Project Report: Whatsapp Lens

Yahya Jabary, 11912007

Code: `https://github.com/sueszli/whatsapp-lens/`

Contents

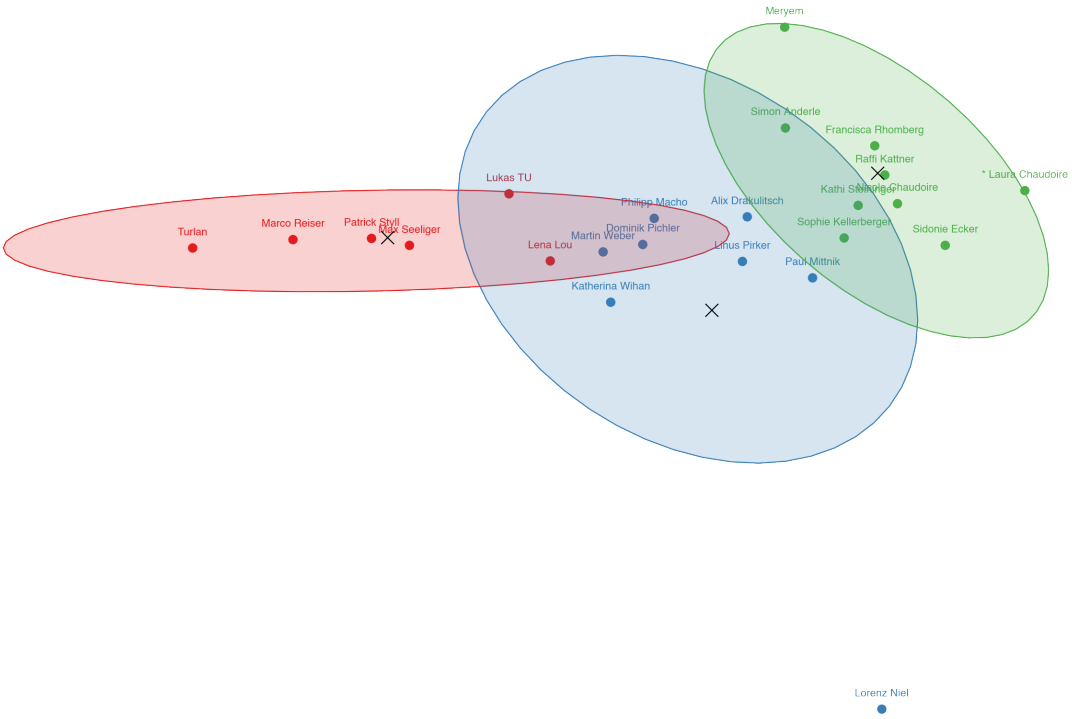


Figure 1: Latent Representation Clustering of WhatsApp Conversations via PCA

Proposal

Assignment 1 - Initiate

In this chapter we discuss the state-of-the-art in WhatsApp analytics, the motivation behind the project, the contribution of the project and a detailed plan of execution with a time estimate for each task.

Motivation

With over 2 billion users WhatsApp has reached the highest user penetration in the European Union[1] [2] [3] and has become the primary means of Internet communication in regions including the Americas, the Indian subcontinent and large parts of Europe and Africa[4].

This ubuiquity has made WhatsApp a rich source of data for social and behavioral analysis. For instance, Kaushal and Chadha[5] have surveyed various sentiment analysis techniques of WhatsApp, while Ngaleka, Uys[6] have studied m-learning and recently, as of 2023 Jannah [7] has explored the utilization of "WhatsApp Business", a Facebook-Business driven platform for customer service and marketing purposes.

However, these studies are mostly of academic interest and lack both practicality and accessibility for the average user in order to gain data driven insights from their own WhatsApp conversations. This is particularly relevant in the context of WhatsApp Business which is increasingly used for customer service and marketing purposes and where customer interactions are a valuable source of feedback and could be quantified and analyzed to improve business processes.

Analyzing WhatsApp conversations can provide valuable insights into:

- Sentiment patterns: The emotional tone of conversations by using sentiment classifiers.
- Temporal patterns: Message frequency, response times, interaction patterns and engagement metrics between participants as an indicator of relationship strength.
- Topic modeling: Extracting underlying themes in conversations to understand how discussion topics evolve over time using techniques such as Latent Dirichlet Allocation (LDA) or more modern models capable of capturing context for Q&A tasks.

Especially given recent advancements in natural language processing with the advent of transformer models, downstream tasks could also use latent representations of the chat data to train models to both classify and cluster relationships based on conversational patterns, as well as predict future interactions with high accuracy.

Contribution

One of the main challenges in applying the mentioned state-of-the-art NLP techniques to this domain however is the lack of both open datasets and accessible parsing tools for WhatsApp chat exports, given that the data comes in an unstructured and proprietary format, increasing the barrier to entry for researchers and practitioners alike.

The project category "Bring your own data" in this course lend themselves itself nicely to bridging this gap by providing a ground for the development of an end-to-end prototype for WhatsApp chat data analysis:

- (1) Data Generator: implementing synthetic data generation tools, a robustness testing dataset and a personal dataset for demonstrative purposes.
  - (1.1) Synthetic dataset: a synthetic conversational data generation tool, based on role-playing large language model agents.
  - (1.2) Robustness dataset: a custom dataset, demonstrating all edge cases and possible media to be handled by the tool.
  - (1.3) Personal dataset: a private dataset consisting of the author's own WhatsApp chat data for demonstrative purposes, not to be shared in the open-source repository.
- (2) Parser: parsing the proprietary format into a standardized CSV format using regular expressions.
- (3) Feature Engineering: validating, preprocessing and extracting key features from the data.
  - (3.1) Validation: ensuring the data is correctly parsed and standardized, including message timestamp normalization, emoji encoding, media message handling, language detection and processing.

[1] Shan, S. The battle between social giants: WeChat and WhatsApp's influence in digital marketing.

[2] WhatsApp-Website. http://blog.whatsapp.com/. Accessed 30 Oct 2024.

[3] Montag, C., Błaszkiewicz, K., Sariyska, R., Lachmann, B. Andone, I., Trendafilov, B. & Markowetz, A. (2015). Smartphone usage in the 21st century: who is active on WhatsApp?. BMC research notes, 8, 1-6.

[4] Metz, Cade (April 5, 2016). "Forget Apple vs. the FBI: WhatsApp Just Switched on Encryption for a Billion People". Wired. ISSN 1059-1028. Archived from the original on April 9, 2017. Retrieved May 13, 2016.

[5] Kaushal, R., & Chadha, R. (2023, March). A Survey of Various Sentiment Analysis Techniques of Whatsapp. In 2023 2nd International Conference for Innovation in Technology (INOCON) (pp. 1-6). IEEE.

[6] Ngaleka, A., & Uys, W. (2013, June). M-learning with whatsapp: A conversation analysis. In International Conference on e-Learning (p. 282). Academic Conferences International Limited.

[7] Jannah, R. (2023). Utilization of whatsapp business in marketing strategy to increase the number of sales through direct interaction with customers. Syntax Idea, 5(4), 488-495.

  - (3.2) Feature Extraction: extracting key features such as message frequency metrics, response time patterns, sentiment scores, topic distributions and user interaction patterns.

- (4) Analytics: performing exploratory data analysis and applying state-of-the-art NLP techniques to the data.
  - (4.1) Sentiment Analysis: training a sentiment classifier on the data to predict the emotional tone of conversations.
  - (4.2) Topic Modeling: applying LDA or more modern models to extract underlying themes in the data.
  - (4.3) Relationship Clustering: clustering relationships based on conversational patterns using latent representations of the chat data.
  - (4.4) Frequency Analysis: analyzing message frequency, response times, interaction patterns and engagement metrics between participants.
  - (4.5) Demographic Analysis: predicting user demographics based on conversational patterns.
- (5) Deployment: deploying the analytics results in a user-friendly and interactive web application.
- (6) Presentation: preparing a final report and a video presentation on YouTube.

Given our time budget of just 3 ECTS credits, equivalent to 75-81 hours of work (25-27 hours per ECTS credit), excluding the time spent on optional lectures and exercises, we constrain each task to a maximum of 15 working hours, ensuring that the project is completed within a reasonable time frame.

We plan to dedicate the largest portion of our time to the Feature Engineering and Analytics tasks, as these are the most critical components of the project, which will provide the most value to the end user in terms of insights gained from the data for our prototype.

Execution

Assignment 2 - Hacking

Synthetic Data Generation

The synthetic data generation component utilizes two instances of the TinyLlama-1.1B-Chat model to simulate a natural conversation between two personas with distinct characteristics. This approach allows for the creation of realistic WhatsApp chat data while maintaining privacy and providing a controlled environment for testing and development. The quantized 1.1B model was chosen for its capacity to run on a consumer-grade GPU such M2 Pro Metal Performance Shaders with 16GB of memory.

```
2024-10-31 03:08:00,Jane Doe,Hey Jane! How's your day going?
2024-10-31 03:10:00,Jane Doe,I had such an amazing weekend at the gym with my friends - it was great having some friendly competition while enjoying all
2024-10-31 03:13:00,John Smith,So excited for Sunday morning workout this AM (and hopefully a good night's sleep too).
2024-10-31 03:10:00,Jane Doe,How about you? Workout? Healthier lifestyle tips/strategies? Food preferences/diets suggestions? Can I tag your friends in t
```

The implementation leverages the Hugging Face Transformers library to load and utilize the language models. Each model instance is configured with specific generation parameters to ensure diverse yet coherent responses. The generation process employs a temperature of 0.9 and top-p sampling of 0.9, striking a balance between creativity and coherence in the generated text.

To maintain conversation authenticity, the system implements realistic temporal patterns through the `get_next_timestamp` function. This function introduces variable time delays between messages, with most intervals falling between 1 second and 10 minutes. Additionally, it incorporates a 1% chance of longer gaps ranging from 1 hour to 1 day, simulating natural conversation breaks.

The personas are carefully crafted to represent distinct personality types, using 2 prompts for this specific example:

- Jane Doe represents a book-loving introvert, programmed to reference literature and use book-related expressions
- John Smith embodies a fitness enthusiast, incorporating workout-related terminology and health-focused language

The conversation generation process is implemented as an iterative loop, where each model takes turns responding to the previous message. The responses are processed to ensure proper formatting and stored in WhatsApp's characteristic timestamp format (`MM/DD/YY, HH:MM`).

The implementation also includes safeguards against common issues in language model outputs, such as response truncation using regular expressions to ensure complete sentences and the removal of special tokens. A repetition penalty of 1.3 helps prevent the models from falling into repetitive patterns or loops, contributing to more natural-sounding conversations.

This approach however is not without limitations, as it occasionally produces nonsensical or off-topic responses. These issues are mitigated through manual filtering and post-processing, ensuring the generated data remains coherent and relevant. A more robust solution could involve fine-tuning the models on WhatsApp chat data to improve the quality of the generated conversations and using larger models to capture more nuanced conversational patterns.

Robustness Dataset

In addition to the synthetic data generation tool and the personal dataset, a robustness dataset was created to test the parser's ability to handle edge cases and various media types. The dataset includes a wide range of scenarios such as: phone numbers, URLs, emojis, media messages, special characters, calls, location sharing, disappearing messages and deleted messages. All of these scenarios are designed to test the parser's ability to correctly extract and encode the data into a standardized format and were handled appropriately in a post-processing step in which they were replaced with placeholders.

Parsing

The parser has been implemented as a Python function that processes WhatsApp chat export plaintext files. The script utilizes regular expressions to extract relevant information from the chat logs and converts them into a structured CSV format.

The function reads the input file line by line, using a regular expression pattern to match the timestamp, author and message content. This pattern is designed to handle the standard WhatsApp chat export format, which typically includes a timestamp, followed by the author's name and then the message content.

The expression used is: `r"(\d{1,2}/\d{1,2}/\d{2,4},\s\d{1,2}:\d{2})\s-\s(?:([^:]+):\s)?(.+)"`.

One of the challenges addressed in the parser is handling multi-line messages. The script maintains a `current_message` dictionary to accumulate message content across multiple lines. This approach ensures that messages spanning multiple lines are correctly captured and preserved in the output.

The parser also handles server messages, which are messages generated by the WhatsApp system rather than by users. These messages are identified by the absence of an author name and are assigned the author "server" in the output CSV.

To ensure data integrity and consistency, the script includes a `validate_csv` function. This function performs several checks on the output CSV file, including verifying the correct number of columns and ensuring that timestamps are in ascending order. These validation steps are crucial for maintaining data quality and preventing errors in subsequent analysis stages.

The main execution block of the script processes all text files in a specified directory, converting each one to a CSV file. This batch processing capability allows for efficient handling of multiple chat export files, which is particularly useful for analyzing conversations across different groups or time periods.

Feature Engineering

The feature extraction module implements a comprehensive set of analyses on WhatsApp chat data through several key components.

The preprocessing function handles the initial data cleaning by converting timestamps to datetime format, categorizing authors, and standardizing media message placeholders. It also removes server messages and poll content to ensure clean data for analysis.

The language detection functionality samples 100 random messages and uses majority voting to determine whether the conversation is in English or German. We only permit the analysis of English and German conversations in our prototype, as these are the languages most commonly used in the author's chat data.

For demographic analysis, the code implements gender classification using a pre-trained transformer model to identify the gender of chat participants based on their names. This model is fine-tuned version of DistilBERT, trained on a large corpus of names and achieves a test accuracy of 100% on a balanced dataset.

The sentiment and toxicity analysis components process messages on a monthly basis, using state-of-the-art transformer models. The sentiment analyzer generates scores indicating the emotional tone of messages, while the toxicity classifier identifies potentially harmful content. Both analyses use sampling to handle large datasets efficiently.

Topic diversity is measured using BERTopic, which combines BERT embeddings with traditional clustering techniques. The implementation uses a multilingual sentence transformer model to generate embeddings and includes parameters for minimum topic size and document frequency thresholds.

The frequency statistics component provides detailed interaction metrics including message ratios, word counts, media usage, emoji frequency, and URL sharing patterns. It also analyzes conversation patterns by measuring response times, conversation initiations, and active hours for both participants. The code defines conversations using a two-hour threshold between messages. This segment is the most sophisticated part of the feature engineering module, as it requires extensive data processing and analysis to extract meaningful insights from the chat data.

Message embeddings are generated using a multilingual MiniLM model, creating vector representations of the entire conversation that can be used for further analysis or visualization.

The implementation makes effective use of modern NLP libraries and handles multilingual content appropriately, while maintaining efficiency through sampling and proper resource management. The code includes appropriate error handling and parameter validation to ensure robust processing of various chat formats and content types.

Here's an explanation of each feature in the result schema:

- `conversation_language`: The primary language detected in the chat (either 'en' or 'de')
- `author_name`: The name of the main chat participant being analyzed
- `partner_name`: The name of the other chat participant
- `author_monthly_sentiments`: Monthly sentiment scores of the author's messages (0-1 scale)
- `partner_monthly_sentiments`: Monthly sentiment scores of the partner's messages (0-1 scale)
- `author_monthly_toxicity`: Monthly toxicity levels in the author's messages (0-1 scale)
- `partner_monthly_toxicity`: Monthly toxicity levels in the partner's messages (0-1 scale)
- `author_gender`: Predicted gender of the author ('m' or 'f')
- `partner_gender`: Predicted gender of the partner ('m' or 'f')
- `topic_diversity`: Score indicating the variety of conversation topics (0-1 scale)
- `total_messages`: Total number of messages in the conversation
- `author_message_ratio`: Proportion of messages sent by the author
- `partner_message_ratio`: Proportion of messages sent by the partner
- `author_avg_word_count`: Average number of words per message from the author
- `partner_avg_word_count`: Average number of words per message from the partner
- `author_media_count`: Number of media messages sent by the author
- `partner_media_count`: Number of media messages sent by the partner
- `author_emoji_count`: Number of emojis used by the author
- `partner_emoji_count`: Number of emojis used by the partner
- `author_url_count`: Number of URLs shared by the author
- `partner_url_count`: Number of URLs shared by the partner
- `author_vocabulary_size`: Number of unique words used by the author
- `partner_vocabulary_size`: Number of unique words used by the partner
- `total_conversations`: Number of distinct chat sessions (separated by 2+ hours of inactivity)
- `total_duration_days`: Total timespan of the conversation in days
- `author_message_freq_s`: Average time between author's messages in seconds
- `partner_message_freq_s`: Average time between partner's messages in seconds
- `author_response_time_s`: Average time taken by author to respond in seconds
- `partner_response_time_s`: Average time taken by partner to respond in seconds
- `author_avg_active_time`: Average hour of the day when author is most active (0-23)
- `partner_avg_active`: Average hour of the day when partner is most active (0-23)
- `author_conversation_initiations`: Number of conversations started by the author
- `partner_conversation_initiations`: Number of conversations started by the partner
- `embeddings`: Vector representation of the entire conversation for semantic analysis

Data Analysis

For data analysis we utilized various data science libraries including pandas for data manipulation, plotnine (a Python implementation of ggplot2) for visualization, and several machine learning tools for advanced analysis.

Our analysis began with loading and preprocessing the conversation data from a CSV file, which contained various metrics such as message counts, response times, and conversation durations. We focused on several key aspects of the conversations, excluding certain columns like author names and toxicity metrics to concentrate on the most relevant features.

For the visualization component, we implemented multiple analytical perspectives. We created scatter plots to examine relationships between total messages and conversation duration, incorporating topic diversity and conversation language as additional dimensions. We also analyzed message ratio distributions across different partner genders using box plots, and investigated response time patterns through various visualizations including violin plots.

In the more advanced analysis section, we implemented dimensionality reduction techniques using both PCA and t-SNE to visualize conversation patterns in a reduced dimensional space. We applied clustering algorithms (KMeans and DBSCAN) to identify groups of similar conversation partners based on their communication patterns. These clusters were visualized using different approaches, including elliptical boundaries and convex hulls to highlight group distinctions.

We also developed interactive visualizations using Altair, creating a dashboard that allows for dynamic exploration of the data through linked views and brush selection. This implementation enabled users to investigate relationships between various metrics such as message ratios, response times, and conversation initiations while maintaining the context of partner gender and sentiment analysis.

Throughout our analysis, we maintained a focus on scientific rigor while implementing various statistical and machine learning techniques to uncover patterns in WhatsApp communication behavior. The combination of traditional statistical visualization with modern machine learning approaches provided a comprehensive understanding of the conversation dynamics in our dataset.

Reproducibility

The project is designed to be reproducible and extensible, with a clear separation of concerns between the data generation, parsing and feature engineering components. The codebase is well-documented and includes detailed instructions for running each module. For increased usability, all weights and datasets are stored in a centralized location, accessible via portable `Path` objects. Additionally a Dockerfile is provided to ensure consistent execution environments across different systems in combination to 2 seperate `pip-compile`'d and `venv` dumped requirements files for any environment. A universal makefile will also ensure the conversion of the `venv` environment into a conda environment for any kind of HPC environment requiring a smaller memory footprint.

Results

Assignment 3 - Deliver

The final prototype was evaluated using the author's personal WhatsApp chat data, which consisted of over 10,000 messages exchanged with 23 contacts, both in English and German. The data spanned a period of 3 years and included a diverse range of conversation topics, from casual social interactions to professional discussions, making it an ideal test case.

Deployed Web Application

A small web application with interactive visualizations using the Vega specification language was deployed on Github Pages to showcase results in a more engaging manner and encourage playfulness with the data.

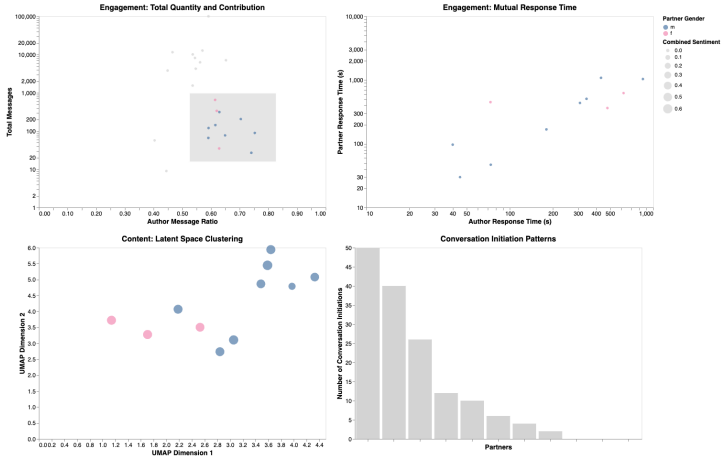


Figure 2: Web Application Screenshot

The application provides 4 linked viewes. The first view is a scatter plot of total messages vs. the author's ratio among the total messages, which can be brushed to highlight specific conversations. This view allows us to filter individual relationships based on the more active participant in the conversation as well as their significance in the author's life. The normal appearance of the scatter plot with a skew towards larger ratios indicates a more active role of the author in most conversations.

The second view is another scatter plot, this time of both the response time of both conversation partners. We can observe a strong linearity, which means that on average, the author seems to mirror the response times of their conversation partners. We couldn't find any psychological explanation for this behavior, but it's an interesting observation nonetheless, worth further investigation in a large scale study.

The third view is a scatter plot of the UMAP clustering of conversation embeddings. Closer points indicate more similar conversations. Additionally, larger points indicate a better combined sentiment score of the conversation. We can observe that the author has mostly neutral to rather positive conversations, which is a good sign for their social life.

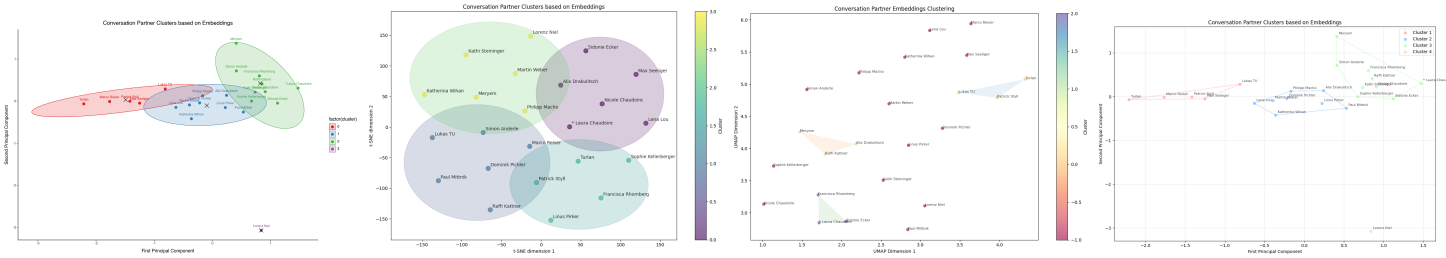
Finally in the last bar plot we can see the number of conversation initiations by the opposite conversation partner. This plot seems to be rather random, with no clear pattern emerging or narrative to be told. However it can serve as an additional metric to gauge the intimacy or importance of a relationship in the author's life.

Data Analysis Results

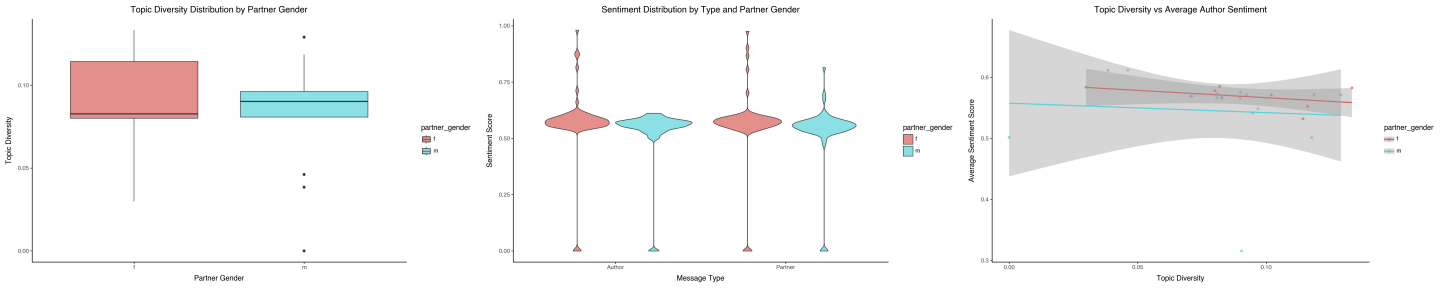
Around 20 plots were generated using the Plotnine library, to gain a deeper understanding of the conversation dynamics in the dataset, before moving on to a final set of 4 interactive visualizations using Altair.

A variety of conversation latent clustering algorithms (t-SNE, PCA, UMAP) were applied to the conversation embeddings, which were generated using a MiniLM model. The results were visualized using scatter plots, with different colors representing different clusters, some created using KMeans and others using DBSCAN. The clusters were then visualized using convex hulls and elliptical boundaries to highlight group distinctions.

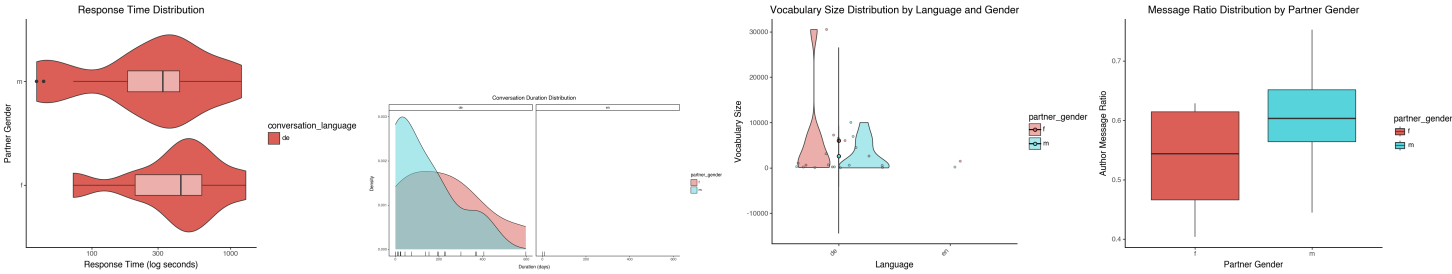
Based on the author's personal knowledge of the contacts, the clusters were found to be meaningful and representative of the different types of relationships in the dataset, namely: professional relationships, male friends, female friends and "Lorenz". It's unclear why conversations with "Lorenz" were clustered separately, but one hypothesis is that they largely consist of the exchange of large fitness related metrics and large messages only consisting of numeric dataframes, as they frequently exchange workout plans and progress reports. Additionally, the fact that male and female friends were clustered separately could indicate that the author has different communication patterns with each group, which is a common phenomenon in social interactions - although surprisingly pronounced in this case.



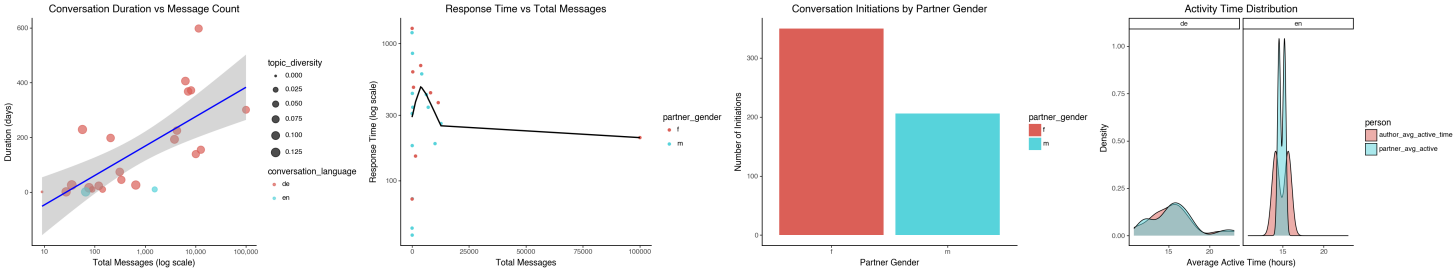
Other visualizations included (1) a boxplots of topic diversity by the partner's gender, which showed a lower median but a much higher third quartile for conversations with women, indicating more diverse topics in these conversations, (2) a violin plot of response times by the the sentiment distribution by gender both from messages by the author and the partner, which showed a slightly positive skew as well as strong outliers in some female partners (most likely the author's girlfriend), (3) a scatter plot of topic diversity and sentiment, fitted with a linear regression line, which didn't show any significant correlation.



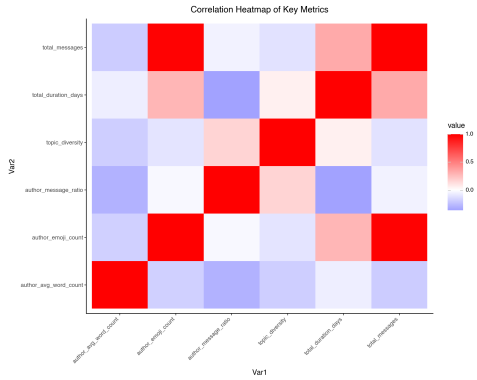
Based on four more observations we were able to conclude that (1) the author responds ~100ms earlier to messages by male aqquaintances, (2) the conversations with male aquaintances are more short-lived and that (3) male partners have a higher vocabulary size on average with a few very significant vemale outliers, as observed in the violin plot of vocabulary size distribution by gender. (4) The author also seems to be more vocal and domninant in conversations with male partners, often surpassing the 60% message ratio threshold, which again indicates a more active role in conversations with male partners.



We also observed (1) a noticible but not significant correlation between the number of messages exchanged and the duration of the conversation, (2) no correlation between the number of messages exchanged and the mutual response time, (3) almost twice as many conversation initiations by female aquaintances, and (4) an almost perfect overlap between the average active times of the author and their conversation partners, namely around 15:00 in the afternoon.



Finally to conclude the analysis, we plotted a correlation heatmap to visualize the relationships between the different features. The heatmap showed the highest correlation between the number of total exchanged messages and the number of emojis used by the author, which was against our intuition, and could be a highly informative feature for conversation classification tasks in WhatsApp chat data and worth further investigation.



Shortcomings & Conclusion

Due to the small sample-size in both participants and exported messages because of insufficient compute (each data processing cycle taking ~2 hours) we can't make any statements on the generalizability of our findings.

Nonetheless, there is a clear potential for this prototype to be developed into a personalized relationship analysis tool for individuals seeking to gain insights into their own communication patterns. Additionally, the models could be further developed to provide forecasting capabilities for predicting future interactions and relationship outcomes based on conversational patterns, which could be highly valuable for personal and professional relationship management.

By discussing the results of this projects with the conversational partners we noticed a lot of enthusiasm and interest in the results, which could be a sign of a potential market for such a tool. However, the ethical implications of such a tool should be carefully considered, as it could potentially infringe on privacy and lead to unintended consequences if not used responsibly.

The time constraints of the project also limited the depth of the analysis and the scope of the features implemented. Future work could focus on expanding the feature set to include more advanced sentiment analysis techniques, such as emotion detection and sarcasm recognition, as well as incorporating more sophisticated topic modeling algorithms to extract deeper insights from the data. We did not overshoot our time budget and the estimations were mostly accurate, but the time spent on the data generation was significantly higher than anticipated, given that running 2 large language models in parallel on a consumer-grade GPU is a much more computationally intensive task than to be expected.

In conclusion: The "Whatsapp Lens" project successfully developed an end-to-end prototype for analyzing WhatsApp chat data, leveraging state-of-the-art NLP techniques and modern data science tools. The project addressed the challenges of data generation, parsing, feature engineering, and analysis, providing valuable insights into conversational patterns and behaviors. We gained data-driven insights into the author's personal chat data, uncovering relationships between message content, sentiment, response times and conversation dynamics, which could be applied to applied psychology, social network analysis and customer relationship management use cases.