IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

WhatsApp Chat Analysis and Deployment on Heroku

MISS. CHANDANI SHINDE[1], MISS. ARPITA DAKHORE[2], MRS.P.S. BIHADE[3]
Gramin Technical and Management Campus, Nanded, India
Computer Engineering

Abstract : In the digital age, communication via instant messaging platforms has become ubiquitous, and WhatsApp stands as one of themost popular platforms worldwide. This research paper presents an end-to-end project that encompasses the collection, analysis,and deployment of WhatsApp chat data using the Heroku cloud platform. The project integrates state-of-the-art (NLP) natural language processing techniques to derive valuable insights from chat conversations.
This innovative project bridges the gap between the digital realm and data-driven decision-making.The Heroku cloud platform's deployment ensures accessibility, making these insights readily available to all.

IndexTerms - **WhatsApp, Chat Analysis, Natural Language Processing, Heroku Deployment, Sentiment Analysis, Topic Modeling, Text Summarization, Data Privacy, Data Handling, Cloud Computing, User Interface.**

INTRODUCTION

In today's fast-paced, interconnected world, the Internet and social media continue to revolutionize the exchange of information. These digital landscapes have streamlined communication, offering simplicity and speed. Platforms like WhatsApp, Facebook, Twitter, and various blogs have become indispensable channels for communication, catering to individuals, corporate entities, organizations, and even governments.
Among these platforms, WhatsApp stands as a global behemoth with two billion users spanning 180 countries. A staggering100 billion messages are exchanged daily on WhatsApp, making it an integral part of people's lives. It serves not only as a conduit for information but also as a platform for sharing emotions and meaningful conversations.

Analyzing WhatsApp group chats offers a profound understanding of its participants. It unveils user activity, message types, and expressive styles, while simultaneously monitoring user involvement. For business organizations, WhatsApp becomes an invaluable tool for analytics, enabling the detailed examination of daily or monthly activities. This social media giant is a goldmine of data, capable of revealing patterns and correlations in private or group conversations. It allows for the evaluation of text sentiment, providing insights into whether the communication is "positive" or "negative." Furthermore, it allows us to dissect human behavior.Sentiment analysis, a well-established field, involves techniques, methods, and tools for detecting subjective information like attitudes and opinions from language. Traditionally, it categorizes text as positive, negative, or neutral. However, these three categories often fall short in capturing the full connotation of the underlying tone.To address this, we delve into emotion analysis, aiming to detect emotions such as joy, sadness, anger, fear, guilt, disgust, and shame. Machine learning algorithms, particularly the LinearSVC classifier from the sklearn library, play a pivotal role in this endeavor.Emotion analysis goes beyond sentiment and offers a deep understanding of user temperament by classifying messages according to seven distinct emotions.

The paper primarily focuses on the top 5 active users and their behavioral patterns, employing appropriate graphical representations for analysisThis research centers on WhatsApp chat files, readily exportable for analysis. The utilization of Python libraries reveals captivating insights, and behavioral analysis is facilitated by machine learning, specifically the LinearSVC classifier.
The paper's key contributions encompass the extraction of insights such as active days, active users, and overall group usage from chat data. The visualizations aid in comprehending the findings, while behavioral analysis hones in on the top 5 users.

BACKGROUND AND RELATED WORK

The advent of digital communication platforms has significantly reshaped the landscape of human interaction and data exchange. In this digital age, instant messaging applications have emerged as dominant channels for communication, facilitating real-time exchanges and connecting users across the globe. Among these platforms, WhatsApp has surged to the forefront, amassing a staggering two billion users in over 180 countries, with more than 100 billion messages transmitted on a daily basis [1] [2]. It has transcended its status as a mere messaging app to become an integral part of people's lives, serving as a platform for sharing not only information but also emotions and meaningful conversations.

The analysis of WhatsApp chat data has grown in importance in parallel with its expanding user base. This data-rich resource holds a trove of insights into human interactions, behavioral patterns, and the dynamics of digital communication. As such, the analysis of WhatsApp chat data offers an invaluable opportunity to understand the temperament, engagement, and activity of users. For corporate entities, organizations, and researchers, this wealth of data is an indispensable resource for conducting analytics and deriving actionable insights [3].

This study stands at the intersection of data analysis, natural language processing (NLP), and cloud computing. Specifically, it delves into sentiment and emotion analysis, providing a nuanced understanding of user expressions, beyond mere polarity categorization. Sentiment analysis, which has historically focused on the categorization of text as positive, negative, or neutral, has evolved to encompass the detection of a wider range of emotions and expressions [4]. This paper advances the field by exploring the user's temperament as a combination of emotions, including joy, sadness, anger, fear, guilt, disgust, and shame. Machine learning algorithms, with a focus on the LinearSVC classifier, are employed for precise categorization and analysis.
The research work detailed in this paper uses WhatsApp chat files, which are easily exportable from the application for analysis. It leverages Python libraries and machine learning tools to unlock significant insights and behavioral patterns of users. By conducting behavioral analysis, the research aims to identify and quantify the patterns and dynamics of users within the WhatsApp group.

The primary contributions of this research paper are threefold:

1. Extraction of Insights: The research unearths valuable insights, including active days, active users, and the overall group usage, gleaned from WhatsApp chats.
2. Visualization: The findings are presented in a visually comprehensible format, facilitating a deeper understanding of the data.
3. Behavioral Analysis: The paper hones in on the top 5 active users within the group chat, employing appropriate, graphical representations to dissect their behavioral patterns.

The subsequent sections of this paper delve deeper into the background and related work (Section 2), system architecture (Section 3), methodology (Section 4), discussion and results (Section 5), and conclude by outlining avenues for future research (Section 6).

SYSTEM ARCHITECTURE

The system architecture of the WhatsApp Chat Analysis Project is designed to facilitate the end-to-end process of collecting, analyzing, and deploying WhatsApp chat data. It encompasses a range of components and technologies to ensure the efficient execution of each stage in the project. The system architecture is outlined as follows:

Data Collection:
WhatsApp Chat Export: WhatsApp chat data is collected by exporting chat histories from the WhatsApp application. Users can create an export file containing the chat data they wish to analyze.

Data Preprocessing:
Data Cleaning: The exported chat data often contains various artifacts such as emojis, images, and irrelevant metadata. Data cleaning involves removing these artifacts to ensure that only text-based messages are considered for analysis.

Text Transformation: To prepare the text data for analysis, various transformations are applied. These may include lowercasing, tokenization, and removal of stop words and special characters.

Exploratory Data Analysis (EDA): EDA is performed to gain initial insights into the data. This includes generating statistics, word cloud visualizations, and identifying any data quality issues.

Natural Language Processing (NLP):
Sentiment Analysis: NLP techniques are employed to analyze the sentiment of messages, categorizing them as positive, negative, or neutral. More advanced sentiment analysis methods may be used for nuanced sentiment detection.

Emotion Analysis: In addition to sentiment, the NLP component performs emotion analysis to identify specific emotional tones in the text, including joy, sadness, anger, fear, guilt, disgust, and shame.
Machine Learning and Classification:

LinearSVC Classifier: Machine learning algorithms are utilized for text classification. The LinearSVC (Support Vector Classifier) from the scikit-learn library is employed to classify messages based on their emotional content.

Web Application Development:

User Interface: A web application is developed to provide users with an interactive platform for uploading their WhatsApp chat data and exploring the analysis results.

Deployment on Heroku: The web application is deployed on the Heroku platform, ensuring accessibility and scalability for users.
The system architecture is designed to accommodate the entire process seamlessly, from data collection to analysis and user

interaction. The utilization of NLP and machine learning techniques enhances the quality and depth of the analysis, while the web application and Heroku deployment make the insights accessible and user-friendly.

This system architecture sets the stage for the subsequent sections that delve into the methodology, results, and further discussions, ultimately highlighting the practicality and significance of this end-to-end WhatsApp chat analysis project.

METHODOLOGY

A. Getting the chat data:

1. Open the WhatsApp chat you want to export.
2. Click on the three dots icon on the top right of the screen.
3. Select the "Export chat" option.
4. Chat data will be exported in a text file format.

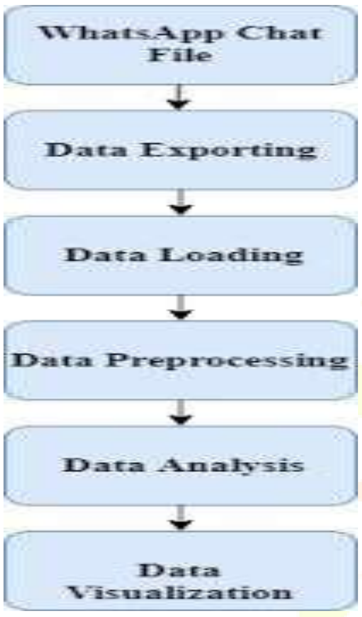

figure 1: flowchart of proposed system

Working Steps to Export chat:

1. Begin by opening the WhatsApp chat analysis web page.
2. Choose your preferred date format.
3. Upload the exported chat file from your WhatsApp group.
4. The data is processed and analyzed by a trained model.
5. Data preprocessing is performed using the trained model.
6. You can select between an overall group analysis or individual analysis.

Prerequisites: Ensure that the following Python libraries are installed on your machine: Streamlit, Matplotlib, Seabor Pandas, Urlextract, Wordcloud, and emoji.

B. Preprocessing:

1. Use Streamlit to create a web application for displaying the chat analysis on a browser.
2. Implement a file upload button using Streamlit to allow users to upload their chat data.
3. Convert the plain text chat data from the text file into a structured format.
4. Convert the plain text data into a Pandas dataframe.
5. Separate the chat messages from the date and time in the text.
6. Further process the data to separate user names and messages.
7. Extract information such as day, date, and time from the date column.

C. System Modules

(a) Install and Import Dependencies:

This module is responsible for installing and importing the necessary Python libraries and packages for the project, including Streamlit, Matplotlib, Pandas, Collections, Seaborn, Emoji, Wordcloud, URLextract, and re. These libraries are essential for data analysis and visualization.

(b) Pre-processing:

This module handles data pre-processing. It formats the raw chat data and separates it into structured components, including date, time, user names, and messages. Data cleaning and transformation occur at this stage.

(c) Export Chat Document from WhatsApp and Upload:

This module guides users on how to export chat data from WhatsApp. It instructs users to export the chat without media, resulting in a document file. Users can then upload this chat document for analysis. It is essentially the user interface for initiating the analysis.

(d) Train Chat Model and Analyze the Data:

This module is the core of the analysis. It involves several sub- modules:

1. Read and Process Data: This step reads the uploaded chat data and processes it. It likely includes tokenization, sentiment analysis, and other natural language processing tasks.
2. Train Machine Learning Model: This is where a machine learning classification model is trained using the processed chat data. The model may be used for tasks such as classifying message types.
3. Model Evaluation: After training the model, it is evaluated to assess its accuracy and performance.
4. Model Serialization: The trained model is serialized, allowing it to be saved and reused for future analyses.
5. Analysis Outputs: The following analyses are conducted:

- Top Statistics: Calculate and display total messages, total words, media shared, and links shared in the chat.
- Monthly Timeline: Generate a timeline showing the frequency of chat messages for each month.
- Daily Timeline: Create a timeline illustrating the frequency of chat messages during each day.

D.Python Libraries and Tools:

1. pandas: Description: Pandas is a powerful library for data manipulation and analysis. It provides data structures (e.g., DataFrames) and functions for data cleaning, transformation, and exploration.

2. NumPy: Description: NumPy is used for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a wide range of mathematical functions.

3. scikit-learn: Description: Scikit-learn is a machine learning library that offers a wide range of machine learning algorithms and tools for tasks such as classification, regression, clustering, and more.

4. NLTK (Natural Language Toolkit):Description: NLTK is a library for natural language processing. It offers tools and resources for text processing and analysis, including sentiment analysis and text classification.

5. spaCy: Description: SpaCy is another popular library for natural language processing. It provides efficient tokenization, part-of-speech tagging, named entity recognition, and more.

6. TextBlob: Description: TextBlob is a simpler library for processing textual data. It offers basic NLP functionalities, including sentiment analysis, noun phrase extraction, and translation.

7. Gensim: Description: Gensim is a library for topic modeling and document similarity analysis. It is commonly used for tasks like latent semantic analysis (LSA) and Latent Dirichlet Allocation (LDA).

8. Matplotlib and Seaborn: Description: Matplotlib and Seaborn are data visualization libraries. They allow you to create various types of charts, plots, and graphs to visualize your analysis results.

9. Plotly: Description: Plotly is a library for creating interactive and dynamic visualizations. It's particularly useful for creating web-based dashboards.

10. Flask: Description: Flask is a web application framework for Python. It can be used to build the user interface for your analysis tool or for creating web-based dashboards to display your results.

11. Heroku CLI: Description: The Heroku Command Line Interface (CLI) is used to interact with the Heroku platform, enabling you to deploy, manage, and scale your application on Heroku.

Heroku Deployment:

The deployment of our analysis tool on the Heroku platform followed a structured process. After creating a Heroku account and selecting an appropriate plan, the Heroku Command Line Interface (CLI) was installed to facilitate interactions with the platform.We carefully configured our project for deployment, ensuring compatibility with Heroku's environment. This involved setting up essential environment variables, verifying dependencies, and making any necessary adjustments.With our project under version control using Git, we created a new Heroku app through the Heroku CLI. This action assigned a unique URL to our application.

To enable deployment, we added the Heroku app as a remote Git repository, allowing us to push our code directly to Heroku. The Heroku platform automatically detected the application type and initiated the build process.

Furthermore, we secured our application by configuring environment variables on Heroku to protect sensitive data, such as API keys and configuration settings.

Architecture Considerations for Hosting on Heroku:

While Heroku abstracts low-level infrastructure management, certain architectural considerations were vital to our project's success:

1. Dynos:

   Heroku employs dynos as lightweight containers for running applications. We carefully selected dyno types based on the specific requirements of our project, distinguishing between web dynos for web applications and worker dynos for background tasks.

2. Add-ons:

   Heroku offers a marketplace of add-ons that extend the functionalities of hosted applications. We evaluated and integrated appropriate add-ons for databases, caching, monitoring, and other key aspects to enhance our application's capabilities.

3. Scaling:

   Heroku simplifies application scaling. Our project allowed for manual and auto-scaling based on changing traffic patterns and resource needs. We meticulously planned the scaling approach to ensure optimal performance.

Scalability and Resource Allocation:

- Scalability and resource allocation played a pivotal role in our project's performance:

1. Manual Scaling:

   We had the flexibility to manually adjust the number of dynos using Heroku CLI or the platform's dashboard. This capability was invaluable for managing workloads that fluctuated over time.

2. Auto-Scaling:

   Heroku's auto-scaling features responded dynamically to predefined metrics, such as request volume, response time, or queue backlog. This ensured that our application could efficiently handle varying workloads without manual intervention.

3. Resource Allocation:

Heroku offered a range of dyno types, each with distinct resource allocations. We carefully selected dyno types in line with our project's resource requirements. When necessary, we considered upgrading to more robust dynos to accommodate resource-intensive tasks.

DISCUSSION AND RESULTS

In this paper, a dashboard was develop to analyze the group chats of the WhatsApp application involves examining conversations within the messaging app to derive insights, patterns, or trends. This could be done for various purposes such as understanding user behavior, sentiment analysis, or even for investigative purposes.

For analyzing the chats data from class group or any business related group is taken from the author's whatsapp application who is the member of that group.

The chat data taken from 10 sept 2023 to 30 oct 2023. Following results are obtained from the analysis.

Total message	Total words	Media shared	link
20795	123568	1403	402

figure 1:shows how many message ,media,links have been sends to this groups by group members.

The line graph for the year 2023 shows that the maximum number of messages were sent in the month of oct and also it can be inferred that the group was the most active during the months oct-sept as seen in Figure 3. Figure 4 shows the bar graph generated by the module to find the top 7 active days of the group.

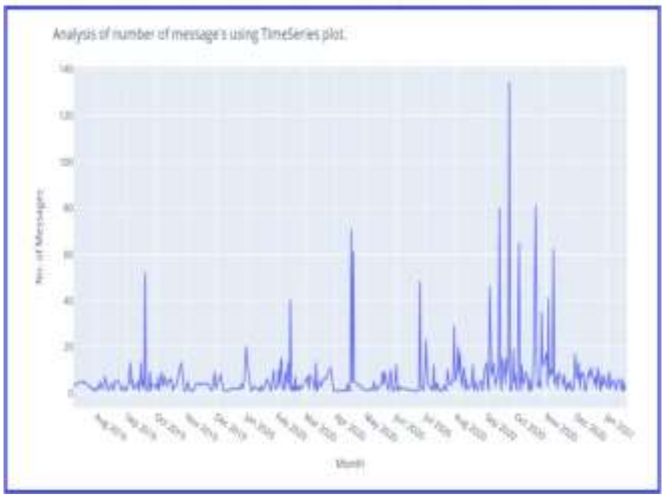

figure 3: a line graph showing the frequency of messages in a particular year

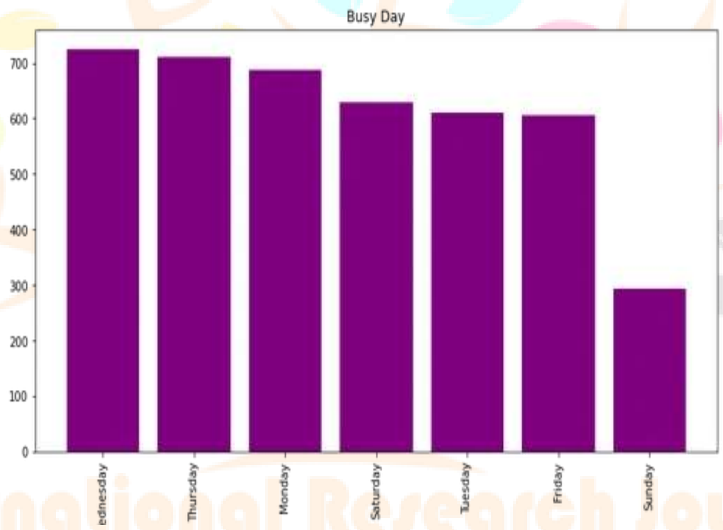

figure 4: a bar graph showing the top 7 active days

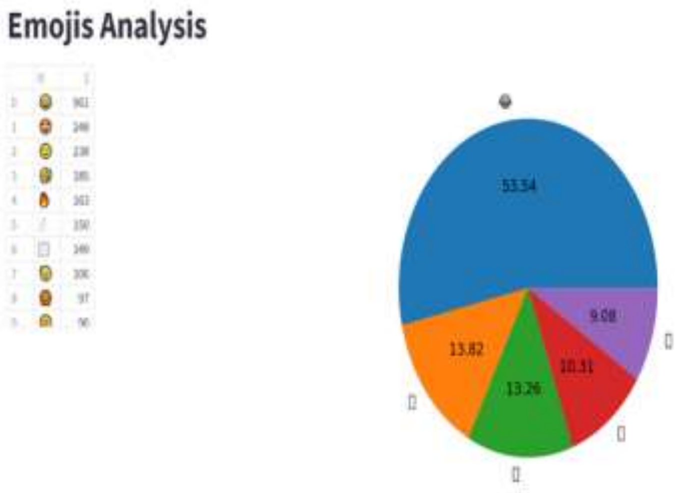

figure 5: emoji distribution

The most frequently used emoji module help to infer the result that the most frequently used emoji in the group are as depicted by a pie chart After working with a few group chats, it was interesting to find that the bigger part of the pie was always taken by the laughing emoji, I guess we can say my group chats are quite happy. Expression is a part of body language to convey your message to another person. While chatting, we use different types of emojis to express different feelings. We will analyze which emoji is used and how many times in a chat

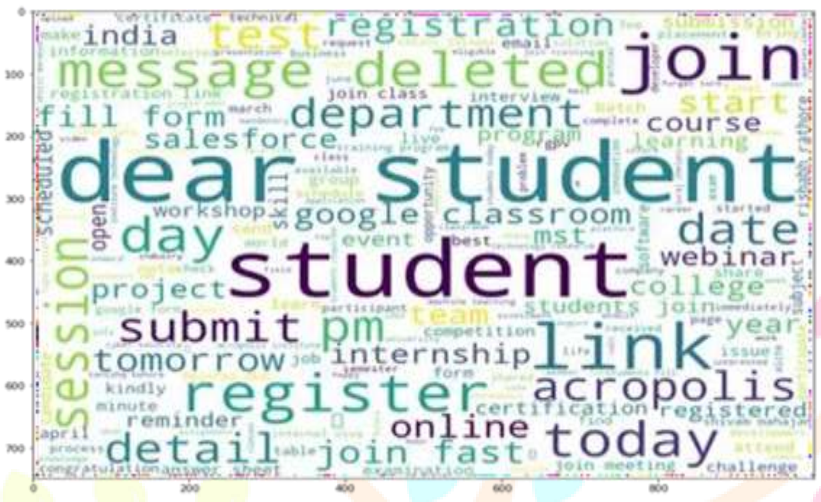

figure 6: top 20 most common words

The question is slightly similar to the upper one, but the context is different, where we have to find the top 20 frequently used words other than stop words. For this, we have to write some custom code where we need a dictionary that stores a word. and its count in total messages. After preparing the dictionary, we can search the top words with the highest frequency. The steps will be similar to the above one, where we must clean the data.

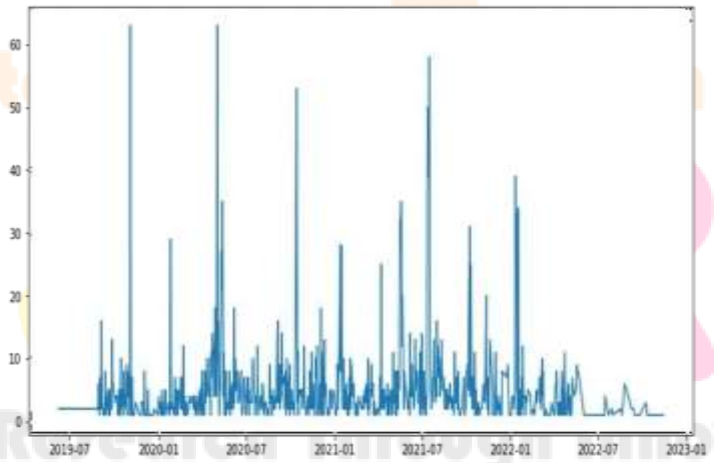

figure 7: daily timeline

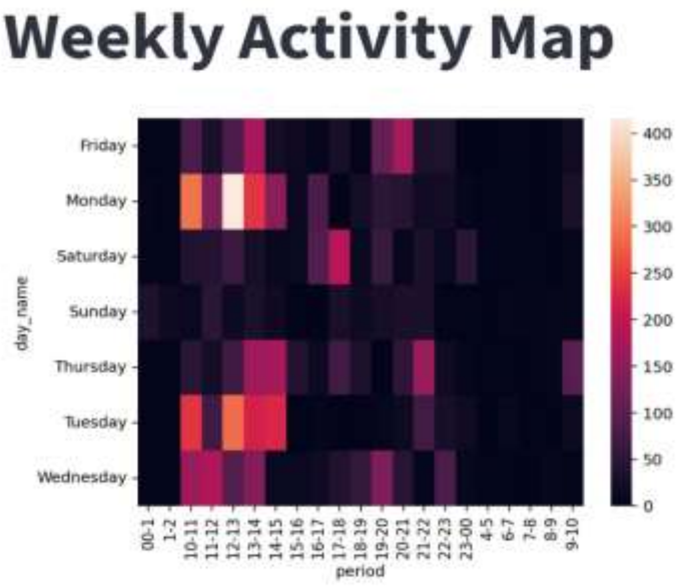

figure 8: weekly activity map

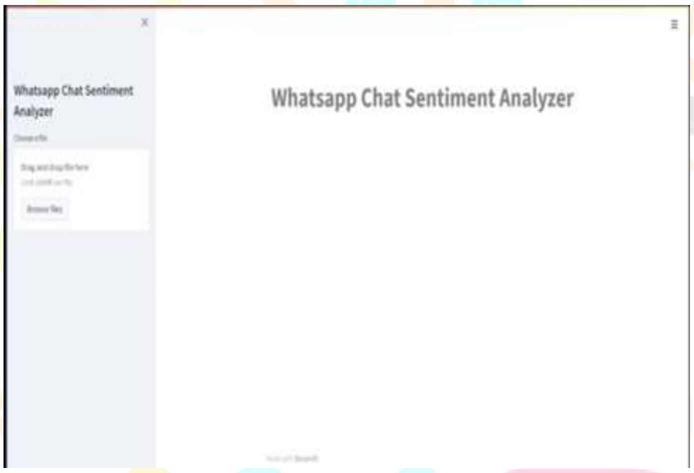
Whatsapp Chat Sentiment Analyzer
Choose a file
Drag and drop file here
Browse files
Whatsapp Chat Sentiment Analyzer

figure 9. screenshot of the homepage

Figure 9 shows the homepage of dashboard that allows the user to upload the WhatsApp chat file in dd/mm/yy 12hr format. for the user to visualize the results.

CONCLUSION AND FUTURE WORK

We have developed and deployed a data analysis project that analyses the WhatsApp chat on a group and individual level. The WhatsApp chat analysis is create to perform chats analysis and study behavioural patterns through WhatsApp group chats. The proposed framework uses data from the WhatsApp application and Python programming language for data analysis. An analysis of the frequency of messages over any period of time, analysis of emojis sent by users, finding the top 7 users and compare their average message length and count of messages sent by them helped us understand their messaging patterns.

We have learned the Data Analysis Life Cycle, data visualizing charts like Bar graphs, and Line charts with their importance in conveying the data. Emojis (body language) Play an important role in the and conversation, and we have learned at a basic level how to analyze emojis with python. In the future, additional features to the analyzer such as finding tha chats data or messages to a particular group, which users send the most relevant and useful messages to the group and who sends unwanted irrelevant messages in the group can be added. Future work may also involve trending analysis where the most trending topic in the group chat is found. The work can also be extended to search when a particular topic was mostly used in group chats or individual chats , how many users were involved in this chats, who were the ghost users when this chat was trending, when was the topic trending, the time period over which the chat was trending and the intensity of the trending chats.

REFRENCES

1. https://www.analyticsvidhya.com
2. https://backlinko.com/whatsapp-users
3. https://whatsanalyze.com
4. https://medium.com/@barklight/cracking-the-conversation973839be5b88
5. https://cran.rproject.org/web/packages/rwhatsapp/vignettes/Text_Analysis_using_WhatsApp_data.ml
6.
7. S. Dahiya, A. Mohta and A. Jain, ”Text Classification based Behavioural Analysis of WhatsApp Chats,” 2020 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2020,pp. 717-724, doi: 10.1109/ICCES48766.2020.9137911.
8. M. Pallavi, M. Nirmala, M. Sravani, M. Shameem, and Dr. K. Soumya, “WHATSAPP CHAT ANALYSIS,”International Research Journal of Modernization in Engineering Technology and Science , vol. 04, no. 05,pp. 32–38, May 2022
9. B. S. R. Chinthapanti, K. Sola, R. Kumar, N. K. Reddy, and Gopichand, “Analysing and predicting the emotion of WhatsApp chats using sentiment analysis,” TEST Engineering & Management, vol 83, pp. 15454 - 15461, April 2020, doi:10.13140/RG.2.2.36456.83209.
10. https://raw.githubusercontent.com/amankharwal/Website data/master/text.txt
11. https://medium.com/mcd-unison/whatsapp-group- chat-analysis-python-5b1c0a4a84bb