid**, **#covaccine**, **#lockdown**, **#homequarantine**, **#quarantinecenter**, **#socialdistancing**, **#stayhome**, **#staysafe** etc. In Figure 5, we present an overview of our collected dataset.



id	created_at	source	original_text	lang	favorite_count	retweet_count	original_author	hashtags	user_mentions	place
125193476721131	Sun Apr 19	<a href="http:	RT @Ash_The	en	0	705	EmpoweringGo	Jehanaba	Ash_TheLoneW	Panjim Goa India
125193473331715	Sun Apr 19	<a href="http:	RT @NGvision	en	0	2646	Ibilola_Amao	lockdown	NGvision2020	London, England
125193466618305	Sun Apr 19	<a href="http:	RT @Barnes_	en	0	5593	cliff_skidmore		Barnes_Law	Texas, USA
125193460755922	Sun Apr 19	<a href="http:	RT @Joelpatri	en	0	108	GMA4Trump_	Covid_19	Joelpatrick1776	Choctaw, OK
125193458229277	Sun Apr 19	<a href="http:	RT @GlblCtzn	en	0	4	Macgirl730	Together,	GlblCtzn	Menomonee Falls, WI

Figure 5. Partial snapshot of Covid-19 tweets corpus.

Chapter 6:

Experiment & Results

Since the entire experiment went through roughly over a span of one year, hence it is quite obvious that experiment was elaborative and time-consuming. In fact, at first setting up the Python environment along with *NLTK* packages for Natural Language Processing experiments is a challenge by itself. It provides useful tools and algorithms such as tokenizing, part-of-speech tagging, stemming, and named entity recognition. For the enhanced deep learning-based experimentation, we need to install Nvidia CUDA Development Toolkit along with Nvidia cuDNN[9] (CUDA Deep Neural Network library) to improve the model performance during the experiment. Not only these installations are significantly time-consuming, but also these require a fair share of resource and computing power also. We have explained a complete list of all the required packages in Appendix section with their respective version details, that we installed for our experiment. While the total project work is segmental and each of the modules can easily be distinguishable from the other once, hence the experiment is a rather dynamic process, where each stage of the experiment is followed by an immediate subsequent result. It is also mentionable that we did not obtain precise and accurate results instantly after each module of the experiment, but it was a constant repeated process to remove the usual programming bugs and refining our results more and more for the optimal betterment achievable.

6.1 Experimental Setup

Initially, the experiments were started with the previous experimental setup after developing the first phase dataset. Executing few experiments, we found that the previous machine was not sufficient to process the preplanned experiments of our project work with the large datasets. However, to overcome this difficulty we used online web servers like- Google Colaboratory, IBM Watson Studio for further processing. Using these servers, we have done lot of experiments with the datasets. But they only provide activated kernel for limited 12Hrs. runtime and again which was not sufficient enough for all of the proposed experimentations. Finally, with the help of the new machine, we overcame this major obstacle of our work. After installing all the required applications along with *nltk*, *matplotlib*[10], *plotly*[11], *scikit-learn*[12], *genism*[13], *tensorflow* (both CPU & GPU versions), *keras* and other relevant packages (Details on Appendix II - page no. 72) related

[9] https://developer.nvidia.com/cudnn
[10] https://matplotlib.org/
[11] https://plotly.com/
[12] https://scikit-learn.org/stable/
[13] https://radimrehurek.com/gensim/

to this project work, we became able to execute all of the required experiments without any further problems.

6.1.1 Hardware Configuration

Here we have listed the hardware configuration of our system.

- Processor: Intel Core i7-10750H
- RAM: 16GB
- SSD: 1TB
- GPU: Nvidia RTX 2060 6GB
- Operating System: Windows 10 Home Edition

6.1.2 Software Configuration

Here we have listed the software configuration of our system.

- Programming Language: Python (3.7.3)
- Application: Anaconda, MS Visual Studio Code, Notepad++
- GPU Application: Nvidia Graphics Driver, Nvidia CUDA Toolkit, Nvidia cuDNN
- Web Browser: Microsoft Edge, Google Chrome, Mozilla Firefox
- Web Server: Google Colaboratory, IBM Watson Studio

6.2 Data Pre-processing

In NLP, text pre-processing is an essential step for cleaning and preparing text data for building a Machine Learning model. Data pre-processing is mainly used for cleaning the raw data by following certain steps to achieve the better result for further evaluations. We used Twitter RESTful API *tweepy*[14] to access data about both Twitter users and what they are tweeting about this pandemic. As we have collected raw tweets for developing the datasets, they contain lots of duplicate tweets having same status ID. However, we have removed all the duplicate tweets from the datasets and found 1,43,903 and 1,20,509 tweets from first phase and second phase Covid-19 datasets. After removing all the duplicate tweets, final size of the datasets was 35MB and 45MB respectively.

The collected tweets were pre-processed by developing a user defined pre-processing function based on NLTK, a Python library for NLP. In the first phase it converts all the tweets into lower case. Then it removes all extra white spaces, numbers, special characters, ASCII characters, URLs, punctuations & stop words from the tweets. Then it converts all ‘covid’ words into ‘covid19’ as all numbers were already removed from the tweets. Tokenization is a way of separating a piece of text into smaller units called tokens. We tokenize each tweet using the function to split the sentence into smaller parts of word. Stemming is the process of reducing a word to its word stem that affixes to suffixes and

[14] https://www.tweepy.org/

prefixes or to the roots of words known as a lemma. Using stemming the pre-processing function has reduced inflected words to their word stem. In Figure 6, we present the overall date wise distribution of tweets from both of the datasets.

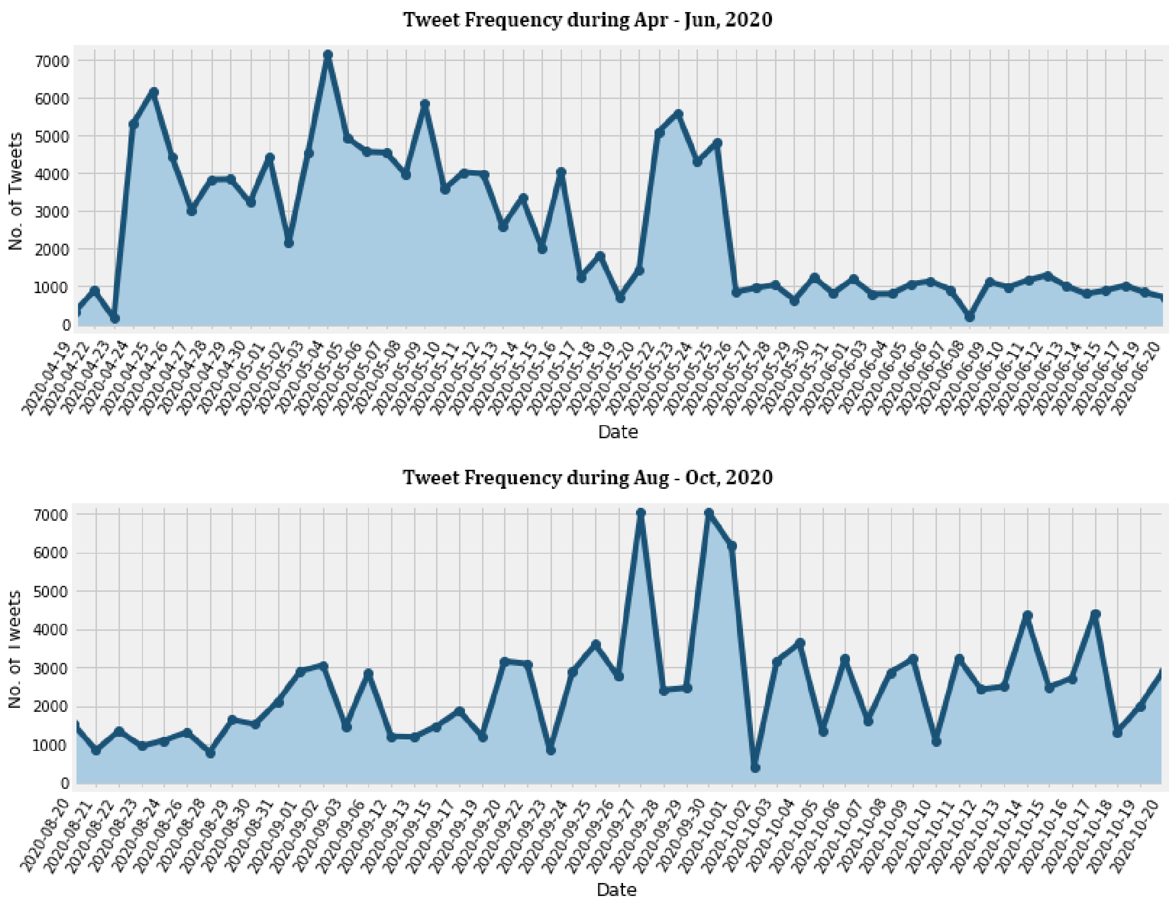


Figure 6. Tweets distribution of the Covid-19 tweets corpora.

6.3 Feature I: Word Trend Detection using N-Gram Analysis

Lexical n-gram models are widely used in Natural Language Processing for statistical analysis & syntax feature mapping. We developed n-gram model to analyze the generated corpus consisting of tokenized words for finding the popularity of words or group of adjacent words. Here the probability of the occurrence of a sequence can be calculated using probability chain rule:

$$P(x_1, x_2, x_3, \dots x_n) \; = \; P(x_1)\, P(x_2 \mid x_1)\, P(x_3 \mid x_1, x_2) \;\dots\; P(x_n \mid x_1, x_2, x_3, \dots\, x_{(n-1)}) \qquad (1)$$

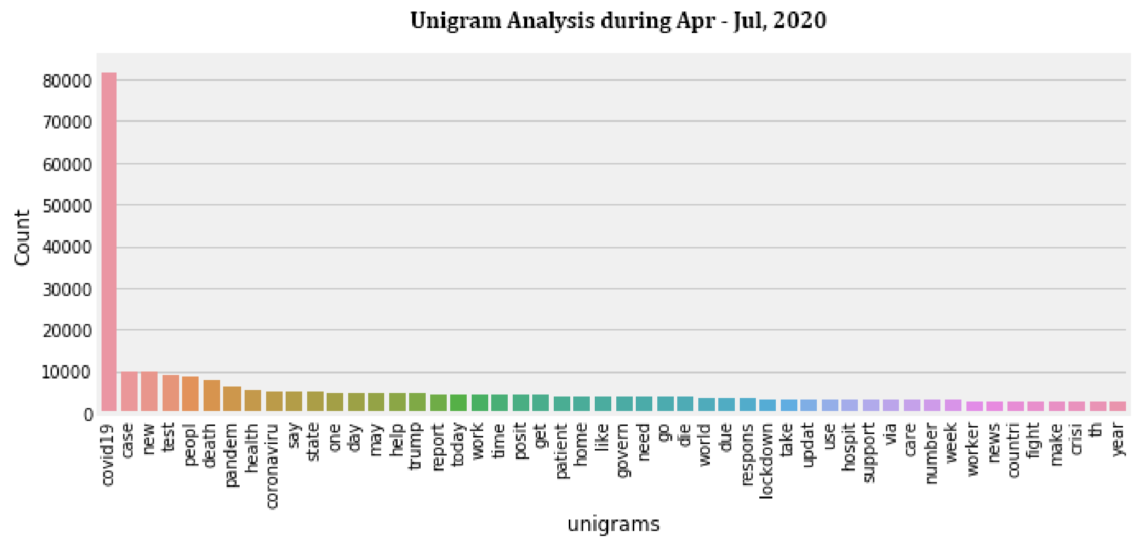
$$= \prod_{i=1}^{n} P(x_i \mid x_1^{(i-1)}) \qquad (2)$$

For example, we can consider a sentence as “Still Covid-19 wave is running”. Now as per the probability chain rule, *P(“Still Covid19 wave is running”) = P(“Still”) x P(“Covid19” | “Still”) x P(“wave” | “Still Covid19”) x P(“is” | “Still Covid19 wave”) x P(“running” | “Still Covid19 wave is”)*.

The probabilities of words in each sentence after applying probability chain rule:

$$P(W_1^n) \;=\; \prod_j P(W_j \mid W_1, W_2, W_3, \ldots W_{(j-1)}) \tag{3}$$

$$= \prod_{j=1}^{n} P(W_j \mid W_1^{(j-1)}) \tag{4}$$



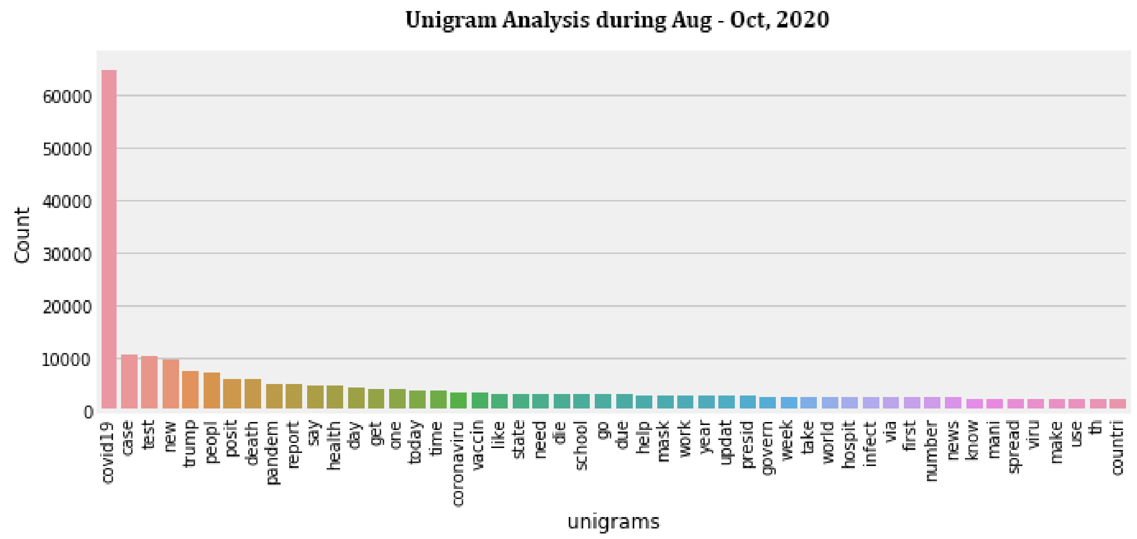


Figure 7. Graphical representation of the popularity for most frequent unigrams.

The assumption that the probability of a word depends only on the previous word is called a Markov assumption. Markov models are the class of probabilistic models that assume we can predict the probability of some future unit without looking too far into the past. The bigram model estimates the probability of a word by using only the conditional probability $P(W_i|W_{i-1})$ of one preceding word on given condition of all the previous words $P(W_i|W_1^{i-1})$ [6].

$$P(W_1, W_2) \;=\; \prod_{i=2} P\,(W_2 \mid W_1) \tag{5}$$

The expression for the bigram probability is –

$$P(W_k \mid W_{(k-1)}) = \frac{count\ (W_{(k-1)}, W_k)}{count\ (W_{(k-1)})} \quad (6)$$

We extracted the most popular unigrams, bigrams and trigrams within the corpora using the n-gram model. The graphical representations of 50 most popular unigrams, bigrams and trigrams along with their popularity are presented in Figure 7, Figure 8 and Figure 9 respectively.

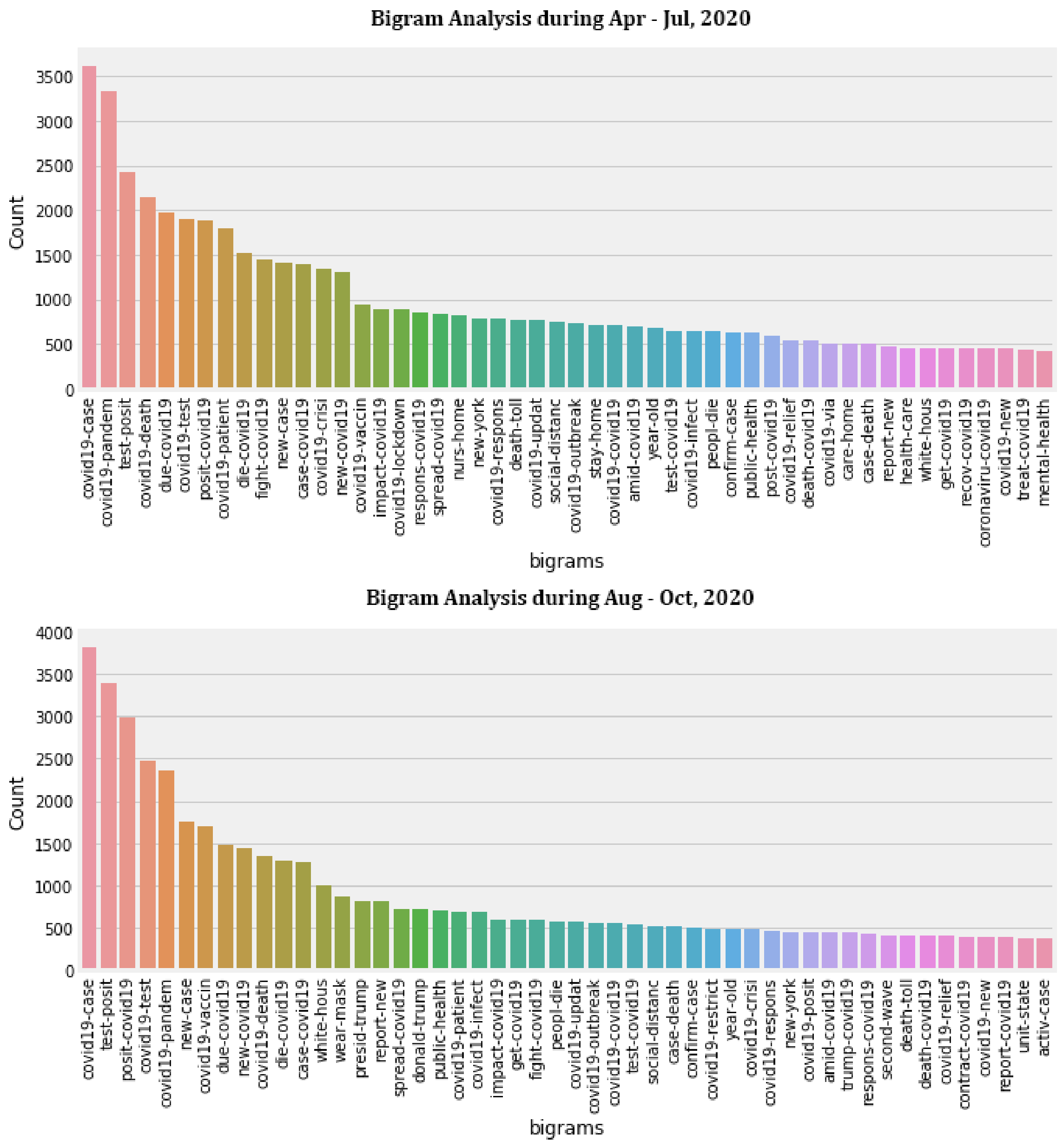


Figure 8. Graphical representation of the popularity for most frequent bigrams.

The n-gram analysis provides a brief idea of the most frequent topics to find out the reason behind a sentiment for better understanding of the human emotions on Covid related tweets. From this analysis, it is evident that there is a significant difference between the popularity of grams identified during the two different time phases as some new grams were detected in the period Aug – Oct, 2020.

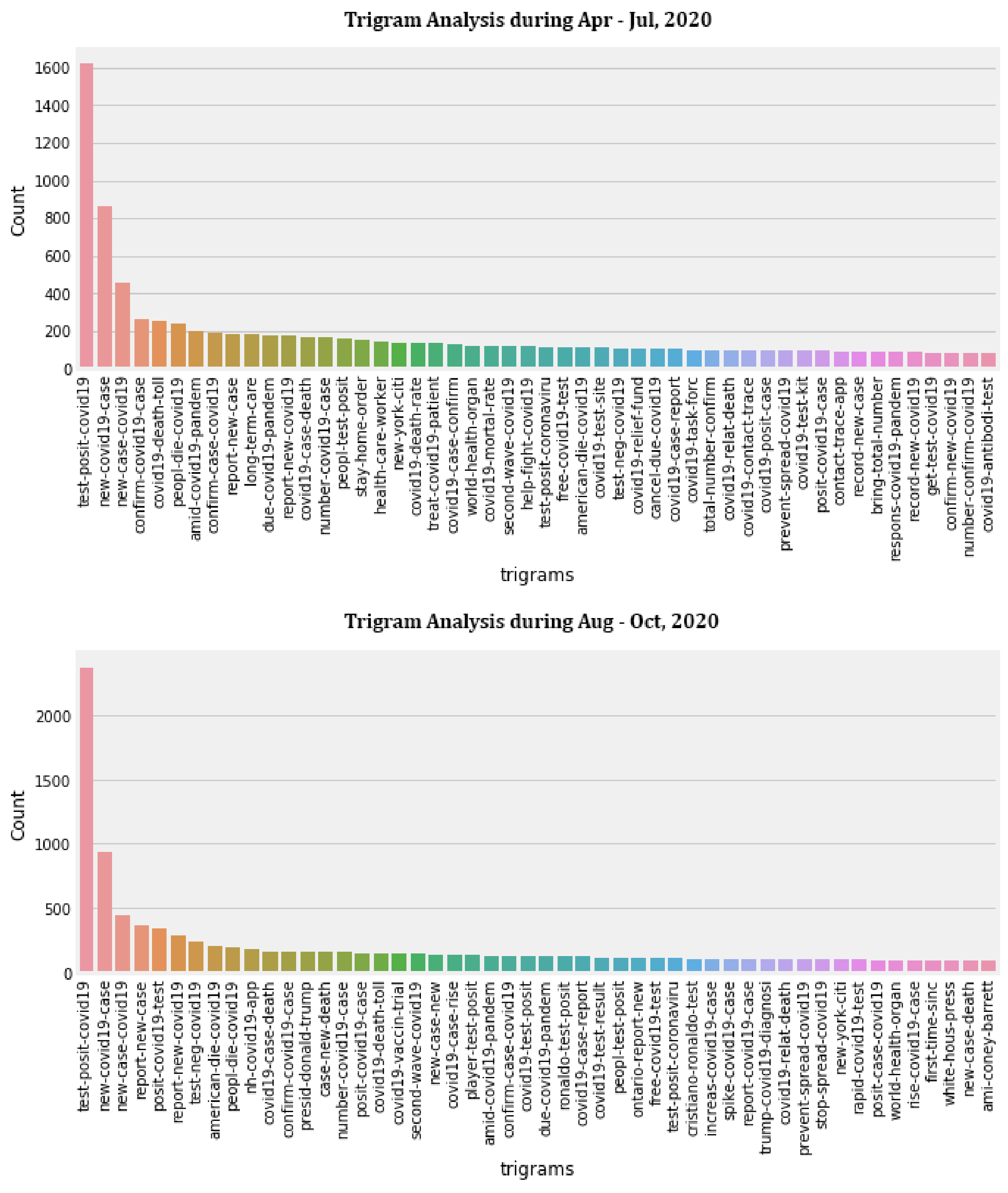


Figure 9. Graphical representation of the popularity for most frequent trigrams.

As the result of this analysis, we found that the popularity of trigrams is lesser than that of bigrams and the unigrams popularity are the highest according to this n-gram model. However, Table 2 represents some of the most frequent unigrams, bigrams and trigrams from both of the datasets along with their popularity count.

Table 2. Most popular n-grams during April - June and August - October, 2020.

	Covid-19 Dataset I (Apr – Jun, 2020)	Covid-19 Dataset II (Aug – Oct, 2020)
unigrams (n = 1)	('covid19', 81696), ('case', 10001), ('new', 9875), ('test', 9204), ('peopl', 8624), ('death', 7930), ('pandem', 6390), ('health', 5495), ('coronaviru', 5295), ('say', 5263)	('covid19', 64672), ('case', 10656), ('test', 10323), ('new', 9797), ('trump', 7605), ('peopl', 7011), ('posit', 5938), ('death', 5836), ('pandem', 5119), ('report', 5114)
bigrams (n = 2)	(('covid19', 'case'), 3612), (('covid19', 'pandem'), 3331), (('test', 'posit'), 2423), (('covid19', 'death'), 2149), (('due', 'covid19'), 1961), (('covid19', 'test'), 1899), (('posit', 'covid19'), 1877), (('covid19', 'patient'), 1800), (('die', 'covid19'), 1519), (('fight', 'covid19'), 1446)	(('covid19', 'case'), 3829), (('test', 'posit'), 3387), (('posit', 'covid19'), 2998), (('covid19', 'test'), 2480), (('covid19', 'pandem'), 2367), (('new', 'case'), 1756), (('covid19', 'vaccin'), 1699), (('due', 'covid19'), 1472), (('new', 'covid19'), 1431), (('covid19', 'death'), 1338)
trigrams (n = 3)	(('test', 'posit', 'covid19'), 1620), (('new', 'covid19', 'case'), 864), (('new', 'case', 'covid19'), 454), (('confirm', 'covid19', 'case'), 258), (('covid19', 'death', 'toll'), 253), (('peopl', 'die', 'covid19'), 237), (('amid', 'covid19', 'pandem'), 196), (('confirm', 'case', 'covid19'), 188), (('report', 'new', 'case'), 187), (('long', 'term', 'care'), 180)	(('test', 'posit', 'covid19'), 2370), (('new', 'covid19', 'case'), 934), (('new', 'case', 'covid19'), 441), (('report', 'new', 'case'), 359), (('posit', 'covid19', 'test'), 340), (('report', 'new', 'covid19'), 278), (('test', 'neg', 'covid19'), 236), (('american', 'die', 'covid19'), 202), (('peopl', 'die', 'covid19'), 193), (('nh', 'covid19', 'app'), 176)

6.4 Feature II: Covid-19 Specific Word Identification

After pre-processing the Bag-of-Words (BOW) model was developed using the frequently occurred words from the word lexicon and we obtained a list of most frequent Covid-19 exclusive words. The word-cloud mainly used for novel visual representation of text data. We present two dense word-clouds in Figure 10 of some of the mostly used words within both first and second phase datasets. The more a specific word appears in the list, the bigger and bolder it appears in the word-cloud.

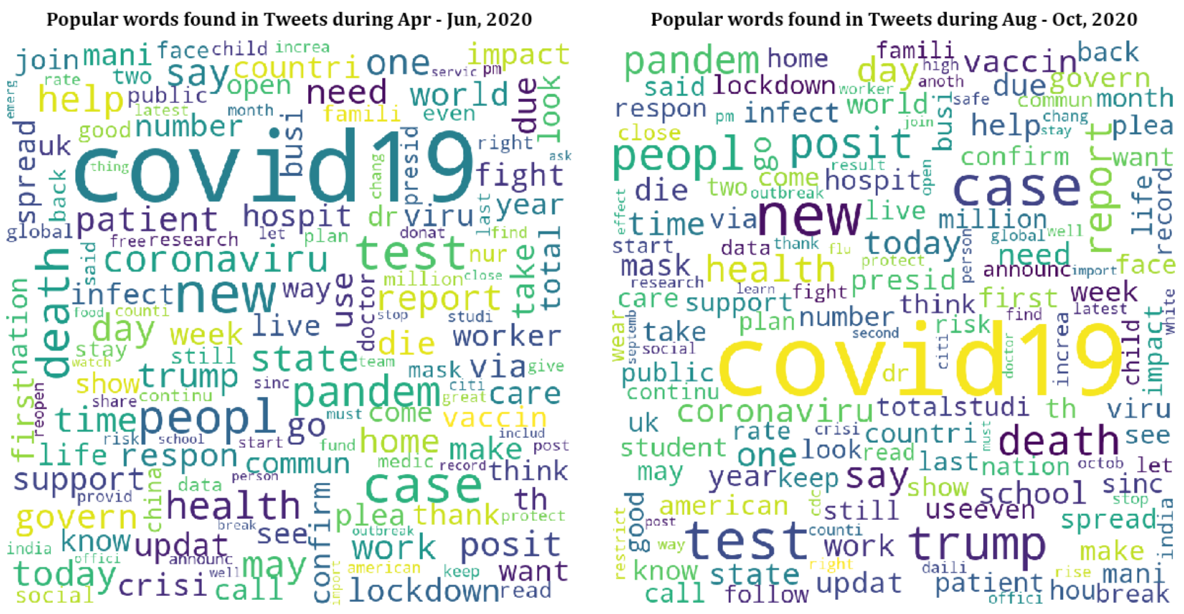


Figure 10. Some of the most popular Covid-19 related words from first and second phase datasets.

6.4.1 Word Popularity & Probability

Within the generated word lexicon, several words have been found at different times in different positions of the tweets. Identifying the most popular words from both of the corpora helps to analyze the impact of those words over the tweet. We found lists of almost 1.38 and 1.13 million words from both of the generated corpora. From the obtained lists of most popular Covid-19 exclusive words we found the frequency of each token. From both of the datasets we calculated the popularity score of each word.

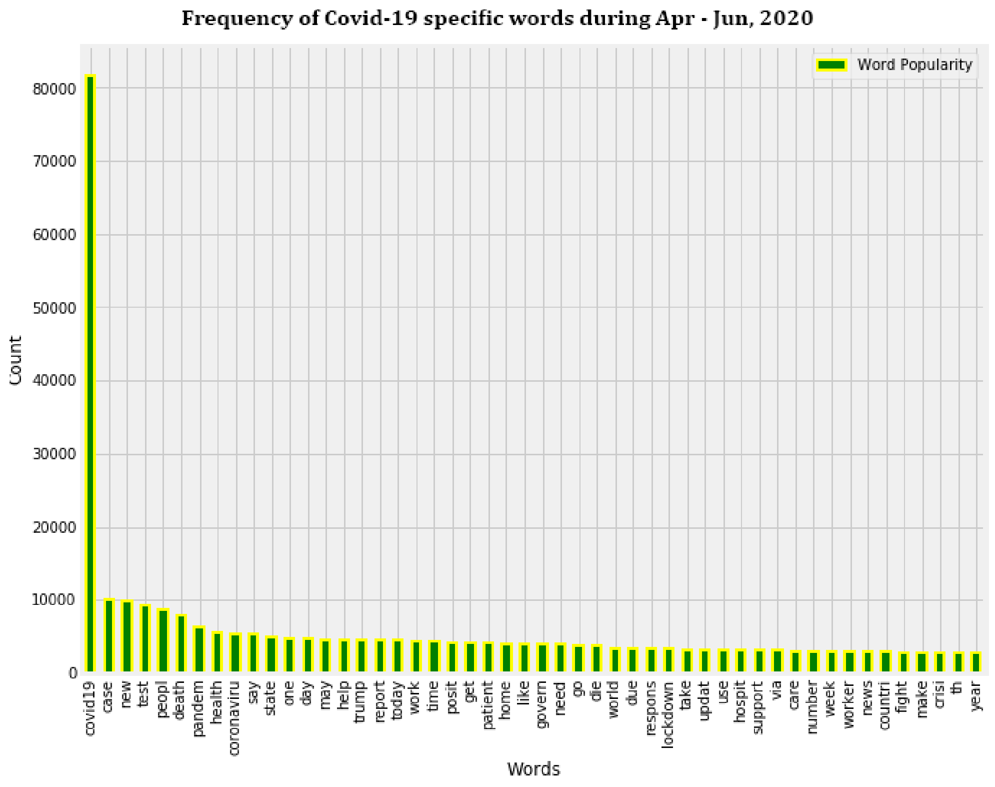
After finding the word popularity, the probability of occurrence for each word was calculated on the basis of total 13,79,835 and 11,33,188 words from both of the generated corpora. Table 3 represents the popularity and probability scores of some most frequent words from both first and second phase datasets.

$$P(W_i) = \frac{count(W_i)}{\sum_{i=0}^{n} count(W_{i=0}^{n})} \tag{7}$$

Table 3. Popularity & Probability of most frequent words.

	Covid-19 Dataset I (Apr – Jun, 2020)				Covid-19 Dataset II (Aug – Oct, 2020)		
	Words	**Popularity**	**Probability**		**Words**	**Popularity**	**Probability**
1	covid19	81696	0.059207	**1**	covid19	64672	0.057071
2	case	10001	0.007248	**2**	case	10656	0.009404
3	new	9875	0.007157	**3**	test	10323	0.009110
4	test	9204	0.006670	**4**	new	9797	0.008646
5	peopl	8624	0.006250	**5**	trump	7605	0.006711

From the analysis we found total 46800 and 39,834 unique words from both of the Covid-19 corpora. In Figure 11, we present the top 50 most popular words along with their popularity count.



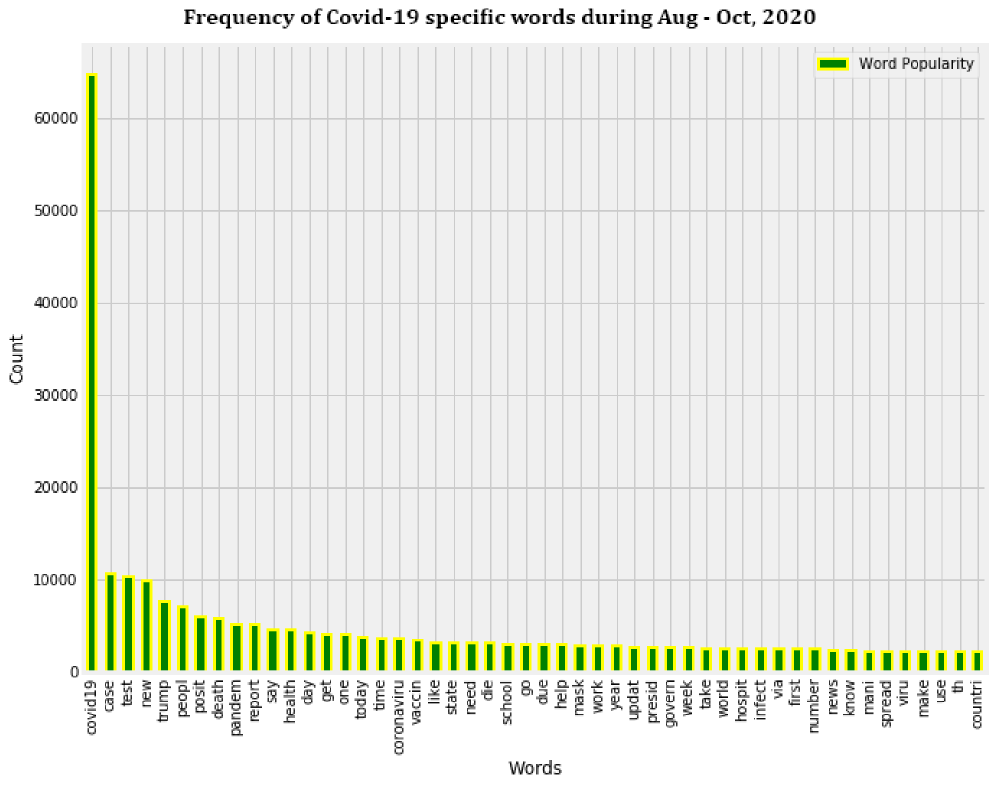


Figure 11. Graphical representation of the popularity for most frequent Covid-19 exclusive words.

6.5 Sentiment Analysis

Nowadays deep learning-based sentiment analysis become more popular for extracting information and text prediction to analyze public behaviors about any event. To measure the trend of public opinions we often use Sentiment Analysis, a specific type of Data Mining through Natural Language Processing (NLP), computational linguistics and text analysis. The subjective information from the social media is analyzed and extracted to classify the text in multiple classes like positive, negative and neutral.

A Sentiment Analyzer is a tool to implement and facilitate Sentiment Analysis tasks using NLTK features and classifiers, especially for teaching and demonstrative purposes. Since both of the datasets were developed by collecting raw tweets without having any sentiment label, the sentiment analysis was required to understand the overall distribution of tweets from different sentiment classes. Here the sentiment polarity of each cleaned and pre-processed tweet was evaluated using the NLTK based Sentiment Analyzer to get the sentiment scores for positive, negative and neutral categories and calculate the compound sentiment scores of each tweet.

6.5.1 NLTK-based Sentiment Classification

The compound scores are computed by summing the valence scores of each word in the lexicon, adjusted according to the rules. Then the valence scores normalized to be between -1 (most extreme negative) and +1 (most extreme positive) for producing a single unidimensional sentiment measure for a given sentence. The more the compound score tends to +1, the more the sentence will be positively sensitive and the more it moves towards -1, the more the sentence will be negatively sensitive. We classified the tweets on the basis of the compound sentiment scores into three different classes i.e., Positive, Negative and Neutral. Then the sentiment polarity ratings were assigned for each tweet based on the algorithm presented in Table 4.

Table 4. Algorithm used for sentiment classification of Covid-19 tweets.

Algorithm Sentiment Classification of Tweets (compound, sentiment):

```
1.  for each k in range (0, len(tweet.index)):
2.     if tweet_k[compound] < 0:
3.        tweet_k[sentiment] ← 0.0       # assigned 0.0 for Negative Tweets
4.     elif tweet_k[compound] > 0:
5.        tweet_k[sentiment] ← 1.0       # assigned 1.0 for Positive Tweets
6.     else:
7.        tweet_k[sentiment] ← 0.5       # assigned 0.5 for Neutral Tweets
8.  end
```

In Figure 12, we present the sentiment distribution on the basis of the compound sentiment scores along with the overall percentage for the appearance of positive, negative and neutral tweets in both first and second phase datasets.

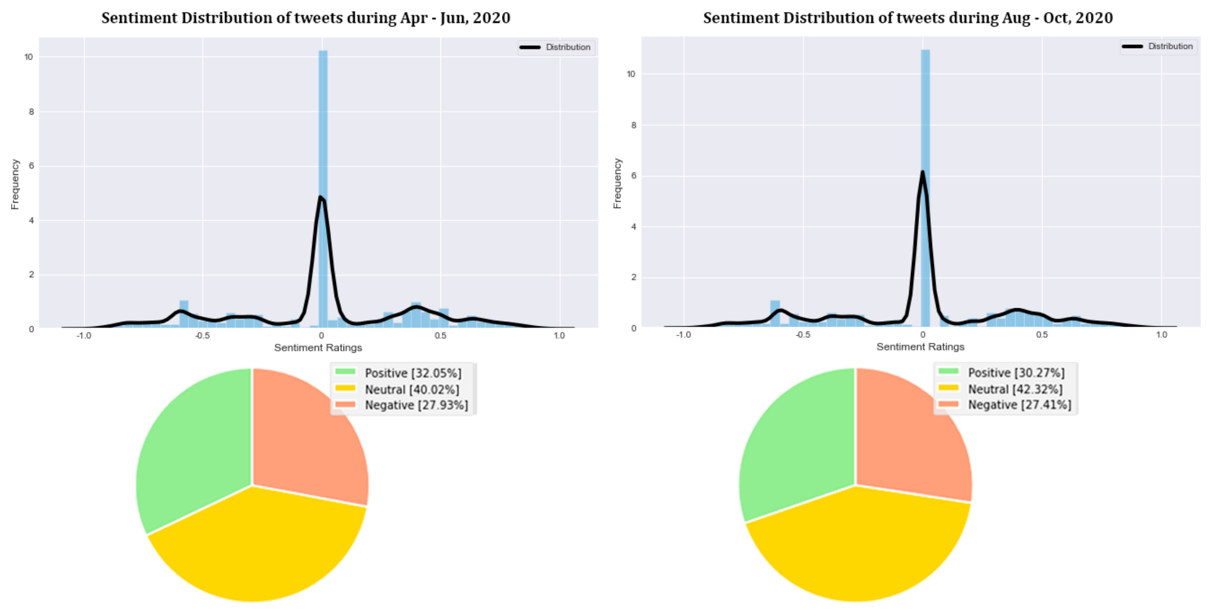


Figure 12. Sentiment distribution of three class polarity along with the percentage of Covid-19 tweets occurred from each class.

6.6 Naïve Bayes Sentiment Model Development

In machine learning, Naïve Bayes classification is a straightforward and powerful algorithm for the classification task. Naïve Bayes classification is based on applying Bayes' theorem with strong independence assumption between the features. Naïve Bayes classification produces good results when we use it for textual data analysis such as Natural Language Processing.

Naïve Bayes models are also known as simple Bayes or independent Bayes. All these names refer to the application of Bayes' theorem in the classifier's decision rule. Naïve Bayes classifier applies the Bayes' theorem in practice. This classifier brings the power of Bayes' theorem to machine learning.

6.6.1 Naïve Bayes Algorithm

Naïve Bayes Classifier uses the Bayes' theorem to predict membership probabilities for each class such as the probability that given record or data point belongs to a particular class. The class with the highest probability is considered as the most likely class. This is also known as the **Maximum A Posteriori (MAP)**.

The **MAP for a hypothesis with 2 events A and B is**

$$\left.\begin{aligned} MAP\ (A) &= max\ (P\ (A\ |\ B)) \\ &= max\ (P\ (B\ |\ A)\ *\ P\ (A))/P\ (B) \\ &= max\ (P\ (B\ |\ A)\ *\ P\ (A)) \end{aligned}\right\} \tag{8}$$

Here, $P(B)$ is evidence probability. It is used to normalize the result. It remains the same, So, removing it would not affect the result. Naïve Bayes Classifier assumes that all the features are unrelated to each other. Presence or absence of a feature does not influence the presence or absence of any other feature.

In real world datasets, we test a hypothesis given multiple evidence on features. So, the calculations become quite complicated. To simplify the work, the feature independence approach is used to uncouple multiple evidence and treat each as an independent one.

6.6.2 Types of Naïve Bayes Algorithm

There are three types of Naïve Bayes algorithm. The three types are listed below:

- ***Gaussian Naïve Bayes:*** When we have continuous attribute values, we made an assumption that the values associated with each class are distributed according to Gaussian or Normal distribution. For example, suppose the training data contains a continuous attribute x. We first segment the data by the class, and then compute the mean and variance of x in each class. Let μ_i be the mean of the values and let σ_i be the variance of the values associated with the ith class. Suppose we have some observation value x_i. Then, the probability distribution of x_i given a class can be computed by the following equation –

$$\boldsymbol{P(x_i|y_i) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(x_i-\mu_i)^2}{2\sigma_j^2}}} \tag{9}$$

- ***Multinomial Naïve Bayes:*** With a Multinomial Naïve Bayes model, samples (feature vectors) represent the frequencies with which certain events have been generated by a multinomial $(p_1, p_2, p_3 \ldots p_n)$ where pi is the probability that event i occurs. Multinomial Naïve Bayes algorithm is preferred to use on data that is multinomially distributed. It is one of the standard algorithms which is used in text categorization classification.

- ***Bernoulli Naïve Bayes:*** In the multivariate Bernoulli event model, features are independent Boolean variables (binary variables) describing inputs. Just like the multinomial model, this model is also popular for document classification tasks where binary term occurrence features are used rather than term frequencies.

6.6.3 Application of Naïve Bayes Algorithm

Naïve Bayes is one of the most straightforward and fast classification algorithms. It is well suited for large volume of data. It is successfully used in various applications such as –

- Spam filtering
- Text classification
- Sentiment analysis
- Recommender systems

It uses Bayes theorem of probability for prediction of unknown class.

6.6.4 Naïve Bayes Sentiment Analysis

The Naïve Bayes sentiment model was proposed based on normalized Naïve Bayes classifier for refining the polarity ratings for each tweet. Naïve Bayes classifier is a generative classifier that performs better than other baseline algorithms on textual fragments for classifying them based on their polarity ratings [3]. In the phase of feature extraction, we extracted the frequent unigrams, bigrams and trigrams from the N-gram model as well as the Covid-19 specific words from the Bag-of-Words model and our proposed model takes those extracted features as its inputs for further computation. By using the algorithm presented in Table 5, we further classify the tweets based on refined sentiment polarities for fine-grained classification of the tweets.

Table 5. Algorithm used for Naïve Bayes fine-grained sentiment classification of Covid-19 tweets.

Algorithm Fine-Grained Sentiment Classification (sentiment, refined_sentiment):

```
1.  for each i in range (0, len(tweet.index)):
2.      if $tweet_i$[sentiment] >= 0.0 and $tweet_i$[sentiment] < 0.25:
3.          $tweet_i$[refined_sentiment] $\leftarrow$ -1.0      # assigned -1.0 for Most Negative Tweets
4.      elif $tweet_i$[sentiment] >= 0.25 and $tweet_i$[sentiment] < 0.5:
5.          $tweet_i$[refined_sentiment] $\leftarrow$ -0.5      # assigned -0.5 for Negative Tweets
6.      elif $tweet_i$[sentiment] == 0.5:
7.          $tweet_i$[refined_sentiment] $\leftarrow$ 0.0       # assigned 0.0 for Neutral Tweets
8.      elif $tweet_i$[sentiment] > 0.5 and $tweet_i$[sentiment] <= 0.75:
9.          $tweet_i$[refined_sentiment] $\leftarrow$ 0.5       # assigned 0.5 for Positive Tweets
10.     else:
11.         $tweet_i$[refined_sentiment] $\leftarrow$ 1.0       # assigned 1.0 for Most Positive Tweets
12. end
```

As both of the large datasets consists almost 235k and 320k tweets, if we classified all the tweets in only three sentiment classes i.e., positive, negative and neutral then it would not have been a through observation of the dataset. The classifier refines the sentiment rating for the large corpora based on the extracted features.

6.6.5 Classification Report & Confusion Matrix

Classification report is another way to evaluate the classification model performance. It displays the precision, recall, f1 and support scores for the model.

- ***Precision:*** Precision can be defined as the percentage of correctly predicted positive outcomes out of all the predicted positive outcomes. It can be given as the ratio of true positives (TP) to the sum of true and false positives (TP + FP). So, Precision identifies the proportion of correctly predicted positive outcome. It is more concerned with the positive class than the negative class.
  Mathematically, precision can be defined as the ratio of TP to (TP + FP).

- ***Recall:*** Recall can be defined as the percentage of correctly predicted positive outcomes out of all the actual positive outcomes. It can be given as the ratio of true positives (TP) to the sum of true positives and false negatives (TP + FN). Recall is also called Sensitivity. Recall identifies the proportion of correctly predicted actual positives.
  Mathematically, recall can be given as the ratio of TP to (TP + FN).

- ***f1-score:*** f1-score is the weighted harmonic mean of precision and recall. The best possible f1-score would be 1.0 and the worst would be 0.0. f1-score is the harmonic mean of precision and recall. So, f1-score is always lower than accuracy measures as they embed precision and recall into their computation. The weighted average of f1-score should be used to compare classifier models, not global accuracy.

- ***Support:*** Support is the actual number of occurrences of the class in the dataset.

A confusion matrix is a tool for summarizing the performance of a classification algorithm. A confusion matrix will give us a clear picture of classification model performance and the types of errors produced by the model. It gives us a summary of correct and incorrect predictions broken down by each category. The summary is represented in a tabular form. Four types of outcomes are possible while evaluating a classification model performance. These four outcomes are described below –

- ***True Positives (TP):*** True Positives occur when we predict an observation belongs to a certain class and the observation actually belongs to that class.

- ***True Negatives (TN):*** True Negatives occur when we predict an observation does not belong to a certain class and the observation actually does not belong to that class.

- ***False Positives (FP):*** False Positives occur when we predict an observation belongs to a certain class but the observation actually does not belong to that class. This type of error is called *Type I error*.

- ***False Negatives (FN):*** False Negatives occur when we predict an observation does not belong to a certain class but the observation actually belongs to that class. This is a very serious error and it is called *Type II error*.

In this experiment we have considered the polar tweets from all positive and negative classes to evaluate the classification accuracy on sentiment prediction of the proposed Naïve Bayes model. Table 6 is representing the classification report to present the performance regarding the classification of tweets along with the different classes from both first and second phase datasets.

Table 6. Classification Report.

		Precision	**Recall**	**F1-Score**	**Support**
Covid-19 Dataset I (Apr – Jun, 2020)	*Positive (1.0)*	0.64	0.63	0.63	9245
	Negative (0.0)	0.58	0.59	0.58	8019
	Avg. / Total	**0.61**	**0.61**	**0.61**	**17264**
Covid-19 Dataset II (Aug – Oct, 2020)	*Positive (1.0)*	0.63	0.62	0.62	7333
	Negative (0.0)	0.58	0.59	0.59	6570
	Avg. / Total	**0.61**	**0.61**	**0.61**	**13903**

In Figure 13, the heatmaps for confusion matrix are presented to identify the differences between predicted and the actual tweets along with the different classes from both first and second phase datasets.

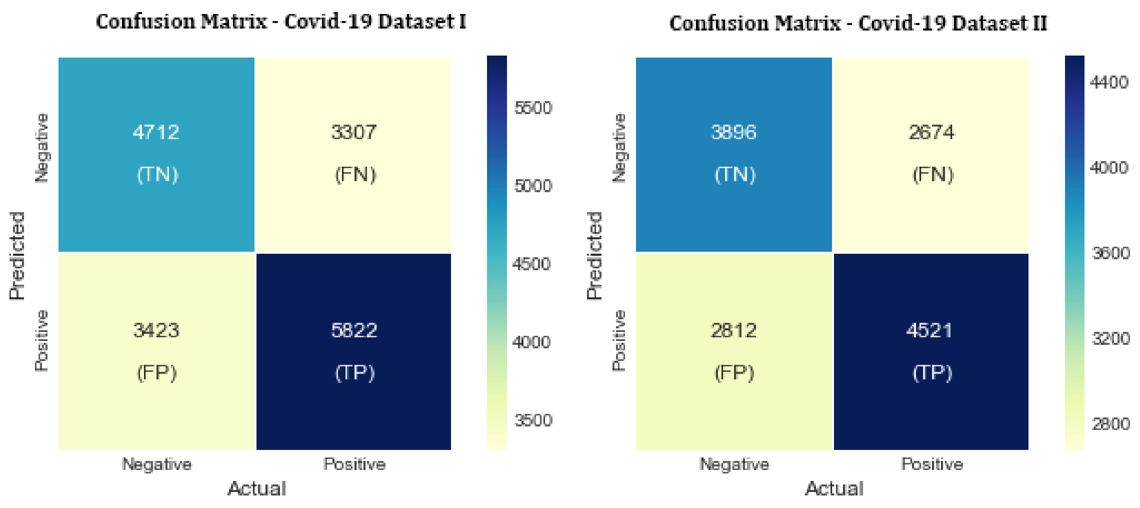


Figure 13. Confusion Matrix.

6.7 Global Sentiment Trend Analysis

6.7.1 Overall Average Sentiment Trend

The scale of the average sentiment scores was provided in the range from -1.0 to 1.0, such as Most Negative (-1.0), Negative (-0.5), Neutral (0.0), Positive (0.5), and Most Positive (1.0) respectively. Using these sentiment scores, a line plot graph was generated for date wise average sentiment trend from all over the world. From the graph it is clearly visible that there is a moderate contrast between the data from different time phases. In Figure 14, we present the global sentiment trend graph for both of the datasets.

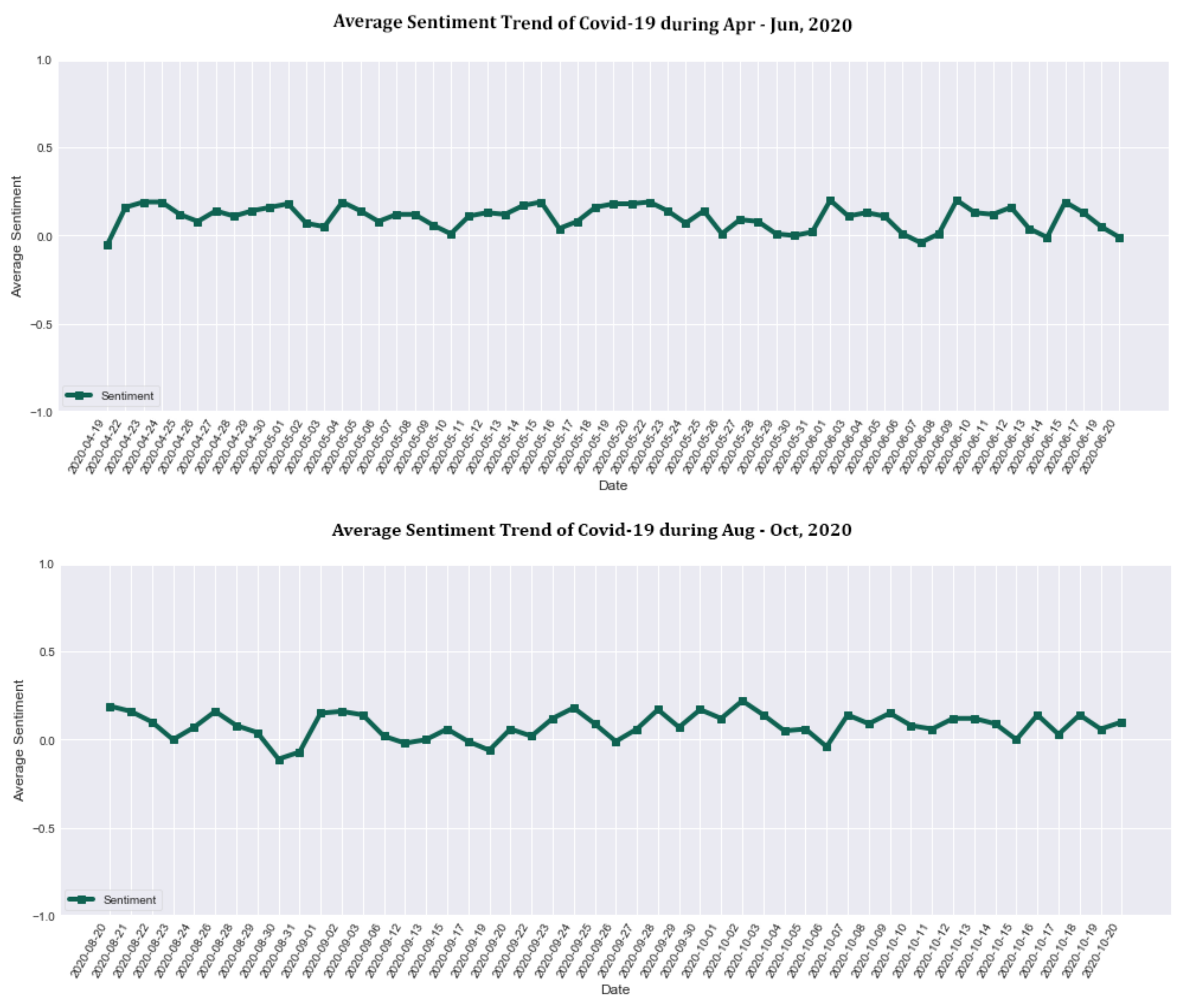


Figure 14. Graphical representation of the Overall Average Sentiment Trend throughout the world.

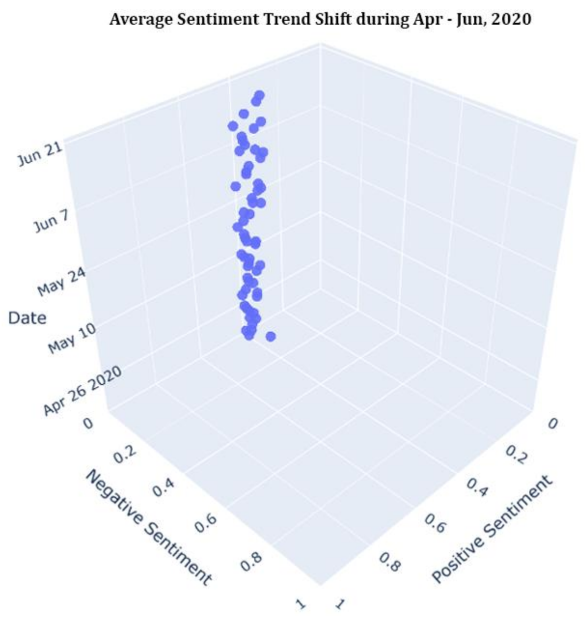
Here the certain drops in the average sentiment during the analysis period due to the massive impact of this pandemic were clearly noticeable. We present some tweets from each sentiment class along with their Naïve Bayes sentiment score in Table 7. From these tweets the various emotions can be identified which may cause the changes of world-wide sentiment trend.

Table 7. Some Covid-19 tweets from different sentiment classes.

Date	Original Tweet	Naïve Bayes Sentiment Rating	Refined Sentiment Rating	Sentiment Class
05-05-2020	**Barbara Walton #FBPE** @LesTroisChenes More Than 60 Doctors in Italy Have Died in COVID-19 Pandemic. Why? Lack of PPE and information. UK just a fraction behind them - far ahead of other countries. We threw our health worker under the #COVID bus and then clapped for them.	0.24	-1	Most Negative
27-04-2020	**Elizabeth Stein** @lizstein A tale of two cities. "More than fifteen thousand people in New York are believed to have died from covid-19. Last week in Washington State, the estimate was fewer than seven hundred people."	0.26	-0.5	Negative
10-05-2020	**Volker Stollorz** @Stollovo "Anyway, I remain a born optimist. And now that I have faced death, my tolerance levels for nonsense and bullshit have gone down even more than before. So, I continue calmly and enthusiastically, although more selectively than before my illness" #COVID19	0.50	0.0	Neutral
26-04-2020	**Muh'd** @smj_esq People are recovering from COVID-19 and it's a good development. I think Government should tell the public the drugs administered on the recovered patients. Just my own thoughts though!	0.73	0.5	Positive
07-05-2020	**CNN Philippines** @cnnphilippines There are now more than 400 health personnel who have recovered from the coronavirus disease, the Department of Health says.	0.10	1.0	Most Positive

6.7.2 Sentiment Trend Shift Detection

At the time of executing the sentiment trend analysis, we found that daily sentiment polarities changed continuously for positive and negative categories. So, the date wise average positive and negative polarities were calculated from both of the datasets to detect the polarity shift during the time phases Apr – Jun and Aug – Oct, 2020. The Identification of Sentiment Trend shift helps us to understand the changes in the daily polarity trend. Here we have presented the average sentiment trend shift. For the visualization of this sentiment trend shift we have used 3D scatter plot to present the date wise for both positive and negative aspects. In Figure 15, we present the graphical representation of the worldwide average sentiment trend shift.



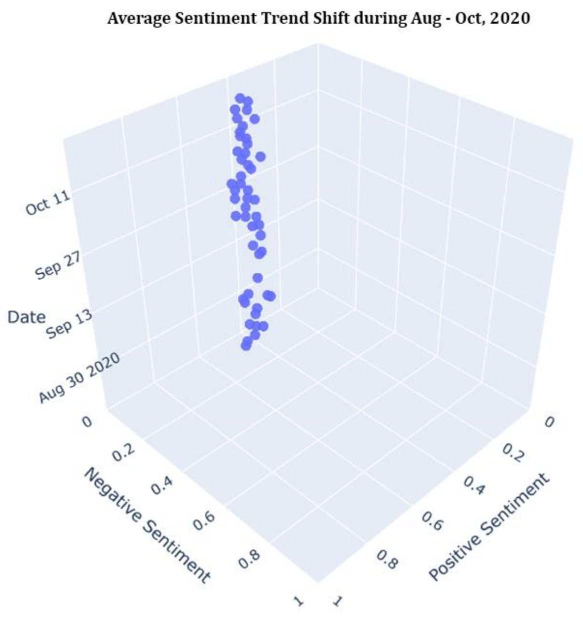


Figure 15. Graphical representation of the Average Sentiment Trend Shift throughout the world.

6.8 Sentiment Modelling using Hybrid Convolutional LSTM

In traditional textual sentiment analysis, LSTM (Long-Short Term Model) network has already been proven to be performing better than the similar neural models as this model provides most feasible solution for prediction task. As a special RNN structure this model can forecast the future prediction based on the various extracted features from the dataset. The data moves through the cell states in LSTM so that this network can accurately recollect or overlook things depending upon the hyperparameters associated with it. For the time series datasets, the information gathered over a progressive period of time frames and generally LSTM are proposed to be an efficient methodology to produce forecasts with these datasets. In LSTM, the memory cell c_t acts as an accumulator of the state information. The self-parameterized controlling gates can modify the LSTM cell. Every time for a new input, if the input gate i_t is on then it's information will be collected to the cell. The past cell information will be forgotten if the forget gate f_t is activated. The output gate o_t controls the flow of the information in the rest of the network by propagating the present cell output c_t to the final state h_t. The three gates and memory cell control the information flow throughout the network to trap the gradient in the cell and prevent them from disappearing too rapidly. The mathematical representation of the status of the gates and cell are presented using the following equations where W denotes the weight matrices, σ denotes the logistic sigmoid activation function and ‘ $\odot$ ’ denotes the Hadamard Product.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci} \odot c_{t-1} + b_i) \quad (10)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf} \odot c_{t-1} + b_f) \quad (11)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (12)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \odot c_t + b_o) \quad (13)$$

$$h_t = o_t \odot tanh(c_t) \quad (14)$$

Although for handling local dependencies between neighbor words, the LSTM has proven powerful but at the time of sentiment classification over large data it had to deal with excessive redundancy. We proposed a hybrid Convolutional LSTM network which takes word embeddings as its input and feeds them into *Convolutional* layers to extract local features. After that, output of the Convolutional model has been transmitted to LSTM network as input to learn long-term dependences between the sequence of words for sentiment classification. We exploit both Convo-Sequential and Convo-Bidirectional LSTM model for sentiment evaluation of the Covid-19 datasets. Figure 16 represent the detailed architecture of the proposed hybrid model.

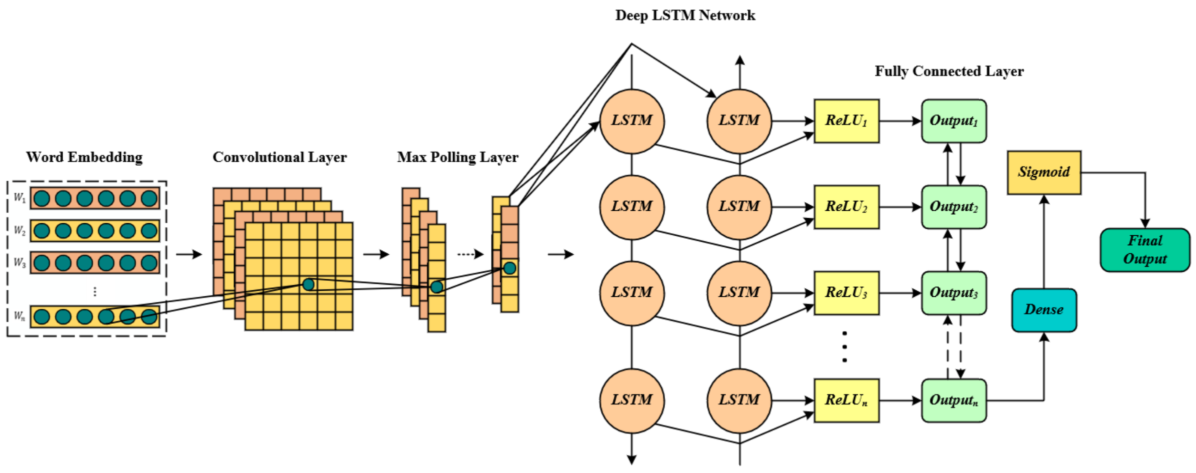


Figure 16. Graphical representation of proposed Hybrid Convolutional LSTM model.

We classified the *positive* and *negative* sentiment rated tweets based on the Naïve Bayes sentiment score assigned by the proposed model. We developed new dataset consisting of the cleaned and pre-processed tweets along with their corresponding *positive* (1.0) and *negative* (0.0) sentiments. Then we created two sets X and y for the cleaned tweets and their sentiment scores respectively and split the dataset into 80:20 ratio i.e., 80% for training (X_train, y_train) and 20% for validation (X_test, y_test) purposes respectively. A large number of Covid-19 exclusive words generated by this model from the new dataset. Then we converted these words into word vectors using *word2vec* by setting the vector dimension as 200 for each collected n-grams within a sentence and developed new *X_train*, *X_test* sets consisting with the calculated word vectors for further processing. From updated training set the word vectors and the respective sentiment scores fed in the model as the first layer of inputs.

In this experiment, the *TensorFlow* framework and *Keras* library were used to add *Sequential* and *Bidirectional* LSTM models with *Embedding*, *Convolutional*, *Max Pooling* and *Dense* layers. The models were trained for 6 epochs with two types of outline activation function with parameters, optimizer, loss and accuracy. The *Embedding* layer used to initialize the words to assign random weights and it learns the embedding to embed all words in the training dataset. In this layer with input length is same as the maximum length of a tweet is used to represent a particular weighted word from a sentence. The *Convolutional layer* with *ReLU* (Rectified Linear Unit) activation function is used to take word embeddings as its input from *Embedding* layer. At the end of this network layer, *Max Pooling* layer used to interpret all the n-gram features to generate the sentence representations. The *Dense* layer, also known as fully connected layer, is used to classify the extracted features of the *Convolutional* layers. Using dense layer, every current input (neuron) in the network layer is connected to every input in the proceeding layer of the network. We used *ReLU* activation function for the initial set of *Dense* layers with 128, 64, and 32 units respectively and *Sigmoid* activation function for the outermost final *Dense* layer with 2 units. The final *Dense* layer classifies the tweets within two sentiment classes for predicting the sentiment scores of each tweet. During the training we used 32 batches, 2 verbose and learning rate as 0.001 for both of the models. We used dropout technique because it prevents the models from overfitting by dropping the irrelevant information from the network which do not contribute in further processing to enhance the performance of the models. Table 8 represents the training and validation accuracy vs. loss using Convo-Sequential LSTM on both the datasets respectively.

Table 8. Training accuracy vs. loss, validation accuracy vs. loss using CNN + Seq-LSTM network.

	Epochs	**Train Loss**	**Train Accuracy**	**Val Loss**	**Val Accuracy**
Covid-19 Dataset I (Apr – Jun, 2020)	*Initially*	16.83%	93.05%	10.13%	96.34%
	2nd	06.65%	97.60%	10.65%	96.33%
	3rd	04.07%	98.59%	11.80%	95.78%
	4th	02.82%	98.98%	13.29%	95.74%
	5th	01.99%	99.32%	15.89%	95.37%
	6th	01.35%	**99.53%**	21.06%	**95.61%**
Covid-19 Dataset II (Aug – Oct, 2020)	*Initially*	18.59%	91.91%	11.26%	96.29%
	2nd	06.90%	97.57%	10.41%	95.99%
	3rd	04.16%	98.54%	12.64%	96.06%
	4th	02.67%	99.02%	11.62%	96.09%
	5th	01.84%	99.35%	16.73%	95.91%
	6th	01.38%	**99.53%**	17.05%	**95.53%**

After completion of the training of the Convo-Sequential LSTM model on Covid-19 Dataset I and Covid-19 Dataset II, finally we achieved 99.53% of overall training accuracy whereas 95.61% and 95.53% of validation accuracy on the testing data respectively. Table 9 represents the training accuracy vs. loss and validation accuracy vs. loss using Convo-Bidirectional LSTM respectively.

Table 9. Training accuracy vs. loss, validation accuracy vs. loss using CNN + Bi-LSTM network.

	Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Covid-19 Dataset I (Apr – Jun, 2020)	*Initially*	18.20%	92.55%	10.51%	96.20%
	2nd	07.18%	97.42%	10.53%	95.88%
	3rd	04.20%	98.51%	11.20%	96.13%
	4th	02.79%	99.03%	12.94%	95.93%
	5th	01.94%	99.31%	14.06%	95.75%
	6th	01.44%	**99.51%**	19.08%	**95.81%**
Covid-19 Dataset II (Aug – Oct, 2020)	*Initially*	18.38%	92.18%	09.93%	96.65%
	2nd	06.76%	97.63%	11.98%	96.19%
	3rd	03.92%	98.65%	10.50%	96.38%
	4th	02.63%	99.07%	12.11%	95.87%
	5th	01.72%	99.44%	14.00%	95.84%
	6th	01.39%	**99.50%**	14.34%	**95.75%**

After completion of the training of the Convo-Bidirectional LSTM model on Covid-19 Dataset I and Covid-19 Dataset II, finally we achieved 99.51% and 99.50% of overall training accuracy whereas 95.81% and 95.75% of validation accuracy on the testing data respectively. We found that Convo-Bidirectional LSTM network performs better than Convo-Sequential LSTM network on both of the datasets because they can learn each token from the sequence based on both the past and the future context of the token.

In Figure 17 and Figure 18, we present the percentages of training accuracy vs. loss and validation accuracy vs. loss, achieved by the Convo-Sequential and Convo-Bidirectional LSTM models on Covid-19 Dataset I and Covid-19 Dataset II during compilation respectively. From the both the figure it is evident that there has been a significant loss difference between the training and testing epochs. This indicates a slight overfitting of the data which can be postulated from the several tweet collection parameters differing from time to time in the tweets streaming phase.

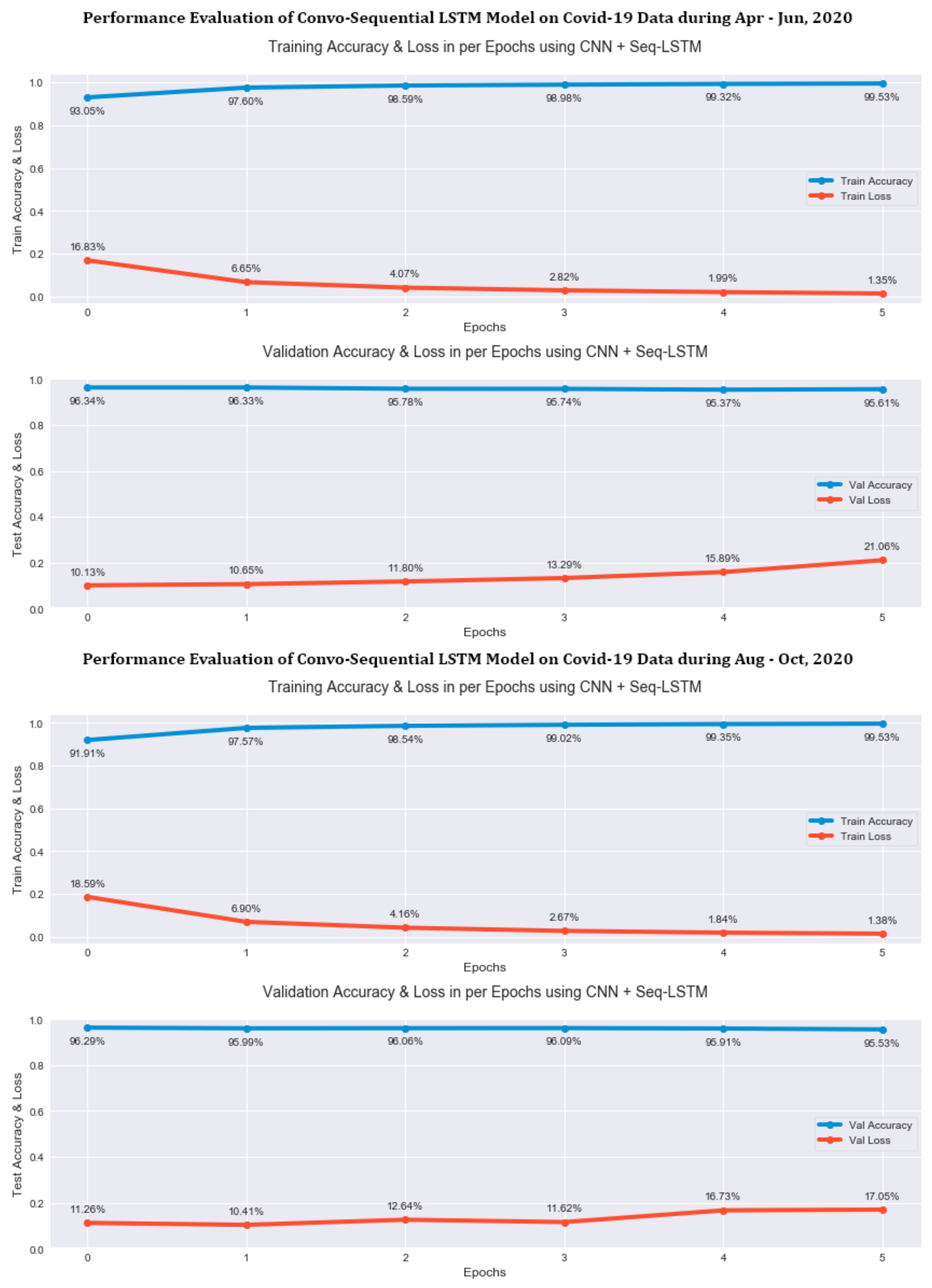


Figure 17. Performance metrics from the training loss vs. accuracy and validation loss vs. accuracy by the proposed Convo-Sequential LSTM model.

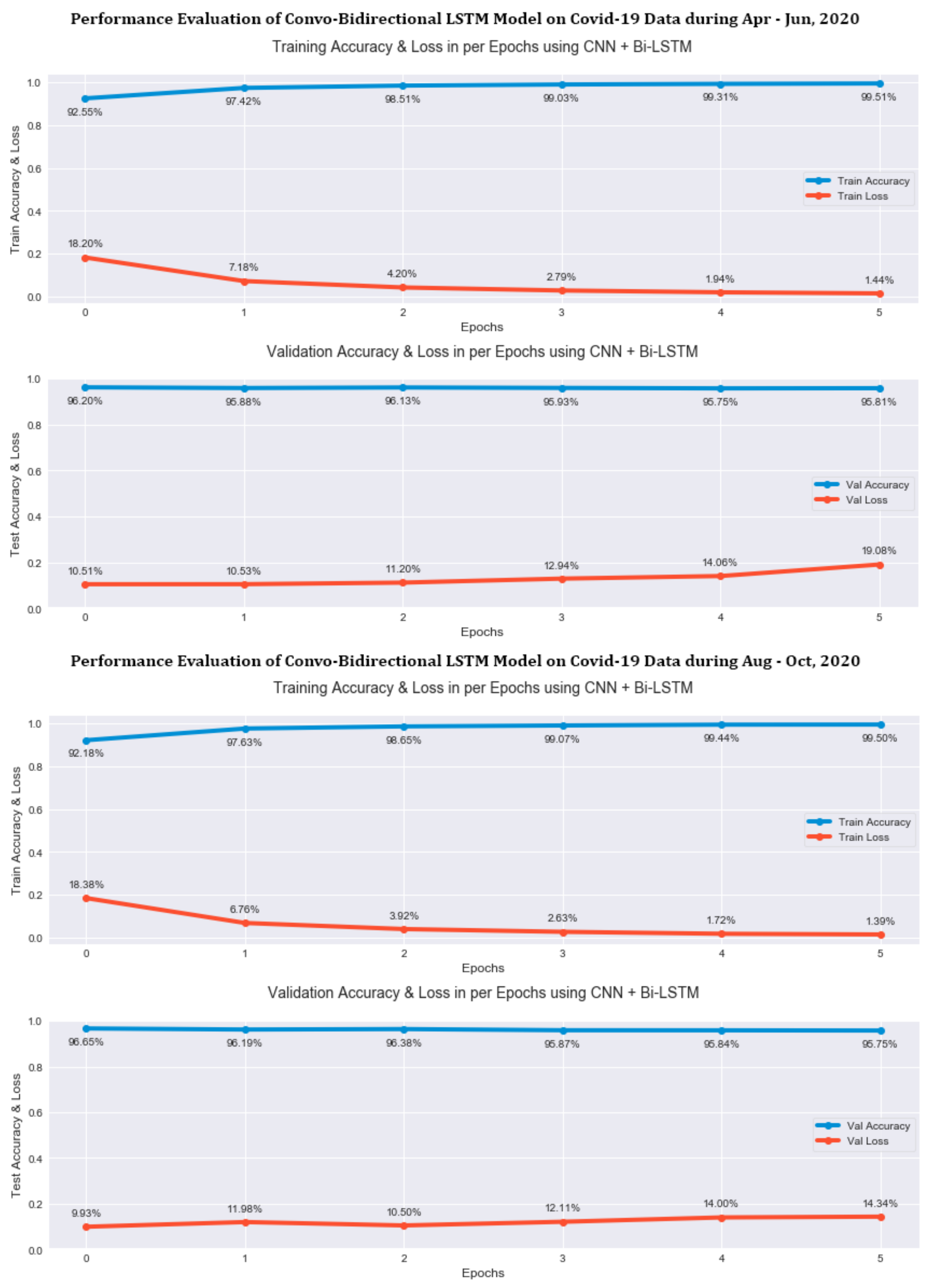


Figure 18. Performance metrics from the training loss vs. accuracy and validation loss vs. accuracy by the proposed Convo-Bidirectional LSTM model.

Chapter 7:

Comparative Performance Analysis

The comparative analysis is presented here to analyze the performance of the proposed deep Convolutional-LSTM models with different previous benchmark experiments on large scale Covid-19 data as well as several public corpora. Although, in our previous experiments we have already train these models on two different datasets collected during the phases Apr – Jun and Aug – Oct, 2020 and the models performed consistently well for sentiment prediction of tweets. Through this study we analyze the efficiency and the consistency of these models compared to the different State-of-Art approaches we found during the phase of literature survey.

7.1 Benchmark Comparison of NLP based Experiments on Covid-19

Here we demonstrated performance comparisons with some recent experiments on Covid-19 using different machine learning and deep neural network models along with any baseline algorithm(s). The main objective for this analysis to identify the comparative accuracy measurements on sentiment prediction achieved by other models with similar large-scale textual data. In Table 10, we present some comparisons of prediction accuracy achieved by various models we identified throughout the survey.

Table 10. Performance comparison between different ML or DNN models on large-scale Covid-19 data.

Machine Learning or Deep Neural Network Models	Overall Sentiment Prediction Accuracy
CNN + Bi-LSTM w. Naïve Bayes Sentiment Model	**95.81%**
Multilingual BERT on single-level classification [47]	95.62%
CNN + Seq-LSTM w. Naïve Bayes Sentiment Model	**95.61%**
CNN w. *GloVe* Embeddings [35]	90.67%
Bayesian Regression w. Tf-Idf [45]	89.40%
ERNIE on Chinese Weibo message [46]	88.00%
Bert on Chinese Weibo message [46]	83.00%
Random Forest [45]	82.20%
LDA + Deep LSTM [38]	81.15%
Logistic Regression w. trigrams + Tf-Idf [41]	81.00%
SVM w. Gaussian Membership based Fuzzy logic [41]	79.00%
LSTM on Chinese Weibo message [46]	78.00%
Deep RNN Model [42]	76.71%

From the analysis, we found that our proposed deep CNN + Bi-LSTM neural network performs better than different State-of-Art machine learning and deep neural models. But the deep CNN + Seq-LSTM neural network achieved slightly lower accuracy on validation data than a similar model. However different datasets with several parameters have been used for many experiments selected for this comparative analysis. The performance of these ML or DNN models may vary to different types of the dataset with multiple parameters. The Multilingual BERT model on single-level classification [47] achieved slight better accuracy than our proposed Convo-Sequential LSTM model. The proposed Convo-Bidirectional network with Naïve Bayes Sentiment Model achieved the highest sentiment prediction accuracy in this detailed comparative survey.

7.2 Comparative Analysis of State-of-Art Experiments on IMDB Dataset

Stanford large movie review dataset, also known as IMDB is an open-source public corpus which is widely used for sentiment analysis. Here we demonstrate performance comparisons with some previous experiments on IMDB review sentiment analysis. The main intention behind this analysis to identify the comparative accuracy measurements achieved by other deep learning models on IMDB movie review dataset. Table 11 represents some comparisons of prediction accuracy achieved from various benchmark experiments. From the analysis we found that both of our proposed hybrid Convolutional LSTM neural networks based on the Naïve Bayes Sentiment model performs better than different State-of-Art deep neural models.

Table 11. Performance comparison between different deep neural network models on IMDB dataset.

Deep Neural Network Models	**Overall Sentiment Prediction Accuracy**
CNN + Bi-LSTM w. Naïve Bayes Sentiment Model	**90.44%**
CNN + Seq-LSTM w. Naïve Bayes Sentiment Model	**90.26%**
Ensemble LSTM + CNN [53]	90.00%
CNN + LSTM w. Combined Kernels [49]	89.50%
CNN [53]	89.30%
CNN - LSTM [51]	89.20%
LSTM [53]	89.00%
CNN + LSTM w. Vanilla or Multiword Pre-processing [52]	88.90%
CNN w. Multiword Pre-processing [52]	87.90%
CNN [51]	87.70%
MLP [51]	86.74%
Vanilla Neural Network [50]	86.67%
LSTM [51]	86.64%
LSTM w. Tuning and Dropout [54]	86.50%
SA-LSTM w. Joint Training [54]	85.30%
Recursive RNN [48]	83.88%

7.3 Comparisons with Benchmark Experiments on Other Public Corpora

We have also compared the performance of our models with previous State-of-Art experiments on different open-source public corpora. For this comparative analysis we considered Amazon customer review and Stanford Sentiment Treebank (SST) datasets. In Table 12, we present the performance comparisons between previous benchmark experiments and our proposed models. From this comparison we can conclude that the proposed models achieved consistent validation accuracy over these datasets for sentiment prediction.

Table 12. Performance comparison with benchmark experiments on public corpora.

Public open-source corpora	**Previous Benchmark results**	**Our results**	
		CNN + Seq-LSTM	**CNN + Bi-LSTM**
Amazon customer review dataset	90.00% [55]	**99.91%**	**99.92%**
Stanford Sentiment Treebank (SST) dataset	86.99% [56]	**90.07%**	**90.25%**

Chapter 8:

Conclusion & Future Scope

In this dissertation, we discussed various recent NLP researches on different large-scale event analysis. We proposed a novel experimental approach, mainly focused on Sentiment Analysis on Covid-19 tweets using deep hybrid network. We analyzed the popularity of group of words using n-gram model and extracted the mostly popular Covid-19 specific words as two main features of the datasets. However, later the Naïve Bayes sentiment model was developed to assign the sentiment scores to the tweets based on their sentiment polarities calculated by NLTK based sentiment analyzer and classify all tweets into positive and negative classes based on their assigned sentiment ratings. Then, using this classified dataset containing the cleaned and pre-processed tweets and their sentiment ratings i.e., 1.0 for positive and 0.0 for negative, we trained the Deep Learning based hybrid LSTM models. The dataset was divided into 80:20 ratio i.e., 80% for training and 20% for testing purposes. With the first and second phase Covid-19 datasets, we trained the proposed Convo-Sequential LSTM and Convo-Bidirectional LSTM models for 6 epochs on certain parameters. Finally, we achieved 95.61% and 95.81% of validation accuracy respectively for first phase dataset whereas on the second phase dataset these models obtained the validation accuracy as 95.53% and 95.75% respectively.

The second wave of coronavirus infection is considered more dangerous than the first because it has already spread around the world. Many countries have taken different strict measures to stop the spread of the new strain of coronavirus. Analyzing this situation, we hope that we will see a lot of such research work shortly as there has been tireless research based on Covid-19 over the last year and a half. We will also continue our relentless efforts to improve this research work. For the future work, we want to extend this work by extracting the tweets from the datasets for which the Convo-Bidirectional LSTM model performs better than Convo-Sequential LSTM model. We will use fine-grained classified tweets to train the hybrid deep learning models to ensure the versatility of our proposed system. Finally, since our work is one of the novel works on such a large-scale pandemic data, we are keen to publish our Covid-19 data with all the components on an open-source data repository platform like Github[15] in near future, so that other researchers feel free to experiment with our findings from this event, and they can compare their achieved results with that of ours.

[15] https://github.com/ArunavaKumar

Chapter 9:

References

[1] Tuli, S., Tuli, S., Tuli, R. and Gill, S.S., 2020. Predicting the Growth and Trend of COVID-19 Pandemic using Machine Learning and Cloud Computing. *Internet of Things*, p.100222.

[2] Dubey, A.D., 2020. Twitter Sentiment Analysis during COVID19 Outbreak. *Available at SSRN 3572023*.

[3] Das, S. and Kolya, A.K., 2017, November. Sense GST: Text mining & sentiment analysis of GST tweets by Naive Bayes algorithm. In 2017 Third International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN) (pp. 239-244). IEEE.

[4] Bhatia, P., Ji, Y. and Eisenstein, J., 2015. Better document-level sentiment analysis from rst discourse parsing. *arXiv preprint arXiv:1509.01599*.

[5] Araujo, M., Reis, J., Pereira, A. and Benevenuto, F., 2016, April. An evaluation of machine translation for multilingual sentence-level sentiment analysis. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing* (pp. 1140-1145).

[6] Jurafsky, D., 2000. *Speech & language processing*. Pearson Education India.

[7] Xia, R., Xu, F., Yu, J., Qi, Y. and Cambria, E., 2016. Polarity shift detection, elimination and ensemble: A three-stage model for document-level sentiment analysis. *Information Processing & Management*, *52*(1), pp.36-45.

[8] Mubarok, M.S., Adiwijaya and Aldhi, M.D., 2017, August. Aspect-based sentiment analysis to review products using Naïve Bayes. In *AIP Conference Proceedings* (Vol. 1867, No. 1, p. 020060). AIP Publishing LLC.

[9] Ma, B., Yuan, H. and Wu, Y., 2017. Exploring performance of clustering methods on document sentiment analysis. *Journal of Information Science*, *43*(1), pp.54-74.

[10] Peng, H., Cambria, E. and Zou, X., 2017, May. Radical-based hierarchical embeddings for chinese sentiment analysis at sentence level. In *The Thirtieth International Flairs Conference*.

[11] Ma, Y., Peng, H., Khan, T., Cambria, E. and Hussain, A., 2018. Sentic LSTM: a hybrid network for targeted aspect-based sentiment analysis. *Cognitive Computation*, *10*(4), pp.639-650.

[12] Rintyarna, B.S., Sarno, R. and Fatichah, C., 2019. Evaluating the performance of sentence level features and domain sensitive features of product reviews on supervised sentiment analysis tasks. *Journal of Big Data*, *6*(1), pp.1-19.

[13] Hu, X., Tang, L., Tang, J. and Liu, H., 2013, February. Exploiting social relations for sentiment analysis in microblogging. In *Proceedings of the sixth ACM international conference on Web search and data mining* (pp. 537-546).

[14] Lee, H., Han, Y., Kim, K. and Kim, K., 2014, November. Sentiment analysis on online social network using probability Model. In *Proceedings of the Sixth International Conference on Advances in Future Internet* (pp. 14-19).

[15] Fornacciari, P., Mordonini, M. and Tomaiuolo, M., 2015, September. Social network and sentiment analysis on Twitter: Towards a combined approach. In *KDWeb* (pp. 53-64).

[16] Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H. and Xu, B., 2016. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. *arXiv preprint arXiv:1611.06639*.

[17] Baecchi, C., Uricchio, T., Bertini, M. and Del Bimbo, A., 2016. A multimodal feature learning approach for sentiment analysis of social network multimedia. *Multimedia Tools and Applications*, *75*(5), pp.2507-2525.

[18] Baziotis, C., Pelekis, N. and Doulkeridis, C., 2017, August. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)* (pp. 747-754).

[19] Yang, Y. and Eisenstein, J., 2017. Overcoming language variation in sentiment analysis with social attention. *Transactions of the Association for Computational Linguistics*, *5*, pp.295-307.

[20] Nakisa, B., Rastgoo, M.N., Rakotonirainy, A., Maire, F. and Chandran, V., 2018. Long short term memory hyperparameter optimization for a neural network based emotion recognition framework. *IEEE Access*, *6*, pp.49325-49338.

[21] Gong, L. and Wang, H., 2018, July. When sentiment analysis meets social network: A holistic user behavior modeling in opinionated data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1455-1464).

[22] Dahal, B., Kumar, S.A. and Li, Z., 2019. Topic modeling and sentiment analysis of global climate change tweets. *Social Network Analysis and Mining*, *9*(1), pp.1-20.

[23] Dang, N.C., Moreno-García, M.N. and De la Prieta, F., 2020. Sentiment analysis based on deep learning: A comparative study. *Electronics*, *9*(3), p.483.

[24] Chauhan, P., Sharma, N. and Sikka, G., 2020. The emergence of social media data and sentiment analysis in election prediction. *Journal of Ambient Intelligence and Humanized Computing*, pp.1-27.

[25] Ji, X., Chun, S.A. and Geller, J., 2013, September. Monitoring public health concerns using twitter sentiment classifications. In *2013 IEEE International Conference on Healthcare Informatics* (pp. 335-344). IEEE.

[26] Coppersmith, G., Dredze, M. and Harman, C., 2014, June. Quantifying mental health signals in Twitter. In *Proceedings of the workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality* (pp. 51-60).

[27] Georgiou, D., MacFarlane, A. and Russell-Rose, T., 2015, July. Extracting sentiment from healthcare survey data: An evaluation of sentiment analysis tools. In *2015 Science and Information Conference (SAI)* (pp. 352-361). IEEE.

[28] Mowery, D.L., Park, Y.A., Bryan, C. and Conway, M., 2016, December. Towards automatically classifying depressive symptoms from Twitter data for population health. In *Proceedings of the Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media (PEOPLES)* (pp. 182-191).

[29] Zhang, Y., Chen, M., Huang, D., Wu, D. and Li, Y., 2017. iDoctor: Personalized and professionalized medical recommendations based on hybrid matrix factorization. *Future Generation Computer Systems*, *66*, pp.30-35.

[30] Vij, A. and Pruthi, J., 2018. An automated psychometric analyzer based on sentiment analysis and emotion recognition for healthcare. *Procedia computer science*, *132*, pp.1184-1191.

[31] Lee, J., Kim, J., Hong, Y.J., Piao, M., Byun, A., Song, H. and Lee, H.S., 2019. Health information technology trends in social media: using twitter data. *Healthcare informatics research*, *25*(2), p.99.

[32] Abualigah, L., Alfar, H.E., Shehab, M. and Hussein, A.M.A., 2020. Sentiment analysis in healthcare: a brief review. *Recent Advances in NLP: The Case of Arabic Language*, pp.129-141.

[33] Arora, P., Kumar, H. and Panigrahi, B.K., 2020. Prediction and analysis of COVID-19 positive cases using deep learning models: A descriptive case study of India. *Chaos, Solitons & Fractals*, *139*, p.110017.

[34] Arpaci, I., Alshehabi, S., Al-Emran, M., Khasawneh, M., Mahariq, I., Abdeljawad, T. and Hassanien, A.E., 2020. Analysis of twitter data using evolutionary clustering during the COVID-19 pandemic. *Computers, Materials & Continua*, *65*(1), pp.193-204.

[35] Das, S., Kolya, A.K.: Predicting the pandemic: sentiment evaluation and predictive analysis from large-scale tweets on Covid-19 by deep convolutional neural network. *Evolutionary Intelligence*. 1–22 (2021).

[36] Kabir, M. and Madria, S., 2020. Coronavis: A real-time covid-19 tweets analyzer. *arXiv preprint arXiv:2004.13932*.

[37] Samuel, J., Ali, G.G., Rahman, M., Esawi, E. and Samuel, Y., 2020. Covid-19 public sentiment insights and machine learning for tweets classification. *Information*, *11*(6), p.314.

[38] Jelodar, H., Wang, Y., Orji, R. and Huang, S., 2020. Deep sentiment classification and topic discovery on novel coronavirus or covid-19 online discussions: Nlp using lstm recurrent neural network approach. *IEEE Journal of Biomedical and Health Informatics*, *24*(10), pp.2733-2742.

[39] Chimmula, V.K.R. and Zhang, L., 2020. Time series forecasting of COVID-19 transmission in Canada using LSTM networks. *Chaos, Solitons & Fractals*, *135*, p.109864.

[40] Zheng, N., Du, S., Wang, J., Zhang, H., Cui, W., Kang, Z., Yang, T., Lou, B., Chi, Y., Long, H. and Ma, M., 2020. Predicting COVID-19 in China using hybrid AI model. *IEEE transactions on cybernetics*, *50*(7), pp.2891-2904.

[41] Chakraborty, K., Bhatia, S., Bhattacharyya, S., Platos, J., Bag, R. and Hassanien, A.E., 2020. Sentiment Analysis of COVID-19 tweets by Deep Learning Classifiers—A study to show how popularity is affecting accuracy in social media. *Applied Soft Computing*, *97*, p.106754.

[42] Al-Shaher, M.A., 2020. A hybrid deep learning and NLP based system to predict the spread of Covid-19 and unexpected side effects on people. *Periodicals of Engineering and Natural Sciences (PEN)*, *8*(4), pp.2232-2241.

[43] Shuja, J., Alanazi, E., Alasmary, W. and Alashaikh, A., 2020. Covid-19 open source data sets: A comprehensive survey. *Applied Intelligence*, pp.1-30.

[44] Lamsal, R., 2020. Design and analysis of a large-scale COVID-19 tweets dataset. *Applied Intelligence*, pp.1-15.

[45] Serrano, J.C.M., Papakyriakopoulos, O. and Hegelich, S., 2020, July. NLP-based feature extraction for the detection of COVID-19 misinformation videos on Youtube. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*.

[46] Yang, Q., Alamro, H., Albaradei, S., Salhi, A., Lv, X., Ma, C., Alshehri, M., Jaber, I., Tifratene, F., Wang, W. and Gojobori, T., 2020. Senwave: Monitoring the global sentiments under the covid-19 pandemic. *arXiv preprint arXiv:2006.10842*.

[47] Li, I., Li, Y., Li, T., Alvarez-Napagao, S., Garcia-Gasulla, D. and Suzumura, T., 2020, December. What Are We Depressed About When We Talk About COVID-19: Mental Health Analysis on Tweets Using Natural Language Processing. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence* (pp. 358-370). Springer, Cham.

[48] Timmaraju, A. and Khanna, V., 2015. Sentiment analysis on movie reviews using recursive and recurrent neural network architectures. *Semantic Scholar*, pp.1-5.

[49] Yenter, A. and Verma, A., 2017, October. Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis. In *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)* (pp. 540-546). IEEE.

[50] Shaukat, Z., Zulfiqar, A.A., Xiao, C., Azeem, M. and Mahmood, T., 2020. Sentiment analysis on IMDB using lexicon and neural networks. *SN Applied Sciences*, *2*(2), pp.1-10.

[51] Ali, N.M., Abd El Hamid, M.M. and Youssif, A., 2019. Sentiment analysis for movies reviews dataset using deep learning models. *International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol*, *9*.

[52] Camacho-Collados, J. and Pilehvar, M.T., 2017. On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis. *arXiv preprint arXiv:1707.01780*.

[53] Minaee, S., Azimi, E. and Abdolrashidi, A., 2019. Deep-sentiment: Sentiment analysis using ensemble of cnn and bi-lstm models. *arXiv preprint arXiv:1904.04206*.

[54] Dai, A.M. and Le, Q.V., 2015. Semi-supervised sequence learning. *arXiv preprint arXiv:1511.01432*.

[55] Guner, L., Coyne, E. and Smit, J., 2019. Sentiment analysis for Amazon. com reviews.

[56] Dong, X.L. and De Melo, G., 2018, July. A helping hand: Transfer learning for deep sentiment analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 2524-2534).

Appendix I:

Previous Work

Sentiment Analysis of Covid-19 Tweets using Evolutionary Classification-Based LSTM Model

Arunava Kumar Chakraborty[1,*], Sourav Das[2], Anup Kumar Kolya[1]

[1] Dept. of Computer Science & Engineering, RCC Institute of Information Technology, Beleghata, Kolkata - 700015, India
[2] Maulana Abul Kalam Azad University of Technology, WB, Salt Lake, Kolkata - 700064, India
{arunava95.akc, sourav.das.research, anup.kolya}@gmail.com

Abstract. As the Covid-19 outbreaks rapidly all over the world day by day and also affects the lives of million, a number of countries declared complete lockdown to check its intensity. During this lockdown period, social media platforms have played an important role to spread information about this pandemic across the world, as people used to express their feelings through the social networks. Considering this catastrophic situation, we developed an experimental approach to analyze the reactions of people on Twitter taking into account the popular words either directly or indirectly based on this pandemic. This paper represents the sentiment analysis on collected large number of tweets on Coronavirus or Covid-19. At first, we analyze the trend of public sentiment on the topics related to Covid-19 epidemic using an evolutionary classification followed by the n-gram analysis. Then we calculated the sentiment ratings on collected tweet based on their class. Finally, we trained the long-short term network using two types of rated tweets to predict sentiment on Covid-19 data and obtained an overall accuracy of 84.46%.

Keywords: Covid-19, Gram Selection, LSTM, Sentiment Analysis.

1 Introduction

On 31st December, 2019 the Covid-19 outbreak was first reported in the Wuhan, Hubei Province, China and it started spreading rapidly all over the world. Finally, WHO announced Covid-19 outbreak as pandemic on 11th March, 2020, when the virus continues to spread [1]. Starting from China, this virus infected and killed thousands of people from Italy, Spain, USA, UK, Brazil, Russia, and other many more countries as well. On 21st August 2020, more than 22.5 million cases of Covid-19 were reported in more than 188 countries and territories, yielding more than 7,92,000 deaths; although 14.4 million people have reported to be recovered. While this pandemic has continued to affect the lives of millions, many countries had enforced a strict lockdown for different periods to break the chain of this pandemic [1]. Since the Covid-19 vaccines are still yet to be discovered, therefore maintaining social distancing is the one and only one solution to check the spreading rate of this virus [2]. During the lockdown period a lot of people have chosen the Twitter to share their expression about this disease so we have been inspired to measure the human sensations about this epidemic by analyzing this huge Twitter data [3]. Initially, we have to face many challenges at the time of streaming the English tweets from the multilingual tweets all over the world as most of the peoples of foreign countries have used their native languages rather than English to express their feelings on social media [3]. However, we have developed our dataset considering the English tweets exclusively on Covid-19 of 160k tweets during April–May, 2020. We found the most popular words from the word corpus. Then we analyzed the trend of tweets

using n-gram model. Further we assigned sentiment scores to our preprocessed tweets based on their sentiment polarity and classified our dataset on basis of their sentiment scores. Finally, we used those tweets and their sentiment ratings to train our LSTM model.
The following sections are furnished as follows: In Section 2 we have described some previous related research works. The architecture of our dataset and proposed pre-processing approach presented in Section 3. The Section 4 consists of Feature A for identifying the Covid-19 specified words based on the word lexicon. In Section 5, we have described Feature B as the trend of tweet words using n-gram model. The evolutionary classification on the sentiment-rated tweets based on their sentiment polarity has given in Section 6. In Section 7 we trained our LSTM model based on the classified tweets including their sentiment ratings. Whereas the Section 8 concludes the future prospects of our research work.

2 Related Works

A machine learning and cloud computing-based Covid-19 prediction model has been developed on May, 2020 to predict the future trend of this epidemic. They mainly used probabilistic distribution functions like Gaussian, Beta, Fisher-Tippet, and Log Normal functions to predict the trend [1].

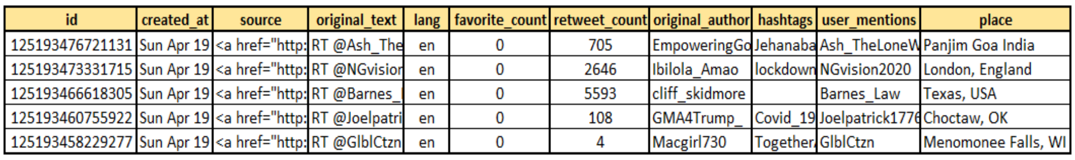
A Covid-19 trend prediction model has introduced on June, 2020 for predicting the number of COVID-19 positive cases in different states of India. The researchers mainly developed a LSTM-based prediction model as LSTM model performs better for time series predictions. They tested different LSTM variants such as stacked, convolutional, and Bidirectional LSTM on the historical data, and based on the absolute error they found that Bi-LSTM gives more accurate results over other LSTM models for short-term prediction [4].

On July, 2020 the evolutionary K-means clustering on twitter data related to Covid-19 has been done by some researchers. They analyzed the tweet patterns using n-gram model. As the result they observed the difference between the occurrences of n-grams from the dataset [5].

Another research work describes a deep LSTM architecture for Message-level and Topic-based sentiment analysis. The authors used LSTM networks augmented with two kinds of attention mechanisms, on top of word embeddings pre-trained on a big collection of Twitter messages [7].
A group of researchers developed LSTM hyperparameter optimization for neural network-based Emotion Recognition framework. In their experiment they found that optimizing LSTM hyperparameters significantly improve the recognition rate of four-quadrant dimensional emotions with a 14% increase in accuracy and the model based on optimized LSTM classifier achieved 77.68% accuracy by using the Differential Evolution algorithm [8].

3 Preparing Covid-19 Dataset

Since this Covid-19 epidemic has affected the entire world, we have collected worldwide Covid-19 related English tweets at a rate of 10k per day in between April 19 and May 20, 2020 to create our dataset of about 160k tweets. The dataset we developed contains the important information about most of the tweets as its attribute. The attributes of our dataset are id [Number], created_at [DateTime], source [Text], original_text [Text], favorite_count [Number], retweet_count [Number], original_author [Text], hashtags [Text], user_mentions [Text], place [Text]. Finally, we have collected 1,61,400 tweets containing the hash-tagged keywords like—*#covid-19*, *#coronavirus*, *#covid*, *#covaccine*, *#lockdown*, *#homequarantine*, *#quarantinecenter*, *#socialdistancing*, *#stayhome*, *#staysafe* etc. In Fig. 1 we have represented an overview of our dataset.



id	created_at	source	original_text	lang	favorite_count	retweet_count	original_author	hashtags	user_mentions	place
125193476721131	Sun Apr 19	<a href="http:	RT @Ash_The	en	0	705	EmpoweringGo	Jehanaba	Ash_TheLoneW	Panjim Goa India
125193473331715	Sun Apr 19	<a href="http:	RT @NGvision	en	0	2646	Ibilola_Amao	lockdown	NGvision2020	London, England
125193466618305	Sun Apr 19	<a href="http:	RT @Barnes_	en	0	5593	cliff_skidmore		Barnes_Law	Texas, USA
125193460755922	Sun Apr 19	<a href="http:	RT @Joelpatri	en	0	108	GMA4Trump_	Covid_19	Joelpatrick1776	Choctaw, OK
125193458229277	Sun Apr 19	<a href="http:	RT @GlblCtzn	en	0	4	Macgirl730	Together	GlblCtzn	Menomonee Falls, WI

Fig.1. Partial snapshot of the Covid-19 tweets corpus.

3.1 Data Pre-Processing

Data pre-processing is mainly used for cleaning the raw data by following certain steps to achieve the better result for further evaluations. We have done the pre-processing on our collected data by developing a user defined pre-processing function based on NLTK (Natural Language Toolkit, a Python library for NLP). Stemming helps to reduce inflected words to their word stem, base, or root form whereas by using Tokenization this function splits each of the sentence into smaller parts of word.

4 Feature A: Covid-19 Specified Words Identification

After pre-processing we have developed the Bag-of-Words (BOW) model using the frequently occurred words from the word lexicon and we obtained a list of most frequent Covid-19 exclusive words. We have represented a dense word-cloud in Fig. 2 of some of the mostly used words within the corpus.

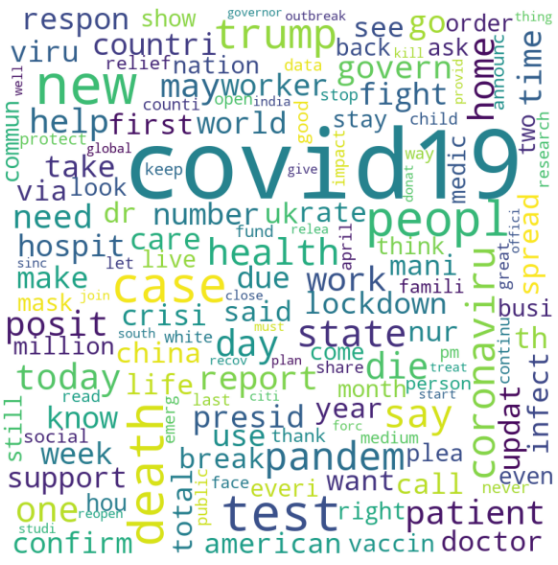


Fig.2. Some of the most popular Covid-19 related words from our corpus.

4.1 Word Popularity

Several words within the generated corpus have been found at different times in different positions of the tweets. Here we have counted the recurrence of each word and presented the top 50 popular words along with their popularity in Fig. 3.

After finding the word popularity, we have calculated the probability of repetition for each word on the basis of total 3,53,704 words from the corpus. Table 1 represents the popularity and probability scores of some most frequent words.

$$P(W_i) = \frac{\text{count}(W_i)}{\sum_{i=0}^{n} \text{count}(W_{i=0}^{n})} \tag{1}$$

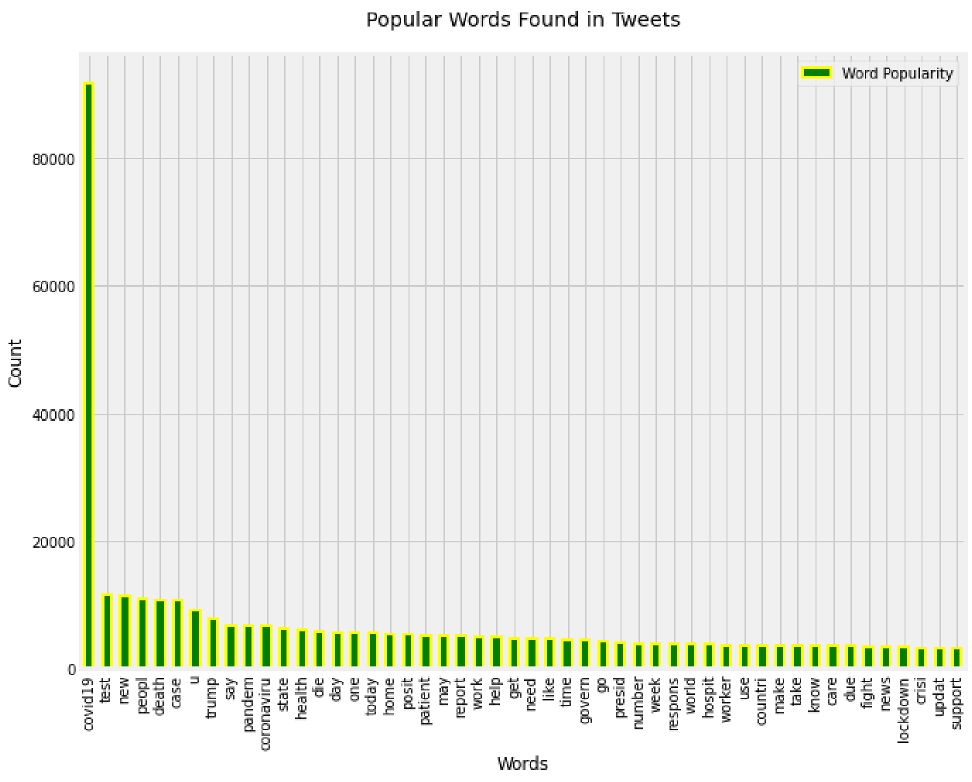


Fig.3. Graphical representation of the popularity for most frequent Covid-19 exclusive words.

Table 13. Popularity & probability of most frequent words.

	Words	Popularity	Probability
0	Covid19	91794	0.259522
1	Test	11663	0.032974
2	New	11305	0.031962
3	People	10834	0.030630
4	Death	10783	0.030486

5 Feature B: Word Popularity Detection using N-gram

Lexical n-gram models are widely used in Natural Language Processing for statistical analysis & syntax feature mapping. We developed n-gram model to analyze our generated corpus consisting of tokenized words for finding the popularity of words or group of adjacent words. Here the probability of the occurrence of a sequence can be calculated using probability chain rule:

$$P(x_1, x_2, x_3, \ldots x_n) = P(x_1)\, P(x_2 \mid x_1)\, P(x_3 \mid x_1, x_2) \ldots P(x_n \mid x_1, x_2, x_3, \ldots x_{(n-1)}) \tag{2}$$

$$= \prod_{i=1}^{n} P(x_i \mid x_1^{(i-1)}) \tag{3}$$

For example, we can consider a sentence as “Still Covid-19 wave is running”. Now as per the probability chain rule, P(“Still Covid19 wave is running”) = P(“Still”) x P(“Covid19” | “Still”) x P(“wave” | “Still Covid19”) x P(“is” | “Still Covid19 wave”) x P(“running” | “Still Covid19 wave is”).
The probabilities of words in each sentence after applying probability chain rule:

$$P(W_1, W_2, W_3, \ldots W_n) = \prod_j P(W_j \mid W_1, W_2, W_3, \ldots W_{(j-1)}) \tag{4}$$

$$= \prod_{j=1}^{n} P(W_j \mid W_1^{(j-1)}) \tag{5}$$

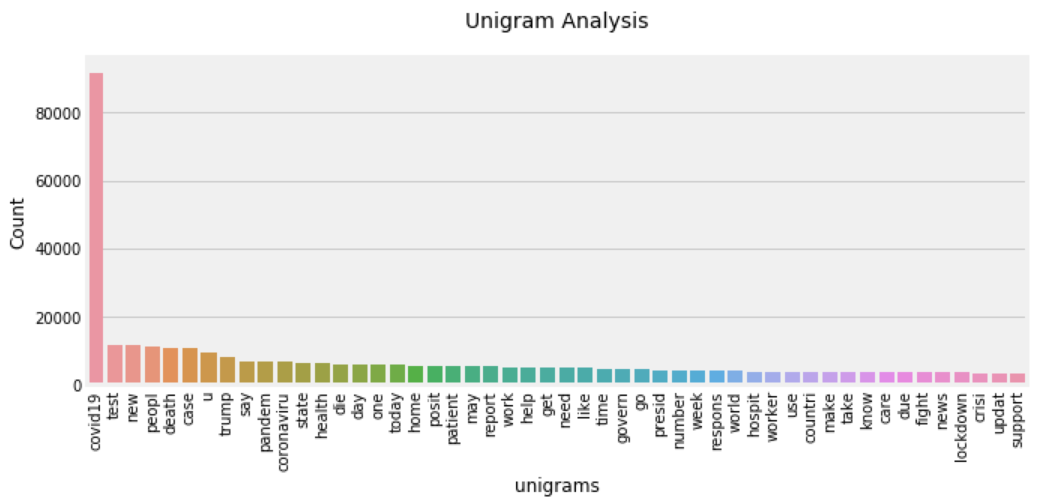


Fig.4. Graphical representation of the popularity for most frequent unigrams.

The bigram model estimates the probability of a word by using only the conditional probability $P(W_i|W_{i-1})$ of one preceding word on given condition of all the previous words $P(W_i|W_1^{i-1})$ [6].

$$P(W_1, W_2) = \prod_{i=2} P(W_2 \mid W_1) \tag{6}$$

The expression for the probability is -

$$P(W_k \mid W_{(k-1)}) = \frac{\text{count}\,(W_{(k-1)}, W_k)}{\text{count}\,(W_{(k-1)})} \tag{7}$$

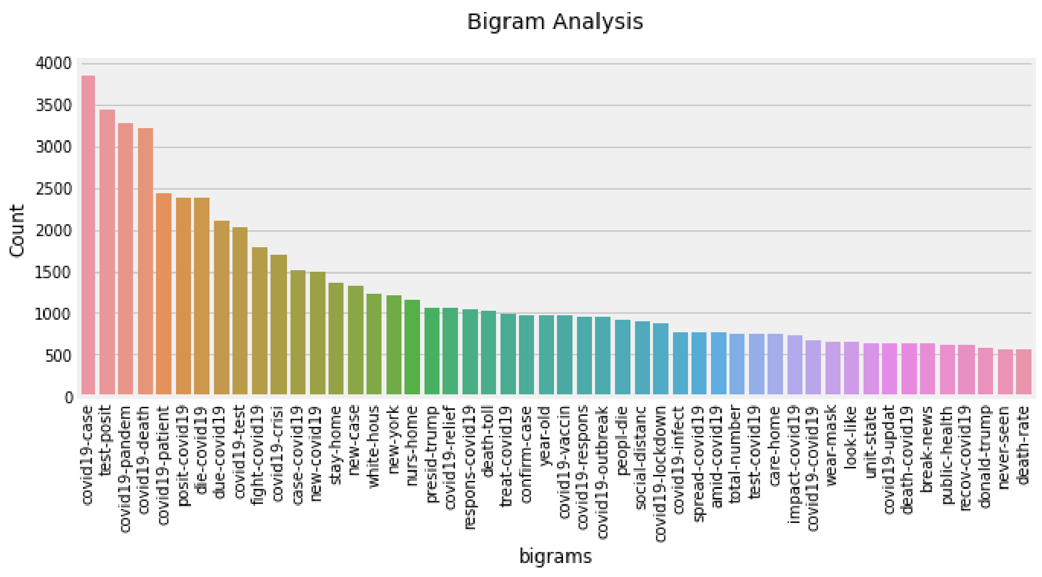


Fig.5. Graphical representation of the popularity for most frequent bigrams.

We have identified the most popular unigrams, bigrams, and trigrams within our corpus using the n-gram model. The graphical representations of 50 most popular unigrams, bigrams, and trigrams along with their popularity are presented in Fig. 4, Fig. 5 and Fig. 6 respectively.

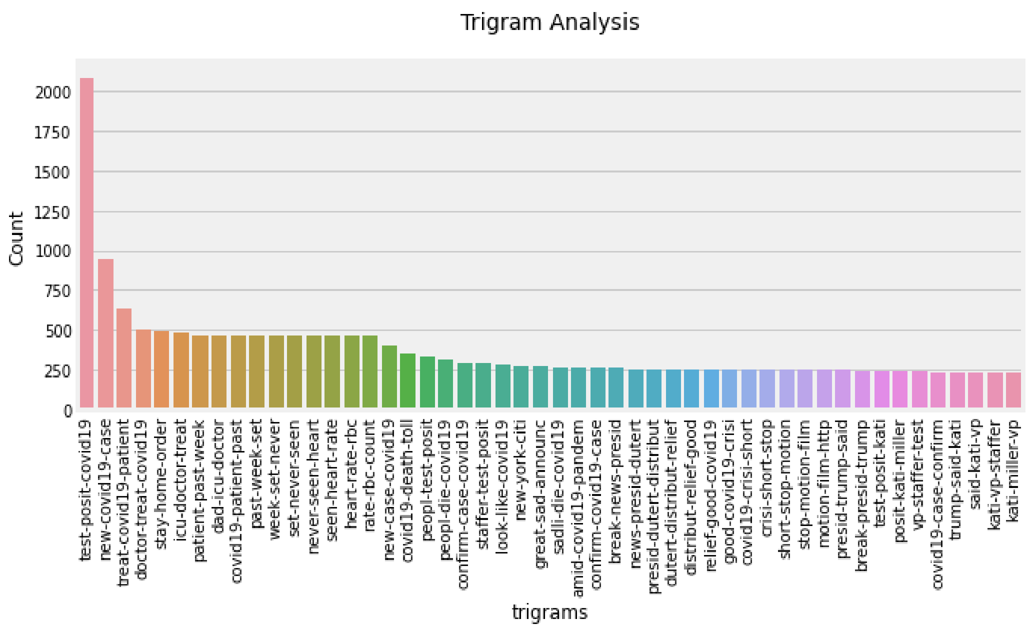


Fig.6. Graphical representation of the popularity for most frequent trigrams.

As the result of this analysis, we found that the popularity of trigrams is lesser than that of bigrams and the unigrams popularity is the highest according to this n-gram model.

6 Sentiment Analysis

To measure the trend of public opinions we use Sentiment Analysis, a specific type of Data Mining through Natural Language Processing (NLP), computational linguistics and text analysis. The subjective information from the social media are analyzed and extracted to classify the text in multiple classes like positive, negative, and neutral.

Here we calculated the sentiment polarity of each cleaned and preprocessed tweet using the NLTK-based Sentiment Analyzer and get the sentiment scores for *positive*, *negative* and *neutral* classes to calculate the *compound* sentiment score for each tweet.

6.1 Sentiment Classification

We have classified the tweets on the basis of the *compound* sentiments into three different classes, i.e. *Positive*, *Negative* and *Neutral*. Then we have assigned the sentiment polarity rating for each tweet based on the algorithm presented in Table 2.

Table 14. Algorithm used for sentiment classification of our Covid-19 tweets.

```
Algorithm Sentiment Classification of Tweets (compound, sentiment):
1. for each k in range (0, len(tweet.index)):
2.     if tweet$_k$[compound] $< 0$:
3.         tweet$_k$[sentiment] = 0.0     # assigned 0.0 for Negative Tweets
4.     elif tweet$_k$[compound] $> 0$:
5.         tweet$_k$[sentiment] = 1.0     # assigned 1.0 for Positive Tweets
6.     else:
7.         tweet$_k$[sentiment] = 0.5     # assigned 0.5 for Neutral Tweets
8. end
```

In Fig. 7, we have represented the sentiment classification with the overall percentage of each positive, negative, and neutral tweet found in the dataset. It can be visualized that the sentiment classes are naturally imbalanced as a large portion of social media users are either negative or neutral against the Covid-19 and the medical details associated with it.

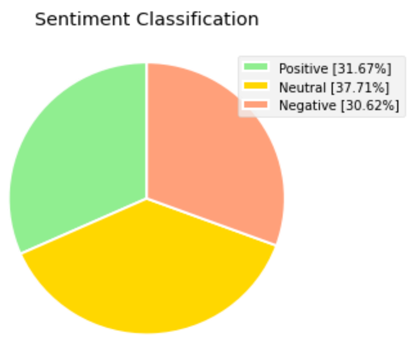


Fig. 719. Sentiment distribution of three class polarity along with the percentage of Covid-19 tweets occurred from each class.

7 Sentiment Modeling using Sequential LSTM

In traditional textual sentiment analysis, LSTM (Long-Short Term Model) network have already been proven to be performing better than the similar neural models [3]. We exploit a sequential LSTM model for sentiment evaluation of our Covid-19 dataset. We developed a new dataset consisting of the cleaned and preprocessed tweets along with their corresponding *positive* (1.0) and *negative* (0.0) sentiments. Then we created two sets *X* and *y* for the cleaned tweets and their sentiment scores, respectively and split the dataset into 80:20 ratio, i.e., 80% for training (*X_train*, *y_train*) and 20% for validation (*X_test*, *y_test*) purposes, respectively. A large number of Covid-19 exclusive words were generated by this model from the new dataset. Then we converted these words into word vectors using *word2vec* by setting the vector dimension as 200 for each collected n-grams within a sentence and developed new *X_train*, *X_test* sets consisting with the calculated word vectors for further processing. From updated training set the word vectors and the respective sentiment scores fed in the model as the first layer of inputs. In this experiment, we used *TensorFlow* framework and *keras* library to add *Sequential* LSTM model with *Dense* layers. We have trained our five-layered model for 30 epochs with two types of outline activation function with parameters, optimizer, loss, and accuracy. We have used *ReLU* (Rectified Linear Unit) activation function for the initial set of *Dense* layers with 128, 64, and 32 units, respectively and *Sigmoid* activation function for the outermost final *Dense* layer with 2 units. During the training we have used 32 batches and 2 verbose for our model. For some epochs Table 3 represents the training accuracy vs. loss and validation accuracy vs. loss, respectively.

Table 3. Training accuracy vs. loss, validation accuracy vs. loss within 30 epochs.

Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Initially	59.93%	67.63%	56.18%	70.45%
5th	42.71%	79.27%	43.75%	78.74%
10th	35.91%	83.38%	39.58%	81.71%
15th	30.43%	86.36%	39.28%	82.81%
20th	26.23%	88.40%	41.04%	83.47%
25th	22.44%	90.15%	41.96%	84.24%
30th	19.38%	**91.67%**	45.57%	**84.46%**

After completion of the training on our model, we have finally achieved 91.67% of overall training accuracy whereas the validation accuracy is 84.46% on the testing data. Table 4 and Table 5 are representing the confusion matrix and classification report to present the differences between predicted and the actual tweets along with the different classes.

Table 4. Confusion Matrix.

Actual	**Predicted**	
	Positive	**Negative**
Positive	8298 (TP)	1941 (FP)
Negative	1946 (FN)	7924 (TN)

Table 5. Classification Report.

	Precision	**Recall**	**F1-Score**	**Support**
Positive (1.0)	0.81	0.81	0.81	10239
Negative (0.0)	0.80	0.80	0.80	9870
Avg / Total	0.81	0.81	0.81	20109

In Fig. 8, we have plotted the percentages of training accuracy vs. loss and validation accuracy vs. loss, achieved by the Sequential LSTM model during compilation. From the figure it is evident that there has been a significant loss difference between the training and testing epochs. This indicates a slight overfitting of the data which can be postulated from the several tweet collection parameters differing from time to time in the tweets streaming phase.

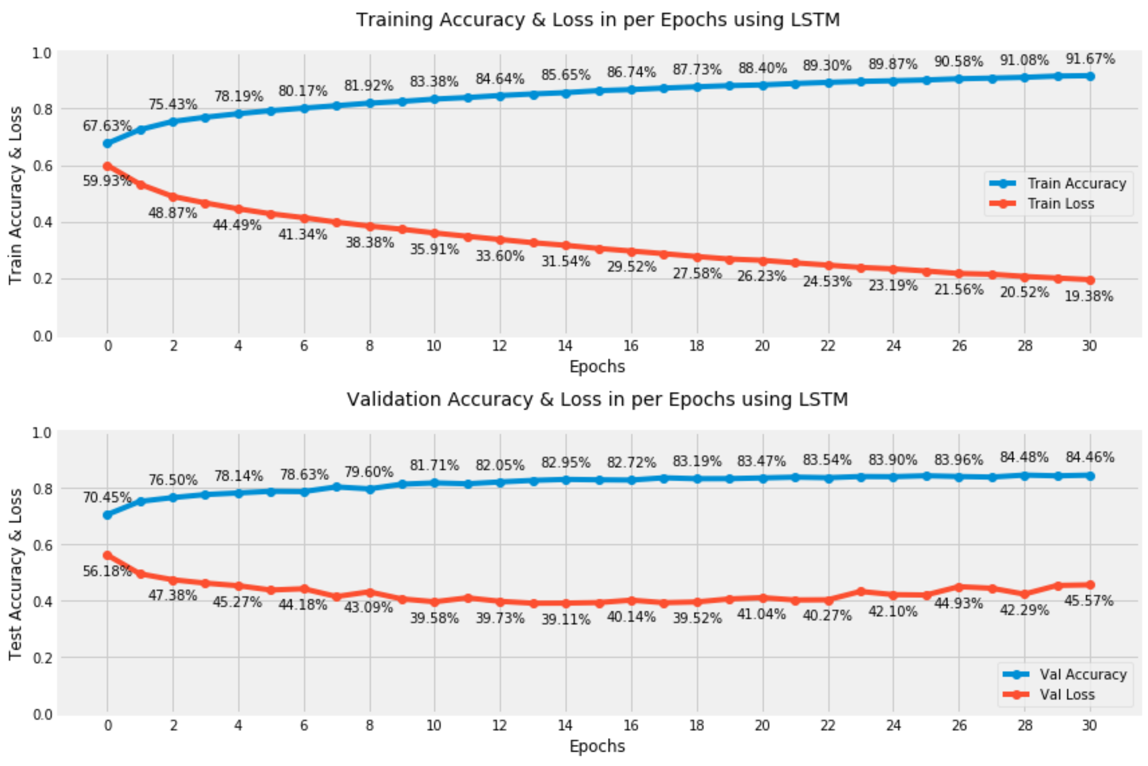


Fig.8. Performance metrics from the training loss vs. accuracy and validation loss vs. accuracy by the proposed model.

8 Conclusion & Future Scope

This experiment is mainly focused on Deep Learning-based Sentiment Analysis on Covid-19 tweets. We extracted the mostly popular words and analyzed the popularity of group of words using n-gram model as two main features of our dataset. However, later we developed a model to assign the sentiment ratings to the tweets based on their sentiment polarities calculated by sentiment analyzer and classify all tweets into *positive* and *negative* classes based on their assigned sentiment ratings. Then, using this classified dataset containing the cleaned and preprocessed tweets and their sentiment ratings, i.e., 1.0 for *positive* and 0.0 for *negative*, we trained our Deep Learning-based LSTM model. We divided the dataset into 80:20 ratio, i.e., 80% for training and 20% for testing purposes. After running 30 epochs on almost 93,474 parameters, we achieved validation accuracy as 84.46%.

For the future work, we want to develop a polarity-popularity model based on the features extracted during this experiment so that we can assign the refined sentiment ratings to the tweet based on the polarity of mostly recurred words [3]. With that data we will train the deep learning model to enhance the validation accuracy of our system.

References

1. Tuli, S., Tuli, S., Tuli, R., Gill, S.S.:. Predicting the growth and trend of COVID-19 pandemic using machine learning and cloud computing. Internet of Things, p. 100222 (2020)
2. Dubey, A.D.: Twitter sentiment analysis during COVID19 outbreak. Available at SSRN 3572023 (2020)
3. Das, S., Das, D., Kolya, A.K.: Sentiment classification with GST tweet data on LSTM based on polarity-popularity model. Sadhana. **45**(1) (2020)
4. Arora, P., Kumar, H., Panigrahi, B.K.: Prediction and analysis of COVID-19 positive cases using deep learning models: a descriptive case study of India. Chaos, Solitons Fractals **139**, 110017 (2020)
5. Arpaci, I.,Alshehabi, S.,Al-Emran, M.,Khasawneh, M.,Mahariq, I.,Abdeljawad, T.,Hassanien, A.E.: Analysis of twitter data using evolutionary clustering during the COVID-19 Pandemic. CMC-Comput. Mat. Cont. **65**(1), 193–203 (2020)
6. Jurafsky, D., 2000. *Speech & language processing*. Pearson Education India.
7. Baziotis, C., Pelekis, N., Doulkeridis, C.: Datastories at semeval-2017 task 4: deep lstm with attention for message-level and topic-based sentiment analysis. In Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017), pp. 747–754 (2017, August)
8. Nakisa, B., Rastgoo,M.N., Rakotonirainy, A.,Maire, F., Chandran, V.: Long short termmemory hyperparameter optimization for a neural network based emotion recognition framework. IEEE Access **6**, 49325–49338 (2018)

Appendix II:

Applications, Python Packages & Source Code

In this section we present A list of Applications & Python packages in alphabetical order that we have used for our thesis experiments, along with their respective versions. We have also included some source codes for the major experiments of this project work. The performance and effectiveness of the codes during the runtime may vary due to any changes in these versions of the following applications and python packages.

Applications

- Anaconda3 2019.07 (Python 3.7.3 64-bit)
- Nvidia CUDA Development 11.2
- Nvidia cuDNN 11.2 64-bit [For extraction only]

Python Packages

beautifulsoup4 == 4.7.1, bokeh == 1.2.0, dash == 1.19.0, en-core-web-sm == 3.0.0, gensim == 3.8.3, image == 1.5.33, keras == 2.4.3, Markdown == 3.3.4, matplotlib == 3.1.0, nltk == 3.4.4, numpy == 1.16.4, pandas == 1.0.3, plotly == 4.14.3, preprocessor == 1.1.3, seaborn == 0.9.0, scikit-learn == 0.24.1, sklearn == 0.0, spacy == 3.0.3, tensorflow == 2.4.0, tensorflow-gpu == 2.4.0, text2vec == 0.1.6, textblob == 0.15.3, tqdm == 4.58.0, tweepy == 3.10.0, tweet-preprocessor == 0.5.0, Unidecode == 1.2.0, vaderSentiment == 3.3.2, wordcloud == 1.8.1.

Source Code

1. <u>Live Tweet Collection from Twitter for preparing Covid-19 Dataset</u>

```
import os
import pandas as pd
import tweepy

#Twitter credentials for the app
consumer_key = '#########################'
consumer_secret = '##################################################'
access_key = '###################-##############################'
access_secret = '#############################################'

#pass twitter credentials to tweepy
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
```

```
auth.set_access_token(access_key, access_secret)
api = tweepy.API(auth)
tweets_data = "TweetsDB.csv"

#columns of the csv file
COLS = ['id', 'created_at', 'source', 'original_text', 'lang',
        'favorite_count', 'retweet_count', 'original_author', 'hashtags',
        'user_mentions', 'place']

#method write_tweets()
def write_tweets(keyword, file):
  if os.path.exists(file):
    df = pd.read_csv('TweetsDB.csv', header=0)
  else:
    df = pd.DataFrame(columns=COLS)
  for page in tweepy.Cursor(api.search, q=keyword, count=200, include_rts=False).pages(50):
    for status in page:
      new_entry = []
      status = status._json
      if status['lang'] != 'en':
        continue
      if status['created_at'] in df['created_at'].values:
        i = df.loc[df['created_at'] == status['created_at']].index[0]
        if status['favorite_count'] != df.at[i, 'favorite_count']
         or status['retweet_count'] != df.at[i, 'retweet_count']:
          df.at[i, 'favorite_count'] = status['favorite_count']
          df.at[i, 'retweet_count'] = status['retweet_count']
        continue
      print(status['id'])
      new_entry += [status['id'], status['created_at'], status['source'], status['text'], status['lang'],
                    status['favorite_count'], status['retweet_count']]
      new_entry.append(status['user']['screen_name'])
      hashtags = ", ".join([hashtag_item['text'] for hashtag_item
               in status['entities']['hashtags']])
      new_entry.append(hashtags)
      mentions = ", ".join([mention['screen_name'] for mention
               in status['entities']['user_mentions']])
      new_entry.append(mentions)
      try:
        location = status['user']['location']
      except TypeError:
        location = ''
      new_entry.append(location)
      single_tweet_df = pd.DataFrame([new_entry], columns=COLS)
      df = df.append(single_tweet_df, ignore_index=True)
```

```
            csvFile = open('TweetsDB.csv', 'a', encoding='utf-8')
    df.to_csv('TweetsDB.csv', mode='a', columns=COLS, index=False, encoding="utf-8")

#declare keywords as a query for three categories
tweet_keywords = '#Covid-19 OR #CoronaVirus OR #Covid OR #Covaccine OR #Covishield OR
                  #Phizer OR #HomeQuarantine OR #LockDown OR #QuarantineCenter OR
                  #SocialDistancing OR #StayHome OR #StaySafe OR #StayAtHomeChallenge'

#call main method passing keywords and file path
write_tweets(tweet_keywords, tweets_data)
```

2. Import the required Packages for the Experiments

```
import numpy as np
import pandas as pd
import preprocessor as p
import collections
import os
import re
import string
import time
import warnings
from collections import Counter
from string import punctuation

#Visualisation
from wordcloud import WordCloud, ImageColorGenerator, STOPWORDS
from PIL import Image
import matplotlib.pyplot as plt
import plotly.offline as py
import plotly.graph_objs as go
import matplotlib
import seaborn as sns

#nltk
from nltk import PorterStemmer
from nltk import FreqDist
from nltk.collocations import *
from nltk.stem import WordNetLemmatizer
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.sentiment.util import *
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.tokenize import PunktSentenceTokenizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Bidirectional
from keras.layers import Embedding, Conv1D, Flatten, MaxPooling1D
from keras.layers import Activation, Dense, Dropout, LSTM
from keras.optimizers import Adam
from keras.utils import to_categorical

from tqdm import tqdm
tqdm.pandas(desc="progress-bar")

warnings.filterwarnings("ignore", category=DeprecationWarning)

plt.style.use('fivethirtyeight')
%matplotlib inline
%config IPCompleter.greedy=True
```

3. Read the Covid-19 Dataset & Remove Duplicate Tweets

```
original_df = pd.read_csv('TweetsDB.csv')

# sorting by first name
original_df.sort_values("original_text", inplace = True)

# dropping ALL duplicte values
original_df.drop_duplicates(subset ="original_text", keep = False, inplace = True)

# changing the date format
original_df['created_at'] = pd.to_datetime(original_df['created_at'],
                                format='%a %b %d %H:%M:%S %z %Y').dt.strftime('%Y-%m-%d')

original_df.sort_values(by='created_at', inplace=True)
original_df.to_csv("TweetsDB.csv", index=False)
original_df = pd.read_csv('TweetsDB.csv')
new_full_df = original_df
```

4. Data Pre-processing

```
# remove urls, RT, handles, and the hashtag from hashtags
def remove_urls(text):
    new_text = re.sub(r'amp;', '', str(text))
    new_text = re.sub(r':', '', str(new_text))
    new_text = re.sub(r',Ä¶', '', str(new_text))
    #replace consecutive non-ASCII characters with a space
    new_text = re.sub(r'[^\x00-\x7F]+',' ', str(new_text))
    new_text = ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)"," ",
                      str(new_text)).split())
    return new_text

# make all text lowercase
def text_lowercase(text):
    return text.lower()

#remove numbers
def remove_numbers(text):
    result = re.sub(r'\d+', '', str(text))
    return result

# remove punctuation
def remove_punctuation(text):
    translator = str.maketrans('', '', string.punctuation)
    return text.translate(translator)

# tokenize
def tokenize(text):
    text = word_tokenize(text)
    return text

# remove stopwords
stop_words = set(stopwords.words('english'))
def remove_stopwords(text):
    text = [i for i in text if not i in stop_words]
    return text

# lemmatize
lemmatizer = WordNetLemmatizer()
def lemmatize(text):
    text = [lemmatizer.lemmatize(token) for token in text]
    return text
```

```
# pre-processing Function
def preprocessing(text):
    text = remove_urls(text)
    text = text_lowercase(text)
    text = remove_numbers(text)
    text = remove_punctuation(text)
    text = tokenize(text)
    text = remove_stopwords(text)
    text = lemmatize(text)
    text = ' '.join(text)
    text = re.sub(r'covid', 'covid19', str(text))
    return text

clean_text = []
for text_data in new_full_df['original_text']:
    pp_text_data = p.clean(str(text_data))
    clean_text_data = preprocessing(pp_text_data)
    clean_text.append(clean_text_data)
new_full_df['clean_tweet'] = clean_text

tokenized_tweet = new_full_df['clean_tweet'].apply(lambda x: x.split())

ps = PorterStemmer()
Stemmed_tweet = tokenized_tweet.apply(lambda x: [ps.stem(i) for i in x])
for i in range(len(Stemmed_tweet)):
    Stemmed_tweet[i] = ' '.join(Stemmed_tweet[i])

new_full_df['clean_tweet'] = Stemmed_tweet
new_full_df['clean_tweet'].replace('', np.nan, inplace=True)
new_full_df['clean_tweet'] = new_full_df['clean_tweet'].str.replace(r'\b\w\b', '').str.replace(r'\s+', ' ')

df = pd.read_csv('TweetsDB.csv')
df["clean_tweet"] = new_full_df['clean_tweet']
df.to_csv("TweetsDB.csv", index=False)
```

5. Token Extraction

```
all_words = ' '.join(str(text) for text in new_full_df['clean_tweet'])

# split into words
tokens = word_tokenize(all_words)

# convert to lower case
tokens = [w.lower() for w in tokens]
```

```
# remove punctuation from each word
table = str.maketrans('', '', string.punctuation)
stripped = [w.translate(table) for w in tokens]

print(tokens[0:], file=open("Covid Tokens.txt", "w"))
```

6. N-gram Analysis

```
# unigram extraction
ug_count = Counter(tokens).most_common()
print(ug_count, file=open("Unigram Analysis.txt", "w"))
print(ug_count)

# bigram extraction
bgs = nltk.bigrams(tokens)
fdist = nltk.FreqDist(bgs)
bg_count = Counter(fdist).most_common()
print(bg_count, file=open("Bigram Analysis.txt", "w"))
print(bg_count)

# trigram extraction
bgs = nltk.trigrams(tokens)
fdist = nltk.FreqDist(bgs)
tg_count = Counter(fdist).most_common()
print(tg_count, file=open("Trigram Analysis.txt", "w"))
print(tg_count)
```

7. Covid-19 Specific Word Identification

```
word_counts = collections.Counter(tokens)
word_counts.most_common()

word_counts_df = pd.DataFrame(word_counts.most_common(),
                                              columns=['Word', 'Popularity'])

probability = []
total_words_count = word_counts_df['Popularity'].sum()
for i in range(0,len(word_counts_df.index)):
    word_probability = word_counts_df['Popularity'].iloc[i] / total_words_count
    probability.append(word_probability)

word_counts_df['Probability'] = probability
```

8. NLTK-based Sentiment Analysis & Classification

```
def nltk_sentiment(sentence):
    nltk_sentiment = SentimentIntensityAnalyzer()
    score = nltk_sentiment.polarity_scores(sentence)
    return score

dataset = new_full_df['clean_tweet'].astype(str)
nltk_results = [nltk_sentiment(row) for row in dataset]
results_df = pd.DataFrame(nltk_results)
text_df = pd.DataFrame(dataset)
sentiment_df = text_df.join(results_df)

sentiment_df["sentiment"] = ""
for i in range(0,len(sentiment_df.index)):
    if sentiment_df['compound'].iloc[i] < 0:
        sentiment_df['sentiment'].iloc[i] = 'neg'
    elif sentiment_df['compound'].iloc[i] > 0:
        sentiment_df['sentiment'].iloc[i] = 'pos'
    else:
        sentiment_df['sentiment'].iloc[i] = 'neu'

new_df = pd.DataFrame()
new_df = pd.read_csv('TweetsDB.csv')
new_df[["compound", "neg", "neu", "pos", "sentiment"]] = sentiment_df[['compound', 'neg',
                                                                       'neu', 'pos', 'sentiment']]
new_df.to_csv("TweetsDB.csv", index=False)

cols = ['Date', 'Text', 'Sentiment']
new_sentiment_df = pd.DataFrame(columns=cols, index=range(len(sentiment_df.index)))
for i in range(0,len(sentiment_df.index)):
    if (sentiment_df.iloc[i]['sentiment'] != 'neu'):
        new_sentiment_df.loc[i].Date = sentiment_df.at[i, 'date']
        new_sentiment_df.loc[i].Text = sentiment_df.at[i, 'clean_tweet']
        new_sentiment_df.loc[i].Sentiment = sentiment_df.at[i, 'sentiment']
new_sentiment_df = new_sentiment_df.dropna()
new_sentiment_df = new_sentiment_df.reset_index(drop=True)
new_sentiment_df = new_sentiment_df
data_df = pd.DataFrame()
data_df[['Sentiment', 'Text']] = new_sentiment_df[['Sentiment', 'Text']]
data_df.to_csv("Data.csv", index=False)

tweet_df = pd.DataFrame()
tweet_df["Text"] = data_df['Text']
tweet_df.to_csv("Tweet Data.csv", index=False)
```

9. Naïve Bayes Sentiment Model

```
tweets = pd.read_csv('Tweet Data.csv', header=0)
data = pd.read_csv('Data.csv', sep=',', names=['Sentiment', 'Text'], dtype=str, header=0)

def split_into_lemmas(tweet):
    ngram_vectorizer = CountVectorizer(ngram_range=(1, 3), token_pattern=r'\b\w+\b', min_df=1)
    analyze = ngram_vectorizer.build_analyzer()
    return analyze(tweet)

bow_transformer = CountVectorizer(analyzer=split_into_lemmas, stop_words='english',
                                  strip_accents='ascii').fit(data['Text'])
text_bow = bow_transformer.transform(data['Text'])
tfidf_transformer = TfidfTransformer().fit(text_bow)
text_tfidf = tfidf_transformer.transform(text_bow)
classifier_nb = MultinomialNB().fit(text_tfidf, data['Sentiment'])
sentiments = pd.DataFrame(columns=['text', 'class', 'prob'])
i = 0
for _, tweet in tweets.iterrows():
    i += 1
    try:
        bow_tweet = bow_transformer.transform(tweet)
        tfidf_tweet = tfidf_transformer.transform(bow_tweet)
        sentiments.loc[i-1, 'text'] = tweet.values[0]
        sentiments.loc[i-1, 'class'] = classifier_nb.predict(tfidf_tweet)[0]
        sentiments.loc[i-1, 'prob'] = classifier_nb.predict_proba(tfidf_tweet)[0][1]
    except Exception as e:
        sentiments.loc[i-1, 'text'] = tweet.values[0]

sentiments.to_csv('Sentiments.csv', encoding='utf-8')
sentiments.head()
```

10. Fine-grained Sentiment Classification

```
trend_df = pd.DataFrame()
trend_df["Date"] = new_sentiment_df['Date']
trend_df["Original_Tweet"] = original_df['original_text']
trend_df["Clean_Tweet"] = new_full_df['clean_tweet']
trend_df["Sentiment"] = sentiments['prob']
trend_df.to_csv('Sentiment Trend.csv', encoding='utf-8')
```

```
trend_df["Refined_Sentiment"] = ""
for i in range(0,len(trend_df.index)):
    if trend_df['Sentiment'].iloc[i] >= 0.0 and trend_df['Sentiment'].iloc[i] < 0.25:
        trend_df['Refined_Sentiment'].iloc[i] = -1.0
    elif trend_df['Sentiment'].iloc[i] >= 0.25 and trend_df['Sentiment'].iloc[i] < 0.5:
        trend_df['Refined_Sentiment'].iloc[i] = -0.5
    elif trend_df['Sentiment'].iloc[i] == 0.5:
        trend_df['Refined_Sentiment'].iloc[i] = 0.0
    elif trend_df['Sentiment'].iloc[i] > 0.5 and trend_df['Sentiment'].iloc[i] < 0.75:
        trend_df['Refined_Sentiment'].iloc[i] = 0.5
    else:
        trend_df['Refined_Sentiment'].iloc[i] = 1.0

trend_df.to_csv('Overall Sentiment Trend.csv', encoding='utf-8')
```

11. Overall Average Sentiment Trend

```
avg_trend_df = pd.DataFrame()
avg_trend_df["Date"] = trend_df['Date']
avg_trend_df["Refined_Sentiment"] = trend_df['Refined_Sentiment']
avg_trend_df = avg_trend_df.set_index(['Date'])
avg_trend_df.Refined_Sentiment = avg_trend_df.Refined_Sentiment.astype('float64')
avg_trend_df = avg_trend_df.groupby('Date')['Refined_Sentiment'].mean().round(2)
avg_trend_df = pd.DataFrame(avg_trend_df)
avg_trend_df.to_csv('Overall Refined Sentiment Trend.csv', encoding='utf-8')
```

12. Sentiment Trend Shift Detection

```
trend_df = pd.read_csv('Sentiment Trend.csv', header=0)
trend_df["Pos_Sentiment"] = ''
trend_df["Neg_Sentiment"] = ''
for i in range(0,len(trend_df.index)):
    if trend_df['Sentiment'].iloc[i] >= 0.0 and trend_df['Sentiment'].iloc[i] <= 0.50:
        trend_df['Neg_Sentiment'].iloc[i] = trend_df['Sentiment'].iloc[i].astype('float64')
    else:
        trend_df['Neg_Sentiment'].iloc[i] = 0.0
for i in range(0,len(trend_df.index)):
    if trend_df['Sentiment'].iloc[i] > 0.50 and trend_df['Sentiment'].iloc[i] < 1.0:
        trend_df['Pos_Sentiment'].iloc[i] = trend_df['Sentiment'].iloc[i].astype('float64')
    else:
        trend_df['Pos_Sentiment'].iloc[i] = 0.0
trend_df['Date'] = pd.to_datetime(trend_df['Date'])
trend_df.sort_values(by='Date', inplace=True)
```

```
avg_trend_df = pd.DataFrame()
avg_trend_df["Date"] = trend_df['Date']
avg_trend_df["Avg_Pos_Sentiment"] = trend_df['Pos_Sentiment'].astype('float64')
avg_trend_df["Avg_Neg_Sentiment"] = trend_df['Neg_Sentiment'].astype('float64')
avg_trend_df['Date'] = pd.to_datetime(avg_trend_df['Date'])
avg_trend_df = avg_trend_df.set_index(['Date'])
avg_trend_df = avg_trend_df.groupby(pd.Grouper(freq='1d')).mean()
avg_trend_df = avg_trend_df.dropna()

avg_trend_df.to_csv('Sentiment Trend Shift.csv', encoding='utf-8')
```

13. Prepare Final Data

```
cols = ['Sentiment', 'Text']
sentiments= pd.read_csv('Sentiments.csv', header=0)
final_data_df = pd.DataFrame(columns=cols, index=range(len(sentiments.index)))
final_data_df["Sentiment"] = ""
for i in range(0,len(sentiments.index)):
    if sentiments['class'].iloc[i] == 'neg':
        final_data_df['Sentiment'].iloc[i] = 0
    else:
        final_data_df['Sentiment'].iloc[i] = 1
final_data_df["Text"] = sentiments['text']
final_data_df.to_csv("Tweet Final Data.csv", index=False)

def ingest():
    data = pd.read_csv('Tweet Final Data.csv')
    data = data[data.Sentiment.isnull() == False]
    data['Sentiment'] = data['Sentiment'].map(int)
    data = data[data['Text'].isnull() == False]
    data.reset_index(inplace=True)
    data.drop('index',axis=1,inplace=True)
    print ('dataset loaded with shape', data.shape)
    return data

data = ingest()

X = data['Text']
Y = to_categorical(data['Sentiment'].values)

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
```

```
all_words=' '.join(x_train)
all_words=word_tokenize(all_words)
dist=FreqDist(all_words)
num_unique_word=len(dist)

tweet_len=[]
for text in x_train:
    word=word_tokenize(text)
    l=len(word)
    tweet_len.append(l)
max_tweet_len=np.max(tweet_len)

max_features = num_unique_word
max_words = max_tweet_len

tokenizer = Tokenizer(num_words=max_features)
tokenizer.fit_on_texts(x_train)
x_train = tokenizer.texts_to_sequences(x_train)
x_test = tokenizer.texts_to_sequences(x_test)

x_train = sequence.pad_sequences(x_train, maxlen=max_words)
x_test = sequence.pad_sequences(x_test, maxlen=max_words)
```

14. Classification Report & Confusion Matrix

```
# create DecisionTree model
classifier = DecisionTreeClassifier()

# train model and make predictions
y_pred = classifier.fit(x_train, y_train).predict(x_test)

# create a classification report
print()
print("\t\t: Classification Report :\t\t")
print("\t\t=========================\t\t")
print()
print(classification_report(y_test, y_pred))

# create a confusion matrix
print()
print("\t\t: Confusion Matrix :\t\t")
print("\t\t====================\t\t")
print()
print(confusion_matrix(y_test.argmax(axis=1), y_pred.argmax(axis=1)))
```

15. Hybrid Convo-Sequential LSTM Model

```
# define the first hybrid model
model1= Sequential()
model1.add(Embedding(max_features, 100, input_length=max_words))
model1.add(Conv1D(64, kernel_size=3, padding='same', activation='relu'))
model1.add(MaxPooling1D(pool_size=2))
model1.add(Dropout(0.25))
model1.add(LSTM(128, return_sequences=True))
model1.add(Dropout(0.2))
model1.add(Flatten())
model1.add(Dense(128, activation='relu'))
model1.add(Dropout(0.1))
model1.add(Dense(64, activation='relu'))
model1.add(Dense(32, activation='relu'))
model1.add(Dense(2, activation='sigmoid'))
model1.compile(loss='binary_crossentropy', optimizer=Adam(lr=0.001), metrics=['accuracy'])
model1.summary()

# train the CNN + Seq-LSTM model
history = model1.fit(x_train, y_train, epochs=6, validation_data=(x_test, y_test),
                                                        batch_size=32, verbose=2)
```

16. Hybrid Convo-Bidirectional LSTM Model

```
# define the second hybrid model
model2= Sequential()
model2.add(Embedding(max_features, 100, input_length=max_words))
model2.add(Conv1D(64, kernel_size=3, padding='same',activation='relu'))
model2.add(MaxPooling1D(pool_size=2))
model2.add(Dropout(0.25))
model2.add(Bidirectional(LSTM(128, return_sequences=True)))
model2.add(Dropout(0.2))
model2.add(Flatten())
model2.add(Dense(128, activation='relu'))
model2.add(Dropout(0.1))
model2.add(Dense(64, activation='relu'))
model2.add(Dense(32, activation='relu'))
model2.add(Dense(2, activation='sigmoid'))
model2.compile(loss='binary_crossentropy', optimizer=Adam(lr=0.001), metrics=['accuracy'])
model2.summary()

# train the CNN + Bi-LSTM model
history = model2.fit(x_train, y_train, epochs=6, validation_data=(x_test, y_test),
                                                        batch_size=32, verbose=2)
```

End