EXPLORATORY DATA ANALYSIS ON WHATSAPP CHAT AND EMOTION DETECTION USING LSTM

A PROJECT REPORT

Submitted by

PURUSHOTTAM KUMAR

(2020178043)

A report of the project
submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment for the award of the degree
of

MASTER OF COMPUTER APPLICATIONS

INFORMATION AND COMMUNICATION ENGINEERING
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600 025

JUNE 2022

ANNA UNIVERSITY

CHENNAI - 600 025

BONA FIDE CERTIFICATE

Certified that this project report titled **EXPLORATORY DATA ANALYSIS ON WHATSAPP CHAT AND EMOTION DETECTION USING LSTM** is the bona fide work of **PURUSHOTTAM KUMAR (2020178043)** who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:	**Dr. T MALA**
DATE:	**ASSOCIATE PROFESSOR**
	PROJECT GUIDE
	DEPARTMENT OF IST, CEG
	ANNA UNIVERSITY
	CHENNAI 600025

COUNTERSIGNED

Dr. S. SRIDHAR
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

In recent times, messages are the primary mode of communication. In today's world, the most popular chat application for communication is Whats App. It has been widely used by all, especially among the business people and youngsters. WhatsApp group chats can be analyzed using several analyzing tools. Authentically users wish to analyse their chat for several purposes.

The main objective of this project is to analyze the WhatsApp group chats to find out the amount of involvement and participation by members. Also, it involves the analysis of the most active date in the group, the number of messages sent on that date, the overall most active user, list of active members in the group, total number of users, number of posts made by each individuals on the group, and the most used word on the platform. Also an analysis of the top 15 members and day/month/year wise messages is done.

The other objective is to predict the emotion of text data in terms of joy, sadness, anger etc. in this project, LSTM (a deep learning algorithm) network architecture have been used to analyse emotions for given text. The experimental evaluations has been performed on emotion data-set from kaggle which contains 16000 records with different emotions. Validation Accuracy is the evaluation metrics used to analyze emotions in this project.

சுருக்கம் (தமிழ்)

சமீப காலங்களில், செய்திகளே முதன்மையான தகவல்தொடர்பு முறையாகும். இன்றைய உலகில், தகவல்தொடர்புக்கான மிகவும் பிரபலமான அரட்டை பயன்பாடு பகிரி ஆகும். இது அனைவராலும், குறிப்பாக வணிகர்கள் மற்றும் இளைஞர்களிடையே பரவலாகப் பயன்படுத்தப்படுகிறது. வாட்ஸ்அப் குழு அரட்டைகளை பல பகுப்பாய்வு கருவிகளைப் பயன்படுத்தி பகுப்பாய்வு செய்யலாம். பல நோக்கங்களுக்காக பயனர்கள் தங்கள் அரட்டையை ஆய்வு செய்ய விரும்பு கிறார்கள்.

உறுப்பினர்களின் ஈடுபாடு மற்றும் பங்கேற்பின் அளவைக் கண்டறிய வாட்ஸ்அப் குழு அரட்டைகளை பகுப்பாய்வு செய்வதே இந்தத் திட்டத்தின் முக்கிய நோக்கமாகும். மேலும், இது குழுவில் மிகவும் செயலில் உள்ள தேதி, அந்த தேதியில் அனுப்பப்பட்ட செய்திகளின் எண்ணிக்கை, ஒட்டுமொத்த மிகவும் செயலில் உள்ள பயனர், குழுவில் செயலில் உள்ள உறுப்பினர்களின் பட்டியல், மொத்த பயனர்களின் எண்ணிக்கை, ஒவ்வொரு நபரும் செய்த இடுகைகளின் எண்ணிக்கை ஆகியவற்றை உள்ளடக்கியது. குழுவில், மற்றும் மேடையில் அதிகம் பயன்படுத்தப்படும் வார்த்தை. முதல் 15 உறுப்பினர்களின் பகுப்பாய்வு மற்றும் நாள்/மாதம்/ஆண்டு வாரியாக

இந்த திட்டத்தில் மகிழ்ச்சி, சோகம், கோபம் போன்றவற்றின் அடிப்படையில் உரைத் தரவின் உணர்ச்சிகளைக் கணிப்பது மற்ற நோக்கமாகும், கொடு க்கப்பட்ட உரைக்கான உணர்ச்சிகளை பகுப்பா ய்வு செய்ய LSTM (ஆழமான கற்றல் வழிமுறை) நெட் வொர்க் கட்டமைப்பு பயன்படுத்தப்பட்டுள்ளது. வெவ் வேறு உணர்வுகளுடன் 16000 பதிவுகளைக் கொண்ட காகில் இருந்து உணர்ச்சித் தரவு-தொகுப்பில் சோதனை மதிப்பீடுகள் செய்யப்பட்டுள்ளன. சரி பார்ப்பு துல்லி யம் என்பது இந்த திட்டத்தில் உணர்ச்சிகளை பகுப் பாய்வு செய்ய பயன்படுத்தப்ப டும் மதிப்பீட்டு அளவீ டுகள் ஆகும்.

ACKNOWLEDGEMENT

I express my sincere thanks and deep sense of gratitude to my guide **Dr. T Mala**, Associate Professor, Faculty of Department of Information Science and Technology, College of Engineering, Guindy for her valuable guidance. She has been a constant source of inspiration and I thanked her for providing me with the necessary counsel and direction to help me complete this project.

My sincere thanks to, **Dr. S. Sridhar**, Professor and Head of the Department, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, Chennai for extending support.

I would like to express my special thanks to **Dr. R. Geetha Ramani**, Professor and the project committee members **Dr. Abirami Murugappan**, Assistant Professor, **Dr. M. Deivamani**, Teaching Fellow, **Dr. T. J. Vijay Kumar**, Teaching Fellow, **Ms. R. L. Jasmine**, Teaching Fellow, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, Chennai for their valuable suggestions and critical reviews during development of this project.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	iii
	ABSTRACT (TAMIL)	iv
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	x
1	**INTRODUCTION**	**1**
	1.1 OVERVIEW	1
	1.2 INTRODUCTION TO WHATSAPP	2
	1.3 INTRODUCTION TO NLP	2
	1.4 INTRODUCTION TO RNN	4
	1.5 LSTM	5
	1.6 APPLICATIONS OF LSTM	6
	1.7 PROBLEM STATEMENT	8
	1.8 PROPOSED SYSTEM	8
	1.9 OBJECTIVES	8
	1.10 ORGANIZATION OF THE REPORT	9
2	**LITERATURE SURVEY**	**10**
	2.1 Analysis of Usage and Impact of WhatsApp Using Text Mining	10
	2.2 Statistical analysis on Group chats of WhatsApp Using R	10
	2.3 Sentiment Analysis in Light of LSTM RNN	11
	2.4 Recognizing Contextual Polarity in Phrase Level Analysis	11
	2.5 Word Level Classification using Centroid Model	12
3	**SYSTEM DESIGN**	**14**
	3.1 **SYSTEM ARCHITECTURE OF EDA AND EMOTION DETECTION SYSTEM**	14
	3.2 MODULE DESCRIPTION	15
	3.2.1 Data Collection	15
	3.2.2 PreProcessing	16
	3.2.3 Exploratory Data Analysis	18
	3.3 FastText Word Vector Model	21

3.4 LSTM 22
3.4.1 Architecture of LSTM 23
3.5 BI-LSTM 25

4 ALGORITHM AND PSEUDO-CODE 26
4.1 EXPLORATORY DATA ANALYSIS USING PYTHON LIBRARIES 26
4.2 EMOTION DETECTION USING LSTM 28
4.3 EMOTION DETECTION USING BI-LSTM 29
4.3.1 Algorithm for BI-LSTM MODEL 29

5 IMPLEMENTATION AND ANALYSIS 31
5.1 EXPLORATORY DATA ANALYSIS 31
5.1.1 Exporting the chat file from WhatsApp 31
5.1.2 Benchmark Dataset of WhatsApp Group Chat 31
5.1.3 Dashboard With Statistical Information 34
5.1.4 Year Wise Message Activity Using Responsive Line Chart 34
5.1.5 Monthly and Weekly message activity 35
5.1.6 Most Active Time in 24 hour During Weeks 36
5.1.7 Most active hours in a day 36
5.1.8 Clustering User activity 37
5.1.9 Top 25 used words in the chat 38
5.1.10Wordcloud of most used words in chat 38
5.1.11Positive sentiment calculation using Polarity Score 39
5.2 EMOTION DETECTION 41
5.2.1 PRE-PROCESSING 41
5.2.2 Model Building of LSTM and Bi-LSTM 43
5.3 Evaluation Results 46

6 CONCLUSION AND FUTURE WORK 48
6.1 CONCLUSION 48
6.2 FUTURE WORK 49

REFERENCES 50

LIST OF FIGURES

Figure no.	Title	Page No
1.1	RNN Neural network	4
1.2	LSTM Gates	5
3.1	EDA & Emotion Detection System using LSTM and BI-LSTM	14
3.2	Separating conversations into timestamp and messages	17
3.3	Transforming timestamp and messages into Data frame	17
3.4	Statistical Information	19
3.5	Day wise message activity throughout year	19
3.6	Total Message sent and Average length of Message by User	20
3.7	Heat Map of Messages during each hour in week	21
3.8	Word cloud of most used words	21
3.9	LSTM Architecture	23
3.10	Bi-LSTM Architecture	25
5.1	Exporting chat file from WhatsApp	31
5.2	Raw Dataset : Default Generated dataset from WhatsApp	32
5.3	Pre-Processed dataset	33
5.4	Statistical information from WhatsApp chats	34
5.5	Line chart of year wise message activity	35
5.6	Month wise and weekday wise message activity	36
5.7	24 hours activity during whole week	36
5.8	Most active hours during a day	37
5.9	Clustering of members on the basis of activities	38
5.10	Top 25 most used words	38
5.11	Wordcloud of words	39
5.12	Positive sentiment on the basis of polarity score	40
5.13	Loading dataset	41
5.14	Pre Processing and Cleaning on Dataset	42
5.15	Label Encoding	42
5.16	Tokenization and intermediate steps	43
5.17	Building Emotion detection model using LSTM	45
5.18	Building Emotion detection model using Bi-LSTM	45
5.19	Accuracy Graph of emotion detection model	46

Figure no.	Title	Page No
5.20	Model Summary	46
5.21	Model Accuracy in each Epoch	47
5.22	Model Loss in each Epoch	47

LIST OF ABBREVIATIONS

LSTM	Long-Short Term Memory
BI-LSTM	Bidirectional Long-Short Term Memory
NLP	Natural Language Processing
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
EDA	Exploratory Data Analysis

CHAPTER 1

INTRODUCTION

Curiosity is one of the major thing which pushed many people to do many astonishing things, Everyone have the curiosity about their image in the other person's mind based on the conversation that took place between them or in a group. In order to make the model more efficient we need lots of data, so we are taking one of the large scale data producers owned by Facebook which is nothing but WhatsApp. WhatsApp claims that nearly 85 billion messages are sent each day. The average user spends more than 4 hours per week on WhatsApp, and is a member of plenty of groups. So the solution this project presents is applying exploratory data analysis to the messages that are exported from the chat. All the messages are exported from WhatsApp export feature available in the app itself. This exported chat has to be pre-processed for better computation and all the messages are separated. Emotion detection or sentiment analysis can be described as techniques, methods, and tools for detecting as well as extracting subjective information, such as attitude and opinions, from language. Until recently, sentiment analysis has been about opinion polarity, i.e., categorization of text as positive, negative or neutral [4]. However, these three conventional categories are not adequate to recognize the connotation of the underlying tone of a sentence

1.1 OVERVIEW

The main goal of exploratory data analysis is to analyze the data using visual techniques and goal of emotion detection is to predict to the mood or point of view of a person based on the language they use with reference to a specific topic, sentence or paragraph. Various methods can be used to analyze the emotion in text whether it is happy, sad, anger etc. These methods are generally categorized as either human based or automated. There are various methods to analyze emotions such as keyword processing, manual processing and Natural Language Processing. Here we are using text mining techniques, python libraries and framework for exploratory data analysis work.

Emotion detection on given text data is the objective of the project. it is crucial for easy and simple detection of human feelings at a specific moment without actually asking them. Further, another key goal is to understand how the proposed LSTM model works and how intermediate layers are working behind. Data-set sentences and their emotions are essential for identifying emotion in text. Joy, surprise, anger, sadness, happy etc based sentence or negations in the sentence make the emotion detection very challenging. The LSTM network with analysis of emotion, knowledge obtained during emotion detection research is the groundwork. In LSTM, different layers need to be employed to understand how the proposed model works and responds to the given text data.[**?**].

1.2 INTRODUCTION TO WHATSAPP

WhatsApp is an instant messaging application that allows users to send text messages, chat and share media files like images, audio and video files. Users can also share documents and applications. With WhatsApp, users have the opportunity to communicate with several other users at the same time in a group. In addition, a user can send a broadcast message to up to two hundred and fifty six (256) users at a single message stance. After installing the application, the sending and receiving of messages seems cost free (in contrast to the original text message function on mobile phones). According to reports, WhatsApp service had up to 2 billion monthly average users as at 2022. Also, a data analysis showed that the use of WhatsApp accounted for 19.83 percents of all smartphone in 2022 as compared to Facebook which takes only 10 percents.

1.3 INTRODUCTION TO NLP

Natural Language Processing is a field of Artificial Intelligence which focuses on enabling the systems for understanding and processing the human languages. As a technology, natural language processing has come of age over the past fifteen years, with products such as Siri, Alexa and Google's voice search employing NLP to understand and respond to user requests. In this project we are using various text mining techniques in NLP for exploring the WhatsApp chats like word-frequency, time series analysis, word-cloud, emojis analysis and statistical manipulation etc.

Text mining is also known as text data mining. It is the process of transforming unstructured text into a structured format to identify meaningful patterns and new insights. By applying advanced analytical techniques, such as Naïve Bayes, Support Vector Machines (SVM), and other deep learning algorithms, we can explore and discover hidden relationships within unstructured data. It involves following operations

- Cleaning
- Information extraction
- Natural Language Processing
- Clustering
- Categorization
- Visualization
- Text Summarization

1.4 INTRODUCTION TO RNN

RNN is a type of Neural Network. The outputs from previous step are fed as input to the current step in RNN. All the inputs and outputs are independent of each other in orthodox neural networks. There is a need to remember the previous words when it is required to predict the next word of a sentence or detect the emotion in case of negation words. This is where RNN came into limelight. RNN solved this issue with the help of a Hidden Layer. Hidden state of RNN remembers some information or details about a sequence. This is one of the main and most important feature of RNN. Other neural networks make prediction on the basis of current word only because they doesn't store the previous word which can cause a huge drawback in model.

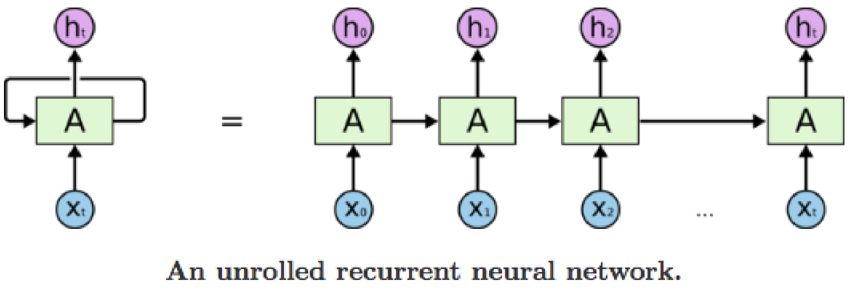


Figure 1.1: RNN Neural network

In Figure 1.1 a normal RNN network control flow has been displayed. For example, we are detecting emotion from a sentence **"I am not happy"**. Other neural network will process on current word only and will store the result. Since they don't have previous memory or any hidden layer so once they will reach to word "happy" they may have forgotten previous negation word "not". Because of this they will detect emotion as joy but in reality it is "not happy" that means it should be sad.

1.5 LSTM

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back-propagation.

The heart of a LSTM network is cell state which provides a bit of memory to the LSTM so it can remember the past. Gates in LSTM are the sigmoid activation functions i.e they output a value between 0 or 1 and in most of the cases it is either 0 or 1. In an LSTM network, three gates are present.

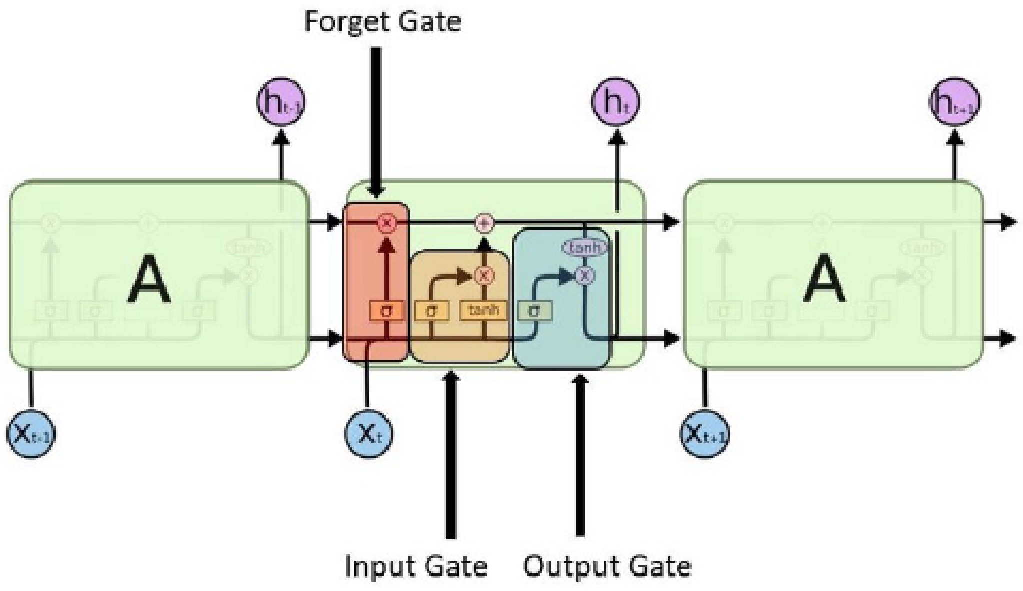


Figure 1.2: LSTM Gates

Input Gate discover which value from input should be used to modify the memory. In figure 1.2, Sigmoid function decides which values to let through 0,1. and tanh function gives weightage to the values.

Forget Gate discovers what details to be discarded from the memory block. It is decided by the sigmoid function. In Figure 1.2, it looks at the previous state (ht-1) and the current input (Xt) and produce an output number between 0(remove this) and 1(keep this) for each number in the cell state (Ct-1).

In **Output Gate** the input and the memory of the block is used to decide the output. Sigmoid function decides which values to let through 0,1. and tanh function gives weightage to the values which are passed deciding their level of importance ranging from -1 to 1 and multiplied with output of Sigmoid.

Formula for calculating Input Gate

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right) \tag{1.1}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C) \tag{1.2}$$

Where σ represents sigmoid function

tanh represents tanh function

W_i is weight of current input gate neuron

h_{t-1} is output of the previous LSTM block (at timestamp t-1)

X_t is input at current timestamp (t)

b_i is biases for the input gates

Formula for calculating Forget Gate

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right) \tag{1.3}$$

Where σ represents sigmoid function

F_t represents Forget Gate

W_f is weight of current Forget Gate neuron

h_{t-1} is output of the previous LSTM block (at timestamp t-1)

X_t is input at current timestamp (t)

B_f is biases for the Forget gate

Formula for calculating Output Gate

$$o_t = \sigma \left(W_o \left[h_{t-1}, x_t \right] + b_o \right) \tag{1.4}$$

$$h_t = o_t * \tanh \left(C_t \right) \tag{1.5}$$

Where	σ	represents sigmoid function
	O_t	represents Output Gate
	W_o	is weight of current Output Gate neuron
	h_{t-1}	is output of the previous LSTM block (at timestamp t-1)
	X_t	is input at current timestamp (t)
	B_o	is biases for the Output gate
	C_t	Final calculated value at timestamp(t)

1.6 APPLICATIONS OF LSTM

LSTMs are mostly used to learn, process, and classify sequential data because these networks can learn long-term dependencies between time steps of data. Common LSTM applications include sentiment analysis, language modeling, speech recognition, and video analysis. In **Speech Recognition** where input is audio and output is text, like Google Assistant, Microsoft Cortana, Apple Siri. Also it is used in **Machine Translation** where input is text and output is also text for ex:- Google Translate.

Apart from this it is also used in **Image Captioning** where input is image and output is text. Here it predict the suitable caption for given image. It is also widely used in **Sentiment Analysis**, **Music Generation/Synthesis** and **Video Activity Recognition**

1.7 PROBLEM STATEMENT

Exploratory data analysis has been performed on WhatsApp chats with different text mining techniques. All visualisation has been done with help of python libraries and WhatsApp chat data. Using mining techniques we have discover interesting insights withing WhatsApp chats.

Emotion detection part has been performed on dataset from kaggle which contains dataset in mainly 4 different emotions. Deep learning techniques has been used for analysis of sentences and emotion detection. There are various approaches to detect emotions from dataset which have been explored but mostly are categorising only in terms of positive, negative and neutral. With the help of deep learning technique we are able to classify them to a specific emotion.

1.8 PROPOSED SYSTEM

For EDA, dataset is a simple text file that has been extracted from WhatsApp groups. As soon as number of text messages will be increased more visualisation effect will also be displayed. Chat from the WhatsApp can be extracted using a feature called export chat and this will mail the exported chat that will contain messages from the beginning and all the undeleted chat will be included in this text file. A lot of pre-processing needs to be done.

Sometimes detecting emotion may be difficult because sentence may include sarcasm, irony, negations, jokes etc. For Emotion Detection, we have used deep learning algorithms named LSTM and BI-LSTM. LSTM learns each word based on the past sentences that have been present in the corpus. They don't simply leave and start from scratch as normal CNN's do. Old neural networks such as CNNs cant do the memory storage of previous context.

1.9 OBJECTIVES

The objective of the proposed exploratory data analysis tool and emotion detection model is as follows :

- To visualise and explore WhatsApp chats to obtain insights by applying text mining techniques.
- To understand how the proposed LSTM and BI-LSTM model acts and respond to word embedding techniques such as Fast-ext word embedding.
- To detect emotion of the given text data in terms of joy, sadness, anger and fear.

1.10 ORGANIZATION OF THE REPORT

Chapter 1 provides a brief introduction to WhatsApp, RNN and LSTM. Different Gates in LSTM has been discussed here.

Chapter 2 gives the survey of various works related to this projects.

Chapter 3 consists System design of the project and module description.

Chapter 4 consists Algorithm and pseudocode related to the models with their outcomes.

Chapter 5 provides the implementation and results with screenshot of inputs and outputs. Also it contains results and analysis of the model summary.

Chapter 6 is the conclusion and future work to be done.

CHAPTER 2

LITERATURE SURVEY

2.1 Analysis of Usage and Impact of WhatsApp Using Text Mining

Ravi shankara et al.[1] conducted a demo Survey analysis on the usage and Impact of WhatsApp Messenger : Various Studies and analysis has been done on the usage and impact of WhatsApp. Some of these studies are for finding the impact of WhatsApp on the students and some are based on for the general public in a local region. In a study of southern part of India was conducted on the age group of between 18 to 23 years to investigate the importance of WhatsApp among youth. Though this study, it was found that students spent 8 hours per day on using WhatsApp and remain online almost 16 hours a day. All the respondents agreed that they are using WhatsApp for communicating with their friends. They also exchange images, audio and video files with their friends using WhatsApp. It was also proved that the only application that the youth uses when they are spending time on their smart phone is WhatsApp. Methods used in this survey is to analyze the intensity of WhatsApp usage and its popular services and to identify the degree of positive or negative impacts of using WhatsApp.

2.2 Statistical analysis on Group chats of WhatsApp Using R

Sunil Joshi [2] proposed the statistical analysis of WhastApp chats by analyzing and classifying chats on various parameters. The dataset of WhatsApp group chat used for analysis is of one year(may, 2019-may, 2020) which consists of 5,5563 records in total and comprises of certain characteristics that define how much a particular person is using WhatsApp chat group, such as the years of usage, duration of usage in a day, the response levels, type of messages posted by each individual in the group (Smiley, Text, Multitude), which age group people are more active and so on. The main attributes set for this analysis are type of message been send, duration of use per year/month/week/day/hour, timestamp (AM/PM), age group of senders, gender (Male/Female). RStudio the most favored IDE for R is been used to perform exploratory data analysis and visualization for the collected data largely because of its open source nature.

2.3 Sentiment Analysis in Light of LSTM RNN

Subarno Pal et al. [3] proposed analysis on Modeling of temporal sequences and their long-range dependencies more accurately, Long Short-Term Memory was designed over simple RNNs. It is a special type of recurrent neural network (RNN) architecture. Text obtained from different sources like user reviews and blogs express user's view or attitude towards the particular product or event etc. Increase in accuracy was achieved by stacking LSTM layer with one upon another. Subarno Paul et al. [4] proposed LSTM layer to be directional to transfer and deliver data both way forward and backward in the network. Splinting the neuron of regular LSTM into two directions, one for positive me direction forward states, and another for negative time direction backward states, the output are not connected to inputs of the opposite direction states. A layered deep LSTM with bidirectional connections is having a better performance in terms of accuracy compared to the simpler versions of LSTM used. Validation accuracy increased and there by validation loss dropped with the increase of complexity of the network. Increase in complexity of the layers clearly increases the computational cost of the network. Conventional LSTM, Deep LSTM and bidirectional deep LSTM were evaluated based on evaluation metrics such as validation loss and validation accuracy.

2.4 Recognizing Contextual Polarity in Phrase Level Analysis

Wilson et al. [4] introduced a new approach for phrase level sentiment analysis. This method determines whether a sentence or the expression is polar or neutral. It illustrates the polarity of polar expressions. There were 15696 subjective expressions from more than 441 documents which is 8964 sentences are annotated with contextual polarity. One of the most common approaches to analyze sentiments is to begin with a dictionary of positive and negative words. In this dictionary, entries are tagged with an apriori prior polarity. This method enables to automatically identify the contextual polarity for large subset of sentiment expression. It achieved better results than the baseline models.

2.5 Word Level Classification using Centroid Model

Felipe et al. [5] addresses the label sparsity problem for polarity classification for twitter by acquiring and exploiting lexical knowledge. Lexical knowledge is acquired in the form of opinion lexicons which is known as polarity lexicon induction Based on word-level classification, two twitter specific polarity lexicon induction was introduced on this paper. Word sentiment associations and tweet centroid model are the two models. Both the models are distant supervision methods that exploit the existing opinion lexicons for building a synthetically labeled data on which message-level polarity classifiers can be trained through partitioned tweet centroids: and annotate sample average (ASA). In word based sentiment association's method, information from automatically annotated tweets and existing hand-made opinion lexicons is combined to induce a Twitter specific opinion lexicon in a supervised fashion. The induced lexicon contains POS, noun, verb adjective with a probability distribution for positive, negative, and neutral polarity classes. Whereas in tweet centroid model, it does not depend on labeled tweets and can perform lexicon induction directly from a given corpus of unlabelled tweets. The tweet centroid model is a distributional representation that exploits the short nature of tweets by treating them as the whole contexts of words. In the tweet centroid model, tweets are represented by sparse vectors using the standard NLP features, such as unigrams and low dimensional word clusters, and words are represented as the centroids of the tweet vectors in which they appear.

CHAPTER 3

SYSTEM DESIGN

This chapter deals with architecture diagram of EDA and text based emotion model and gives the detailed module design and algorithm used as depicted in Figure 3.1.

3.1 SYSTEM ARCHITECTURE OF EDA AND EMOTION DETECTION SYSTEM

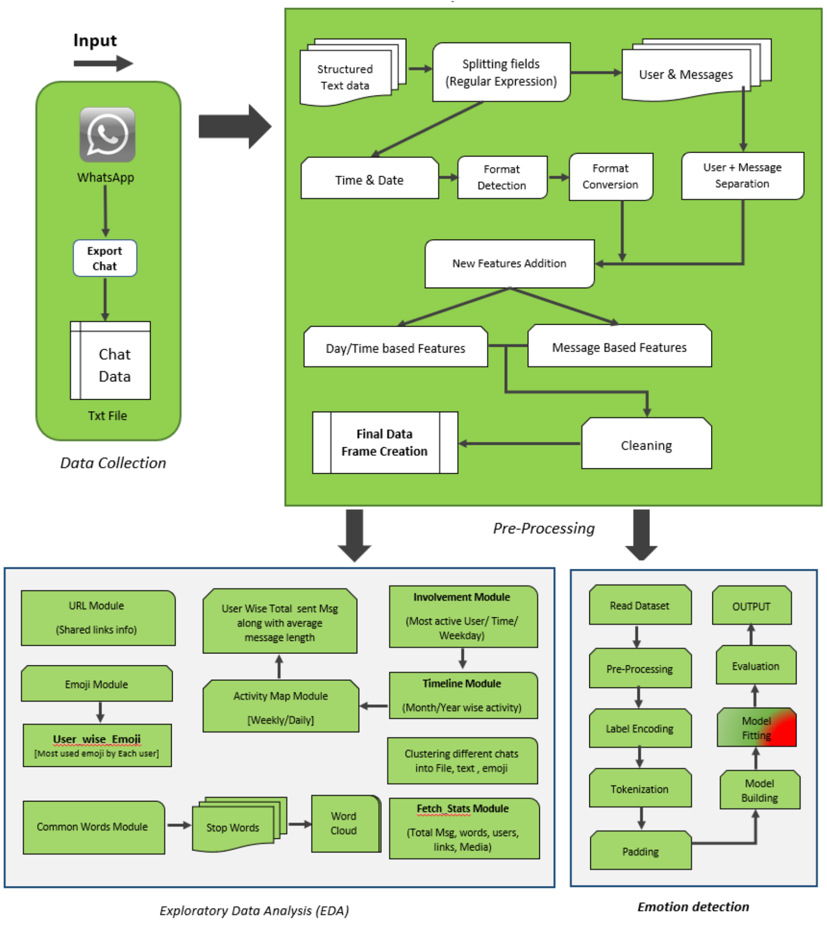

Input
WhatsApp
Export Chat
Chat Data
Txt File
Data Collection
Structured Text data
Splitting fields (Regular Expression)
User & Messages
Time & Date
Format Detection
Format Conversion
User + Message Separation
New Features Addition
Day/Time based Features
Message Based Features
Final Data Frame Creation
Cleaning
Pre-Processing
URL Module (Shared links info)
User Wise Total sent Msg along with average message length
Involvement Module (Most active User/ Time/ Weekday)
Emoji Module
Activity Map Module [Weekly/Daily]
Timeline Module (Month/Year wise activity)
User_wise_Emoji [Most used emoji by Each user]
Clustering different chats into File, text , emoji
Common Words Module
Stop Words
Word Cloud
Fetch_Stats Module (Total Msg, words, users, links, Media)
Exploratory Data Analysis (EDA)
Read Dataset
Pre-Processing
Label Encoding
Tokenization
Padding
OUTPUT
Evaluation
Model Fitting
Model Building
Emotion detection

Figure 3.1: EDA & Emotion Detection System using LSTM and BI-LSTM

The above architectural diagram is used to perform EDA on the chat data and to analyze the emotion from any text. Preprocessing of dataset is done. For Exploratory data analysis, a WhatsApp group chat has been considered as benchmark and for emotion detection a dataset with 4 different emotion has been taken. Emotion detection model was being implemented using LSTM and BI-LSTM network architectures.

3.2 MODULE DESCRIPTION

1. Data Collection
2. Preprocessing
3. Exploratory Data Analysis
4. FastText Word Vector Model
5. LSTM
6. BI-LSTM

3.2.1 Data Collection

For Exploratory Data Analysis, any WhatsApp exported chat can be considered as dataset. By default WhatsApp exports all the chats in form of text file. All the exported chats has a fix pattern **[Timestamp-Username-Messages]** as a sequential string, so here regular expressions has been used for extracting information from it. There are two different options whether to include media and whether to not. In case media are included then file size be increased hence it is preferred to export the chats without using media files. Once chats will be exported then it is uploaded as raw dataset.

The chat in WhatsApp application can be exported by following :-

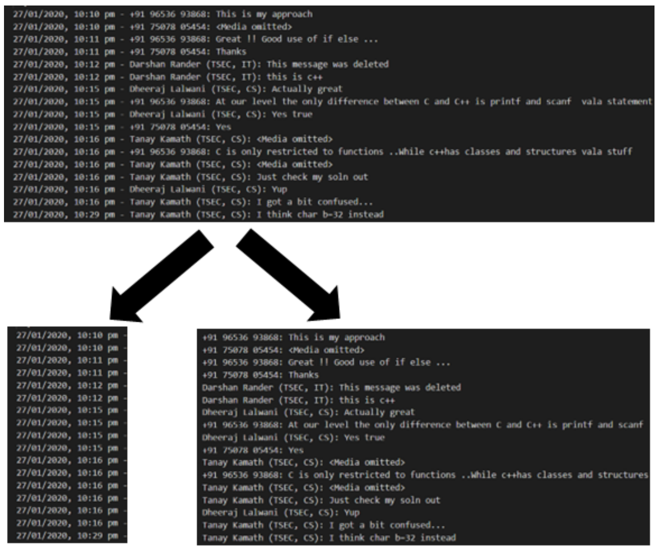
Launch the WhatsApp on phone. Then open the individual or group chat. After that tap on More options or three dots. Click More and then tap on Export chat. At last select export without media.

3.2.2 PreProcessing

Once exported chat is loaded to the application then the very first requirement is to extract all the information like Date, Time, Day, User, Message, Url etc. Then after Date related manipulations and cleaning process on messages by removing stop-words and punctuations are being followed. Following are six major sub-modules under PreProcessing :

1. Splitting Fields
2. Data frame creation
3. Date format conversion
4. Time format conversion (24 hour)
5. Feature addition
6. Cleaning

Splitting Fields : Using Regular Expression Time-stamp can be extracted from given conversation. In Figure 3.2 chats is split into timestamp and message.



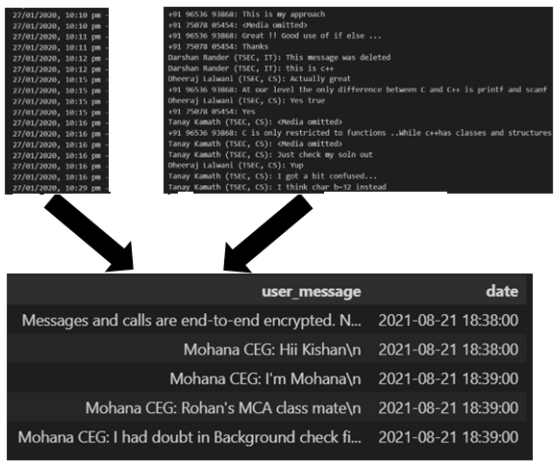
```
27/01/2020, 10:10 pm - +91 96536 93868: This is my approach
27/01/2020, 10:10 pm - +91 75078 05454: <Media omitted>
27/01/2020, 10:11 pm - +91 96536 93868: Great !! Good use of if else ...
27/01/2020, 10:11 pm - +91 75078 05454: Thanks
27/01/2020, 10:12 pm - Darshan Rander (TSEC, IT): This message was deleted
27/01/2020, 10:12 pm - Darshan Rander (TSEC, IT): this is c++
27/01/2020, 10:15 pm - Dheeraj Lalwani (TSEC, CS): Actually great
27/01/2020, 10:15 pm - +91 96536 93868: At our level the only difference between C and C++ is printf and scanf  vala statement
27/01/2020, 10:15 pm - Dheeraj Lalwani (TSEC, CS): Yes true
27/01/2020, 10:15 pm - +91 75078 05454: Yes
27/01/2020, 10:16 pm - Tanay Kamath (TSEC, CS): <Media omitted>
27/01/2020, 10:16 pm - +91 96536 93868: C is only restricted to functions ..While c++has classes and structures vala stuff
27/01/2020, 10:16 pm - Tanay Kamath (TSEC, CS): <Media omitted>
27/01/2020, 10:16 pm - Tanay Kamath (TSEC, CS): Just check my soln out
27/01/2020, 10:16 pm - Dheeraj Lalwani (TSEC, CS): Yup
27/01/2020, 10:16 pm - Tanay Kamath (TSEC, CS): I got a bit confused...
27/01/2020, 10:29 pm - Tanay Kamath (TSEC, CS): I think char b-32 instead
```

```
27/01/2020, 10:10 pm -
27/01/2020, 10:10 pm -
27/01/2020, 10:11 pm -
27/01/2020, 10:11 pm -
27/01/2020, 10:12 pm -
27/01/2020, 10:12 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:29 pm -
```

```
+91 96536 93868: This is my approach
+91 75078 05454: <Media omitted>
+91 96536 93868: Great !! Good use of if else ...
+91 75078 05454: Thanks
Darshan Rander (TSEC, IT): This message was deleted
Darshan Rander (TSEC, IT): this is c++
Dheeraj Lalwani (TSEC, CS): Actually great
+91 96536 93868: At our level the only difference between C and C++ is printf and scanf
Dheeraj Lalwani (TSEC, CS): Yes true
+91 75078 05454: Yes
Tanay Kamath (TSEC, CS): <Media omitted>
+91 96536 93868: C is only restricted to functions ..While c++has classes and structures
Tanay Kamath (TSEC, CS): <Media omitted>
Tanay Kamath (TSEC, CS): Just check my soln out
Dheeraj Lalwani (TSEC, CS): Yup
Tanay Kamath (TSEC, CS): I got a bit confused...
Tanay Kamath (TSEC, CS): I think char b-32 instead
```

Figure 3.2: Separating conversations into timestamp and messages

Data frame creation : Once timestamp and messages are extracted then creation of data-frame will help in further processing. In Figure 3.3 pandas library has been used to create data-frame from separated timestamp and messages.



```
27/01/2020, 10:10 pm -
27/01/2020, 10:10 pm -
27/01/2020, 10:11 pm -
27/01/2020, 10:11 pm -
27/01/2020, 10:12 pm -
27/01/2020, 10:12 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:15 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:16 pm -
27/01/2020, 10:29 pm -
```

```
+91 96536 93868: This is my approach
+91 75078 05454: <Media omitted>
+91 96536 93868: Great !! Good use of if else ...
+91 75078 05454: Thanks
Darshan Rander (TSEC, IT): This message was deleted
Darshan Rander (TSEC, IT): this is c++
Dheeraj Lalwani (TSEC, CS): Actually great
+91 96536 93868: At our level the only difference between C and C++ is printf and scanf
Dheeraj Lalwani (TSEC, CS): Yes true
+91 75078 05454: Yes
Tanay Kamath (TSEC, CS): <Media omitted>
+91 96536 93868: C is only restricted to functions ..While c++has classes and structures
Tanay Kamath (TSEC, CS): <Media omitted>
Tanay Kamath (TSEC, CS): Just check my soln out
Dheeraj Lalwani (TSEC, CS): Yup
Tanay Kamath (TSEC, CS): I got a bit confused...
Tanay Kamath (TSEC, CS): I think char b-32 instead
```

user_message	date
Messages and calls are end-to-end encrypted. N...	2021-08-21 18:38:00
Mohana CEG: Hii Kishan\n	2021-08-21 18:38:00
Mohana CEG: I'm Mohana\n	2021-08-21 18:39:00
Mohana CEG: Rohan's MCA class mate\n	2021-08-21 18:39:00
Mohana CEG: I had doubt in Background check fi...	2021-08-21 18:39:00

Figure 3.3: Transforming timestamp and messages into Data frame

Date format conversion : Since WhatsApp is locally installed on each mobile hence it follow the custom date format according to mobile device. Therefore while exporting it can have one of these formats - **[DD/MM/YYYY, DD/MMM/YY, MM/DD/YYYY, MM/DD/YY etc]**. By deafult python uses date format as **YYYY/MM/DD**,So to handle all formats of date, it is converted from any date format to this specific format.

Time format conversion (24 hour) : Just like date, time is also present in custom format in each mobile device. Few chats may have time in 12 hour format and few may have in 24 hour format. For simplicity all the times has been converted to 24 hour clock format.

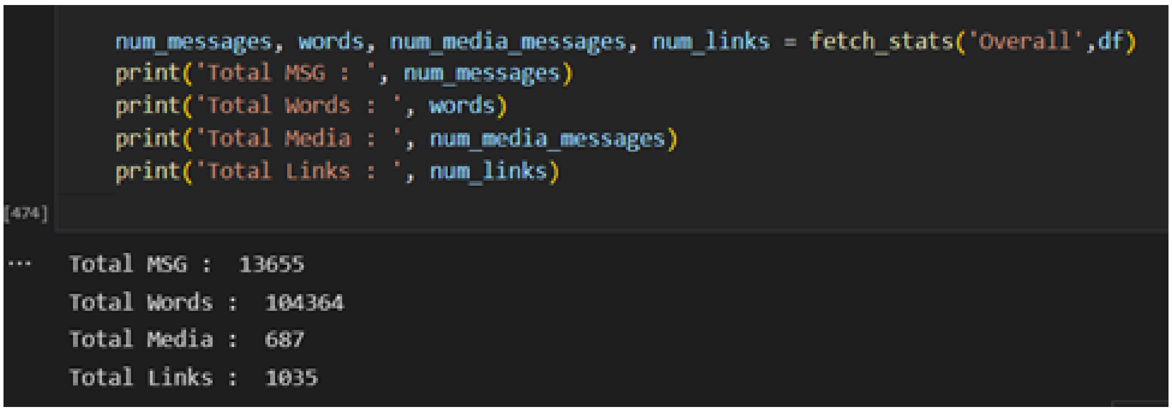
Feature addition : Once regular expression has been performed on timestamp and messages, now other meaningful information can be extracted from all fields. For example Day, Date, Hour, Minute, Day Part, Period etc. has been extracted from timestamp. Similarly URLs, emojis, word count, media count etc. are also extracted from the messages part.

Cleaning : This step includes removal of punctuation, stop words and unnecessary empty data of less than length=2 data.

3.2.3 Exploratory Data Analysis

This module is mainly used for visualisation and observation. Here we have used bar graph , line chart, pie chart etc. on the basis of count, most engagement etc. This modules has various sub-modules like :

Fetch Stats : This module is used for filtering the users, URLs, Message and words and their counting etc. by going through all the messages. To identify an URL URLSurf library and regular expression has been used. In Figure 3.4 statistical information from chats has been calculated.



```
num_messages, words, num_media_messages, num_links = fetch_stats('Overall',df)
print('Total MSG : ', num_messages)
print('Total Words : ', words)
print('Total Media : ', num_media_messages)
print('Total Links : ', num_links)
[474]
...  Total MSG :  13655
     Total Words :  104364
     Total Media :  687
     Total Links :  1035
```

Figure 3.4: Statistical Information

Timeline : This module is responsible for analysing the time in a day or weekday during which most conversation has been occurred. It also helps in predicting that which tine us best to send msg so that we can get response. In Figure 3.5, an day wise message activity has been displayed.

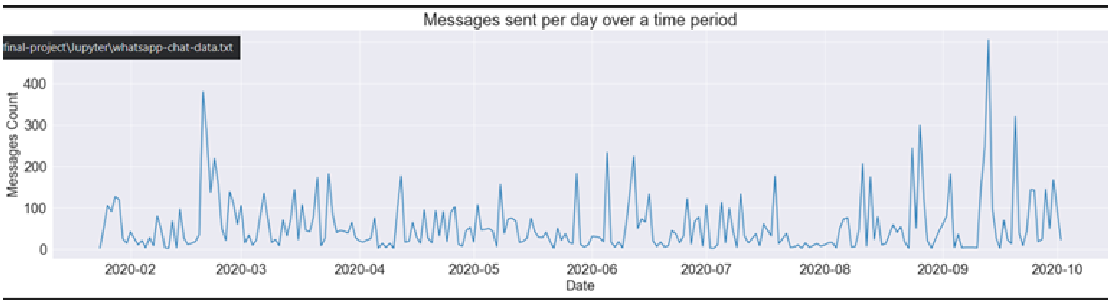


Figure 3.5: Day wise message activity throughout year

In figure 3.5, day wise message activity throughout the year has been displayed and higher raised line states that most conversation occur in chats in a that day. The intuition behind it is count total number of message day wise and sort them in descending order and plot it on graph.

Top Users and Message Length : This module is responsible for performing the text mining techniques like classification on WhatsApp chats. After performing these instructions, 2D data frame with relevant records will be evaluated. Mainly two different functionality has been performed here which are :

- Separating user message and applying count to identify busiest user.
- Fetching top users from chat by most no of messages and average length of messages.

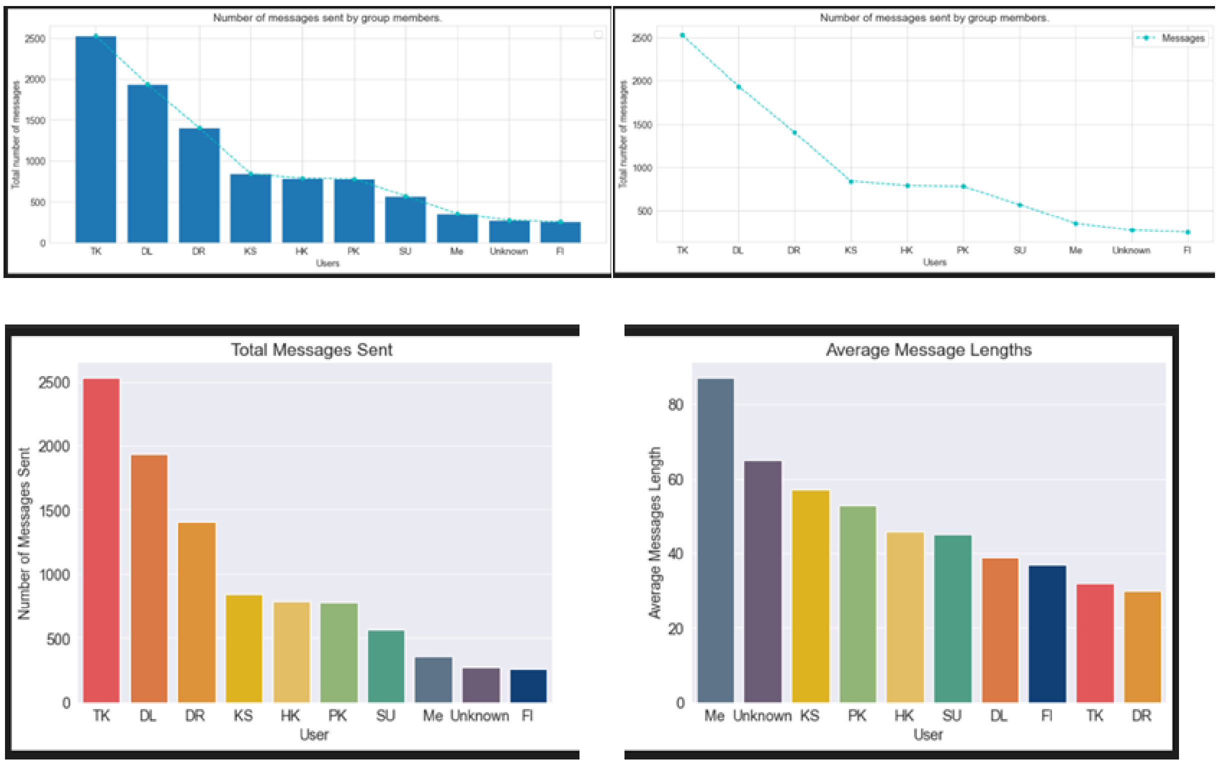


Figure 3.6: Total Message sent and Average length of Message by User

Top 10 members with most sent messages has been displayed in Figure 3.6 and average word length has been displayed here to detect who send long message normally. Here matplotlib bar chart and line chart has been used for visualization.

Activity Map shows the message frequency during all 24 hours in week in terms of color. More brighter the color states more no of messages sent. Heatmap from seaborn library has been used in Figure 3.7 to display the participation of members in group chat in 24 hours of the day during weeks.

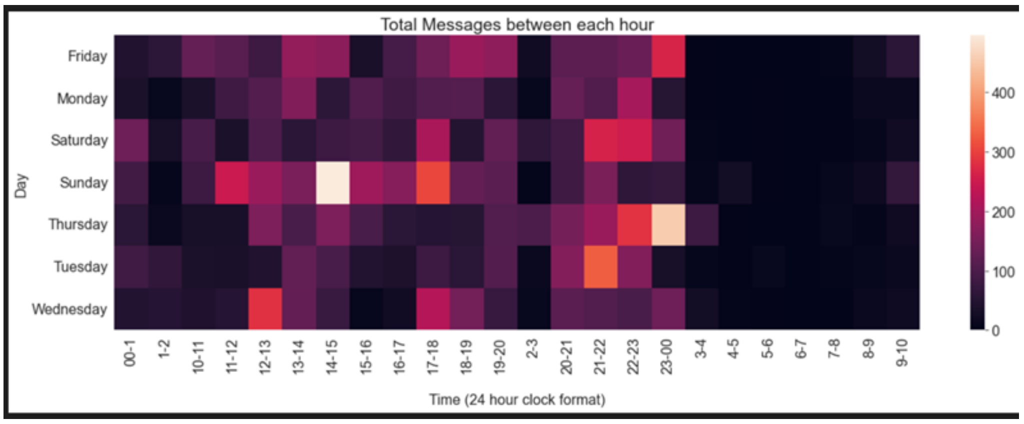


Figure 3.7: Heat Map of Messages during each hour in week

Word Cloud is a powerful visual representation object for text processing, which shows the most frequent word with bigger and bolder letters, and with different colors. The smaller the the size of the word the lesser it's important.

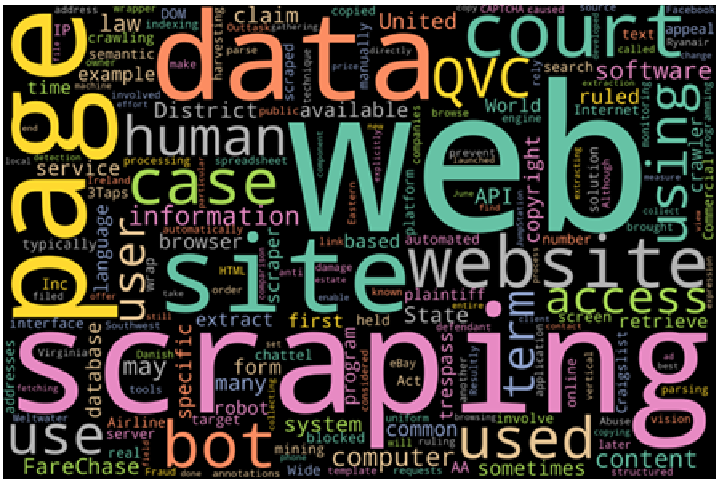


Figure 3.8: Word cloud of most used words

Figure 3.8 states the most used words which has been filtered after removing punctuation and stopwords.

3.3 FastText Word Vector Model

Word embedding techniques were considered as an important factor in sentiment evaluation. In this project, fastText word embedding technique is used. FastText is word embedding method that is an extension of the word2vec model. Instead of using words to build word embedding, FastText goes one level deeper. This deeper level consists of part of words and characters. For example, the word, "artificial" with n=3, the fastText representation of this word is $\langle ar, art, rti, tif, ifi, fic, ici, ial, al \rangle$, where the angular brackets indicate the beginning and end of the word.

There are two main advantages to this approach. First, generalization is possible as long as new words have the same characters as known ones. Second, less training data is needed since more information can be extracted from each piece of text.

3.4 LSTM

LSTM stands for long short-term memory networks, used in the field of Deep Learning. It is a variety of recurrent neural networks (RNNs) that are capable of learning long-term dependencies, especially in sequence prediction problems. LSTM has feedback connections, i.e., it is capable of processing the entire sequence of data.

The role of an LSTM model is held by a memory cell known as a **cell state** that maintains its state over time. The cell state is the horizontal line that runs through the top of the below diagram. Information can be added to or removed from the cell state in LSTM and is regulated by gates. These gates optionally let the information flow in and out of the cell. It contains a point-wise multiplication operation and a sigmoid neural net layer that assist the mechanism. The sigmoid layer gives out numbers between zero and one, where zero means nothing should be let through, and one means everything should be let through.

3.4.1 Architecture of LSTM

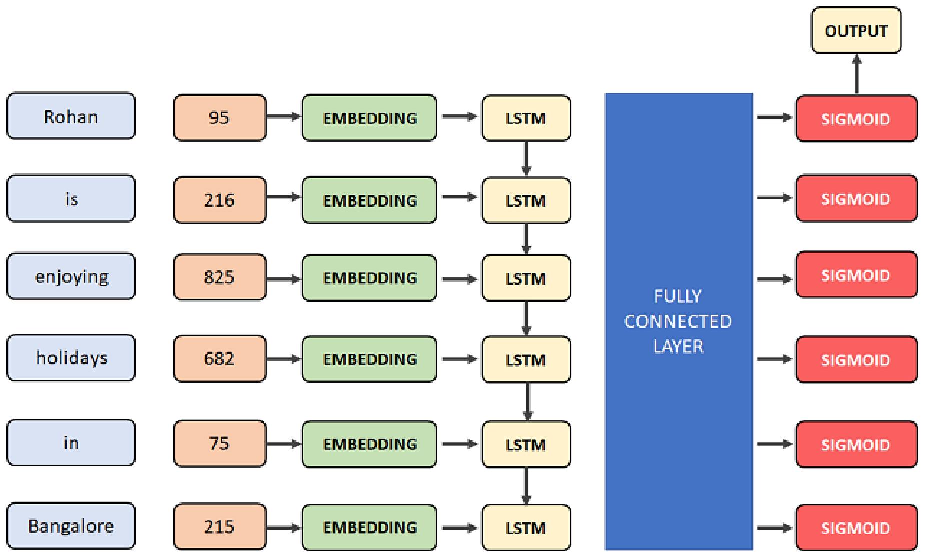


Figure 3.9: LSTM Architecture

Figure 3.9 shows that LSTM network has mainly 5 layers. but more layers can be hyper tuned in order to improve the accuracy of model. Before passing data to model, it is necessary to convert the words into tokens with integer. following are 5 layers of LSTM:

- Embedding Layer
- LSTM Layer
- Fully Connected Layer
- Sigmoid Activation Layer
- Output Layer

Embedding Layer is responsible for converting word tokens into embedding of a specific size like 256, 512 etc. This layer maps a sequence of word indices to embedding vectors and learns the word embedding during training

LSTM Layer is responsible for processing data sequentially and keep its hidden state through time. It keep previous memory and process with current memory and on the basis of intermediate operation it decides whether to keep current memory or remain with old memory.

Fully Connected Layer in a neural networks are those layers where all the inputs from one layer are connected to every activation unit of the next layer. Similarly here fully connected layer maps output of LSTM layer to a specified output.

Sigmoid Activation Layer is responsible for converting all output values in the range of 0 to 1. It means it can either let no flow or complete flow of information throughout the gates.

Output Layer is the final layer in the architecture. it is responsible for generating the output which is obtained from the output of the last sigmoid layer. The size of output is 2D array of real numbers. The first dimension is indicating the number of samples in the batch given to the LSTM layer. The second dimension is the dimensionality of the output space defined by the units parameter in Keras LSTM implementation.

3.5 BI-LSTM

Bidirectional LSTM (BiLSTM) is a recurrent neural network used primarily on natural language processing. Unlike standard LSTM, the input flows in both directions, and it's capable of utilizing information from both sides.BiLSTM adds one more LSTM layer, which reverses the direction of information flow. Briefly, it means that the input sequence flows backward in the additional LSTM layer. Then we combine the outputs from both LSTM layers in several ways, such as average, sum, multiplication, or concatenation. Figure 3.10 shows the basic architecture of BI-Directional LSTM network.

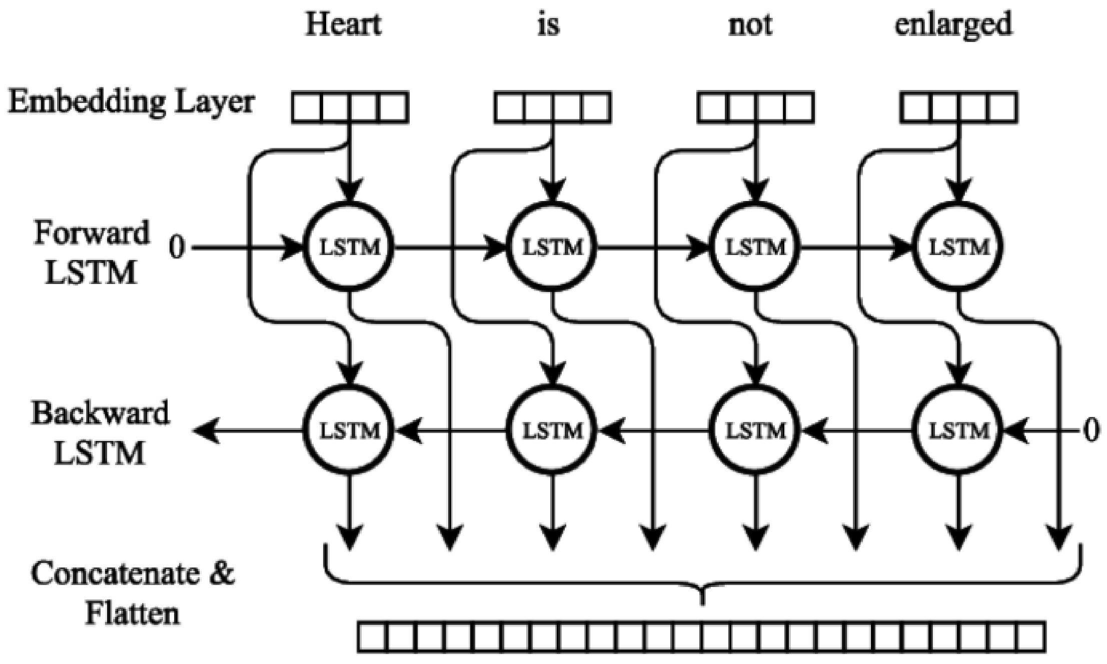


Figure 3.10: Bi-LSTM Architecture

This chapter is about system design and architecture of the project and its implementation where each module details has been discussed. In next chapter, Algorithms and Pseudo-Code has been discussed. For EDA, NLP and statistical approach has been discussed and for Emotion Detection, LSTM and BI-LSTM algorithm has been stated.

CHAPTER 4

ALGORITHM AND PSEUDO-CODE

This section explains in details the various modules in the system. Each module includes the input for the module, process flow for the module and output for the module in detail.

4.1 EXPLORATORY DATA ANALYSIS USING PYTHON LIBRARIES

Python programming language has a rich set of libraries for various work. Here we have used streamlit framework to build the interface. During EDA we are using various libraries like Numpy, Pandas, Seaborn, Matplotlib, Emoji and nltk etc. Exploratory analysis on WhatsApp comes under the category of text mining in Natural Language Processing. Here we are using feature extraction, regular expressions operations, clustering and classification etc. on pre-processed WhatsApp data. Various kind of charts like bar-chart, pie chart, column chart, line chart and scatter plots comes under Matplotlib library. Pandas and Numpy library has been used as they contains lot of methods to support Data-frame related stuff. Final Output is generated as various kind of charts and graph in form of visualization.

Following are methods implemented in helper module. Each methods take dataset as argument and perform some operations on it.

1. emoji_extract(df)
2. most_common_words(df)
3. message_cluster(df)

4.1 Algorithm for Exploratory Data Analysis

```
1: Start
2: Import numpy, pandas, matplotlib etc.
3: data = read_dataset().
4: time_stamp, message = split_data(data).
5: time, date = split_timestamp(time_stamp).
6: user, msg = split_message(message)
7: pre_data = pre_processing(data)
8: pre_data ← This is the modified dataset after Pre-Processing
9: analyse_url(msg)
10: analyse_emoji(msg)
11: analyse_sentiment(msg)
12: time_series(msg)
13: activity_map(daily/weekly)
14: statistic_calculation(msg)
15: scatter_plot(user_activity))
16: word_cloud(most_common_word)
17: End
```

INPUT: WhatsApp Chat File (Txt).

OUTPUT: bar_chart, pie_chart, scatter_chart, word_cloud etc.

In algorithm 4.1 initially all the required libraries has been imported. After that dataset has been stored in dataframe. **Split_data( )** method take data frame as input and return the timestamp and message separately as output. **Split_timestamp()** is responsible for splitting the timestamp into date and time and Once all information are separated then **pre_processing( )** function take raw dataframe and return the modified and prepossessed dataset. **analyse_url()** extracts the urls and **analyse_emoji()** extracts the emoji from chat. To check the polarity score **analyse_sentiment()** has been used which returns answer between 0 to 1. **time_series(msg)** method returns the line chart of day wise messages throughout whole year and to display result in form of Heatmap **activity_map()** has been used. Method **statistic_calculation()** return the statistical details. At last **word_cloud()** returns the wordcloud display of most used text.

4.2 EMOTION DETECTION USING LSTM

Long Short Term Memory networks also known as LSTMs – are a special kind of RNN, capable of learning long-term dependencies. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer. The central role of an LSTM model is held by a memory cell known as a 'cell state' that maintains its state over time. The cell state is the horizontal line that runs through the top of the below diagram. Information can be added to or removed from the cell state in LSTM and is regulated by gates. These gates optionally let the information flow in and out of the cell. It contains a pointwise multiplication operation and a sigmoid neural net layer that assist the mechanism.

4.2 Algorithm for LSTM MODEL

```
 1: Start
 2: import numpy,pandas,LabelEncoder,train_test_split,Tokenizer
 3: from keras.layers import Dense, LSTM, Embedding, Dropout, Activation,
    Conv1D, MaxPooling1D
 4: df_train = pd.read_csv('train.txt', names=['Text', 'Emotion'])
 5: y_train = LabelEncoder().fit_transform(y_train)
 6: tokenizer = Tokenizer()
 7: tokenizer.fit_on_texts(pd.concat([X_train, X_test], axis=0))
 8: X_train = pad_sequences(sequences_train, maxlen=256 truncating='pre')
 9: model = Sequential()
10: model.add(Embedding(vocabSize, embedding_size, input_length=maxlen))
11: model.add(Dropout(0.25))
12: model.add(Conv1D(filters, kernel_size, padding='valid', activation='relu'))
13: model.add(MaxPooling1D(pool_size=pool_size))
14: model.add(LSTM(lstm_output_size))
15: model.add(Dense(4))
16: model.add(Activation('softmax'))
17: emotion = Predict(sentence)
18: End
```

INPUT: A text sequence or a sentence

OUTPUT: Got Accuracy of 83% to 95% in Emotion Prediction depending the sentence context.

4.3 EMOTION DETECTION USING BI-LSTM

These are like an upgrade over LSTMs. In bidirectional LSTMs, each training sequence is presented forward and backward so as to separate recurrent nets. Both sequences are connected to the same output layer. Bidirectional LSTMs have all the information about every point in given sequence, everything before and after it. Conventional recurrent neural networks are only capable of using the previous context to get information. Whereas, in bidirectional LSTMs, the information is obtained by processing the data in both directions within two hidden layers, pushed toward the same output layer. This helps bidirectional LSTMs access long-range context in both directions.

FastText is another word embedding method that is an extension of the word2vec model. Instead of learning vectors for words directly, fastText represents each word as an n-gram of characters. So, for example, take the word, “artificial” with n=3, the fastText representation of this word is **[ar, art, rti, tif, ifi, fic, ici, ial, al]**, This helps capture the meaning of shorter words and allows the embeddings to understand suffixes and prefixes. Once the word has been represented using character n-grams, a skip-gram model is trained to learn the embeddings. This model is considered to be a bag of words model with a sliding window over a word because no internal structure of the word is taken into account. As long as the characters are within this window, the order of the n-grams doesn’t matter. fastText works well with rare words. So even if a word wasn’t seen during training, it can be broken down into n-grams to get its embedding.

4.3.1 Algorithm for BI-LSTM MODEL

```
import Bidirectional,Tokenizer, Dense, Embedding, Dropout
df_train = pd.read_csv('train.txt', names=['Text', 'Emotion'])
y_train = LabelEncoder().fit_transform(y_train)
tokenizer.fit_on_texts(pd.concat([X_train, X_test], axis=0))
X_train = pad_sequences(sequences_train, maxlen=256 truncating='pre')
model = Sequential()
model.add(Embedding(input_dim,output_dim,input_len,weights,trainable))
model.add(Bidirectional(LSTM(128, return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(256, return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(128, return_sequences=False)))
model.add(Dense(6,activation='softmax'))
model.compile(loss,optimizer='adam', metrics='accuracy')
emotion = Predict(sentence)
```

INPUT: A sentence or a text sequence.

OUTPUT: Got Accuracy of 87% to 92% in Emotion Prediction depending the sentence context.

In this chapter, three algorithms has been discussed. For EDA, mainly python libraries such as pandas, numpy , seaborn, matplotlib etc. has been used and for implementation streamlit framework has been used. For text based emotion detection LSTM and BI-LSTM algorithm has been used and both the algorithm takes a sentence as input and predict emotion. In

In next chapter, Implementation of project has been discussed with input and output interface. Also LSTM and BI-LSTM Model summary has been shown in next chapter.

CHAPTER 5

IMPLEMENTATION AND ANALYSIS

5.1 EXPLORATORY DATA ANALYSIS

This section includes the implementation of Exploratory Data Analysis on WhatsApp Chat. It involves various visualization tools like bar chart, pie-chart, Heat-map, Line-chart, scatter bubble plot etc.

5.1.1 Exporting the chat file from WhatsApp

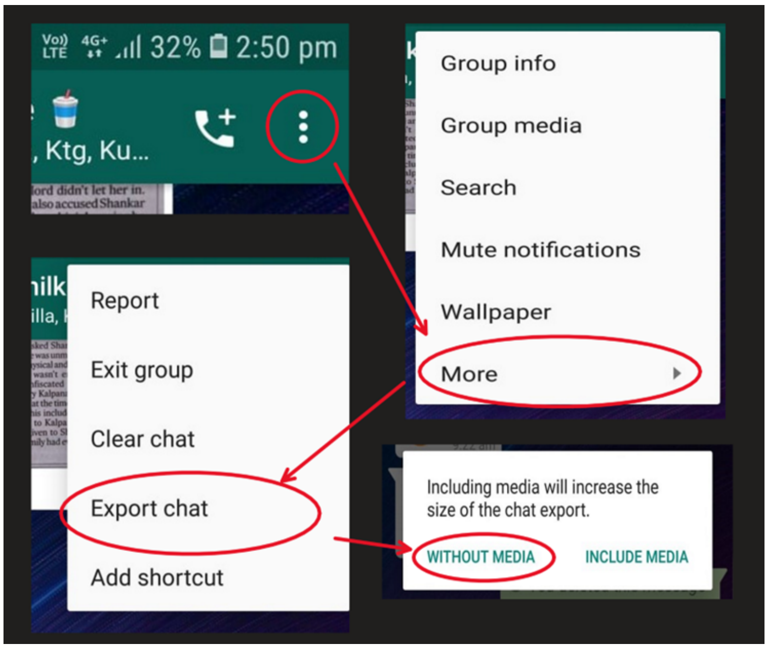


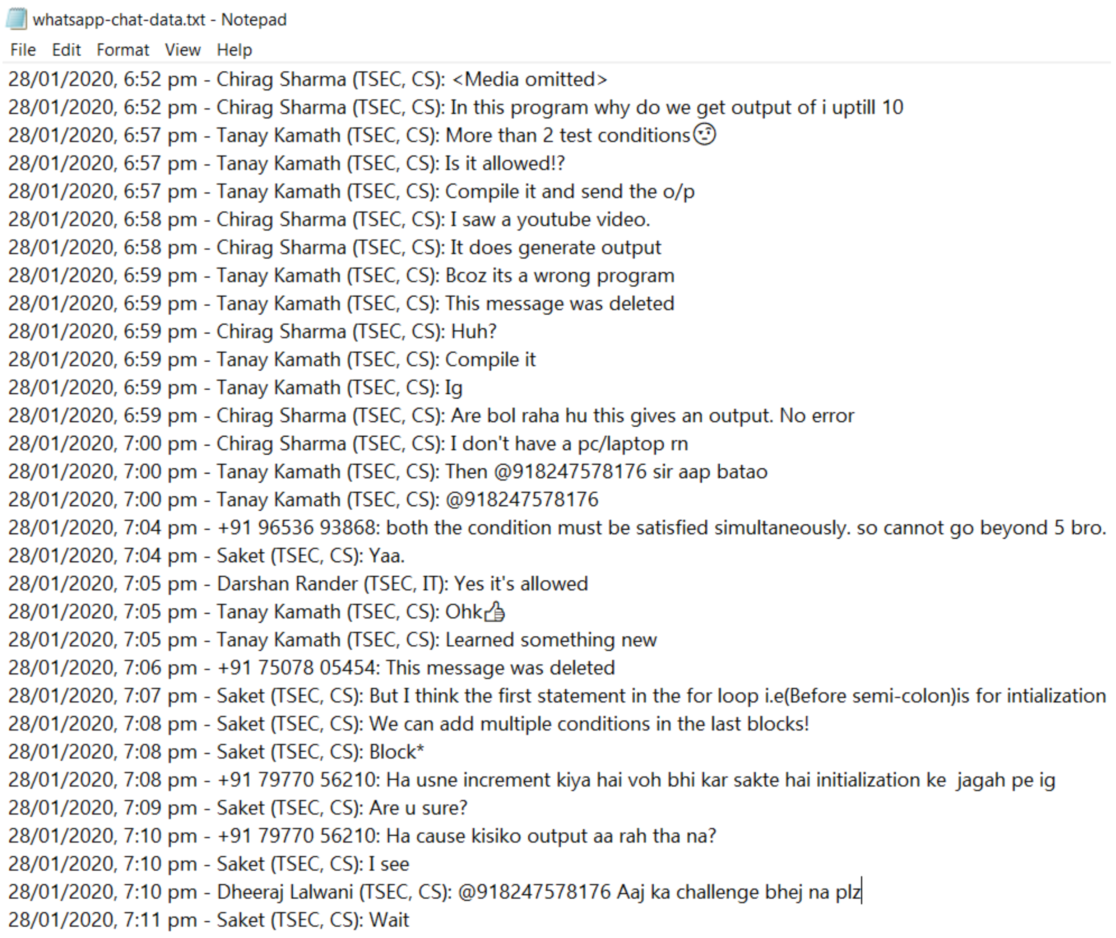
Figure 5.1: Exporting chat file from WhatsApp

For Exporting the chat, Launch the WhatsApp on phone > open the individual or group chat > tap on More options or three dots > Click More, then tap on Export chat > Choose export without media.

5.1.2 Benchmark Dataset of WhatsApp Group Chat

Initial raw dataset or the exported whats-app chat file has a fix format of **[Timestamp-MemberName-Messages]**. This dataset contains around 12000 records. After uploading chat file, it is pre-processed through several stages and finally pre-processed dataset is generated on which further operations are being performed. Multiple regular expressions has been used during pre-processing.

Raw Dataset

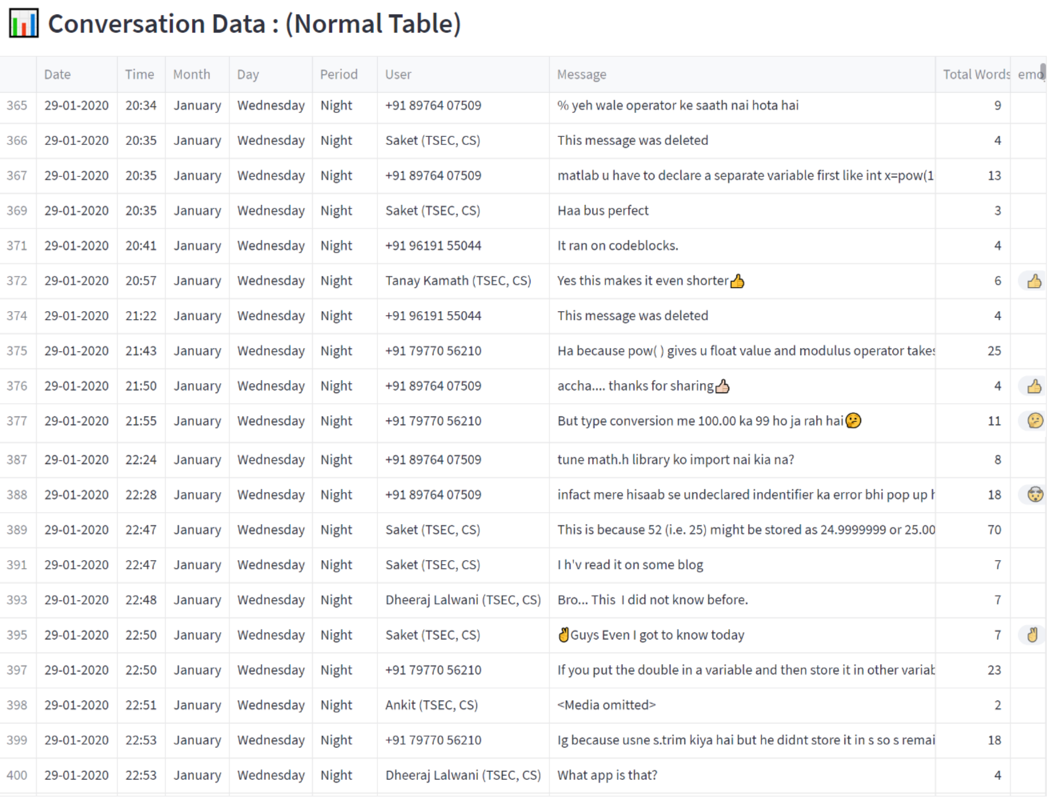


```
28/01/2020, 6:52 pm - Chirag Sharma (TSEC, CS): <Media omitted>
28/01/2020, 6:52 pm - Chirag Sharma (TSEC, CS): In this program why do we get output of i uptill 10
28/01/2020, 6:57 pm - Tanay Kamath (TSEC, CS): More than 2 test conditions😕
28/01/2020, 6:57 pm - Tanay Kamath (TSEC, CS): Is it allowed!?
28/01/2020, 6:57 pm - Tanay Kamath (TSEC, CS): Compile it and send the o/p
28/01/2020, 6:58 pm - Chirag Sharma (TSEC, CS): I saw a youtube video.
28/01/2020, 6:58 pm - Chirag Sharma (TSEC, CS): It does generate output
28/01/2020, 6:59 pm - Tanay Kamath (TSEC, CS): Bcoz its a wrong program
28/01/2020, 6:59 pm - Tanay Kamath (TSEC, CS): This message was deleted
28/01/2020, 6:59 pm - Chirag Sharma (TSEC, CS): Huh?
28/01/2020, 6:59 pm - Tanay Kamath (TSEC, CS): Compile it
28/01/2020, 6:59 pm - Tanay Kamath (TSEC, CS): Ig
28/01/2020, 6:59 pm - Chirag Sharma (TSEC, CS): Are bol raha hu this gives an output. No error
28/01/2020, 7:00 pm - Chirag Sharma (TSEC, CS): I don't have a pc/laptop rn
28/01/2020, 7:00 pm - Tanay Kamath (TSEC, CS): Then @918247578176 sir aap batao
28/01/2020, 7:00 pm - Tanay Kamath (TSEC, CS): @918247578176
28/01/2020, 7:04 pm - +91 96536 93868: both the condition must be satisfied simultaneously. so cannot go beyond 5 bro.
28/01/2020, 7:04 pm - Saket (TSEC, CS): Yaa.
28/01/2020, 7:05 pm - Darshan Rander (TSEC, IT): Yes it's allowed
28/01/2020, 7:05 pm - Tanay Kamath (TSEC, CS): Ohk👍
28/01/2020, 7:05 pm - Tanay Kamath (TSEC, CS): Learned something new
28/01/2020, 7:06 pm - +91 75078 05454: This message was deleted
28/01/2020, 7:07 pm - Saket (TSEC, CS): But I think the first statement in the for loop i.e(Before semi-colon)is for intialization
28/01/2020, 7:08 pm - Saket (TSEC, CS): We can add multiple conditions in the last blocks!
28/01/2020, 7:08 pm - Saket (TSEC, CS): Block*
28/01/2020, 7:08 pm - +91 79770 56210: Ha usne increment kiya hai voh bhi kar sakte hai initialization ke  jagah pe ig
28/01/2020, 7:09 pm - Saket (TSEC, CS): Are u sure?
28/01/2020, 7:10 pm - +91 79770 56210: Ha cause kisiko output aa rah tha na?
28/01/2020, 7:10 pm - Saket (TSEC, CS): I see
28/01/2020, 7:10 pm - Dheeraj Lalwani (TSEC, CS): @918247578176 Aaj ka challenge bhej na plz
28/01/2020, 7:11 pm - Saket (TSEC, CS): Wait
```

Figure 5.2: Raw Dataset : Default Generated dataset from WhatsApp

Pre Processed Dataset

Real world chat conversations contains lot of shortcut words, missing values, blank spaces and multiple punctuation which are not useful in exploration and visualisation. Pre-processing need to be performed to clean the dataset and extracting and separating from the data.



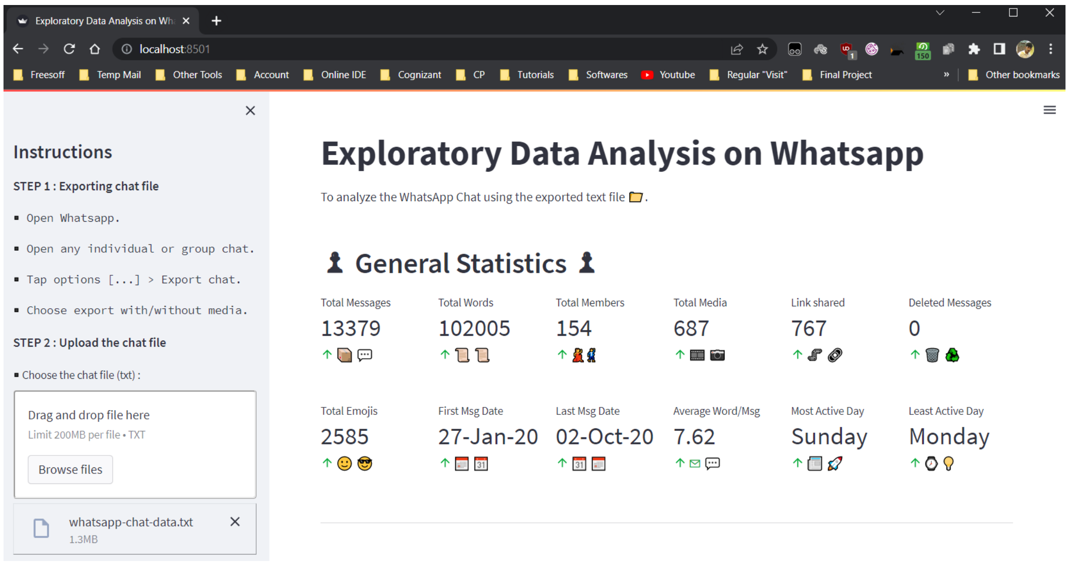
📊 Conversation Data : (Normal Table)

	Date	Time	Month	Day	Period	User	Message	Total Words	emo
365	29-01-2020	20:34	January	Wednesday	Night	+91 89764 07509	% yeh wale operator ke saath nai hota hai	9	
366	29-01-2020	20:35	January	Wednesday	Night	Saket (TSEC, CS)	This message was deleted	4	
367	29-01-2020	20:35	January	Wednesday	Night	+91 89764 07509	matlab u have to declare a separate variable first like int x=pow(1	13	
369	29-01-2020	20:35	January	Wednesday	Night	Saket (TSEC, CS)	Haa bus perfect	3	
371	29-01-2020	20:41	January	Wednesday	Night	+91 96191 55044	It ran on codeblocks.	4	
372	29-01-2020	20:57	January	Wednesday	Night	Tanay Kamath (TSEC, CS)	Yes this makes it even shorter👍	6	👍
374	29-01-2020	21:22	January	Wednesday	Night	+91 96191 55044	This message was deleted	4	
375	29-01-2020	21:43	January	Wednesday	Night	+91 79770 56210	Ha because pow( ) gives u float value and modulus operator take	25	
376	29-01-2020	21:50	January	Wednesday	Night	+91 89764 07509	accha.... thanks for sharing👍	4	👍
377	29-01-2020	21:55	January	Wednesday	Night	+91 79770 56210	But type conversion me 100.00 ka 99 ho ja rah hai🤔	11	🤔
387	29-01-2020	22:24	January	Wednesday	Night	+91 89764 07509	tune math.h library ko import nai kia na?	8	
388	29-01-2020	22:28	January	Wednesday	Night	+91 89764 07509	infact mere hisaab se undeclared indentifier ka error bhi pop up	18	😳
389	29-01-2020	22:47	January	Wednesday	Night	Saket (TSEC, CS)	This is because 52 (i.e. 25) might be stored as 24.9999999 or 25.00	70	
391	29-01-2020	22:47	January	Wednesday	Night	Saket (TSEC, CS)	I h'v read it on some blog	7	
393	29-01-2020	22:48	January	Wednesday	Night	Dheeraj Lalwani (TSEC, CS)	Bro... This I did not know before.	7	
395	29-01-2020	22:50	January	Wednesday	Night	Saket (TSEC, CS)	✌Guys Even I got to know today	7	✌
397	29-01-2020	22:50	January	Wednesday	Night	+91 79770 56210	If you put the double in a variable and then store it in other varia	23	
398	29-01-2020	22:51	January	Wednesday	Night	Ankit (TSEC, CS)	<Media omitted>	2	
399	29-01-2020	22:53	January	Wednesday	Night	+91 79770 56210	Ig because usne s.trim kiya hai but he didnt store it in s so s remai	18	
400	29-01-2020	22:53	January	Wednesday	Night	Dheeraj Lalwani (TSEC, CS)	What app is that?	4	

Figure 5.3: Pre-Processed dataset

In Figure 5.3 pre-processed form of raw dataset has been displayed. where new features has been added by extracting the data, time, day_time, user, messages separately from the chats. Apart from this emojis, urls and statistical information has also been extracted from the raw chat file. On the pre-processed dataset all the EDA process has been preformed.

5.1.3 Dashboard With Statistical Information



Instructions

STEP 1 : Exporting chat file

- `Open Whatsapp.`
- `Open any individual or group chat.`
- `Tap options [...] > Export chat.`
- `Choose export with/without media.`

STEP 2 : Upload the chat file

- Choose the chat file (txt) :

Drag and drop file here
Limit 200MB per file • TXT

whatsapp-chat-data.txt
1.3MB

Exploratory Data Analysis on Whatsapp

To analyze the WhatsApp Chat using the exported text file.

General Statistics

Total Messages	Total Words	Total Members	Total Media	Link shared	Deleted Messages
13379	102005	154	687	767	0

Total Emojis	First Msg Date	Last Msg Date	Average Word/Msg	Most Active Day	Least Active Day
2585	27-Jan-20	02-Oct-20	7.62	Sunday	Monday

Figure 5.4: Statistical information from WhatsApp chats

Figure 5.4 states all the statistical details which are being calculated with the help of numpy and pandas libraries. **Total Message** has been calculated via counting total no of records in dataset. Similarly **total words** has also been counted. Date of first line is considered as first message date and date at last line as last message date. Using urlsurf library, urls has been counted withing message. For counting total media shared , a pattern name <Media Omitted> has been used. For counting the total deleted message **deleted** word pattern has been used as WhatsApp use it by default. Emoji library has been used to detect emojis within the text message. All the information has been fetched using python libraries such as numpy, pandas and RegEx. All the above operations has been performed on pre-processed dataset

Dashboard has been designed using streamlit library. Here uploading chat file is given in sidebar of web interface and all the visualised data are displayed in main body section.

5.1.4 Year Wise Message Activity Using Responsive Line Chart

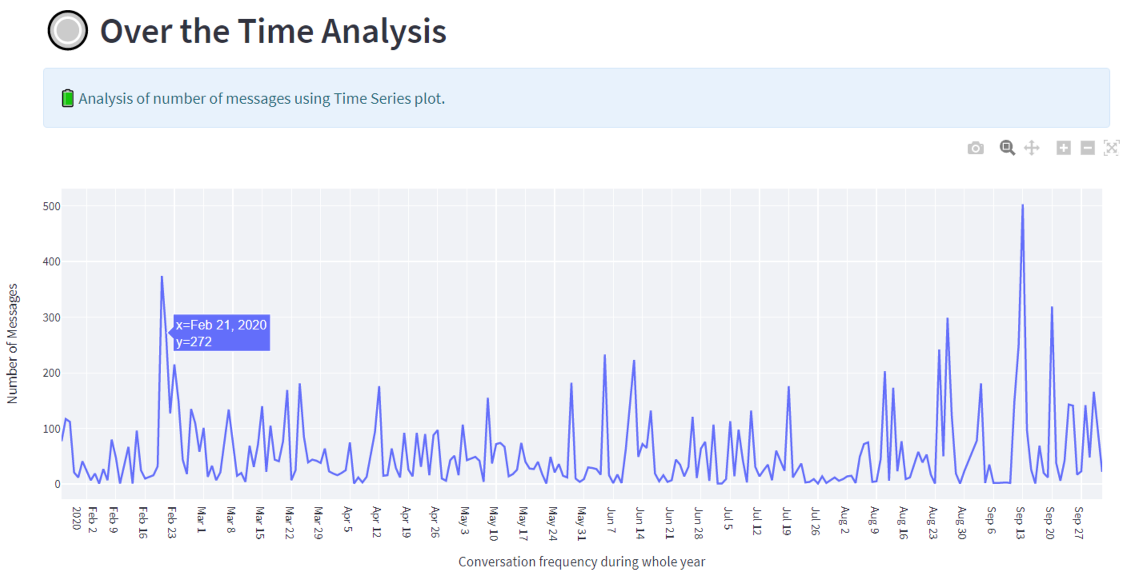


Figure 5.5: Line chart of year wise message activity

Frequency of message per day has been displayed in figure 5.5 using responsive line chart. First using the value_count() method frequency of daily messages has been counted. Here x-axis represents date of messages and y-axis represents total messages sent.

5.1.5 Monthly and Weekly message activity

In figure 5.6 (left side), Number of messages during each month has been displayed using a bar graph. Using value_count() method frequency of messages has been counted month wise and being displayed using bar chart where x and y-axis represents month names and number of messages respectively.

In part-2 (right side) frequency of messages during each day of week has been displayed. First messages has been grouped on the basis of day and then it is displayed in the descending order of most messages sent.

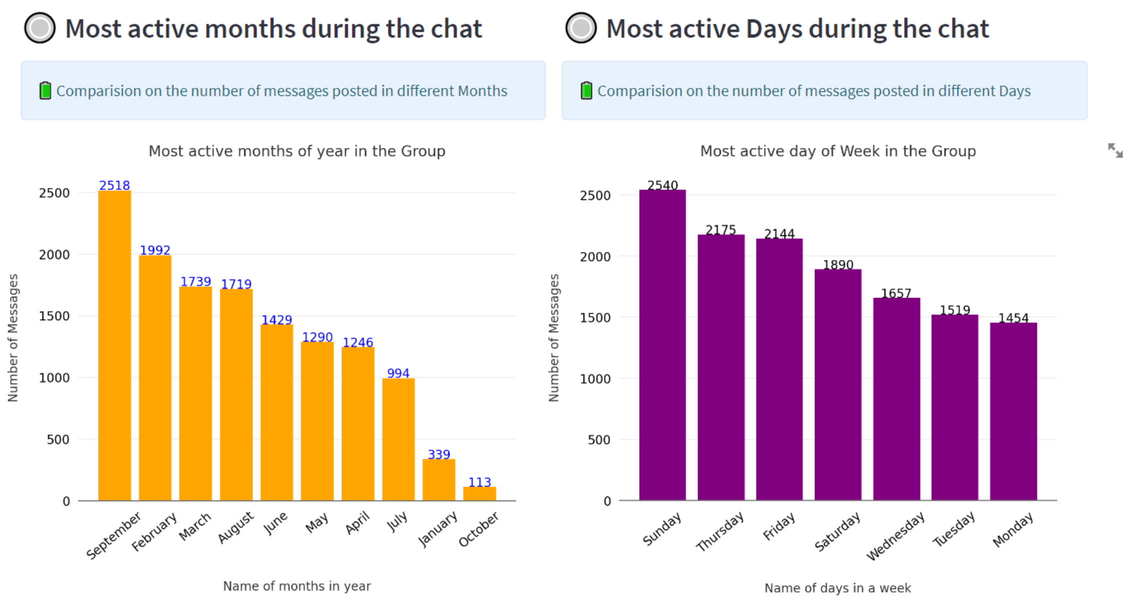


Figure 5.6: Month wise and weekday wise message activity

5.1.6 Most Active Time in 24 hour During Weeks

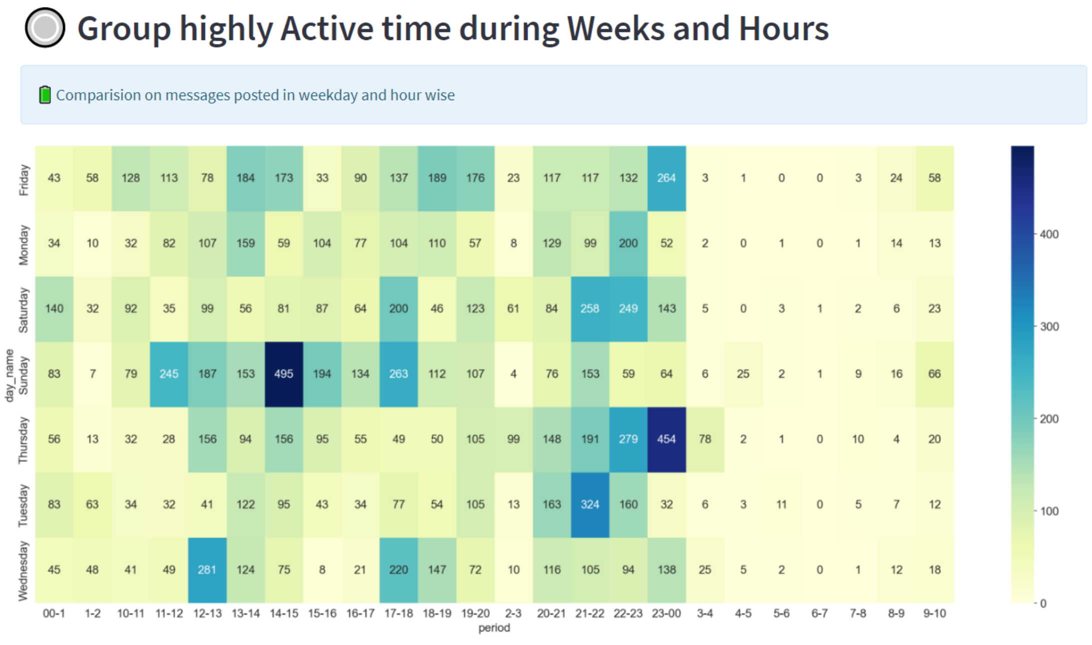


Figure 5.7: 24 hours activity during whole week

5.1.7 Most active hours in a day

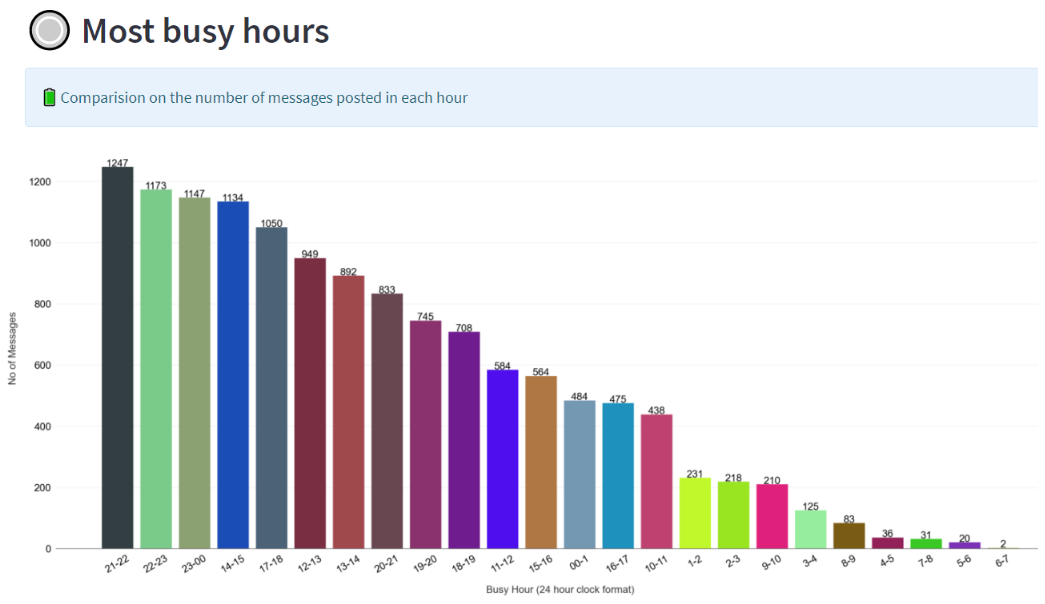


Figure 5.8: Most active hours during a day

In Figure 5.8 most active hours in a day has been calculated. First frequency of messages has been grouped using time period like 1-2, 1-3 etc. Then according to descending order it has been plotted graph where x-axis represents the time during the day (in 24 hour clock) format and y-axis represents the number of messages during that hour. Using seaborn library , color pattern of bars are being displayed. It will help as what is perfect time to send messages in which there is more chance of getting reply as most number of messages have been sent at that time.

5.1.8 Clustering User activity

In Figure 5.9 Clustering has been preformed on member overall activity. Here scatter bubble plotting has been used to show the clustering bubble. For representing the bubbles in 3d planes, three fields total_message, word_count and letter_count has been considered as x-axis, y-axis and z-axis.

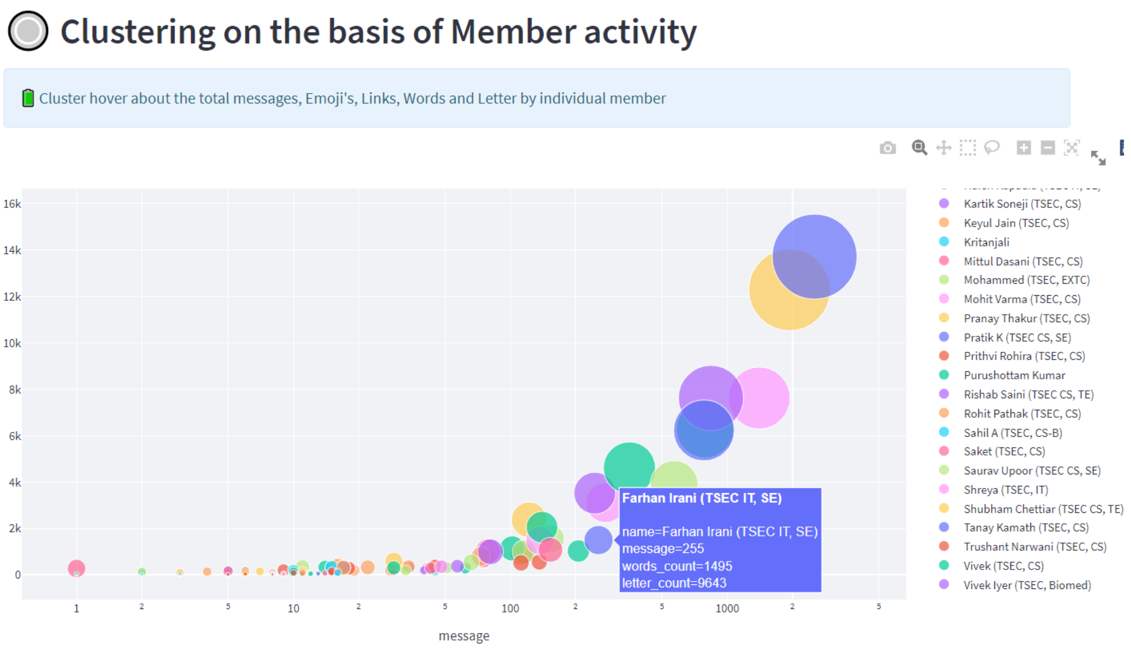


Figure 5.9: Clustering of members on the basis of activities

5.1.9 Top 25 used words in the chat

Most frequent used word has been displayed in Figure 5.10 with their count. Cleaning, punctuation removal, tokenization and stopwords has been applied. at last frequency has been counted and displayed.

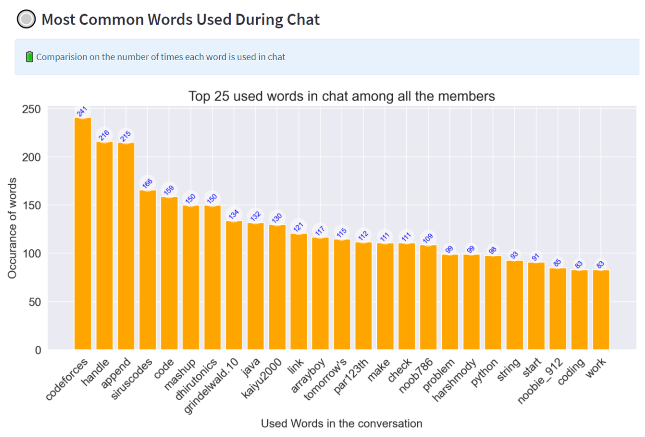
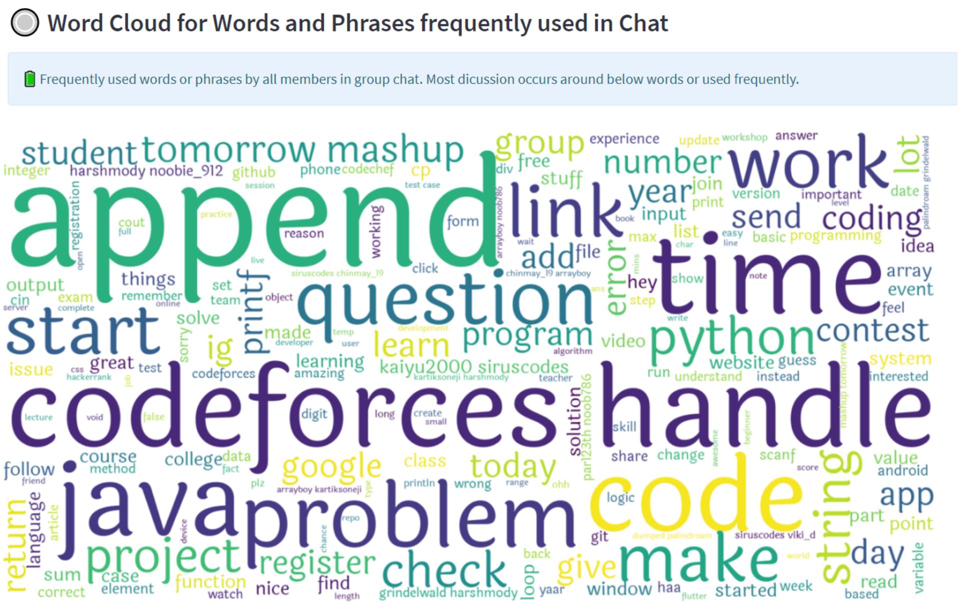


Figure 5.10: Top 25 most used words

5.1.10 Wordcloud of most used words in chat



Word Cloud for Words and Phrases frequently used in Chat

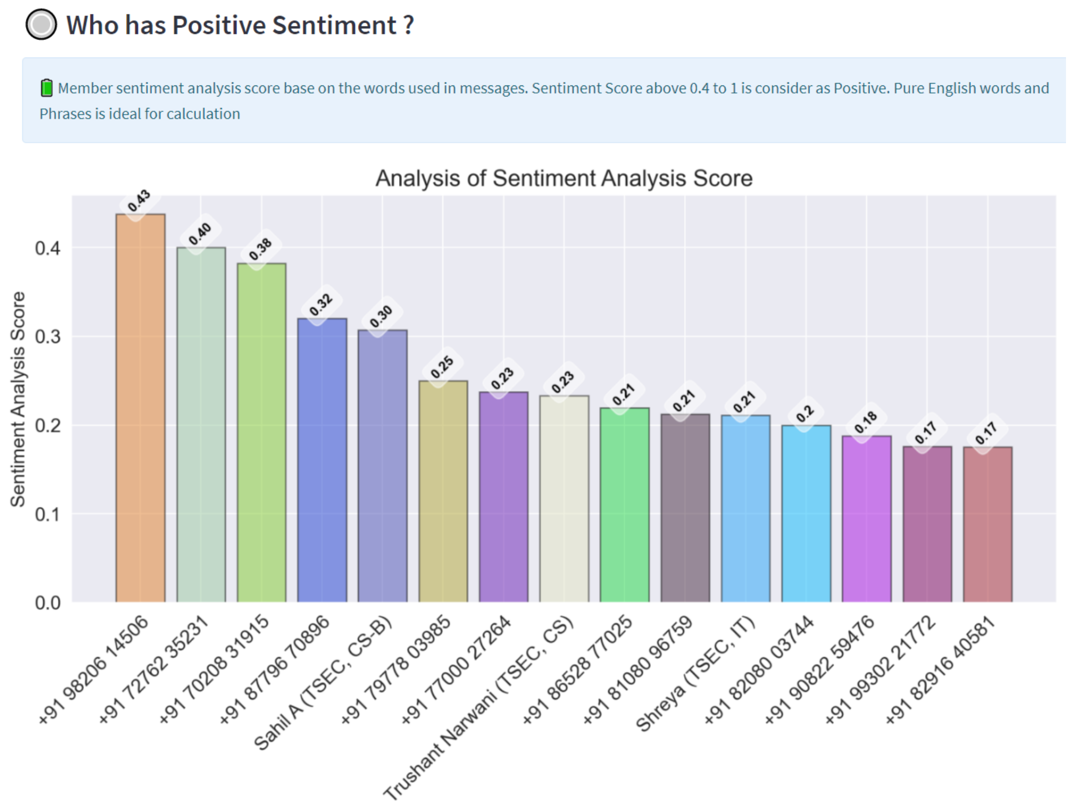
Frequently used words or phrases by all members in group chat. Most dicussion occurs around below words or used frequently.

Figure 5.11: Wordcloud of words

In Figure 5.11, Wordcloud has been displayed for the top 100 most frequent used words. Word cloud is a technique for visualising frequent words in a text where the size of the words represents their frequency. More bigger the font size means most number of times it has occur.

5.1.11 Positive sentiment calculation using Polarity Score

Polarity score is used to analyse sentiment in texts. It takes a text as input and return the value which lies in the range of [-1,1] where 1 means positive statement and (-1) means a negative statement and score 0 means neutral. There are a few NLP libraries existing in Python such as Spacy, NLTK, gensim, TextBlob, etc. that can be used to calculate polarity score.



Who has Positive Sentiment ?

Member sentiment analysis score base on the words used in messages. Sentiment Score above 0.4 to 1 is consider as Positive. Pure English words and Phrases is ideal for calculation

Figure 5.12: Positive sentiment on the basis of polarity score

Figure 5.12 describes the polarity score using builtin library sentiment analyser. Since builtin analyser work mostly on English words but in chats most bilingual words has been used. That's why most message may have polarity as 0 because of neutral and hence polarity score is low here.

This section deals with implementation of Exploratory Data Analysis on chat using python libraries. Matplotlib has been used to display bar-chart, scatter-plot, pie-chart and line chart etc. while streamlit components has been used to build the web interface. In next section, Emotion detection model implementation has been presented.

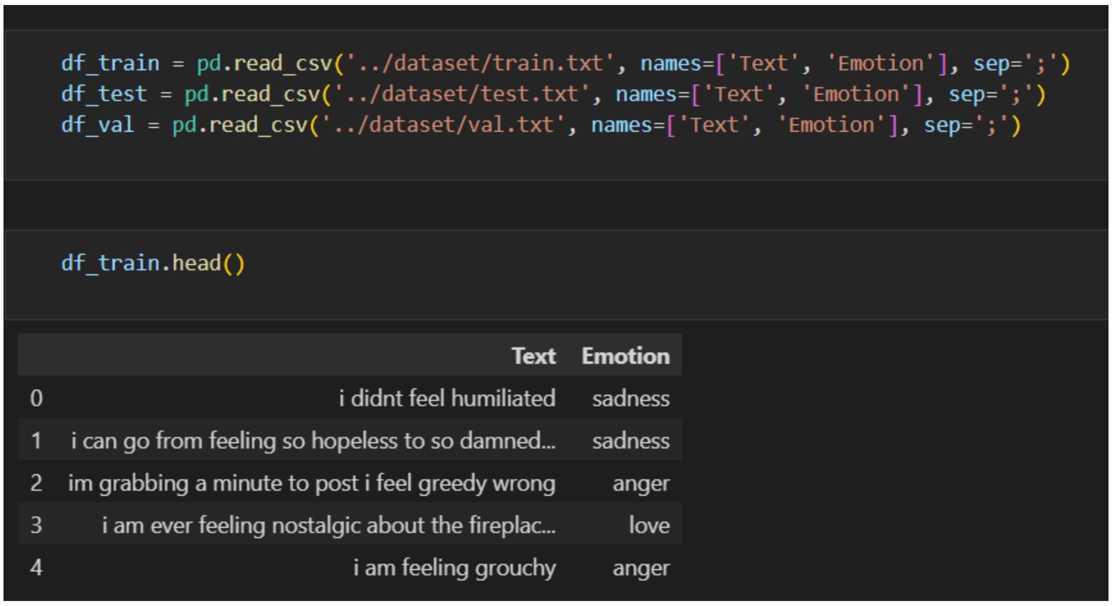
5.2 EMOTION DETECTION

This section deals with implementation of emotion detection model based on deep learning techniques by using python language and packages.

5.2.1 PRE-PROCESSING

Emotion dataset is loaded and punctuation are removed. At last cleaning is performed. Here three text files has been taken.

1. **Train.txt :** Model will be trained with this dataset which contains 16000 records
2. **Val.txt :** provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters. It contains 2000 records.
3. **Test.txt :** provide an unbiased evaluation of a final model fit on the training dataset.

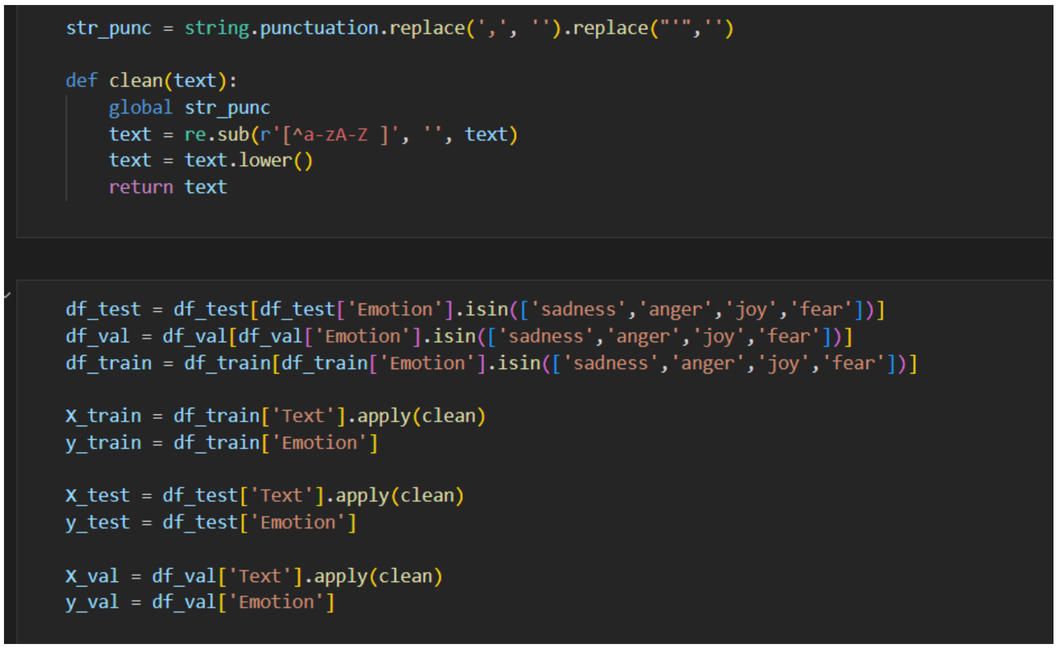


```
df_train = pd.read_csv('../dataset/train.txt', names=['Text', 'Emotion'], sep=';')
df_test = pd.read_csv('../dataset/test.txt', names=['Text', 'Emotion'], sep=';')
df_val = pd.read_csv('../dataset/val.txt', names=['Text', 'Emotion'], sep=';')
```

```
df_train.head()
```

```
                                               Text  Emotion
0                             i didnt feel humiliated  sadness
1   i can go from feeling so hopeless to so damned...  sadness
2    im grabbing a minute to post i feel greedy wrong    anger
3        i am ever feeling nostalgic about the fireplac...     love
4                                i am feeling grouchy    anger
```

Figure 5.13: Loading dataset



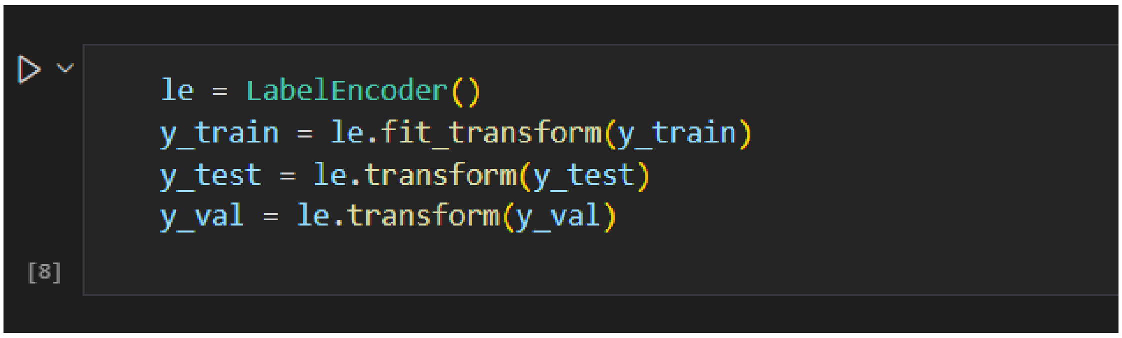
```
str_punc = string.punctuation.replace(',', '').replace("'",'')

def clean(text):
    global str_punc
    text = re.sub(r'[^a-zA-Z ]', '', text)
    text = text.lower()
    return text
```

```
df_test = df_test[df_test['Emotion'].isin(['sadness','anger','joy','fear'])]
df_val = df_val[df_val['Emotion'].isin(['sadness','anger','joy','fear'])]
df_train = df_train[df_train['Emotion'].isin(['sadness','anger','joy','fear'])]

X_train = df_train['Text'].apply(clean)
y_train = df_train['Emotion']

X_test = df_test['Text'].apply(clean)
y_test = df_test['Emotion']

X_val = df_val['Text'].apply(clean)
y_val = df_val['Emotion']
```

Figure 5.14: Pre Processing and Cleaning on Dataset

Figure 5.14 describes the punctuation removal and text clean operation. Only 4 emotions based records are kept to maintain balance among the dataset.

Label Encoding is a popular encoding technique for handling the categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.



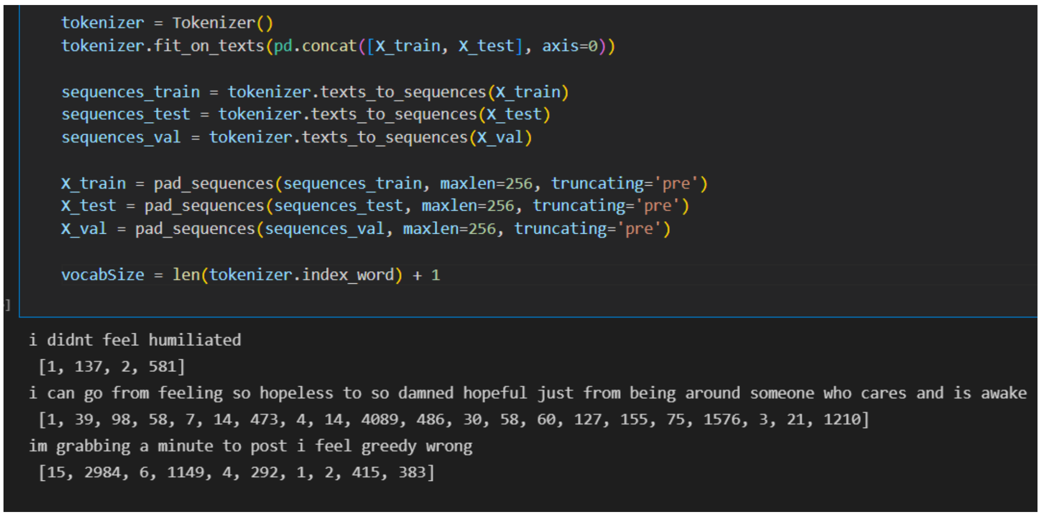
```
le = LabelEncoder()
y_train = le.fit_transform(y_train)
y_test = le.transform(y_test)
y_val = le.transform(y_val)
```

Figure 5.15: Label Encoding

Since training dataset is converted to numeric form, but to establish the relationship between labels so that it could be understood by the algorithm, it need to be converted it to 2D Matrix form (usually Binary Matrix). We use to_categorical() to transform our training data before we pass it to model. If training data uses classes as numbers, to_categorical() will transform those numbers in proper vectors for using with model

Tokenization is the task of splitting a sequence of text into units with semantic meaning. These units are called tokens. The difficulty in tokenization lies on how to get the ideal split so that all the tokens in the text have the correct meaning, and there are no left out tokens. Following steps has been carried out for the process of tokenization:

First **vocabulary** has been created to integer mapping dictionary in such a way that frequently occurring words are assigned lower indexes. then words has been encoded in numbers and they are labelled. Then sentence length has been analyzed for getting rid of extremely long and short sentences. At last padding and truncating of remaining sentences has been done to deal with short and long sentence of specific length.



```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(pd.concat([X_train, X_test], axis=0))

sequences_train = tokenizer.texts_to_sequences(X_train)
sequences_test = tokenizer.texts_to_sequences(X_test)
sequences_val = tokenizer.texts_to_sequences(X_val)

X_train = pad_sequences(sequences_train, maxlen=256, truncating='pre')
X_test = pad_sequences(sequences_test, maxlen=256, truncating='pre')
X_val = pad_sequences(sequences_val, maxlen=256, truncating='pre')

vocabSize = len(tokenizer.index_word) + 1
```

```
i didnt feel humiliated
 [1, 137, 2, 581]
i can go from feeling so hopeless to so damned hopeful just from being around someone who cares and is awake
 [1, 39, 98, 58, 7, 14, 473, 4, 14, 4089, 486, 30, 58, 60, 127, 155, 75, 1576, 3, 21, 1210]
im grabbing a minute to post i feel greedy wrong
 [15, 2984, 6, 1149, 4, 292, 1, 2, 415, 383]
```

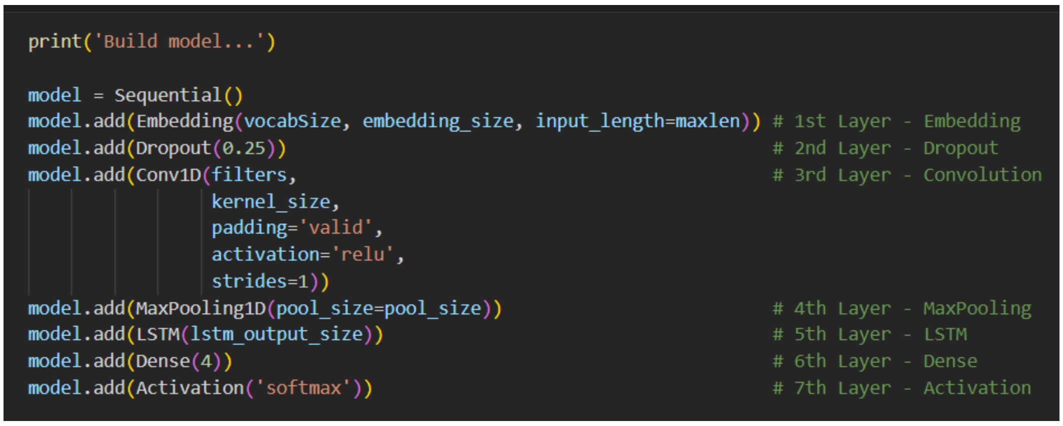
Figure 5.16: Tokenization and intermediate steps

5.2.2 Model Building of LSTM and Bi-LSTM

Since input is in form of text sequences that's why Sequential model is selected. This model contains 7 intermediate layers which are following :

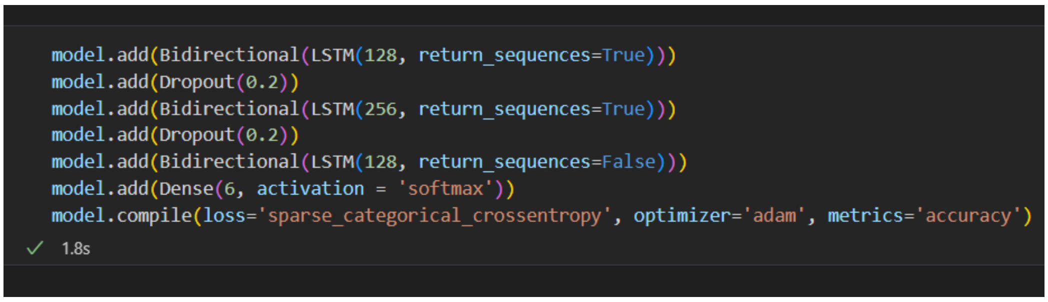
The first layer is the **Embedded layer** that uses 200 length vectors to represent each word. This layer maps word indices to vectors. The second layer is **Dropout layer**, It has been used because with the greater number of parameters there may be chance of causing over-fitting. It prevents all neurons in a layer from synchronously optimizing their weights. It helps in increasing accuracy score. Here it can max dropout 25% bits. The third layer is **Convolution layer** which is mainly used for feature extraction with respect to filters. If Case Kernel Size = 3 then it will extract size of 1x3 in case of 1D and 3x3 in case of 2D. A kernel is a small matrix, with its height and width smaller than the training dataset to be convoluted. It is also known as a convolution matrix or convolution mask.

An **activation function** is the last component of the convolutional layer to increase the non-linearity in the output. Generally, **ReLu** function or **Tanh** functions are two most used activation function in a convolution layer. The next layer is **Max-pooling layer**. Pooling layer is used to reduce the size of the input image and speed up computation. The next layer is the **LSTM layer** with 128 memory units (smart neurons) that represent the dimensionality of outer space. It mainly used to classifying, processing and making predictions based on time series data. The next layer is **Dense layer**, Since this is a classification problem we use a Dense output layer with a single neuron and **sigmoid** activation function to make 0 or 1 predictions for the each emotions. **Softmax** converts a vector of values to a probability distribution. The elements of the output vector are in range (0, 1) and sum to 1.



```
print('Build model...')

model = Sequential()
model.add(Embedding(vocabSize, embedding_size, input_length=maxlen)) # 1st Layer - Embedding
model.add(Dropout(0.25))                                             # 2nd Layer - Dropout
model.add(Conv1D(filters,                                            # 3rd Layer - Convolution
                 kernel_size,
                 padding='valid',
                 activation='relu',
                 strides=1))
model.add(MaxPooling1D(pool_size=pool_size))                         # 4th Layer - MaxPooling
model.add(LSTM(lstm_output_size))                                    # 5th Layer - LSTM
model.add(Dense(4))                                                  # 6th Layer - Dense
model.add(Activation('softmax'))                                     # 7th Layer - Activation
```

Figure 5.17: Building Emotion detection model using LSTM



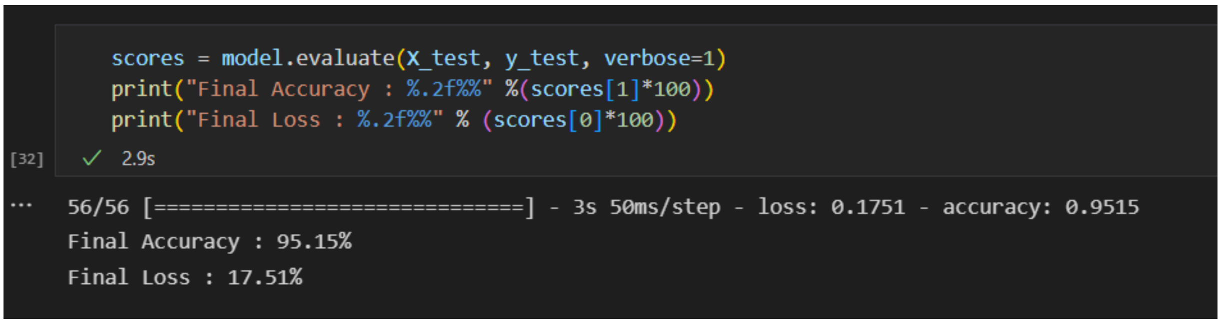
```
model.add(Bidirectional(LSTM(128, return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(256, return_sequences=True)))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(128, return_sequences=False)))
model.add(Dense(6, activation = 'softmax'))
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics='accuracy')
```

Figure 5.18: Building Emotion detection model using Bi-LSTM

In Figure 5.18, BI-Directional LSTM model has been displayed. To make an LSTM layer as Bidirectional, an extra layer as Bi-directional is added to LSTM. In this BI-LSTM model 4 intermediate layers have been used which are **LSTM Layer**, **Bidirectional** Layer, **Dropout** Layer and **Dense** Layer. At last model has been compiled with hyper tune and model is built.

5.3 Evaluation Results

The overall accuracy of model is found to be 95.15 % after iteration of 30 with a loss of 17.51 %

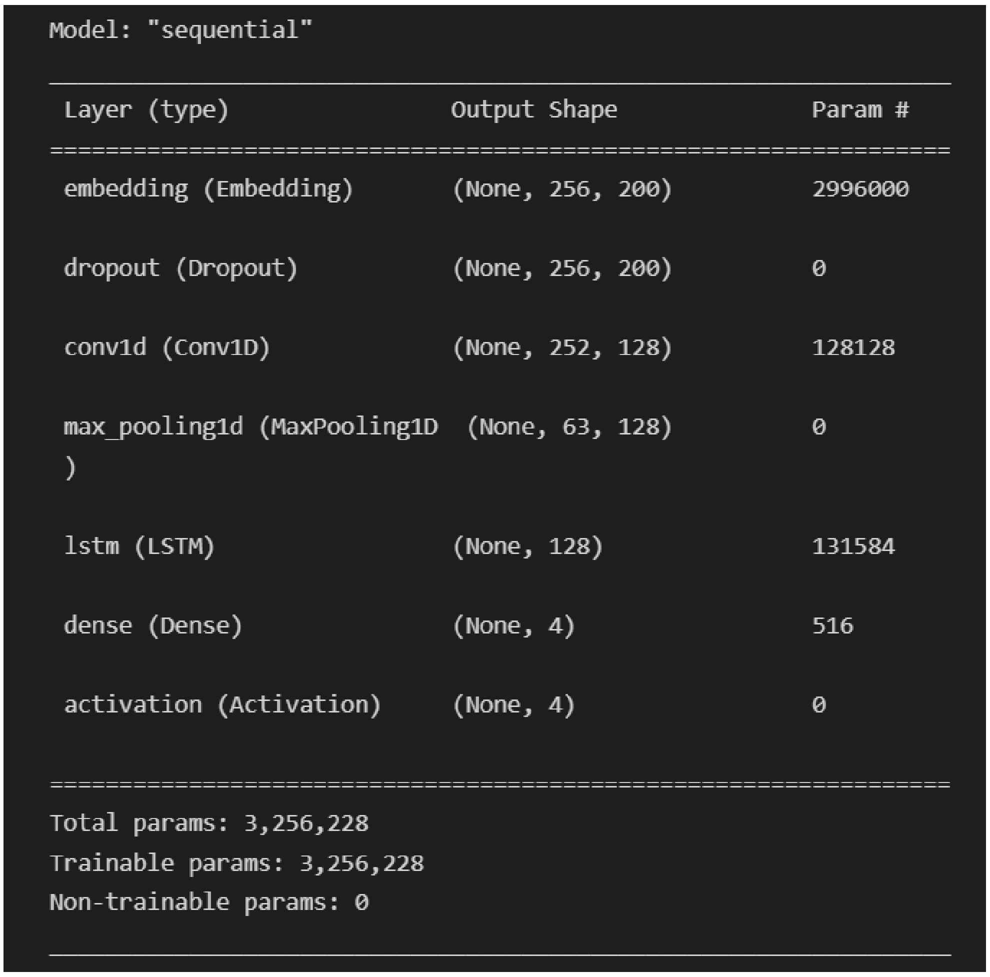


```
scores = model.evaluate(X_test, y_test, verbose=1)
print("Final Accuracy : %.2f%%" %(scores[1]*100))
print("Final Loss : %.2f%%" % (scores[0]*100))
```

```
[32]  ✓ 2.9s
```

```
56/56 [==============================] - 3s 50ms/step - loss: 0.1751 - accuracy: 0.9515
Final Accuracy : 95.15%
Final Loss : 17.51%
```

Figure 5.19: Accuracy Graph of emotion detection model



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 256, 200)          2996000

 dropout (Dropout)           (None, 256, 200)          0

 conv1d (Conv1D)             (None, 252, 128)          128128

 max_pooling1d (MaxPooling1D  (None, 63, 128)          0
 )

 lstm (LSTM)                 (None, 128)               131584

 dense (Dense)               (None, 4)                 516

 activation (Activation)     (None, 4)                 0

=================================================================
Total params: 3,256,228
Trainable params: 3,256,228
Non-trainable params: 0
_________________________________________________________________
```

Figure 5.20: Model Summary

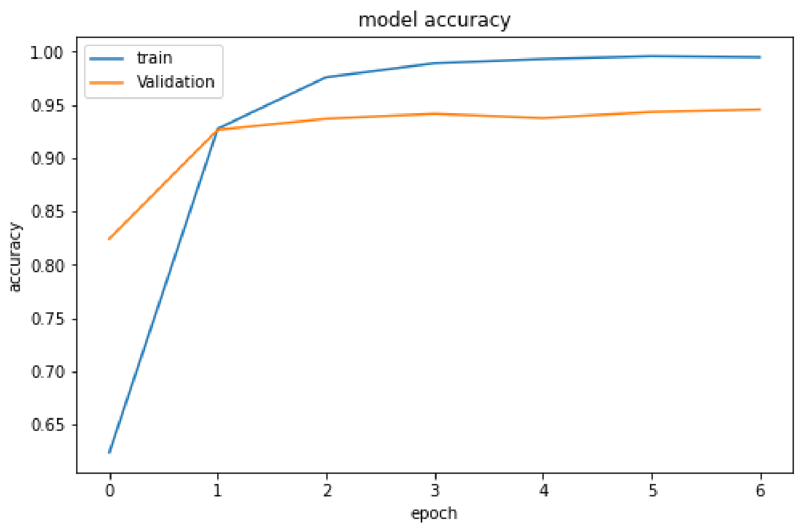


Figure 5.21: Model Accuracy in each Epoch

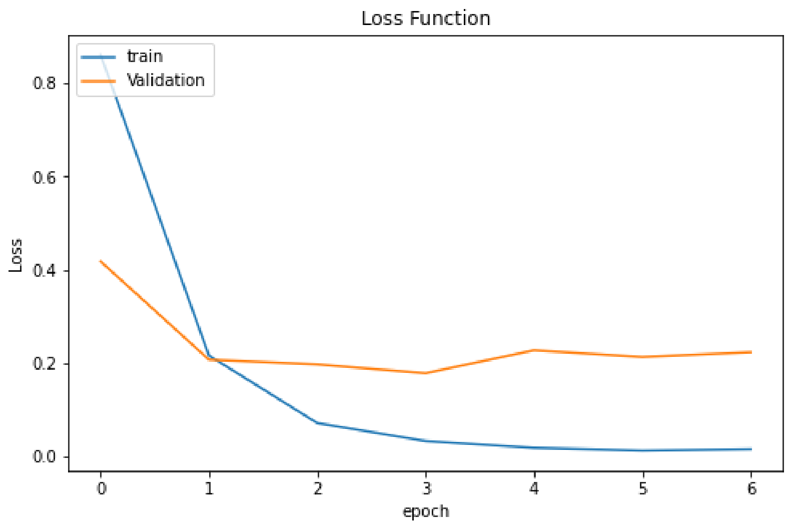


Figure 5.22: Model Loss in each Epoch

Figure 5.21 shows the relation between number of iteration and model accuracy during training, Whereas Figure 5.22 shows the relation between the iteration and model loss. The model is trained up to 16000 iterations with average 95.15 and loss of 17.51 %. in Figure 5.21 it can be seen that towards the end training accuracy is slightly higher than validation accuracy and in Figure 5.22 training loss is slightly lower than validation loss. This hints at overfitting and more epochs are required to be trained so the gap should widen.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

In conclusion, this report states that exploratory data analysis or EDA is a gateway to the unexplored field of WhatsApp Chats. It allows to analyze bilingual user. It was able to use the Python programming language and its libraries, to create an analysis of a WhatAapp group chat and visually represent the top 10 and top 20 users in the chat group, analyze the emojis and most active day/month/year etc. At the end of the work the expected results were obtained and the analysis was able to show the level of participation of the all individuals on the given WhatsApp group.

Emotion Detection using LSTM model needed several intermediate layers for passing memory to get the desired output. In this project seven intermediate layers are used to make the sequential model and to predict the emotion in 4 different categories **['sadness','anger','joy','fear']**. Dataset of 16000 records has been used but most of records have either joy or sadness.

In this sequential model, word embedding layer need to be employed to understand how the model works and responds to the reviews or textual data. The unsupervised word embedding learning algorithm is used.

6.2 FUTURE WORK

In Exploratory data analysis, although we have seen lot of insights like day wise messages, most active time in group, most used words during conversation etc. A rise in hate speech, cyber-bulling, heckling and increased impudence on social media has been observed. This Chat Analyzer can be used as a medium to give a user an insight on their online behavior as they communicate with their peers. It allows a user to keep a check on their emotions by analyzing them. Still we can add feature like **time wise topic modeling** (which will state about discussed topics between two time), **code recognition** (in case programming language code is shared in chat), **feature extraction for products** (to get details if user discuss about purchasing any product), **Malicious activity** (bullying, abusing or narcotics etc discussion).

In Emotion detection, majority of records belong to either joy and sadness, therefore more no of records should be added for more accurate and diverse output. Also till now it is predicting the emotions in four different categories, it must be improved to categorise in at least 15 emotions. Also use of some google API for detecting the emotion in any language can also be implemented. Performance can be improved with deep learning model via applying different word embedding techniques and various hyper parameter. By using pre-trained NLP named Bi-directional Encoder Representation from Transformer (BERT) with training on English Wikipedia (2500 millions words) and Book Corpus (800 million words), it can show most accurate results.

REFERENCES

[1] IS Srajan, K Ravishankara, and Vaisakh Dhanush. Whatsapp chat analyzer. *International Journal of Engineering Research & Technology*, 9(5):897–900, 2020.

[2] Sunil Joshi. Sentiment analysis on whatsapp group chat using r. *Data, Engineering and Applications*, pages 47–55, 2019.

[3] Subarno Pal, Soumadip Ghosh, and Amitava Nag. Sentiment analysis in the light of lstm recurrent neural networks. *International Journal of Synthetic Emotions (IJSE)*, 9(1):33–39, 2018.

[4] Huang Zou, Xinhua Tang, Bin Xie, and Bing Liu. Sentiment classification using machine learning techniques with syntax features. pages 175–179, 2015.

[5] Alec Yenter and Abhishek Verma. Deep cnn-lstm with combined kernels from multiple branches for imdb review sentiment analysis. pages 540–546, 2017.

[6] Chinthapanti Bharath Sai Reddy, G Gopichand, Kumar MV Rakesh, O Nikhil Kumar, and S Kowshik. Analysing and predicting the emotion of whatsapp chats using sentiment analysis. *TEST Engineering & Management*, 83:15454–15461, 2020.

[7] G RajKumar, P Brundha, and V Selina Annie Retna. People's behaviour analysis in chat message using natural language processing. pages 1128–1133, 2021.

[8] Astha Mohta, Atishay Jain, and Sonika Dahiya. Text classification based behavioural analysis of whatsapp chats. pages 717–724, 2020.

[9] Blessing Nwamaka Iduh. Whatsapp network group chat analysis using python programming. *International Journal of Education*, pages 124–126, 2018.

[10] J Andjelic. Whatsapp statistics: Revenue, usage, and history. *Fortunly.*, pages 101–103, 2019.

[11] George Dominic Ewur and Johnson Yeboah. The impact of whatsapp messenger usage on students performance in tertiary institutions in ghana. *Journal of Education and practice*, 5(6):157–164, 2014.