WhatsApp Chat Analyzer: A Streamlit-Based NLP Approach

Rohan B, Gunashree R and Rohith S

Abstract—The widespread use of WhatsApp as a primary communication platform generates vast volumes of textual data daily. This data, when analyzed effectively, can reveal key insights into communication trends, sentiment, and user behavior. The WhatsApp Text Analyzer is a data science project aimed at transforming raw WhatsApp chat exports into structured and interpretable formats using Python and Streamlit. It provides comprehensive statistical analysis, visualizations, and a sentiment classification model to evaluate chat data. By offering user-centric insights such as activity patterns, most common words, and emoji usage, this tool serves as a valuable application in digital behavior analysis. This paper presents the methodology, implementation, and results of the WhatsApp Text Analyzer and suggests directions for further enhancement, including sentiment accuracy improvements and cloud-based deployment.

1. Introduction

In the digital era, instant messaging applications have become an integral part of human communication. WhatsApp, one of the most widely used messaging platforms, generates a massive volume of textual data every day. These chats contain valuable insights that can be analyzed for various purposes, including sentiment analysis, user behavior analytics, content evaluation, and trend prediction. The WhatsApp Text Analyzer project aims to offer a robust framework for analyzing exported WhatsApp chat data, transforming raw text into meaningful visualizations and statistics.

2. Project Objectives and Motivation

The primary objective of this project is to develop an interactive tool capable of analyzing WhatsApp chat data. The motivations behind this project include:

- **Understanding Communication Patterns**: Identify who is the most active, when users are most active, and how conversation trends evolve over time.
- **Performing Sentiment Classification**: Using basic machine learning to label chats as positive, negative, or neutral.
- **Visual Analysis**: Generate visualizations such as word clouds, emoji analysis, heatmaps, and timelines for enhanced interpretation.
- **User-Centric Insights**: Analyze statistics both overall and for individual users.

3. Requirement Specification

3.1. Software Requirements

1. Python 3.8 or later
2. Streamlit for Web App Development
3. Required Python packages listed in requirements.txt

3.2. Hardware Requirements

1. A computer capable of running Python scripts
2. Minimum 4GB RAM (8GB recommended)
3. Stable internet connection (for Streamlit UI deployment, optional)

3.3. Functional Requirements

1. Ability to upload WhatsApp chat files.
2. Parse and preprocess the chat text.
3. Perform statistical analysis and text classification.
4. Display interactive charts and visualizations.

3.4. Non-Functional Requirements

1. The application should be user-friendly.
2. Should produce results within a reasonable time (seconds for typical chats)
3. Portable and easy to set up using requirements.txt.

4. Dataset Details and Preprocessing Steps

The dataset used in this project is an exported WhatsApp chat file (in .txt format). The chat text contains timestamps, usernames, and messages.

4.1. Preprocessing

1. **Date Time Extraction:** Regex pattern matches date and time formats.
2. **User Message Segregation:** Messages are split into user and content using regex.
3. **Feature Engineering:** Additional features like day, month, year, hour, minute, and period (hourly intervals) are extracted.
4. **Data Cleaning:** Removal of empty messages and standardization of columns.

This transforms raw chat logs into a structured *pandas.DataFrame* for further analysis.

5. Methodology

5.1. Preprocessing and Data Cleaning (Source: *preprocessor.py*)

This step is essential to convert the unstructured chat data into a structured format. The exported WhatsApp chat file is first read as plain text. Using regular expressions, date-time stamps and sender information are separated from message content. The preprocessor script breaks down messages into individual records, filtering out system-generated texts and irrelevant content. Features such as hour, day, month, and the messaging period (hourly) are engineered from the timestamp for further filtering and plotting.

5.2. Statistical Analysis (Source: *helper.py*)

After cleaning, statistical evaluations are performed on the structured data. Metrics such as the number of messages, total words, shared media count, and hyperlink frequency are calculated. User activity is analyzed to determine the most active users. Libraries such as matplotlib and seaborn are used to create informative visuals, including bar charts, heatmaps, and line graphs. WordCloud is used to create a visual cluster of the most frequent terms, enhancing topic visibility.

5.3. Sentiment Classification (Source: *helper.py*)

A basic sentiment classification module is developed to showcase machine learning capabilities. CountVectorizer is used to convert textual data into numerical format. The Multinomial Naive Bayes algorithm is employed to classify messages into positive, negative, or neutral categories. Since the dataset is unlabeled, random labels are used for demonstration, which limits the model's accuracy. However, this placeholder can be substituted with a real labeled dataset and a more advanced model like BERT or VADER.

5.4. User Interface and Visualization (Source: *app.py*)

The application is built using Streamlit to offer a simple and interactive web-based interface. Users can upload their exported chat files directly into the browser. The app provides dynamic charts and data summaries, including timeline plots, emoji analysis, top words used, and sentiment distribution. The sidebar allows users to filter results by specific users, making the tool flexible and personalized. The

integration of backend analytics with frontend presentation offers a complete user experience.

6. Results and Analysis

.

6.1. Sample Outputs

1. **Total Messages:** Displays total count of messages exchanged.
2. **Top Words:** Bar chart of most frequently used words (excluding stopwords).
3. **Word Cloud:** A visual cluster of words to identify key discussion topics.
4. **User Activity:** Bar charts showing most active users.
5. **Timelines:** Line graphs for monthly and daily message volumes.
6. **Heatmap:** Time-based heatmap showing peak activity periods.
7. **Emoji Usage:** Data table and pie chart displaying popular emojis.

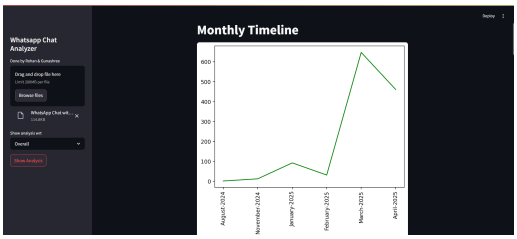

Figure 1. Sample Output of WhatsApp chat plot.

6.2. Activity Map

It shows the busy days and months. We have used the matplotlib library to plot the graph, the number of messages in a particular month or day are mapped to the particular day or month.

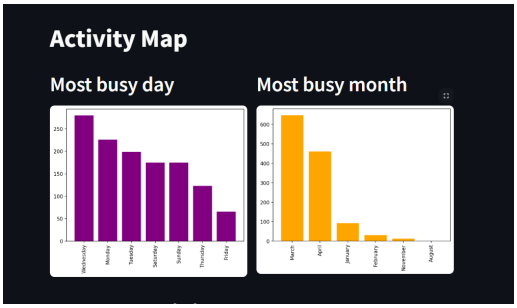

Figure 2. Activity Map

6.3. Daily Timeline

It gives the frequency of messages in a day we have used matplotlib to plot the graph and the days are taken and the count of messages are calculated and plotted

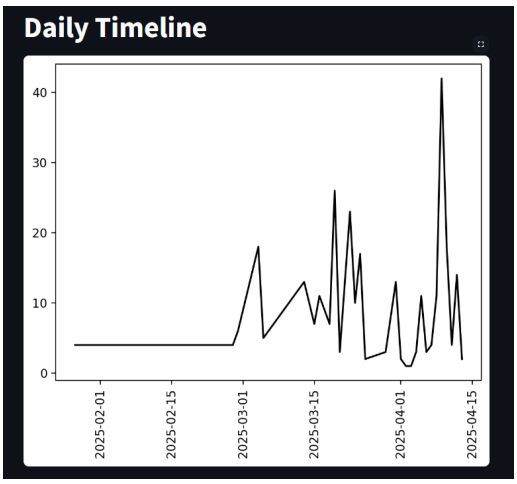

Figure 3. Daily Timeline

6.4. Top Statistics

It shows the statistics like total messages, words, and images links shared. We have converted the whole chat file into a data frame and then separated the words and messages and used URL extract to find links.

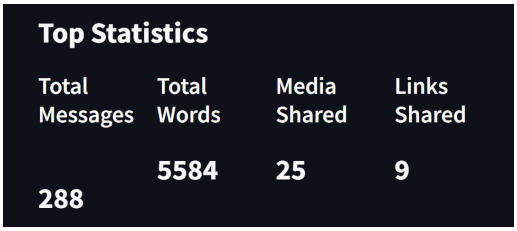

Figure 4. Top Statistics

6.5. Most Common words

It shows the most commonly used word we have used matplotlib to plot the graph and the top frequently used words are displayed. Is shows the busy users and their contribution to chat we have used matplotlib to plot the graph and the users and how frequently the chat is calculated and plotted

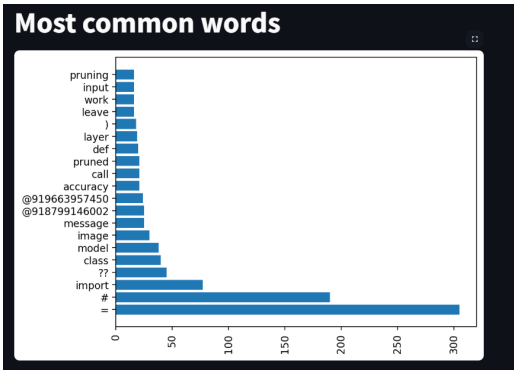

Figure 5. Most Common words

6.6. Most Busy Users

The graph displays the most active users in a WhatsApp, showing who sent the most messages.It visually represents user engagement based on message count and percentage share.

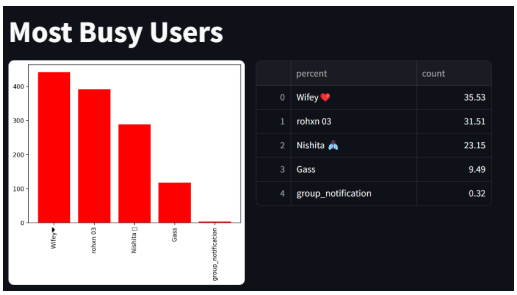

Figure 6. Most Busy Users

6.7. Emoji Analysis

It shows the most commonly used emoji's. We have used the Emoji library to select or distinguish the emoji'sfrom the messages and plotted the pie chart using matplotlib

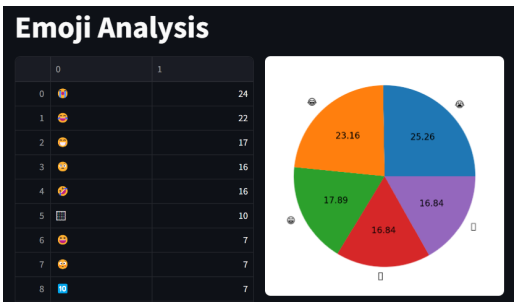

Figure 7. Emoji Analysis

7. Conclusion and Future Work

In the digital era, instant messaging applications have become an integral part of human communication. WhatsApp, one of the most widely used messaging platforms, generates a massive volume of textual data every day. These chats contain valuable insights that can be analyzed for various purposes, including sentiment analysis, user behavior analytics, content evaluation, and trend prediction. The WhatsApp Text Analyzer project aims to offer a robust framework for analyzing exported WhatsApp chat data, transforming raw text into meaningful visualizations and statistics.

Future Enhancements:

- Integrate real sentiment analysis using models like BERT or VADER.
- Multi-language support.
- User behavior tracking over longer periods..
- Export insights as downloadable reports.
- Deploy on cloud with chat file storage and dashboard access.

8. References

1. https://www.statista.com/number-of-monthly-active-whatsapp-users
2. Ravishankara K, Dhanush, Vaisakh, Srajan I S. "International Journal of Engineering Research & Technology (IJERT)," ISSN: 2278-0181, Vol. 9 Issue 05, May 2020.
3. Dr. D. Lakshminarayanan, S. Prabhakaran. "DogoRangsang Research Journal," UGC Care Group I Journal, Vol-10 Issue-07 No. 12, July 2020.
4. F. Meng Cai. "PubMed Central," PMCID: PMC7944036, PMID: 33732917.
5. Abdullah, A., Brobst, S., Pervaiz, I., Umer, M., and A. Nisar. "Learning dynamics of pesticide abuse through data mining." Proceedings of Australian Workshop on Data Mining and Web Intelligence, New Zealand, January 2004.
6. https://www.interaction-design.org/literature/topics/web-design
7. Radhika, Narendiran. "Kind of Crops and Small Plants Prediction using IoT with Machine Learning," International Journal of Computer & Mathematical Sciences, April 2018, pp. 93–97.
8. http://www.statista.com/statistics/260819/numberof-monthly-activeWhatsApp-users. Number of monthly active WhatsApp users worldwide from April 2013 to February 2016 (in millions). *Journal of Innovative Research in Computer and Communication Engineering*, January 2017, pp. 318–323.
9. Ahmed, I., Fiaz, T., Aijaz, K. "Mobile phone to youngsters: Necessity or addiction," *African Journal of Business Management*, Vol. 5 (32), pp. 12512–12519, 2011.
10. Mike Dickson. "An examination into yahoo messenger 7.0 contact identification," *Digital Investigation*, ScienceDirect, Vol. 3, Issue 3, pp. 159–165, 2006.
11. Akash Raj N, Balaji Srinivasan, Deepit Abhishek D, Sarath Jeyavanth J, Vinith Kannan A. "IoT based Agro Automation System using Machine Learning Algorithms," *International Journal of Innovative Research in Science, Engineering and Technology*, November 2016, pp. 19938–19342.
12. D. Ramesh, B. Vishnu Vardhan. "Analysis of Crop Yield Prediction Using Data Mining Techniques," *IJRET: International Journal of Research in Engineering and Technology (IJERT)*, 2015.