

Advanced NLP Project

WhatsApp Chat Reports

[HTTPS://GITHUB.COM/YUNVS/WHATSAPPREPORTS](https://github.com/yunvs/whatsappreports)

YUNUS RENZ, Oct. 21 2022

Inhalte

- Vorgehensweise
- Technologien und Libraries
- Aufbau des Repository
- Verwendung des Programms
- Statistiken & Grafiken
- Ausgabe
- Probleme und Adaptionen

Vorgehensweise

- Heranschaffen von Testdaten (Chatverläufen)
 - hauptsächlich eigene Chats mit Familie und Freunden benutzt
- Projektplanung
 - welche Features und welche Statistiken sollen verwendet werden
 - wie sollen der Code aufgeteilt sein und wie sollen Daten gespeichert werden
 - wie sollen die erhobenen Daten dargestellt und ausgegeben werden
 - welche Libraries soll ich benutzen
- Umsetzung, Probleme erkennen und Lösungen suchen, Projekt zu Ende bringen

Technologien und Libraries,

die in diesem Projekt benutzt werden

Programming language:

- `python3`

Package manager:

- `pip3`
- `anaconda3/conda3`

benötigte Versionen findet
man in `requirements.txt`

- `pandas`: Datenanalyse und Datenmanipulation
- `textblob-de`: Stimmungsanalyse auf Deutsch
- `unidecode`: ASCII-Transliterationen
- `emojis`: Emoji Handhabung
- `numpy`: Numerische Berechnungen
- `matplotlib`: Graphik-Bibliothek für Datenvisualisierung
- `wordcloud`: WordCloud-Erzeugung
- `fpdf`: PDF-Erzeugung und Bearbeitung
- `nltk`: Verarbeitung natürlicher Sprache, Stoppwörter

Technologien und Libraries,

die in diesem Projekt benutzt werden

Weitere verwendete, in python enthaltene Module:

- `re` (Regular Expressions), `os` (Betriebssystem-Funktionalität), `timeit` (Zeitmessung)

Verwendete python Libraries, final nicht mehr verwendet:

- `germansentiment` und `nlptown`: Stimmungsanalyse auf Deutsch
- `emoji` und `emoji-unicode`: Emoji Handhabung
- `plotly` und `seaborn`: Graphik-Bibliothek für Datenvisualisierung
- `spacy` und `stanza`: Verarbeitung natürlicher Sprache
- `pyPdf`, `PyPDF2` and `ReportLab`: PDF-Erzeugung und Bearbeitung

Aufbau des Repository

kurz zusammengefasst

- Inhalte, nach Logik, auf drei Verzeichnisse geteilt
 - **data**, **database** und **utils**
- Code, nach Aufgabe, auf neun Skripte verteilt
 - Dateneingabe, Verarbeitung, Analyse und Ausgabe
- `main.py`: Hauptdatei, die kompletten Code ausführt
- `README.md`: Leitfaden, der Repository detailliert beschreibt
- `requirements.txt`: Datei, die notwendige Python-Module, mit entsprechender Version auflistet

```
$ tree WhatsAppReports/  
WhatsAppReports  
├── data  
│   ├── output  
│   │   ├── images  
│   │   │   ├── page1  
│   │   │   └── senderpages  
│   └── sample_chat.txt  
├── database  
│   ├── constants.py  
│   └── variables.py  
├── utils  
│   ├── analyzer.py  
│   ├── converter.py  
│   ├── getter.py  
│   ├── helper.py  
│   ├── plotter.py  
│   └── texter.py  
├── README.md  
├── main.py  
└── requirements.txt  
  
7 directories, 12 files
```


Data Ordner

Speicherort für die erstellten Grafiken und den PDF Report

```
$ tree WhatsAppReports/  
WhatsAppReports  
├── data  
│   ├── output  
│   │   ├── images  
│   │   │   ├── page1  
│   │   │   └── senderpages  
│   └── sample_chat.txt  
├── database  
│   ├── constants.py  
│   └── variables.py  
├── utils  
│   ├── analyzer.py  
│   ├── converter.py  
│   ├── getter.py  
│   ├── helper.py  
│   ├── plotter.py  
│   └── texter.py  
├── README.md  
├── main.py  
└── requirements.txt
```

7 directories, 12 files

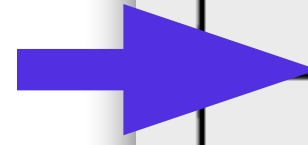
- **output**: Ordner, in dem erstellte Grafiken (WordClouds, Graphen, etc.) und der finale PDF Report gespeichert werden
 - **images/page1**: Grafiken für die erste Seite (Erste Seite: generelle Übersicht und Infos über den Chat)
 - **images/senderpages**: Grafiken für die restlichen Seiten (je zwei Seiten absenderspezifische Informationen pro Sender)
- **sample_chat.txt**: Beispiel Chatverlauf, zum Testen des Code
- falls dem Programm eine **.zip** Datei übergeben wird, wird die daraus entstandene Datei **_chat.txt** genannt und in diesem Verzeichnis gespeichert

Database Ordner

Datenspeicher, auf den alle Dateien zugreifen können

- `constants.py`: Datei, in der fest sehende Begriffe sind
 - z.B.: Bezeichnung von Statistiken oder Grafiken
- `variables.py`: Datei, in der sich Platzhalter befinden
 - anfangs leer, werden beim Durchlaufen des Programmes von anderen Funktionen gefüllt
- wenn `helper.export_database()` Funktion aufgerufen wird, werden alle befüllten Variablen in ein neues Verzeichnis **database/exports** in Form von `.csv` Dateien exportiert

```
$ tree WhatsAppReports/  
WhatsAppReports  
├── data  
│   ├── output  
│   │   └── images  
│   │       ├── page1  
│   │       └── senderpages  
│   └── sample_chat.txt  
├── database  
│   ├── constants.py  
│   └── variables.py  
├── utils  
│   ├── analyzer.py  
│   ├── converter.py  
│   ├── getter.py  
│   ├── helper.py  
│   ├── plotter.py  
│   └── texter.py  
├── README.md  
├── main.py  
└── requirements.txt  
  
7 directories, 12 files
```

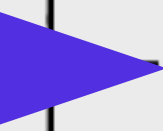


Utils Ordner

alle Funktionen, die von dem Programm benötigt werden

```
$ tree WhatsAppReports/
WhatsAppReports
├── data
│   ├── output
│   │   └── images
│   │       ├── page1
│   │       └── senderpages
│   └── sample_chat.txt
├── database
│   ├── constants.py
│   └── variables.py
├── utils
│   ├── analyzer.py
│   ├── converter.py
│   ├── getter.py
│   ├── helper.py
│   ├── plotter.py
│   └── texter.py
├── README.md
├── main.py
└── requirements.txt

7 directories, 12 files
```



- `getter.py`: Dateneingabe und Verarbeitung
 - nimmt Dateipfad entgegen, prüft die Datei auf Typ sowie Richtigkeit und unzippt (wenn nötig) die Datei
- `converter.py`: Umwandlung und Vorbereitung
 - wandelt Chat Rohdaten mit `regex` zuerst zu einer verschalteten Liste und danach in ein `pandas.DataFrame`
- `analyzer.py`: Analyse des Chatverlaufes
 - analysiert und berechnet Statistiken mit NLP-Modulen für den generellen Chat und für jeweils jeden Sender

Utils Ordner (Fortführung)

alle Funktionen, die von dem Programm benötigt werden

```
$ tree WhatsAppReports/  
WhatsAppReports  
├── data  
│   ├── output  
│   │   └── images  
│   │       ├── page1  
│   │       └── senderpages  
│   └── sample_chat.txt  
├── database  
│   ├── constants.py  
│   └── variables.py  
├── utils  
│   ├── analyzer.py  
│   ├── converter.py  
│   ├── getter.py  
│   ├── helper.py  
│   ├── plotter.py  
│   └── texter.py  
├── README.md  
├── main.py  
└── requirements.txt
```

7 directories, 12 files

- `plotter.py`: Datenvisualisierung
 - erstellt WordCloud und zeichnet mit `matplotlib` um Graphen; für ganzen Chat und jeweils jeden Sender
- `texter.py`: Text Verfassung und Report Erstellung
 - erstellt eine übersichtliche Darstellung der erfassten Statistiken, Grafiken und Texten in Form eines PDF Reports
- `helper.py`: Hilfsfunktionen (z.B.: Zeitmessung, Export, ...)
 - enthält wichtige Funktionen für alle Skripte
 - aber kein Teil der eigentlichen Aufgabe des Programs

Verwendung des Programms

WhatsApp Chats analysieren

1. **Chat aus WhatsApp exportieren:** Kontakt auswählen; auf Namen klicken; Chat exportieren auswählen und ohne Medien exportieren auswählen
2. **Datei importieren:** Dateipfad (.txt- oder .zip-Datei) in `main.py` angeben oder Datei ausführen und Dateipfad, wenn aufgefordert, im Terminal eingeben
3. **Chat verarbeiten lassen:** Programm zeigt durchlaufenden Schritte und benötigte Zeit an; Nach Fertigstellung Ausgabe der Dauer; Programm schließt sich
4. **Ergebnisse:** Programm speichert leserfreundlicher PDF-Bericht, der die erstellten Statistiken, Grafiken und Texte enthält im **data/output** Ordner; Erste Seite: Generelle Infos; Danach: je Sender zwei Seiten mit absenderspezifischen Infos

Statistiken

Allgemeine Statistiken über den Chat und alle Sender

- Anzahl der Nutzer
- Gesamtzahl der Nachrichten, Medien und Emojis
- Durchschnittliche Nachrichtenlänge, Anzahl der Wörter und Zeichen
- Längste und kürzeste Nachricht
- Gesamtzahl der einzelnen Medientypen
- Gesamtzahl an unterschiedlichen Emojis
- Gesamtzahl der positiven und negativen Nachrichten
- Durchschnittliches Sentiment

Statistiken

Absenderspezifische Statistiken für jeden Absender

- Gesamtzahl der Nachrichten, Medien und Emojis pro Absender
- Gesamtzahl der unterschiedlichen Medien pro Absender
- Gesamtzahl der unterschiedlichen Wörter und Emojis pro Absender
- Meist verwendete Wörter und Emojis pro Absender
- Durchschnittliche Nachrichtenlänge, Anzahl von Wörtern und Zeichen pro Absender
- Längste und kürzeste Nachricht pro Absender
- Gesamtzahl der positiven und negativen Nachrichten pro Absender
- Durchschnittliches Sentiment pro Absender

Statistiken

Zeitbasierte Statistik über die Zeiten der Nachrichten

Im Allgemeinen und pro Absender:

- Datum der ersten und letzten Nachricht
- Zeitspanne zwischen der ersten und der letzten Nachricht
- Datum des Tages mit den meisten Nachrichten
- Tag und Stunde mit den meisten Nachrichten
- Anzahl der Tage mit und ohne Nachrichten
- Längste Zeitspanne ohne Nachrichten

Statistiken

NLP Statistik über linguistische Features

Im Allgemeinen und pro Absender:

- Anzahl der verschieden Wörtern, Lemmas, Stoppwörtern
- Durchschnittliche Wort- und Satzlänge
- Anzahl der erkannten Eigennamen
- Anzahl der Wortarten (POS-Tags)
- Meist verwendete Lemmas und Stoppwörter (mit Frequenzen)

Grafiken

Unterschiedliche grafische Ausgaben

- **WordClouds** mit dem `wordcloud`-Modul
- **Heatmaps, Balken-, Kreis-, Liniendiagramme, Box- und Violinplots** mit dem `matplotlib`-Modul
- **PDFs** mit dem `fpdf`-Moduls
 - dieses enthält verfasste Texte, Bilder (der oben genannten Grafiken) und Zellen und Tabellen

Ausgabe

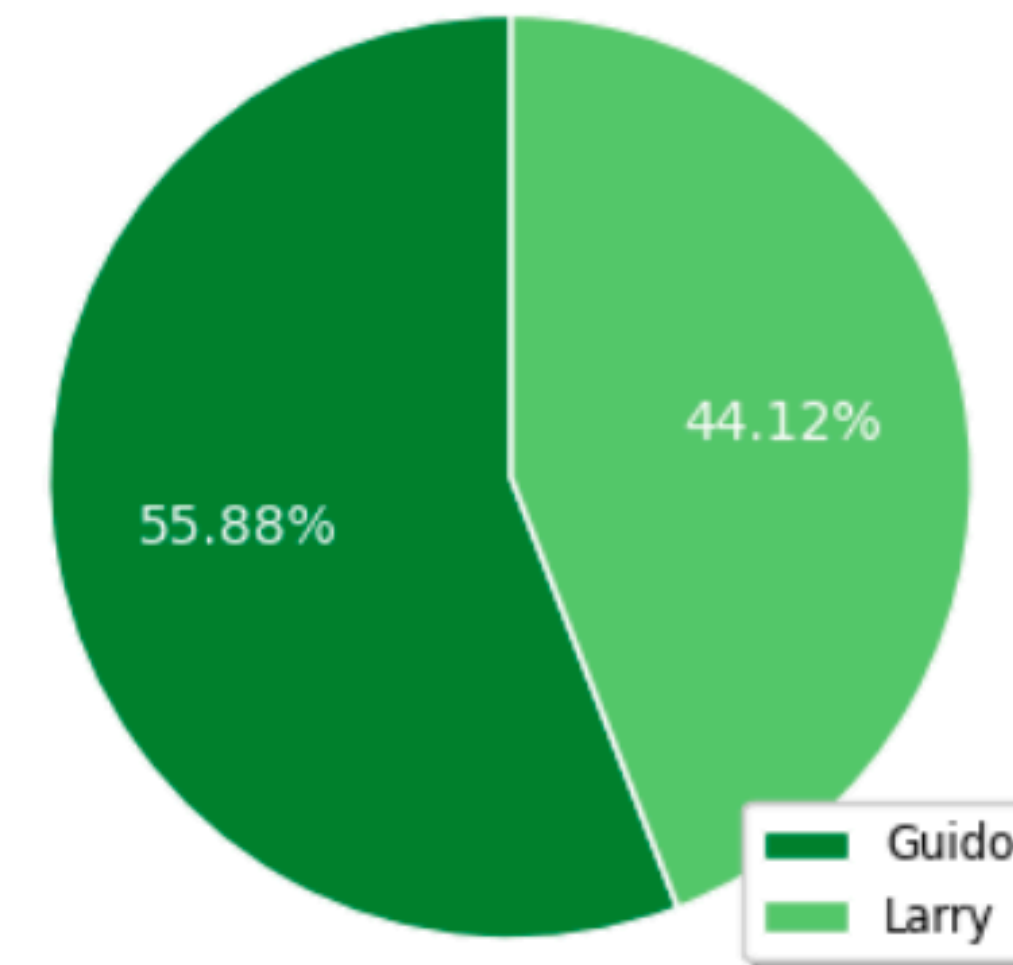
WhatsApp Chat Statistics for Guido and Larry

This conversation contains
204 texts, 41 media and 46 emojis

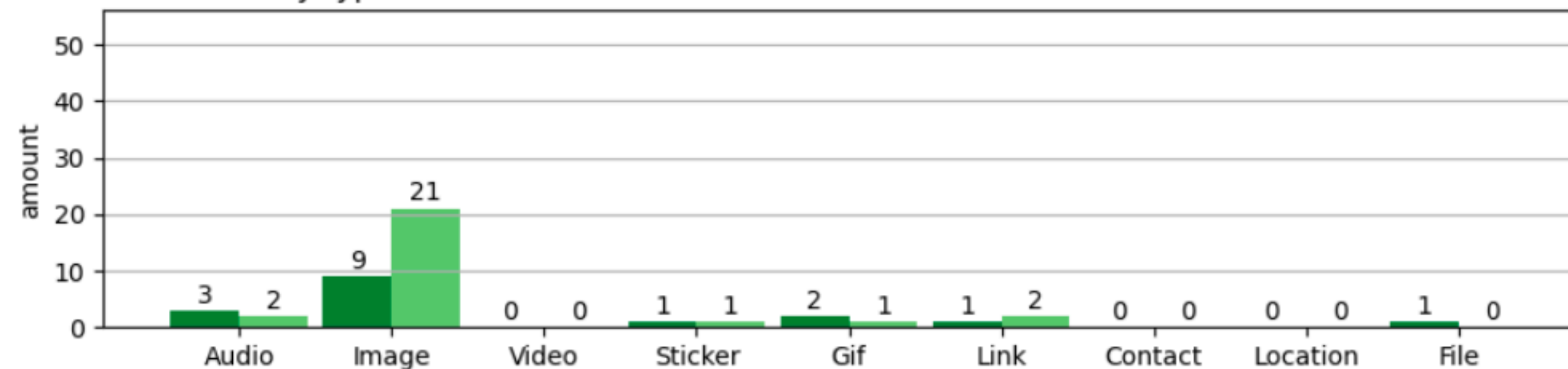
Avg. message length: 4.35 words (18.4 chars)
Longest message: 14 words (69 chars)
Guido and Larry deleted 0 messages.

At lest one message sent on 65 days.
Not a single message sent on 177 days.
Longest period without any messages: 19 days.

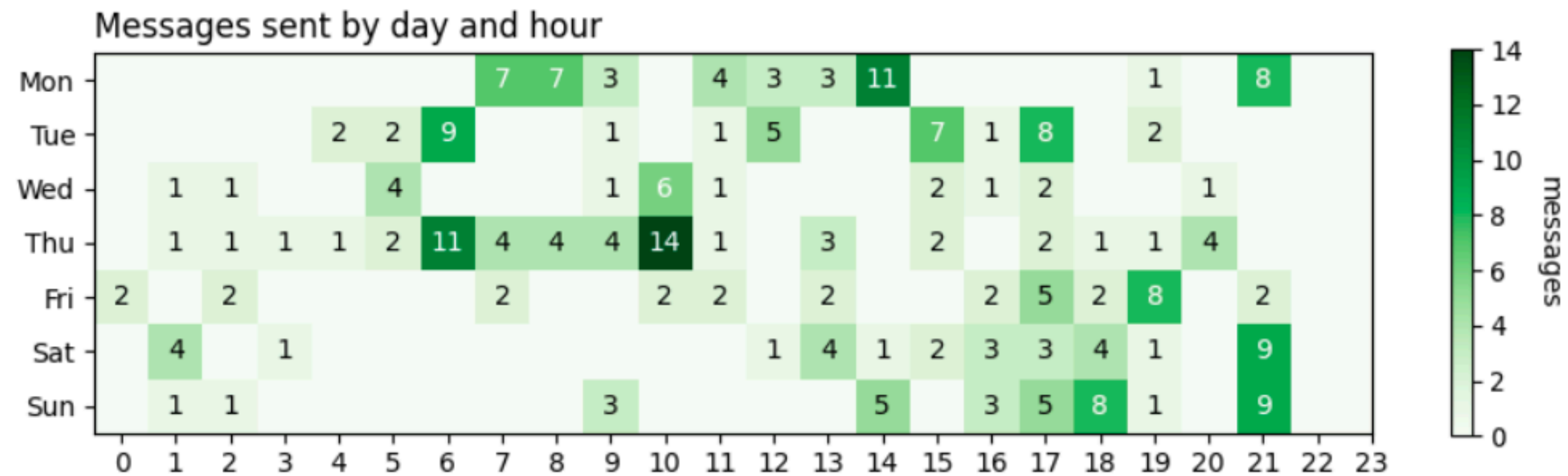
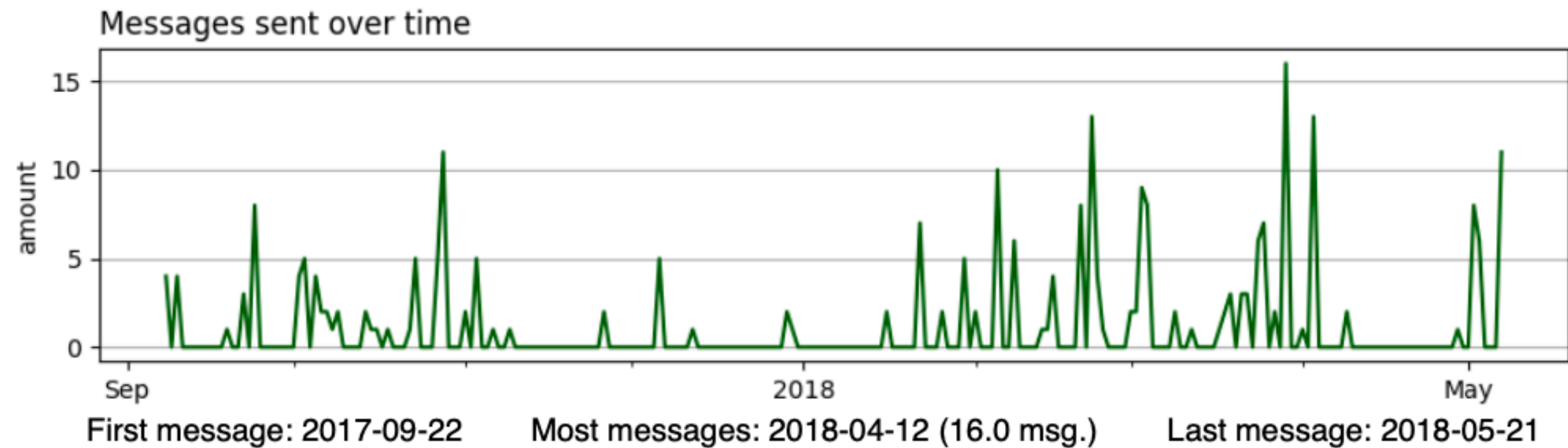
Messages sent



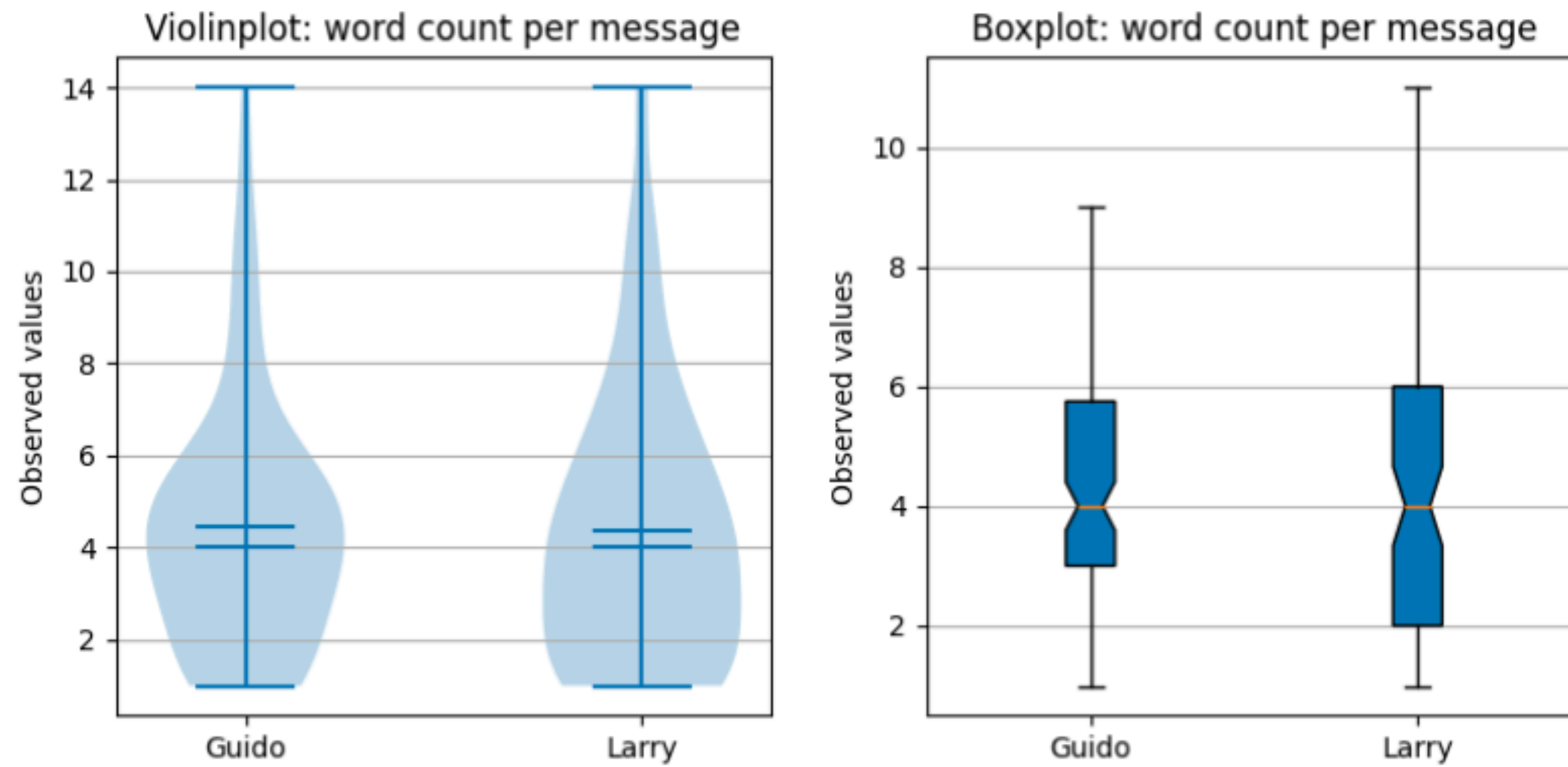
Media sent by type and sender



Ausgabe



Ausgabe



Ausgabe



Top 20 Words	Frequency
ich	54
du	29
bin	20
hab	13
ja	12
die	12
noch	10

Top 18 Emojis	Frequency
<u>kissing heart</u>	14
<u>heart eyes</u>	7
<u>tada</u>	6
<u>clap</u>	2
<u>thumbsup</u>	2
<u>two hearts</u>	2
<u>hugs</u>	2

Ausgabe

Natural Language Processing Analysis

Amount of words: 486; distinct ones: 252
Amount of lemmas: 201; distinct ones: 151
Amount of stopwords: 284; distinct ones: 102
Amount of non-stopword words: 202; distinct ones: 150
Average word length: 4.34 chars
Amount of recognized named entities: 31

POS Tags and counts

NOUN	53	DET	23	ADJ	12
PRON	89	AUX	41	ADV	97
VERB	67	SPACE	40	ADP	35
PROPN	38	X	13	PART	6
CCONJ	6	SCONJ	6		

Top 20 Lemmas	Frequency
hab	13
zuhause	3
danke	3
ok	3
out	3

Top 20 Stopwords	Frequency
ich	38
du	13
bin	11
ja	11
noch	10

Ausgabe

WhatsApp Chat Statistics for Guido

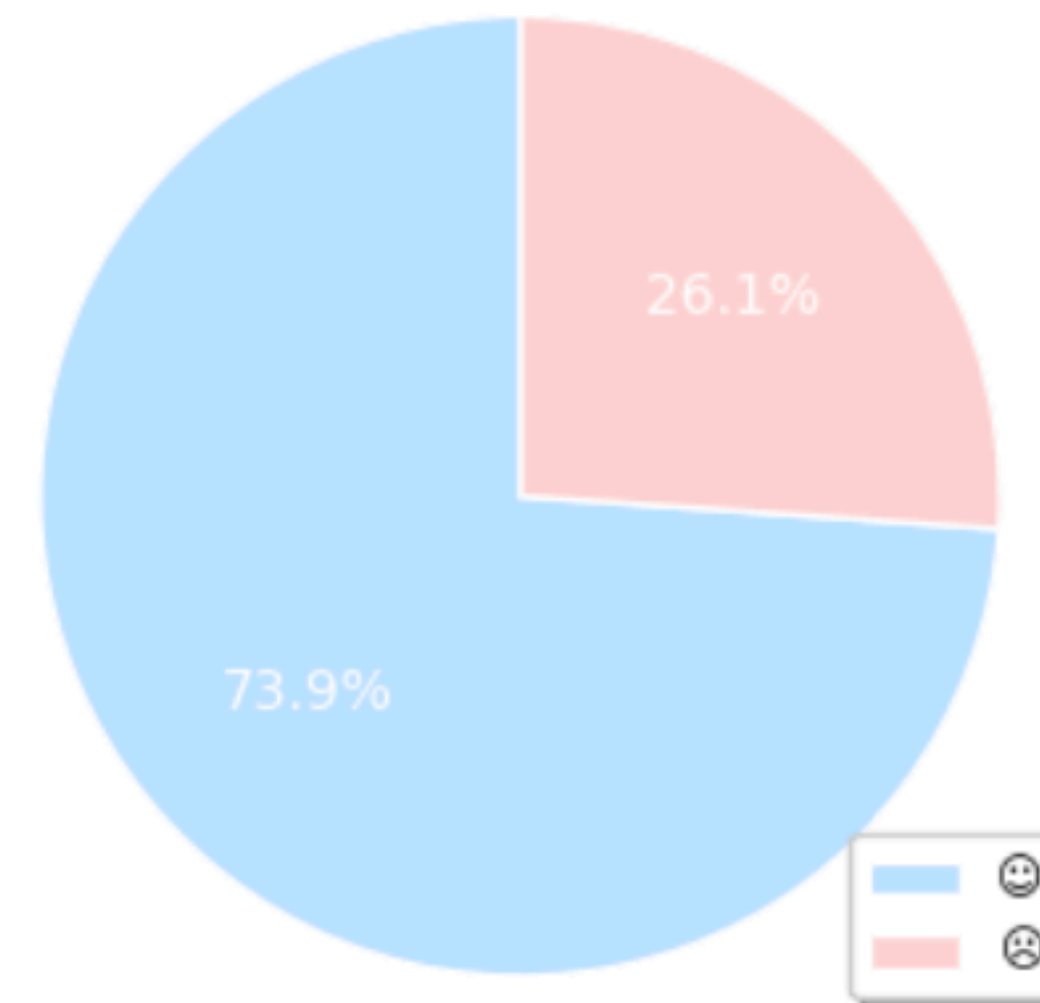
Messages Guido sent contain
114 texts, 16 media and 20 emojis

Media types and amount sent

Audios	3	Contacts	0
Images	9	Locations	0
Videos	0	Files	1
Stickers	1	Links	1
Gifs	2		

Avg. message length: 4.4 words (18.6 chars)
Longest message: 11 words (61 chars)
0 messages were deleted by Guido.

Sentiment of rated messages



TOP 20 Words	Frequency
ich	38
du	13
hab	13
ia	11

TOP 9 Emojis	Frequency
heart eyes	7
tada	6
open mouth	1
dolls	1

Ausgabe

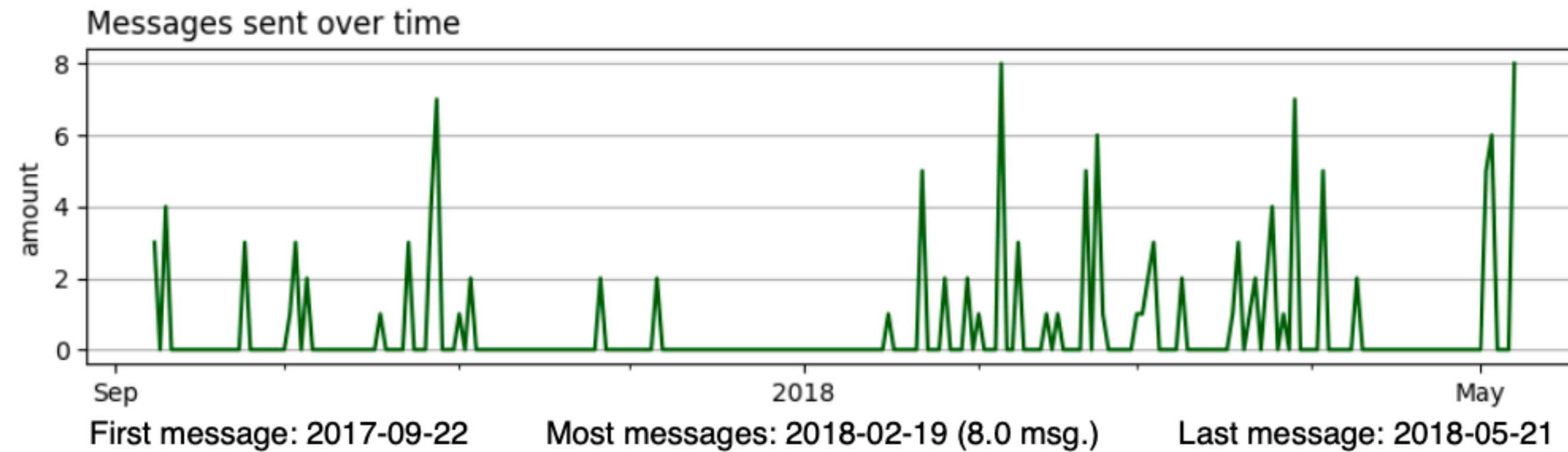
au	15
bin	9
guido	7
dich	7
ok	6
in	6
der	5
das	5
toll	4
ist	4
es	4
nicht	4
schule	4
super	4
habe	4
bitte	4
bist	4
hast	4
mich	4

clap	2
hugs	2
two hearts	2
thumbsup	2
grinning	1
pray	1
yum	1
umbrella	1

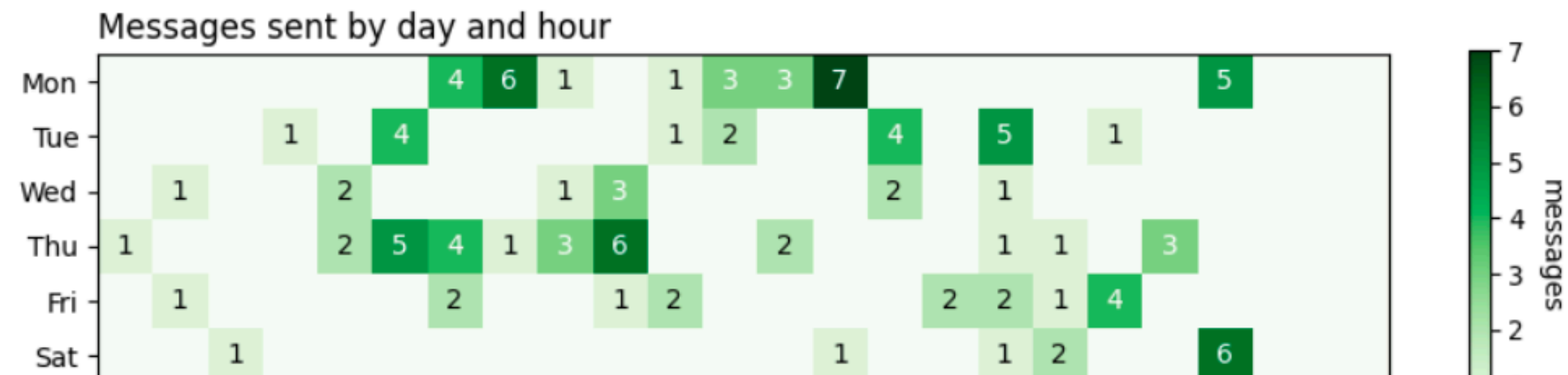
In total Larry used 208 distinct words and 9 distinct emojis.

Ausgabe

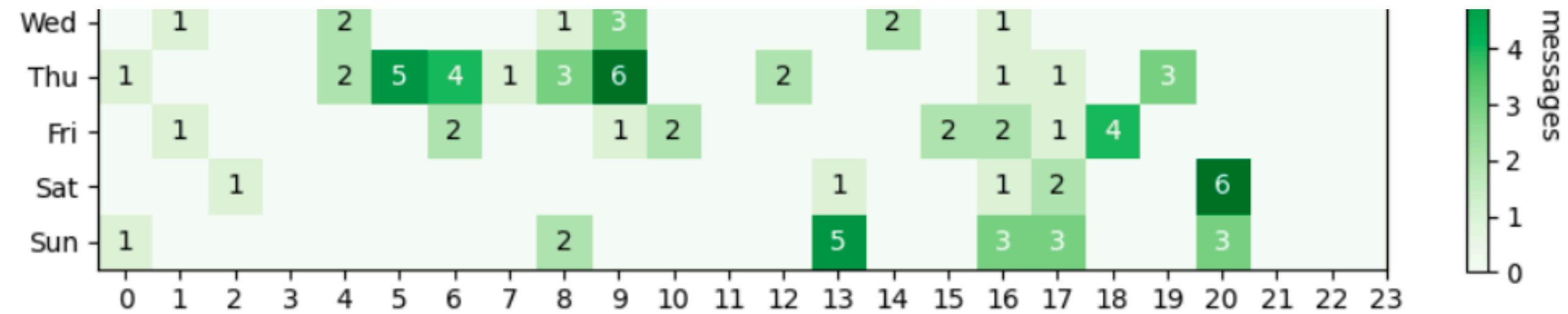
WhatsApp Chat Statistics for Guido



Guido sent at least one message on 44 days and no messages on 198 days.
The longest period without any messages from Guido was 40 days.



Ausgabe



Probleme und Adaptionen

Stimmungsanalyse Problem: großer Datensätzen mit Kurznachrichten, wenig Modelle

Fix: Textblob-de und Nachrichten ausschließen

PDF Problem: Emoji Darstellung nicht unterstützt, nur Latin Buchstaben

Fix: emojis Transliterationen + Link zu Google img search und unicodecode

Emoji Problem: neuere Emoji nicht in allen Modellen vorhanden

Fix: emojis Transliterationen unterstützen (fast) alle Emojis

NLP Problem: Tools brauchen mehr Input als nur ein paar Wörter

Fix: weniger dafür aussagekräftigere Statistiken

QUELLEN:

Adobe Systems Incorporated. (2016). PDF Reference fifth edition: Adobe® Portable Document Format Version 1.6. Internet-Archive. <https://web.archive.org/web/20160304052118/http://partners.adobe.com/public/developer/en/pdf/PDFReference16.pdf>

Dean, B. (2022, January 5). WhatsApp 2022 User Statistics. Backlinko. <https://backlinko.com/whatsapp-users>

Singh, M. (2020, October 30). WhatsApp is now delivering roughly 100 billion messages a day. TechCrunch. <https://techcrunch.com/2020/10/29/whatsapp-is-now-delivering-roughly-100-billion-messages-a-day/?guccounter=1>

Chapagain, Anish. Hands-On Web Scraping with Python: Perform advanced scraping operations using various Python libraries and tools such as Regex, and others. Packt Publishing Ltd, 2019.

Lutz, Mark. Programming python. " O'Reilly Media, Inc.", 2001

Walkowska, J. (2010). An NLP-Oriented Analysis of the Instant Messaging Discourse. Text, Speech and Dialogue, 576–583. https://doi.org/10.1007/978-3-642-15760-8_73

**Vielen Danke,
für eure Aufmerksamkeit**

**Gibt es noch Fragen?
Are there any Questions?**