DESIGN AND IMPLEMENTATION OF WHATS APP CHAT ANALYSER USING PYTHON

This project report is submitted to
Rashtrasant Tukadoji Maharaj Nagpur University
in the partial fulfilment of the requirement for the award of the degree
of

Bachelor of Engineering in Computer Technology

by

Ms. Ritika Rai (CT18113)
Mr. Tanmay Dhawale (CT18105)
Ms. Pranali Shyamkuwar (CT18001)
Mr. Vinay Vardhan Reddy (CT18111)
Ms. Himani Rokade (CT17049)

Under the guidance of

Ms. Vaishali Malekar

Assistant Professor

2021-2022

DEPARTMENT OF COMPUTER TECHNOLOGY
KAVIKULGURU INSTITUTE OF TECHNOLOGY AND SCIENCE
RAMTEK – 441 106

DEPARTMENT OF COMPUTER TECHNOLOGY

KAVIKULGURU INSTITUTE OF TECHNOLOGY AND SCIENCE

RAMTEK – 441 106

CERTIFICATE

This is to certify that the project report entitled **'Design and Implementation of WhatsApp Chat Analyser by using Python'** carried out by Ms. Ritika Rai (CT18113), Mr. Tanmay Dhawale (CT18105), Ms. Pranali Shyamkuwar (CT18001), Mr. Vinay Vardhan Reddy (CT18111) and Ms. Himani Rokade (CT17049) of the B.E. Fourth year of Computer Technology, during the academic year 2021-2022, in the partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering (Computer Technology)** offered by the **Rashtrasant Tukadoji Maharaj Nagpur University,** Nagpur.

Ms. Vaishali Malekar

Guide

Dr. Vilas P. Mahatme

Head of the Department

Dr. Avinash N. Shrikhande

Principal

Date:

Place: Ramtek

DECLARATION

We declare that

1. The work contained in this project report has been done by us under the supervision of our guide.
2. The work has not been submitted to any other Institute for any degree or diploma.
3. We have followed the guidelines provided by the Institute in preparing the project report.
4. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
5. Whenever we have used material (data, theoretical analysis, figures and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references. Further, we have taken permission from the copyright owners of the sources, whenever necessary.

Project-mates

ACKNOWLEDGEMENT

We are grateful to our respected guide **Ms. Vaishali Malekar** for her kind, disciplined and invaluable guidance which inspired us to solve all the difficulties that came across during completion of project.

We express our special thanks to **Dr. Vilas P. Mahatme,** Head of the Department, for his kind support, valuable suggestion and allowing us to use all facilities that are available in the Department during this project.

Our sincere thanks are due to **Dr. Avinash N. Shrikhande,** Principal, for extending all the possible help and allowing us to use all resources that are available in the Institute.

We are also thankful to our **Parents** and **Friends** for their valuable cooperation and standing with us in all difficult conditions.

Project-mates

ABSTRACT

The most used and efficient method of communication in recent times is an application called WhatsApp. WhatsApp chats consists of various kinds of conversations held among group of people. This chat consists of various topics. This information can provide lots of data for latest technologies such as machine learning. The most important thing for a machine learning model is to provide the right learning experience which is indirectly affected by the data that we provide to the model. This tool aims to provide in depth analysis of this data which is provided by WhatsApp. Irrespective of whichever topic the conversation is based our developed code can be applied to obtain a better understanding of the data. The advantage of this tool is that is implemented using simple python modules such as pandas, matplotlib, seaborn and sentiment analysis which are used to create data frames and plot different graphs, where then it is displayed in the flutter application which is efficient and less resources consuming algorithm, therefor it can be easily applied to largest dataset.

Keywords: WhatsApp chat data, Pandas, matplotlib, Word Cloud, sentiment analyser, Stream lit application etc.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This tool is based on data analysis and processing. The first step in implementing a machine learning algorithm is to understand the right learning experience from which the model starts improving on. Data pre-processing plays a major role when it comes to machine learning. In order to make the model more efficient it needs lots of data, so this turned focus primarily on one of the largescale data producers owned by Facebook which is nothing but WhatsApp. WhatsApp claims that nearly 55 billion messages are sent each day. The average user spends 195 minutes per week on WhatsApp, and is a member of plenty of groups. With this treasure house of data right under ones very noses, it is but imperative that this embark on a mission to gain insights on the messages in phones are forced to bear witness to.

WhatsApp Group Chat Analysis using Python

Today one of the trendy social media platforms is one and only WhatsApp. It is one of the favourite social media platforms among all because of its attractive features. It has more than 2B users worldwide and according to one survey an average user spends more than 195 minutes per week on WhatsApp. How terrible the above statement. Leave all these things and understand what actually the meaning of WhatsApp analyser.

WhatsApp Analyzer means analysing WhatsApp group activities. It tracks the conversation and analyses how much time people are spending or saying it as wasting on WhatsApp. The aim of this article is to provide step by step guide to build a WhatsApp analyser using python. Here different python libraries are used which helps to extract useful information from raw data.

WhatsApp provides the feature of exporting chats, so first export the chat and save the file. The python program is created that will extract the Date, Username of Author, Time, Messages from exported chat file and creating a data frame, and storing all data in it step in implementing a machine learning algorithm is to understand the right learning experience from which the model starts improving on. Data pre-processing plays a major role when it comes to machine learning. In order to make the model more efficient there is a need of lots of data, so focus is turned primarily on one

of the largescale data producers owned by Facebook which is nothing but WhatsApp. WhatsApp claims that nearly 55 billion messages are sent each day. The average user spends 195 minutes per week on WhatsApp, and is a member of plenty of groups. With this treasure house of data right under our very noses, it is but imperative that this embark on a mission to gain insights on the messages which our phones are forced to bear witness to.

1.1 Machine Learning

Machine learning is closely related to the fields of statistics and data science, which provide a range of tools and methods for data analysis and inference from data. It is also related to robotics and intelligent automation. These fields help shape the context in which people relate to many machine learning applications, and inform the opportunities and challenges associated with it. Machine learning also supports progress in these fields, as an underlying technology for both Artificial Intelligence and data science. At its most basic level, machine learning involves computers processing a large amount of data to predict outcomes. This process of data handling and prediction has strong links to the overlapping fields of data science and statistics, which seek to extract insights from data. Statistical approaches can inform how machine learning systems deal with the probabilities or uncertainty in decision-making, while processing and analysis techniques from data science feed into machine learning. Despite the recent attention given to, and hype surrounding, machine learning, fundamental ideas in the field are not so new, with early papers being published over sixty years ago. Within the last decade, even the past five years, the field of machine learning has made revolutionary advances. These advances have been driven in part by the availability of large amounts of data and the accessibility of computing power, but also underpinned by algorithmic advance. computers processing a large amount of data to predict outcomes. This process of data handling and prediction has strong links to the overlapping fields of data science and statistics, which seek to extract insights from data. Statistical approaches can inform how machine learning systems deal with the probabilities or uncertainty in decision-making, while processing and analysis techniques from data science feed into machine learning. Despite the recent attention given to, and hype surrounding, machine learning, fundamental ideas in the field are not so new, with early papers being published over sixty years ago. Within the last

decade, even the past five years, the field of machine learning has made revolutionary advances. These advances have been driven in part by the availability of large amounts of data and the accessibility of computing power, but also underpinned by algorithmic advances achieved by revisiting and re-envisioning the simple neural networks put forward in the 1940s and 1950s.

Drawing further insights from physiology and neuroscience, artificial neural networks have been created in which hundreds of layers of processing allow systems to perform more complicated tasks. These so-called deep learning techniques have been responsible for some of the more high-profile recent advances in Artificial Intelligence research, such as the Alpha Go systems victory over Lee Sedol, acknowledged as the strongest human player at the game of Go, in March 2016. These fields help shape the context in which people relate to many machine learning applications, and inform the opportunities and challenges associated with it. Machine learning also supports progress in these fields, as an underlying technology for both Artificial Intelligence and data science. At its most basic level, machine learning involves computers processing a large amount of data to predict outcomes. This process of data handling and prediction has strong links to the overlapping fields of data science and statistics, which seek to extract insights from data. Statistical approaches can inform how machine learning systems deal with the probabilities or uncertainty in decision-making, while processing and analysis techniques from data science feed into machine learning. Despite the recent attention given to, and hype surrounding, machine learning, fundamental ideas in the field are not so new, with early papers being published over sixty years ago. Within the last decade, even the past five years, the field of machine learning has made revolutionary advances. These advances have been driven in part by the availability of large amounts of data and the accessibility of computing power, Classical machine learning is often categorized by how an algorithm learns to become more accurate in its predictions. There are four basic approaches: supervised learning, unsupervised learning, semi supervised learning and reinforcement learning. The type of algorithm data scientists chooses to use depends on what type of data want to predict

1.1.1 Supervised learning

In this type of machine learning, data scientists supply algorithms with labelled training data and define the variables user want the algorithm to assess for correlations. Both the input and the output of the algorithm is specified. Supervised learning is an approach to creating Artificial Intelligence, where a computer algorithm is trained on input data that has been labelled for a particular output. The model is trained until it can detect the underlying patterns and relationships between the input data and the output labels, enabling it to yield accurate labelling results when presented with never-before-seen data.

Supervised learning is good at classification and regression problems, such as determining what category a news article belongs to or predicting the volume of sales for a given future date. In supervised learning, the aim is to make sense of data within the context of a specific question.

How does supervised learning work

Like all machine learning algorithms, supervised learning is based on training. During its training phase, the system is fed with labelled data sets, which instruct the system what output is related to each specific input value. The trained model is then presented with test data: This is data that has been labelled, but the labels have not been revealed to the algorithm. The aim of the testing data is to measure how accurately the algorithm will perform on unlabelled data.

Supervised machine learning requires the data scientist to train the algorithm with both labelled inputs and desired outputs. Supervised learning algorithms are good for the following tasks:

- **Binary classification:** Dividing data into two categories.
- **Multi-class classification:** Choosing between more than two types of answers.
- **Regression modelling:** Predicting continuous values.
- **Assembling:** Combining the predictions of multiple machine learning models to produce an accurate prediction.

1.1.2 Unsupervised learning

This type of machine learning involves algorithms that train on unlabelled data. The algorithm scans through data sets looking for any meaningful connection. The data that algorithms train on as well as the predictions or recommendations user output are predetermined.

Unsupervised learning refers to the use of Artificial Intelligence algorithms to identify patterns in data sets containing data points that are neither classified nor labelled. The algorithms are thus allowed to classify, label and/or group the data points contained within the data sets without having any external guidance in performing that task. In other words, unsupervised learning allows the system to identify patterns within data sets on its own. In unsupervised learning, an Artificial Intelligence system will group unsorted information according to similarities and differences even though there are no categories provided. Unsupervised learning algorithms can perform more complex processing tasks than supervised learning systems. Additionally, subjecting a system to unsupervised learning is one way of testing Artificial Intelligence.

However, unsupervised learning can be more unpredictable than a supervised learning model. While an unsupervised learning Artificial Intelligence system might, for example, figure out on its own how to sort cats from dogs, it might also add unforeseen and undesired categories to deal with unusual breeds, creating clutter instead of order.

Artificial Intelligence systems capable of unsupervised learning are often associated with generative learning models, although user may also use a retrieval-based approach. Chatbots, self-driving cars, facial recognition programs. Expert systems and robots are among the systems that may use either supervised or unsupervised learning approaches, or both. Unsupervised machine learning algorithms do not require data to be labelled. User sift through unlabelled data to look for patterns that can be used to group data points into subsets. Most types of deep learning, including neural networks, are unsupervised algorithms. Unsupervised learning algorithms are good for the following tasks:

1. **Clustering:** Splitting the dataset into groups based on similarity.
2. **Anomaly detection:** Identifying unusual data points in a data set.

3. **Association mining:** Identifying sets of items in a data set that frequently occur together.
4. **Dimensionality reduction:** Reducing the number of variables in a data set.

Unsupervised vs. supervised learning

Comparing supervised versus unsupervised learning, supervised learning uses labelled data sets to train algorithms to identify and sort based on provided labels. The input object, or sample, has a corresponding label so that the algorithms learn to identify and classify those input objects which match with the same label.

In other words, the algorithms create maps from given inputs to specific outcomes based on what user learn from training data that has been labelled by machine learning engineers or data scientists.

Moreover, supervised learning uses both labelled training data and labelled validation data. This allows the accuracy of supervised learning outputs to be checked for accuracy in a way that unsupervised learning cannot be measured. Machine learning engineers or data scientists may opt to use a combination of labelled and unlabelled data to train their algorithms. This in-between option is appropriately called semi-supervised learning.

1.1.3 Semi-supervised learning

This approach to machine learning involves a mix of the two preceding types. Data scientists may feed an algorithm mostly labelled training data, but the model is free to explore the data on its own and develop its own understanding of the data set. Today Machine Learning algorithms can be broadly classified into three categories, Supervised Learning, Unsupervised Learning and Reinforcement Learning. Casting Reinforced Learning aside, the primary two categories of Machine Learning problems are Supervised and Unsupervised Learning. The basic difference between the two is that Supervised Learning datasets have an output label associated with each tuple while Unsupervised Learning datasets do not. The most basic disadvantage of any Supervised Learning algorithm is that the dataset has to be hand-labelled either by a Machine Learning Engineer or a Data Scientist. This is a

very costly process, especially when dealing with large volumes of data. The most basic disadvantage of any Unsupervised Learning is that it is application spectrum is limited**.**

To counter these disadvantages, the concept of Semi-Supervised Learning was introduced. In this type of learning, the algorithm is trained upon a combination of labelled and unlabelled data. Typically, this combination will contain a very small amount of labelled data and a very large amount of unlabelled data. The basic procedure involved is that first, the programmer will cluster similar data using an unsupervised learning algorithm and then use the existing labelled data to label the rest of the unlabelled data. The typical use cases of such type of algorithm have a common property. The acquisition of unlabelled data is relatively cheap while labelling the said data is very expensive.

Intuitively, one may imagine the three types of learning algorithms as Supervised learning where a student is under the supervision of a teacher at both home and school, Unsupervised learning where a student has to figure out a concept himself and Semi-Supervised learning where a teacher teaches a few concepts in class and gives questions as homework which are based on similar concepts.

A Semi-Supervised algorithm assumes the following about the data

- **Continuity Assumption:** The algorithm assumes that the points which are closer to each other are more likely to have the same output label.
- **Cluster Assumption:** The data can be divided into discrete clusters and points in the same cluster are more likely to share an output label.
- **Manifold Assumption:** The data lie approximately on a manifold of much lower dimension than the input space. This assumption allows the use of distances and densities which are defined on a manifold.

Practical applications of Semi-Supervised Learning

- **Speech Analysis:** Since labelling of audio files is a very intensive task, Semi-Supervised learning is a very natural approach to solve this problem.
- **Internet Content Classification:** Labelling each webpage is an impractical and unfeasible process and thus uses Semi-Supervised learning algorithms. Even the Google search algorithm uses a variant of Semi-Supervised learning to rank the relevance of a webpage for a given query.

- **Protein Sequence Classification:** Since DNA strands are typically very large in size, the rise of Semi-Supervised learning has been imminent in this field.

Semi-supervised learning works by data scientists feeding a small amount of labelled training data to an algorithm. From this, the algorithm learns the dimensions of the data set, which it can then apply to new, unlabelled data. The performance of algorithms typically improves when user train on labelled data sets. But labelling data can be time consuming and expensive. Semi-supervised learning strikes a middle ground between the performance of supervised learning and the efficiency of unsupervised learning. Some areas where semi-supervised learning is used include:

- **Machine translation:** Teaching algorithms to translate language based on less than a full dictionary of words.
- **Fraud detection:** Identifying cases of fraud when you only have a few positive examples.
- **Labelling data:** Algorithms trained on small data sets can learn to apply data labels to larger sets automatically.

1.1.4 Reinforcement learning

Data scientists typically use reinforcement learning to teach a machine to complete a multi-step process for which there are clearly defined rules. Data scientists program an algorithm to complete a task and give it positive or negative cues as it works out how to complete a task. But for the most part, the algorithm decides on its own what steps to take along the way. Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what

to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

Main points in Reinforcement learning

- Input: The input should be an initial state from which the model will start
- Output: There are many possible outputs as there are a variety of solutions to a particular problem
- Training: The training is based upon the input, The model will return a state and the user will decide to reward or punish the model based on its output.
- The model keeps continues to learn.
- The best solution is decided based on the maximum reward.

Types of Reinforcement: There are two types of Reinforcement

- **Positive** –
  Positive Reinforcement is defined as when an event, occurs due to a particular behaviour, increases the strength and the frequency of the behaviour. In other words, it has a positive effect on behaviour.
  0. Advantages of reinforcement learning are:
  1. Maximizes Performance
  2. Sustain Change for a long period of time
  3. Too much Reinforcement can lead to an overload of states which can diminish the results
- **Negative** –
  Negative Reinforcement is defined as strengthening of behaviour because a negative condition is stopped or avoided.

Advantages of reinforcement learning

- Increases Behaviour
- Provide defiance to a minimum standard of performance
- It Only provides enough to meet up the minimum behaviour

Various Practical applications of Reinforcement Learning

- Reinforcement Learning can be used in robotics for industrial automation.

- Reinforcement Learning can be used in machine learning and data processing
- Reinforcement Learning can be used to create training systems that provide custom instruction and materials according to the requirement of students.

Reinforcement Learning can be used in large environments in the following situations

- A model of the environment is known, but an analytic solution is not available;
- Only a simulation model of the environment is given the subject of simulation-based optimization.
- The only way to collect information about the environment is to interact with it.

Reinforcement learning works by programming an algorithm with a distinct goal and a prescribed set of rules for accomplishing that goal. Data scientists also program the algorithm to seek positive rewards -- which it receives when it performs an action that is beneficial toward the ultimate goal -- and avoid punishments -- which it receives when it performs an action that gets it farther away from its ultimate goal. Reinforcement learning is often used in areas such as:

- **Robotics:** Robots can learn to perform tasks the physical world using this technique.
- **Video gameplay:** Reinforcement learning has been used to teach bots to play a number of video games.
- **Resource management:** Given finite resources and a defined goal, reinforcement learning can help enterprises plan out how to allocate resources.

1.2 Applications of machine learning

Today, machine learning is used in a wide range of applications. Perhaps one of the most well-known examples of machine learning in action is the recommendation engine that powers Facebook news feed.

Facebook uses machine learning to personalize how each members feed is delivered. If a member frequently stops to read a particular group posts, the recommendation engine will start to show more of that group activity earlier in the feed.

Behind the scenes, the engine is attempting to reinforce known patterns in the members online behaviour. Should the member change patterns and fail to read posts from that group in the coming weeks, the news feed will adjust accordingly. Facebook uses machine learning to personalize how each members feed is delivered. If a member frequently stops to read a particular groups post, the recommendation engine will start to show more of that group activity earlier in the feed.

Behind the scenes, the engine is attempting to reinforce known patterns in the members online behaviour. Should the member change patterns and fail to read posts from that group in the coming weeks, the news feed will adjust accordingly.

In addition to recommendation engines, other uses for machine learning include the following:

- **Customer relationship management**. Software can use machine learning models to analyse email and prompt sales team members to respond to the most important messages first. More advanced systems can even recommend potentially effective responses**.**
- **Business intelligence.** Analytics vendors use machine learning in their software to identify potentially important data points, patterns of data points and anomalies.
- **Human resource information systems.** Systems can use machine learning models to filter through applications and identify the best candidates for an open position.
- **Self-driving cars.** Machine learning algorithms can even make it possible for a semi-autonomous car to recognize a partially visible object and alert the driver.
- **Virtual assistants.** Smart assistants typically combine supervised and unsupervised machine learning models to interpret natural speech and supply context.

1.3 Advantages and Disadvantages of machine learning

- Machine learning has seen use cases ranging from predicting customer behaviour to forming the operating system for self-driving cars.
- When it comes to advantages, machine learning can help enterprises understand their customers at a deeper level. By collecting customer data and correlating it

with behaviours over time, machine learning algorithms can learn associations and help teams tailor product development and marketing initiatives to customer demand.

- Some companies use machine learning as a primary driver in their business models. Uber, for example, uses algorithms to match drivers with riders. Google uses machine learning to surface the ride advertisements in searches.
- When it comes to advantages, machine learning can help enterprises understand their customers at a deeper level. By collecting customer data and correlating it with behaviours over time, machine learning algorithms can learn associations and help teams tailor product development and marketing initiatives to customer demand.
- But machine learning comes with disadvantages. First and foremost, it can be expensive. Machine learning projects are typically driven by data scientists, who command high salaries. These projects also require software infrastructure that can be expensive.
- Some companies use machine learning as a primary driver in their business models. Uber, for example, uses algorithms to match drivers with riders. Google uses machine learning to surface the ride advertisements in searches.
- Some companies use machine learning as a primary driver in their business models. Uber, for example, uses algorithms to match drivers with riders. Google uses machine learning to surface the ride advertisements in searches.
- There is also the problem of machine learning bias. Algorithms trained on data sets that exclude certain populations or contain errors can lead to inaccurate models of the world that, at best, fail and, at worst, are discriminatory. When an enterprise bases core business processes on biased models it can run into regulatory and reputational harm.

How to choose the right machine learning model

The process of choosing the right machine learning model to solve a problem can be time consuming if not approached strategically.

Step 1: Align the problem with potential data inputs that should be considered for the solution. This step requires help from data scientists and experts who have a deep understanding of the problem.

Step 2: Collect data, format it and label the data if necessary. This step is typically led by data scientists, with help from data wranglers.

Step 3: Chose which algorithm(s) to use and test to see how well user perform. This step is usually carried out by data scientists.

Step 4: Continue to fine tune outputs until user reach an acceptable level of accuracy. This step is usually carried out by data scientists with feedback from experts who have a deep understanding of the problem.

Importance of human interpretable machine learning

Explaining how a specific ML model works can be challenging when the model is complex. There are some vertical industries where data scientists have to use simple machine learning models because it is important for the business to explain how every decision was made. This is especially true in industries with heavy compliance burdens such as banking and insurance.

Complex models can produce accurate predictions, but explaining to a lay person how an output was determined can be difficult.

1.4 Future of machine learning

While machine learning algorithms have been around for decades, user have attained new popularity as Artificial Intelligence has grown in prominence. Deep learning models, in particular, are most advanced Artificial Intelligence applications.

Machine learning platforms are among enterprise technologies most competitive realms, with most major vendors, including Amazon, Google, Microsoft, IBM and others, racing to sign customers up for platform services that cover the spectrum of machine learning activities, including data collection, data preparation, data classification, model building, training and application deployment.

As machine learning continues to increase in importance to business operations and Artificial Intelligence becomes more practical in enterprise settings, the machine learning platform wars will only intensify.

Machine learning platforms are among enterprise technologies most competitive realms, with most major vendors, including Amazon, Google, Microsoft, IBM and others, racing to sign customers up for platform services that cover the

spectrum of machine learning activities, including data collection, data preparation, data classification, model building, training and application deployment.

As machine learning continues to increase in importance to business operations and Artificial Intelligence becomes more practical in enterprise settings, the machine learning platform wars will only intensify.

Continued research into deep learning and Artificial Intelligence is increasingly focused on developing more general applications. Today Artificial Intelligence models require extensive training in order to produce an algorithm that is highly optimized to perform one task. But some researchers are exploring ways to make models more flexible and are seeking techniques that allow a machine to apply context learned from one task to future, different tasks.

How has machine learning evolved

1642 - Blaise Pascal invents a mechanical machine that can add, subtract, multiply and divide.

1679 - Gottfried Wilhelm Leibniz devises the system of binary code.

1834 - Charles Babbage conceives the idea for a general all-purpose device that could be programmed with punched cards.

1842 - Ada Lovelace describes a sequence of operations for solving mathematical problems using Charles Babbage theoretical punch-card machine and becomes the first programmer.

1847 - George Boole creates Boolean logic, a form of algebra in which all values can be reduced to the binary values of true or false.

1936 - English logician and cryptanalyst Alan Turing proposes a universal machine that could decipher and execute a set of instructions. His published proof is considered the basis of computer science.

1952 - Arthur Samuel creates a program to help an IBM computer get better at checkers the more it plays.

1959 - MADALINE becomes the first artificial neural network applied to a real-world problem: removing echoes from phone lines.

1985 - Terry Sejnowski and Charles Rosenberg artificial neural network taught itself how to correctly pronounce 20,000 words in one week.

1997 - IBM Deep Blue beat chess grandmaster Garry Kasparov.

1999 - A CAD prototype intelligent workstation reviewed 22,000 mammograms and detected cancer 52% more accurately than radiologists did.
2006 - Computer scientist Geoffrey Hinton invents the term deep learning to describe neural net research.
2012 - An unsupervised neural network created by Google learned to recognize cats in YouTube videos with 74.8% accuracy.
2014 - A chatbot passes the Turing Test by convincing 33% of human judges that it was a Ukrainian teen named Eugene Gottman.
2014 - Google AlphaGo defeats the human champion in Go, the most difficult board game in the world.
2016 - Lip Net, DeepMind Artificial Intelligence system, identifies lip-read words in video with an accuracy of 93.4%.
2019 - Amazon controls 70% of the market share for virtual assistants in the U.S.

1.5 Chat Analysing

Today one of the trendy social media platforms is…. guess what, One and only WhatsApp. It is one of the favourite social media platforms among all of them because of its attractive features. It has more than 2B users worldwide and according to one survey an average user spends more than 195 minutes per week on WhatsApp. How terrible the above statement is.

WhatsApp Analyzer means analysing WhatsApp group activities. It tracks conversation and analyses how much time spending or saying it as wasting on WhatsApp. The aim of this article is to provide step by step guide to build WhatsApp analyser using python. Here in this project used different python libraries which help to extract useful information from raw data.

WhatsApp Analyzer means analysing WhatsApp group activities. It tracks conversation and analyses how much time spending or saying it as wasting on WhatsApp. The aim of this article is to provide step by step guide to build WhatsApp analyser using python. Different python libraries are used which help to extract useful information from the raw data. WhatsApp provides the feature of exporting chats, so export the chat and save the file. I, create a python program that will extract the Date, Username of Author, Time, Messages from exported chat file and creating a data frame, and storing all data in it.

Data Analysis with Python

Data Analysis is the technique to collect, transform, and organize data to make future predictions, and make informed data-driven decisions. It also helps to find possible solutions for a business problem. There are six steps for Data Analysis. User are

- Ask or Specify Data Requirements
- Prepare or Collect Data
- Clean and Process
- Analyse
- Share
- Act or Report

Each step has its own process and tools to make overall conclusions based on the data.

1.6 Challenges for Analysing Chat

Many challenges this will face while analysing WhatsApp chat. For analyse the chat have to export chat from the WhatsApp and this want chat in the proper format like time and date must have the same format like to use in our programming, otherwise this cannot analyse chat.

Operational feasibility is mainly concerned with issues like whether the system will be used if it is developed and implemented, whether there will be resistance from the users which will affect the possible application benefits. It is the ability to utilize, support and perform the necessary tasks of a system or program. It includes everyone who creates, operates or uses the system or program. It is the measure of how well a proposed system solves the problem and takes advantages of the opportunities identified during the scope definition and problem analysis phases. This system helps in many ways. It shows the number of users using WhatsApp and gives the data information of their sharing data. Which is organized in Pie-chart and Bar chart.

Operational feasibility is mainly concerned with issues like whether the system will be used if it is developed and implemented, whether there will be resistance from the users which will affect the possible application benefits. It is the ability to utilize, support and perform the necessary tasks of a system or program. It includes everyone who creates, operates or uses the system or program.

Many challenges this will face while analysing WhatsApp chat. For analyse the chat have to export chat from the WhatsApp and wants chat in the proper format like time and date must have the same format like to use in our programming, otherwise this cannot analyse chat.

1.7 Motivation

The motivation behind to build this WhatsApp chat analyser is people around the world. Maximum people used WhatsApp user do not even know how much time user spend on WhatsApp. due to this user do not even aware about that and user avoid many things due to WhatsApp.

1.8 Organization of Report

Chapter one contains the introduction of the proposed system of transfer learning for image classification and describes the various problems and describes the motivation and scope for the project.

Chapter two includes the literature review which refers to the study that has been carried out in the domain of transfer learning and neural network over the previous years and also various technologies which are used for classification of image.
Chapter three explains proposed approach and system architecture which includes the detail of how the system has been developed with different components and modules.
Chapter four discusses the results that were generated from the proposed system of transfer learning and showcases every necessary output needed to describe the proposed system precisely.
Chapter five contains conclusion about the proposed system and it also describe limitation of study and future scope of proposed system.

CHAPTER 2
LITERATURE REVIEW

Recent years have seen much discussion of machine intelligence and what this means for the health, productivity, and wellbeing. In such discussion, machine learning apparently promises to save lives, address global challenges such as climate change and add trillions of dollars to the global economy through increasing productivity while doing so it also fundamentally changes the nature of work, and shapes, or defines, the choices people make in everyday life. In this decade the upcoming technologies are mainly dependent on data. This data can only be obtained if there is some research applied on the context of the requirements of the tool. Since a lot of machine learning enthusiasts develop models which helps solve multiple problems the requirements of appropriate data are very large scale this project aims to provide a better understanding towards various types of chats. This analysis proves to be better input to machine learning models which essentially explore the chat data. These models require proper learning instances which provides better accuracy for these models. The project ensures to provide an in-depth exploratory data analysis on various types of WhatsApp chats.

Ahmed, I., Fiaz (2011) discuss 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 user are using WhatsApp for communicating with their friends. User 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 the users are spending time on their smart phone is WhatsApp. Methods used in this survey is to analyse the intensity of WhatsApp usage and its popular services and to identify the degree of positive or negative impacts of using WhatsApp.

Aharony, N. (2016) presents analytical study to evaluate the Effectiveness of WhatsApp Application in Karachi. The Study will be an important research work for exploring the possibilities of emergence of WhatsApp as the leading mobile messaging application in Pakistan. With the advancement of digital technology and the mergence of mobile phones in Pakistan, the communication scenario has completely changed The increasing trend of smart phones and social networking applications in Pakistan has made communication faster and easier than at any time in history. Now, people may not have enough money to eat, enough place to sleep and enough dress to wear but have mobile phone in their pockets to interact with their family members, friends and customers. With the changing scenario, use of quantitative and qualitative research techniques has also increased with the passage of time. Procedures were devised for the measurement of nature and effect of communication devises on human behaviour. During the same period, smart phones and instant messaging application like WhatsApp, Viber and Skype took over the world of communication in Pakistan.

D. Radha, R. Jayaparvathy, D. Yamini (2016) presents the dataset of WhatsApp group chat used for analysis is of 1 year 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, 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, age group of senders, gender. RStudio the most favoured IDE for R is been used to perform exploratory data analysis and visualization for the collected data largely because of its open-source nature.

Jessica Ho, Ping Ji, Weifang Chen, Raymond Hsieh (2009) WhatsApp provides its users with various forms of communications, namely user-to-user communications, broadcast messages, and group chats. When communicating, users may exchange plain text messages, as well as multimedia files, contact cards, and geolocation information. Each user is associated with its profile, set of information that includes his/her WhatsApp name, status line, and avatar. The profile of each user is stored on a central system, from which it is downloaded by other WhatsApp users that include that user in

their contacts. The central systems provide also other services, like user registration, authentication, and message relay.

Mike Dickson (2006) In this research the forensically acquire and analyse the device-stored data and network traffic of 20 popular instant messaging applications for Android. It was able to reconstruct some or the entire message content from 16 of the 20 applications tested, which reflects poorly on the security and privacy measures employed by these applications but may be construed positively for evidence collection purposes by digital forensic practitioners. This work shows which features of these instant messaging applications leave evidentiary traces allowing for suspect data to be reconstructed or partially reconstructed, and whether network forensics or device forensics permits the reconstruction of that activity. This shows that in most cases were able to reconstruct or intercept data such as: passwords, screenshots taken by applications, pictures, videos, audio sent, messages sent, sketches, profile pictures and more.

C. Boyle (2017) This article presents a study on the nodes positioning confidence in mobile collaborative scenarios. The confidence of nodes positioning is an indicator that allows providing appropriate services and information to mobile users. In mobile ad-hoc networks, nodes with positioning capabilities could share their position with their neighbours in order to allow these latter to determine their own location collaboratively and increasing their positioning confidence. User study the fluctuation of this confidence in mobile scenarios populated by nodes with and without positioning capabilities. This analyses the effect of changing the maximum communication range of the nodes, the number of nodes with positioning capabilities present in the environment. This also studied the impact that produces the degree of nodes mobility. The findings of this study are presented and discussed.

2.1Aims and Objectives

- To learn python technology and its modules.
- To design and implement WhatsApp chat analyser using python.
- To convert WhatsApp conversation into text format and give the brief analysis of conversation.

CHAPTER 3

PROPOSED APPROACH AND SYSTEM ARCHITECTURE

Data pre-processing, the initial part of the project is to understand implementation and usage of various python-built modules. The above process helps us to understand why different modules are helpful rather than implementing those functions from scratch by the developer. These various modules provide better code representation and user understandability. The following libraries are used such as NumPy, SciPy pandas, csv, matplotlib, sys, re, emoji, nltk, seaborn etc.

Exploratory data analysis, first step in this to apply a sentiment analysis algorithm which provides positives negative and neutral part of the chat and is used to plot pie chart based on these parameters. To plot a line graph which shows author and message count of each date, to plot a line graph which shows author and message count of each author, ordered graph of date vs message count, media sent by authors and their count, Display the message which is di not have authors, plot graph of hour vs message count.

The above process helps us to understand why different modules are helpful rather than implementing those functions from scratch by the developer. These various modules provide better code representation and user understandability. The following libraries are used such as NumPy, SciPy pandas, csv, matplotlib, sys, re, emoji, nltk seaborn etc.

3.1 Flow Chart

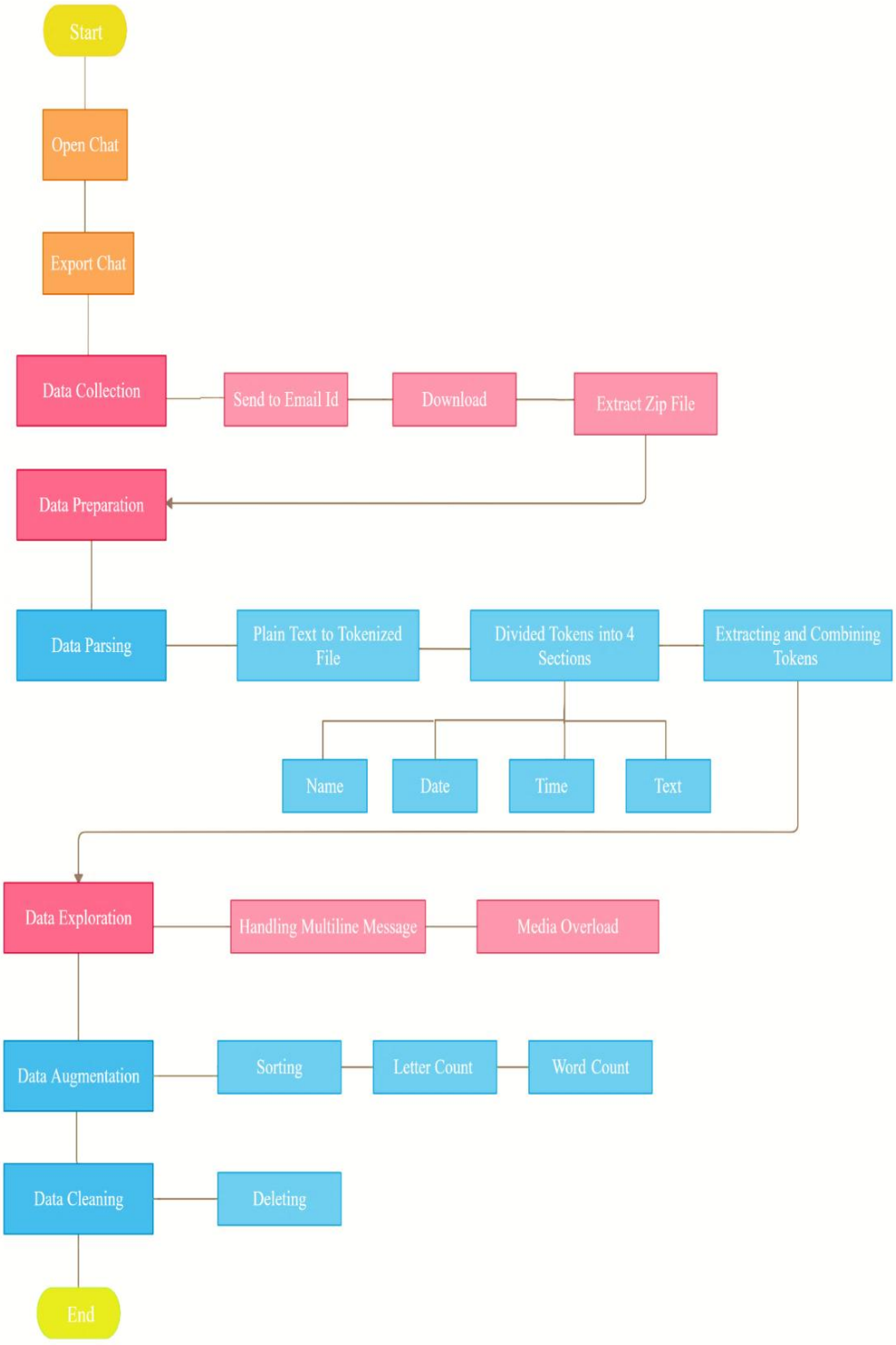

Fig 3.1 System architecture

3.2 Data Collection

First off, it requires a WhatsApp conversation to analyse. Open a WhatsApp conversation you wish to analyze preferably a group chat since user tend to be larger and use the Export Chat functionality to send the entire conversation in text format to your email ID. When prompted by WhatsApp, ensure that the user do not export any media otherwise it might take ages to export. Download the exported chat from the email inbox.

3.3 Data Preparation

Just like raw vegetables have to be cooked and garnished with a variety of spices to make them palatable to humans, so also this plain text file will have to be parsed and tokenized in a meaningful manner in order to be served. First, in order to detect if a line of text is a new message or belongs to a multi-line message, user will have to check if that line begins with a Date and Time, for which th will need a little bit of regular expression matching. Once again, it will require some more regular expression matching. Objective is to detect the author of this message. While there could be a variety of patterns depending on how you have saved your friends names in phone contacts app.

Now user have been able to identify the Date, Time, Author and Message tokens in a single message, it is time to split each line based on the separator tokens like commas, hyphens, colons and the spaces, so that the required tokens can be extracted and saved in a data frame. This time, invert things by highlighting the separator tokens instead of the Date, Time, Author and Message tokens.

3.4 Data Parsing

Parsing is defined as the processing of a piece of python program and converting these codes into machine language. In general, it can say parse is a command for dividing the given program code into a small piece of code for analysing the correct syntax. User have come to the last stage of data parsing, for which the user will have to read the entire WhatsApp text file, identify and extract tokens from each line and capture all data in tabular format within a list.

3.5 Data Exploration

Exploring data sets and developing deep understanding about the data is one of the most important skills every data scientist should possess. People estimate that the time spent on these activities can go as high as 80% of the project time in some cases. Python has been gaining a lot of ground as preferred tool for data scientists lately, and for the right reasons. Ease of learning, powerful libraries with integration of C/C++, production readiness and integration with web stack are some of the main reasons for this move lately. Using Pandas data frame user can show the number of entries, unique entries, most frequently occurring entries and frequency of the most frequently occurring entries for each column in the data frame. NumPy, Matplotlib, Seaborn, and Pandas are the powerful libraries to perform data exploration in Python. User do not collect new data, rather transform the already present data. number of entries, unique entries, most frequently occurring entries and frequency of the most frequently occurring entries for each column in the data frame. Through this process, functions like sorting, letter count and word count will be very easy.

3.6 Data Augmentation

Data augmentation is the process of increasing the amount and diversity of data. User do not collect new data, rather transform the already present data. Ease of learning, powerful libraries with integration of C/C++, production readiness and integration with web stack number of entries, unique entries, most frequently occurring entries and rather transform the already present data. frequency of the most frequently occurring entries for each column in the data frame. frequency of the most frequently occurring entries for each column in the data frame. Through this process, functions like sorting, letter count and word count will be very easy.

3.7 Data Cleaning

Data cleaning or cleansing is the process of detecting and correcting or removing corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

3.8 Naive Bayes algorithm

It is a classification technique based on Bayes Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Step 1: Convert the data set into frequency.

Step 2: Creating likelihood table by finding the probabilities.

Step 3: Use Naïve Bayes theorem equation to calculate posterior probability.

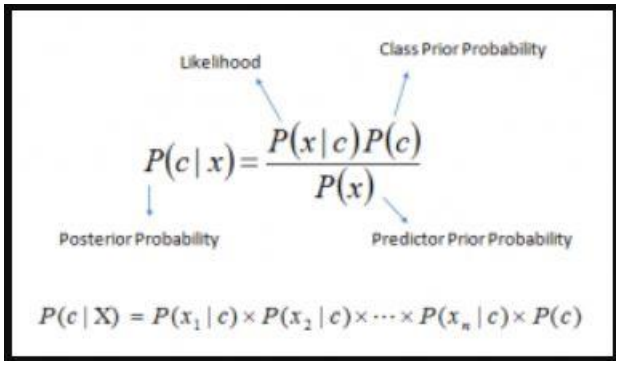

Fig 3.1 Naïve Bayes theorem equation.

Figure 3.1 defines

- $P(c|x)$ is the posterior probability of class (c, target) given predictor (x, attributes).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

CHAPTER 4

IMPLEMENTATION

It is the measure of the specific technical solution and the availability of the technical resources and expertise. It is one of the first studies that must be conducted after tool has been identified. A technical study of feasibility is an assessment of the logistical aspects of business operation. This is considered with specifying equipment and software that will successfully satisfy the user requirement. The technical needs of the system may vary considerably but should include the facility to produce outputs in a given time, response time under certain conditions and the ability to process a certain amount of transaction at a certain speed. The proposed system is developed by using Jupyter software. Jupyter is non-profit organization created to develop open-source software, open standards, and services for interactive computing across dozens of programming languages. The idea is to implement a data processing code using python to make better sense of WhatsApp group chat data.

4.1 Artificial Intelligence

The evolution in Artificial Intelligence has advanced the development of human society in own time, with dramatic revolutions shaped by both theories and techniques. However, the multidisciplinary and fast-growing features make Artificial Intelligence is a field in which it is difficult to be well understood. Find the area is in the sustainable development and its impact continues to grow. From the perspective of reference behaviour, the decrease in self-references indicates that the Artificial Intelligence is becoming more and more open-minded. The influential papers/researchers/institutions identified outline landmarks in the development of this field.

4.2 Machine Learning

Machine learning is a field of study and is concerned with algorithms that learn from examples.

Classification is a task that requires the use of machine learning algorithms that learn how to assign a class label to examples from the problem domain. An easy-to-understand example is classifying emails as spam or not spam.

There are many different types of classification tasks that you may encounter in machine learning and specialized approaches to modelling that may be used for each.

In this tutorial, you will discover different types of classification predictive modelling in machine learning.

- Classification predictive modelling involves assigning a class label to input examples.
- Binary classification refers to predicting one of two classes and multi-class classification involves predicting one of more than two classes.
- Multi-label classification involves predicting one or more classes for each example and imbalanced classification refers to classification tasks where the distribution of examples across the classes is not equal.
    - Classification Predictive Modelling
    - Binary Classification
    - Multi-Class Classification
    - Multi-Label Classification
    - Imbalanced Classification

Classification Predictive Modelling

In machine learning, classification refers to a predictive modelling problem where a class label is predicted for a given example of input data.

Examples of classification problems include:

- Given an example, classify if it is spam or not.
- Given a handwritten character, classify it as one of the known characters.
- Given recent user behaviour, classify as churn or not.

A model will use the training dataset and will calculate how to best map examples of input data to specific class labels. As such, the training dataset must be sufficiently representative of the problem and have many examples of each class label.

Class labels are often string values, e.g., spam, not spam, and must be mapped to numeric values before being provided to an algorithm for modelling. This is often referred to as label encoding, where a unique integer is assigned to each class label, e.g. spam = 0, no spam = 1.

There are many different types of classification algorithms for modelling classification predictive modelling problems.

A model will use the training dataset and will calculate how to best map examples of input data to specific class labels. As such, the training dataset must be sufficiently representative of the problem and have many examples of each class label.

There is no good theory on how to map algorithms onto problem types; instead, it is generally recommended that a practitioner use controlled experiments and discover which algorithm and algorithm configuration results in the best performance for a given classification task.

Binary Classification

Binary classification refers to those classification tasks that have two class labels.
Examples include:

- Email spam detection.
- Churn prediction.
- Conversion prediction.

Typically, binary classification tasks involve one class that is the normal state and another class that is the abnormal state.
For example, not spam is the normal state and spam is the abnormal state. Another example is cancer not detected is the normal state of a task that involves a medical test and cancer detected is the abnormal state.

The class for the normal state is assigned the class label 0 and the class with the abnormal state is assigned the class label. It is common to model a binary classification task with a model that predicts a Bernoulli probability distribution for each example. The Bernoulli distribution is a discrete probability distribution that covers a case where an event will have a binary outcome as either a 0 or 1. For classification, this means that the model predicts a probability of an example belonging to class 1, or the abnormal state.
Popular algorithms that can be used for binary classification include:

- Logistic Regression
- k-Nearest Neighbours
- Decision Trees
- Support Vector Machine

- Naive Bayes

Some algorithms are specifically designed for binary classification and do not natively support more than two classes; examples include Logistic Regression and Support Vector Machines.

Multi-Class Classification

Multi-class classification refers to those classification tasks that have more than two class labels.

Examples include

- Face classification.
- Plant species classification.
- Optical character recognition.

Unlike binary classification, multi-class classification does not have the notion of normal and abnormal outcomes. Instead, examples are classified as belonging to one among a range of known classes. The number of class labels may be very large on some problems. For example, a model may predict a photo as belonging to one among thousands or tens of thousands of faces in a face recognition system.

Problems that involve predicting a sequence of words, such as text translation models, may also be considered a special type of multi-class classification. Each word in the sequence of words to be predicted involves a multi-class classification where the size of the vocabulary defines the number of possible classes that may be predicted and could be tens or hundreds of thousands of words in size. The number of class labels may be very large on some problems. For example, a model may predict a photo as belonging to one among thousands or tens of thousands of faces in a face recognition system.

It is common to model a multi-class classification task with a model that predicts a Multimodule probability distribution for each example. The Multimodule distribution is a discrete probability distribution that covers a case where an event will have a categorical outcome, e.g. K in $\{1, 2, 3, \ldots, K\}$. For classification, this means that the model predicts the probability of an example belonging to each class label.

Many algorithms used for binary classification can be used for multi-class classification.

Popular algorithms that can be used for multi-class classification include:

- k-Nearest Neighbours.
- Decision Trees.
- Naive Bayes.
- Random Forest.
- Gradient Boosting.

Algorithms that are designed for binary classification can be adapted for use for multi-class problems.

Multi-Label Classification

Multi-label classification refers to those classification tasks that have two or more class labels, where one or more class labels may be predicted for each example.
Consider the example of photo classification, where a given photo may have multiple objects in the scene and a model may predict the presence of multiple known objects in the photo, such as bicycle, apple, person, etc.
This is unlike binary classification and multi-class classification, where a single class label is predicted for each example.

It is common to model multi-label classification tasks with a model that predicts multiple outputs, with each output taking predicted as a Bernoulli probability distribution. This is essentially a model that makes multiple binary classification predictions for each example.

Classification algorithms used for binary or multi-class classification cannot be used directly for multi-label classification. Specialized versions of standard classification algorithms can be used, so-called multi-label versions of the algorithms, including:

- Multi-label Decision Trees
- Multi-label Random Forests
- Multi-label Gradient Boosting

Another approach is to use a separate classification algorithm to predict the labels for each class. Next, take a closer look at a dataset to develop an intuition for multi-label classification problems.

It is common to model multi-label classification tasks with a model that predicts multiple outputs, with each output taking predicted as a Bernoulli probability distribution. This is essentially a model that makes multiple binary classification predictions for each example. The function make_multilabel_classification () can be used to generate a synthetic multi-label classification dataset.

Imbalanced Classification

Imbalanced classification refers to classification tasks where the number of examples in each class is unequally distributed. Typically, imbalanced classification tasks are binary classification tasks where the majority of examples in the training dataset belong to the normal class and a minority of examples belong to the abnormal class.

Examples include:

- Fraud detection.
- Outlier detection.
- Medical diagnostic tests.

These problems are modelled as binary classification tasks, although may require specialized techniques. Specialized techniques may be used to change the composition of samples in the training dataset by under sampling the majority class or oversampling the minority class.

4.3 Tools and Technologies

This section describes various technologies those are being used in the development of the proposed system. The functions and packages are explained along with features and components. Every module and its use in the proposed system has been highlighted.

4.3.1 Regex

This module provides regular expression matching operations similar to those found in Perl.

Both patterns and strings to be searched can be Unicode strings as well as 8-bit strings. However, Unicode strings and 8-bit strings cannot be mixed: that is, you cannot match a Unicode string with a byte pattern or vice-versa; similarly, when asking for a substitution, the replacement string must be of the same type as both the pattern and the search string.

Regular expressions use the backslash character \ to indicate special forms or to allow special characters to be used without invoking their special meaning. This collides with Python usage of the same character for the same purpose in string literals; for example, to match a literal backslash, one might have to write \\\ as the pattern string, because the regular expression must be \\, and each backslash must be expressed as \\ inside a regular Python string literal. Also, please note that any invalid escape sequences in Python usage of the backslash in string literals now generate a Deprecation Warning and in the future this will become a Syntax Error. This behaviour will happen even if it is a valid escape sequence for a regular expression.

Both patterns and strings to be searched can be Unicode strings as well as 8-bit strings. However, Unicode strings and 8-bit strings cannot be mixed: that is, you cannot match a Unicode string with a byte pattern or vice-versa; similarly, when asking for a substitution, the replacement string must be of the same type as both the pattern and the search string.

The solution is to use Python raw string notation for regular expression patterns; backslashes are not handled in any special way in a string literal prefixed with r. So, \n is a two-character string containing \ and n, while \n is a one-character string containing a newline. Usually, patterns will be expressed in Python code using this raw string notation.

It is important to note that most regular expression operations are available as module-level functions and methods on compiled regular expressions. The functions are shortcuts that do not require you to compile a regex object first, but miss some fine-tuning parameters.

Regular expressions use the backslash character \ to indicate special forms or to allow special characters to be used without invoking their special meaning. This

collides with Python usage of the same character for the same purpose in string literals; for example, to match a literal backslash, one might have to write \\\\ as the pattern string, because the regular expression must be \\, and each backslash must be expressed as \\ inside a regular Python string literal. Also, please note that any invalid escape sequences in Python usage of the backslash in string literals now generate a 1 and in the future this will become a Syntax Error. This behaviour will happen even if it is a valid escape sequence for a regular expression.

A regular expression specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression or if a given regular expression matches a particular string, which comes down to the same thing.

It is important to note that most regular expression operations are available as module-level functions and methods on compiled regular expressions. The functions are shortcuts that do not require you to compile a regex object first, but miss some fine-tuning parameters

Regular expressions can be concatenated to form new regular expressions; if *A* and *B* are both regular expressions, then *AB* is also a regular expression. In general, if a string *p* matches *A* and another string *q* matches *B*, the string *pq* will match AB. This holds unless *A* or *B* contain low precedence operations; boundary conditions between *A* and *B*; or have numbered group references. Thus, complex expressions can easily be constructed from simpler primitive expressions like the ones described here. For details of the theory and implementation of regular expressions, consult the Friedl book, or almost any textbook about compiler construction.

A brief explanation of the format of regular expressions follows. For further information and a gentler presentation, consult the Regular Expression HOWTO.

Regular expressions can contain both special and ordinary characters. Most ordinary characters, like A, a, or 0, are the simplest regular expressions; user simply match themselves. You can concatenate ordinary characters, so last matches the string last. In the rest of this section, this will write REs in this special style, usually without quotes, and strings to be matched in single quotes.

Some characters, like or are special. Special characters either stand for classes of ordinary characters, or affect how the regular expressions around them are interpreted.

Repetition qualifiers *, +, ?, {m, n} etc, cannot be directly nested. This avoids ambiguity with the non-greedy modifier suffix? and with other modifiers in other implementations. To apply a second repetition to an inner repetition, parentheses may be used. For example, the expression matches any multiple of six a character.

Some characters, like or are special. Special characters either stand for classes of ordinary characters, or affect how the regular expressions around them are interpreted.

4.3.2 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented application package interface for embedding plots into applications using general-purpose graphic user interface toolkits like Tkinter, wxPython, Qt and GTK+.There is also a procedural pylab interface based on a state machine like OpenGL, designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of matplotlib.

Matplotlib was originally written by John D. Hunter, has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib lead developer shortly before John Hunter death in August 2012 and further joined by Thomas Caswell.

As of 23 June 2017, matplotlib 2.0.x supports Python versions 2.7 through 3.6. Matplotlib 1.2 is the first version of matplotlib to support Python 3. x. Matplotlib 1.4 is the last version of matplotlib to support Python 2.6. In Fig shows output of matplotlib.

4.3.3 NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the

competing Num array into Numeric, with extensive modifications. NumPy is open-source software and has many contributors

• Data Operations in NumPy

The most important object defined in NumPy is an N-dimensional array type called ND array. It describes the collection of items of the same type. Items in the collection can be accessed using a zero-based index. The most important object defined in NumPy is an N-dimensional array type called ND array. It describes the collection of items of the same type. Items in the collection can be accessed using a zero-based index.

• Traits

NumPy targets the Python reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since both are interpreted, and both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available, SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and

Using NumPy in Python gives functionality comparable to MATLAB since both are interpreted, and both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, LAPACK for efficient linear algebra computations. Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with

multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy 21 array as universal data structure in OpenCV for images, extracted feature points, filter
kernels and many more vastly simplifies the programming workflow and debugging.

• The ND array Data Structure

The core functionality of NumPy is its ND array, for n-dimensional array, data structure. These arrays are stride views on memory. In contrast to Python built-in list data structure which, despite the name, is a dynamic array, these arrays are homogeneously typed: all elements of a single array must be of the same type.
Such arrays can also be views into memory buffers allocated by C/C++, Cpython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries notably BLAS and LAPACK. NumPy has built-in support for memory-mapped ND arrays.
The core functionality of NumPy is its ND array, for n-dimensional array, data structure. These arrays are stride views on memory. In contrast to Python built-in list data structure which, despite the name, is a dynamic array, these arrays are homogeneously typed: all elements of a single array must be of the same type.

• Limitations

Inserting or appending entries to an array is not as trivially possible as it is with Python lists. The np.pad(...) routine to extend arrays actually creates new arrays of the desired shape and padding values, copies the given array into the new one and returns it. NumPy np. concatenate [a1, a2] operation does not actually link the two arrays but returns a new one, filled with the entries from both given arrays in sequence. Reshaping the dimensionality of an array with np. reshape is only possible as long as the number of elements in the array does not change. These circumstances originate from the fact that NumPy arrays must be views on contiguous memory buffers. A replacement package called Blaze attempts to overcome this limitation.

Algorithms that are not expressible as a vectorized operation will typically run slowly because must be implemented in pure Python, while vectorization may increase

memory complexity of some operations from constant to linear, because temporary arrays must be created that are as large as the inputs. Runtime compilation of numerical code has been implemented by several groups to avoid these problems; open-source solutions that interoperate with NumPy include SciPy. Weave, numexpr and Numba. Cpython is a static-compiling alternative.

4.3.4 Pandas

In computer programming, Pandas is a software library written for the Python programming for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term panel data, an econometrics term for data sets that include observations over multiple time periods for the same individuals.

In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

• Library Features

Following are the features of pandas.

1. Data Frame object for data manipulation with integrated indexing.
2. Tools for reading and writing data between in-memory data structures and different file formats.
3. Data alignment and integrated handling of missing data.
4. Reshaping and pivoting of data sets.
5. Label-based slicing, fancy indexing, and sub-setting of large data sets.
6. Data structure column insertion and deletion.
7. Group by engine allowing split-apply-combine operations on data sets.
8. Data set merging and joining.
9. Hierarchical axis indexing to work with high-dimensional data in a lower dimensional data structure.
10. Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.
11. Provides data filtration.

12. The library is highly optimized for performance, with critical code paths written in python or C.

2.3.5 Date Time

The datetime module supplies classes for manipulating dates and times. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.

See also

Module calendar

General calendar related functions.

Module time

Time access and conversions.

Module zone info

Concrete time zones representing the IANA time zone database.

Package dateutil

Third-party library with expanded time zone and parsing support.

Aware and Naive Objects

Date and time objects may be categorized as aware or naive depending on whether or not user include time zone information. With sufficient knowledge of applicable algorithmic and political time adjustments, such as time zone and daylight-saving time information, an aware object can locate itself relative to other aware objects. An aware object represents a specific moment in time that is not open to interpretation.

A naive object does not contain enough information to unambiguously locate itself relative to other date/time objects. Whether a naive object represents Coordinated Universal Time, local time, or time in some other time zone is purely up to the program, just like it is up to the program whether a particular number represents metres, miles,

or mass. Naive objects are easy to understand and to work with, at the cost of ignoring some aspects of reality.

For applications requiring aware objects, datetime and time objects have an optional time zone information attribute, tzinfo, that can be set to an instance of a subclass of the abstract tzinfo class. These tzinfo objects capture information about the offset from UTC time, the time zone name, and whether daylight saving time is in effect.

Only one concrete tzinfo class, the time zone class, is supplied by the datetime module. The time zone class can represent simple time zones with fixed offsets from UTC, such as UTC itself or North American EST and EDT time zones. Supporting time zones at deeper levels of detail is up to the application. The rules for time adjustment across the world are more political than rational, change frequently, and there is no standard suitable for every application aside from UTC.

Common Properties

The date, datetime, time, and time zone types share these common features:

1.Objects of these types are immutable.

2.Objects of these types are hash able, meaning that user can be used as dictionary keys.

3.Objects of these types support efficient pickling via the pickle module.

4.3.6 Emoji

For the past several years, this have seen a huge digital transformation in almost every field and aspect of users life. This can easily observe that how this is now more dependent on technologies than this were ever before. This can see this in every part of the daily life and observe it in daily routines. One part of life where this can see the huge impact of digital transformation and the role of technology is communication. Technologies have created a bridge, or this can say, filled the gap so that now this is more comfortable and find it more convenient to communicate with anyone. With the help of technology, this can communicate with anyone and from any part of the world.

One important change can see in the communication process after the digital transformation is the use of messages, text messages, emails etc, is increased so much. This can now see that messages have been an important part of users life, and if this have to communicate or pass a piece of information to someone or publish an important notice, the most common way choose to do it is through messages. It can be text messages, messages on a particular application, emails etc., but the one thing that has become much common in all of these is, use of emojis in every messaging application and even in emails & in text messages too.

Emojis: Emojis are very small digital images used to express an idea or emotion of the sender. Thus, Emojis are very helpful in saving time and making the sender elaborate their mood easily. Emojis are also very helpful for receivers as, with the help of emojis, user can easily understand with what mood the sender has sent the message.

Emojis have become so important that user have also been introduced in programming languages and used by developers for so long. This is going to study the Python Emoji module in this tutorial, and this will learn how this can use this module to print emojis through a Python program.

Python Emoji Module

The use of Emojis in Python has also become very common, and have many ways by which can use & print emojis through a Python program. This can print emojis through their CLDR names, Unicode or using the emoji module. Using Emoji Module is the most common way of all the mentioned ones, and this is going to use the Emoji Module in a Python program to print emojis. But first, this will learn in brief about the Python Emoji Module and its installation process.

4.3.7 Word Cloud

Word Cloud or Tag Clouds is a visualization technique for texts that are natively used for visualizing the tags or keywords from the websites. These keywords typically are single words that depict the context of the webpage the word cloud is being made from. These words are clustered together to form a Word Cloud. Each word in this cloud has a variable font size and colour tone. Thus, this representation helps to determine words of prominence. A bigger font size of a word portrays its prominence

more relative to other words in the cluster. Word Cloud can be built in varying shapes and sizes based on the creators vision. The number of words plays an important role while creating a Word Cloud. More number of words does not always mean a better Word Cloud as it becomes cluttery and difficult to read. A Word Cloud must always be semantically meaningful and must represent what it is meant for. Although, there are different ways by which Word Clouds can be created but the most widely used type is by using the Frequency of Words in user corpus. And thus, this will be creating user Word Cloud by using the Frequency type.

Fig 4.1 Word cloud

CHAPTER 5

RESULTS AND DISCUSSIONS

In computer programming, Pandas is a software library written for the Python programming for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term panel data, an econometrics term for data sets that include observations over multiple time period.

5.1 Imported libraries

```
import re
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import *
import datetime as dt
from matplotlib.ticker import MaxNLocator
import regex
import emoji
from seaborn import *
from heatmap import heatmap
from wordcloud import WordCloud , STOPWORDS , ImageColorGenerator
from nltk import *
from plotly import express as px
```

Fig 5. 1 Imported libraries.

Figure 5.1 shows the libraries which are imported. There are so many libraries are used some of them are re, pandas as pd, NumPy as np and so on.

WhatsApp provides us the feature of exporting chats, in step 2, this will create a python program that will extract the Date, Username of Author, Time, Messages from exported chat file and creating a data frame, and storing all data in it.

5.2 Extracted data

```
### Regex pattern to extract username of Author.

def FindAuthor(s):
    patterns = [
        '([w]+):',                          # First Name
        '([w]+[s]+[w]+):',                # First Name + Last Name
        '([w]+[s]+[w]+[s]+[w]+):',      # First Name + Middle Name + Last Name
        '([+]d{2} d{5} d{5}):',           # Mobile Number (India no.)
        '([+]d{2} d{3} d{3} d{4}):',    # Mobile Number (US no.)
        '([w]+)[u263a-U0001f999]+:',      # Name and Emoji
    ]
    pattern = '^' + '|'.join(patterns)
    result = re.match(pattern, s)
    if result:
        return True
    return False


### Extracting Date, Time, Author and message from the chat file.
```

Fig 5.2 Extracting date, time, Author and message from the chat file.

Figure 5.2 using regex pattern used to extract username of author. It shows extracting date, time, Author and message from the chat file. This pattern contains First Name, Middle Name, Last Name, Mobile Number, Name and at last Emoji.

5.3 Creating data frame

```
def getDataPoint(line):
    splitLine = line.split(' - ')
    dateTime = splitLine[0]
    date, time = dateTime.split(', ')
    message = ' '.join(splitLine[1:])
    if FindAuthor(message):
        splitMessage = message.split(': ')
        author = splitMessage[0]
        message = ' '.join(splitMessage[1:])
    else:
        author = None
    return date, time, author, message

### Finally creating a dataframe and storing all data inside that dataframe.
```

Fig 5.3 Creating a data frame and storing all data inside that data frame.

Figure 5.3 shows the creation of data frame and storing all data inside that data frame. The figure shows, the source code which returns date, time, author, message.

```
parsedData = [] # List to keep track of data so it can be used by a Pandas dataframe
### Uploading exported chat file
conversationPath = 'WhatsApp Chat with TE Comp 20-21 Official.txt' # chat file
with open(conversationPath, encoding="utf-8") as fp:
    ### Skipping first line of the file because contains information related to something about
    fp.readline()
    messageBuffer = []
    date, time, author = None, None, None
    while True:
        line = fp.readline()
        if not line:
            break
        line = line.strip()
        if startsWithDateAndTime(line):
            if len(messageBuffer) > 0:
                parsedData.append([date, time, author, ' '.join(messageBuffer)])
            messageBuffer.clear()
            date, time, author, message = getDataPoint(line)
            messageBuffer.append(message)
        else:
            messageBuffer.append(line)
```

Fig 5.4 Parsing of data.

Figure 5.4 shows the parsing of data i.e.it list to keep track of data so it can be used by a Pandas data frame then uploaded the chat files which is exported from WhatsApp and then skipping first line of the file.

```
df = pd.DataFrame(parsedData, columns=['Date', 'Time', 'Author', 'Message']) # Initialising a pa
### changing datatype of "Date" column.
df["Date"] = pd.to_datetime(df["Date"])
```

	Date	Time	Author	Message
0	2019-07-03	6:49 PM	None	Shubham Patil changed this group's icon
1	2019-06-29	2:31 PM	None	Pranjali Jadhav created group "SE Comp 2019-20...
2	2019-07-03	8:18 AM	None	Pranjali Jadhav added you
3	2019-07-03	8:28 AM	None	Pranjali Jadhav added Deepak, Shravani Naik, +...
4	2019-07-03	8:37 AM	None	Pranjali Jadhav added Shubham Patil, +91 [redacted]
...	...	...	...	...
3040	2021-01-21	10:03 PM	None	Prasanna Shinde's security code changed. Tap f...
3041	2021-01-22	11:07 AM	+91 [redacted]	<Media omitted>
3042	2021-01-22	12:02 PM	None	+91 [redacted]'s security code changed. Tap f...
3043	2021-01-22	1:53 PM	+91[redacted]	Hello Grads!!! _Greetings from AptiTech Educat...
3044	2021-01-22	1:54 PM	+91 [redacted]	Dear All, Greetings !!! Do share the followi...

Fig 5.5 Extracted WhatsApp chat accordingly to time, date and messages

Figure 5.5 shows the extracted WhatsApp chat according to time, date and message. It mainly contains time, date, author name and at least message.

5.4 Word Cloud

```
### Word Cloud of mostly used word in our Group
text = " ".join(review for review in df.Message)
wordcloud = WordCloud(stopwords=STOPWORDS, background_color="white").generate(text)
  ### Display the generated image:
plt.figure( figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

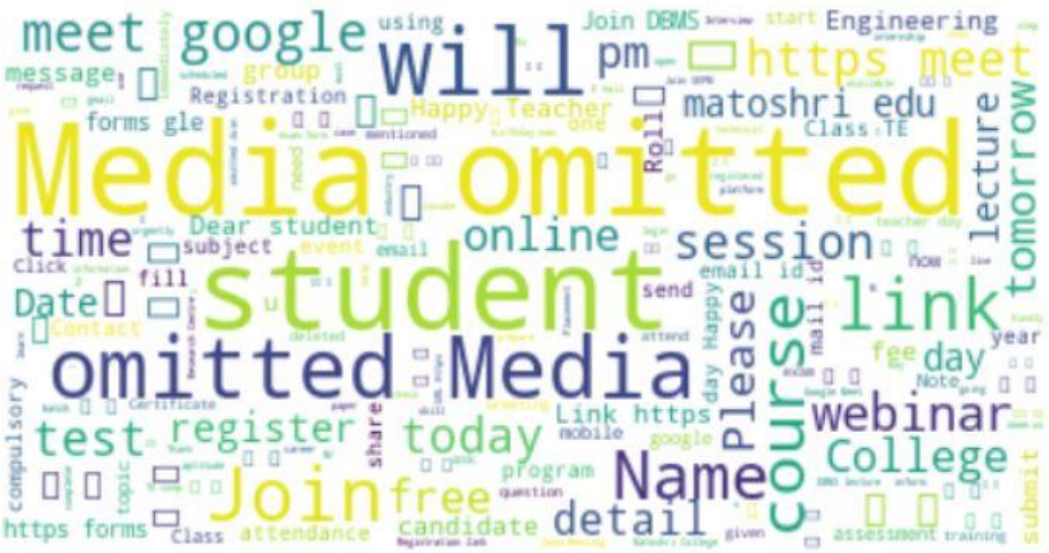

Fig 5.6 Word cloud result

Figure 5.6 shows the word cloud result. This word cloud shows the most frequently used word used in the exported chat from WhatsApp. Whichever word used most frequently would shows in larger size.

5.5 Analysis

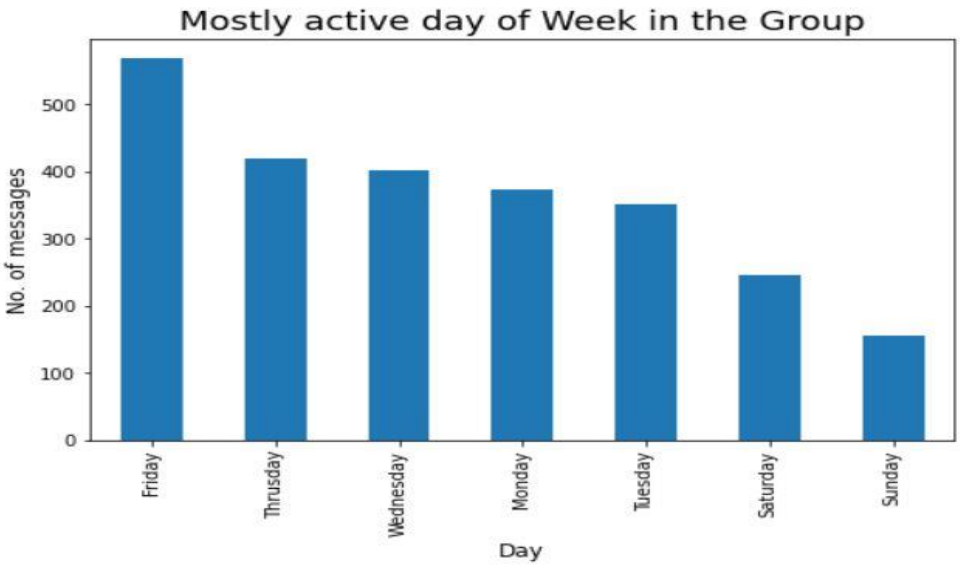

Fig 5.7 Most active day of Week in the Group.

Figure 5.6 shows Mostly active day of week in the group in the form of graph. The graph is between days and number of messages sends in the form of words.

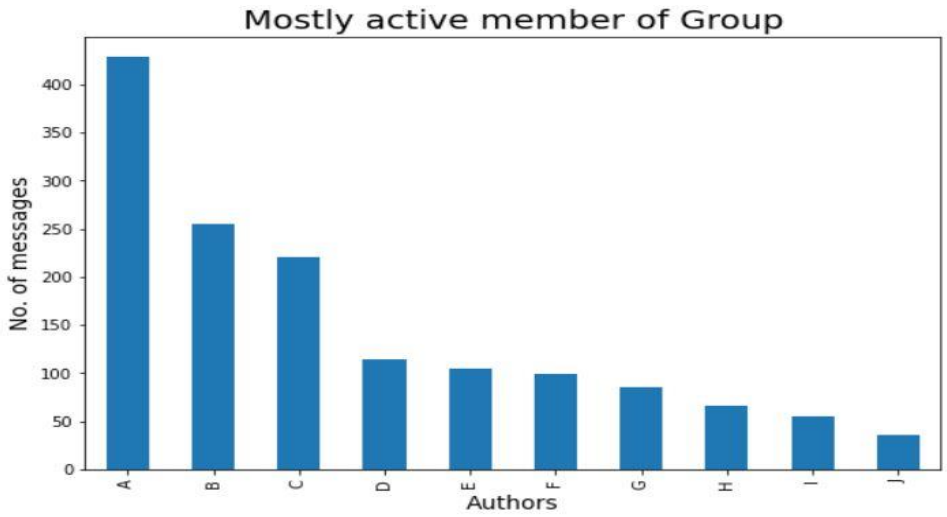

Fig 5.8 Most active member of group.

Figure 5.8 shows the most active members of group on the basis of WhatsApp chat exported. This is in the form of graph between authors and number of messages sent by the members.

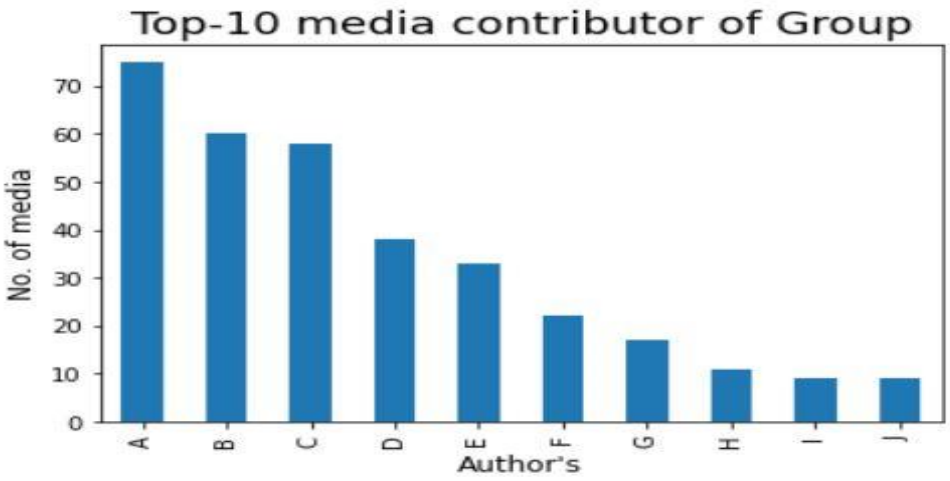

Fig 5.9 Top-10 media contributor of Group

Figure 5.9 shoows the top-10 media contributor of group in the form of bar-graph. This bar grap is between authors and number of medias sent in the group.

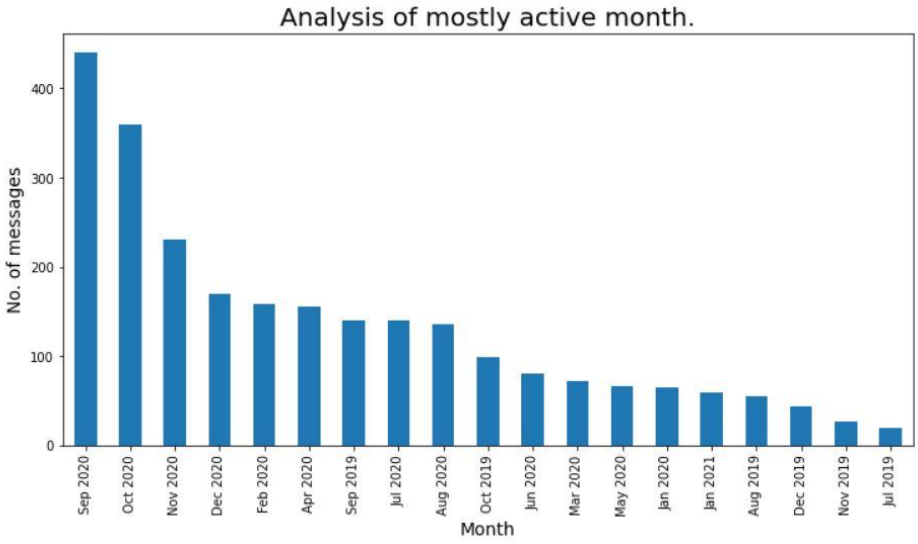

Fig 5.10 Analysis of mostly active month.

Figure 5.10 shows the analysis of mostly active month in the form of bar garph. The bar bar graph is between months and number of mesasge send in which month by the members.

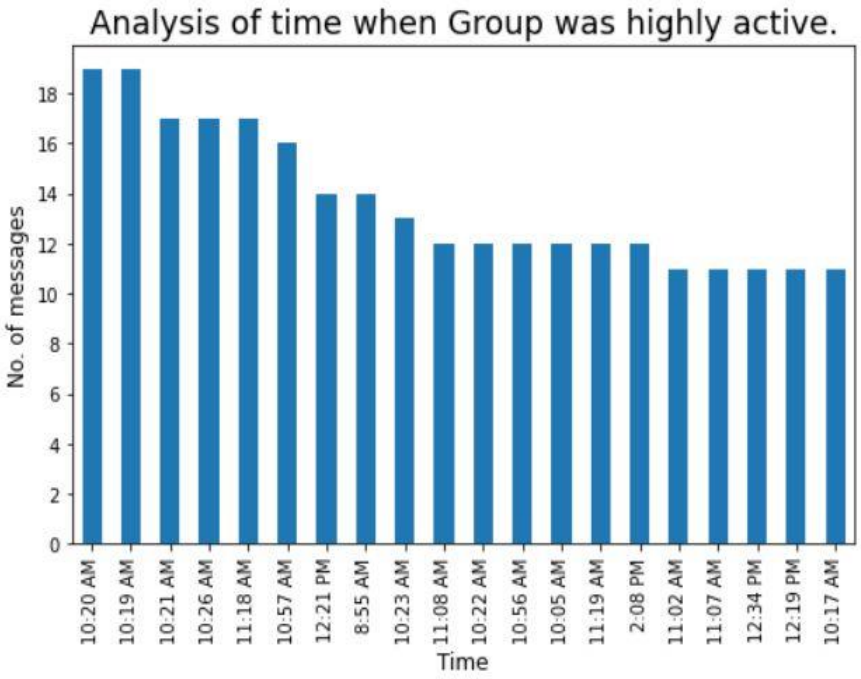

Fig 5.11 Analysis of time when group was highly active.

Figure 5.11 shows the analysis of time when group was highly active. This is based on the chat exported by WhatsApp in the form of bar graph. The bar graph is between time and number of messages.

CHAPTER 6

CONCLUSION

In conclusion, it can be said that the capabilities of the WhatsApp application and the power of the python programming language in implementing whatever network data analysis intended, cannot be overemphasized. This work was able to discuss the WhatsApp application and its libraries, to create an analysis of a WhatsApp group chat and visually represent the top 10 and top 20 users in the chat groups. A pseudocode of the plot was given and at the end, visual representation of the plot was implemented. Also, an analysis of the top 10 and top 20 users were done. The system was done with python, and the python libraries that were implemented includes, NumPy, Pandas, Matplotlib and Seaborn. At the end of the work expected results were obtained and the analysis was able to show the level of participation of the various individuals on the given WhatsApp group. On serious note this system has the ability to analyse any WhatsApp group data input into it.

6.1 Limitations of study

- This cannot analyse chat which are in 12-hour format, this can only analyse chat which are in 24-hour format.
- This cannot analyse chat if the user deletes its chats, because without data this cannot analyse chats.

6.2 Future scope of the work

WhatsApp chat analyser can be very useful for parents as now a days every mobile phone has become a necessity for people of all ages. Children just like teens, are addicted to mobile phones. User play games and chat with their friends all the time. Use of cell phones can lead children to engage in inappropriate behaviours. Texting and sending inappropriate text are a growing problem with children.

References

1. Ahmed, I., Fiaz (2011). “Mobile phone to youngsters: Necessity or addiction”, *African Journal of Business Management*, 5(32), 12512-12519.

2. Aharony, N. (2016). “The Importance of the WhatsApp Family Group: An Exploratory Analysis”, *Aslib Journal of Information Management*, 68(2), 1-37.

3. D. Radha, R. Jayaparvathy, D. Yamini (2016). “Analysis on Social Media Addiction using Data Mining Technique”, *International Journal of Computer Applications*, 139(7), 0975 – 8887.

4. Jessica Ho, Ping Ji, Weifang Chen, Raymond Hsieh (2009). “Identifying google talk”, *IEEE International Conference on Intelligence and Security Informatics*, 9(7), 285-290.

5. Mike Dickson (2006). “An examination into AOL instant messenger 5.5 contact identification.”, *Digital Investigation, ScienceDirect*, 3(4), 227-237.

6. Mike Dickson (2006). “An examination into yahoo messenger 7.0 contact identification”, *Digital Investigation, ScienceDirect*, 3(3), 159-165.

7. C. Boyle (2017), “Messaging App Usage Worldwide: eMarketer’s Updated Forecast, Leader board and Behavioural Analysis”, *eMarketer, Tech. Rep*, 2(6), 175-185.