Hanna Tuomi
4.032 Final Project

Interface Design

0 Referencing dictionary idea

1 Instant read: text messages

2 Color palette
    ➔ reference messenger apps? (telegram blue)
presentation notes/next steps

1. Data limitations
   - Messages from my perspective
   - Chose words I know I use significantly
     - alternative ways of selecting words
   - Didn’t want to cherry-pick messages to show as examples
     - Could be cool to have pop-up of the interaction

2. Design / Interaction
   - Making transition between timing & frequency smoother
     - animation of bars collapsing into bar
   - Being able to see messages or examples when hovering over the bars
   - Colors ⇒ what other color could help?
     - Highlighting myself?
   - Some glitches on entry to the program
meep

filler word to either a) get someone's attention, or b) to fill awkward space

When did this word get said?

How often was this word said?

HT
MY
YP
AC
JL

0 7 15 22 29
beep

Filler word to get someone's attention

- HT
- D
- AC
- JC

When did this word get said?
How often was this word said?
Yikes on Bikes - Used when something is particularly cringey

- HT
- MY
- YP
- VL
- Beep

Timing vs. Frequency Graph
<table>
<thead>
<tr>
<th>Yikes on Bikes</th>
<th>🤔 - emoji response when something is particularly interesting or exciting, but is self-explanatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meep</td>
<td>HH</td>
</tr>
<tr>
<td>😓</td>
<td>D</td>
</tr>
<tr>
<td>:)</td>
<td>YY</td>
</tr>
<tr>
<td>oh oh</td>
<td>FF</td>
</tr>
<tr>
<td>Beep</td>
<td>SS</td>
</tr>
</tbody>
</table>
Yikes on Bikes

Meep

::

:'

oh oh

Beep

:)' - emoji response when you feel slight disappointment or want to express struggle

Hanna Tuomi
Dyusha
Derik Kaufman
Vivian Li
Yooni Park
Dyusha Gritsevsky
Rachel Zhang
Kevin Ye
Emily Huang
Fiona Gillespie
Julia Fikinskii
Miki Hansen
Danny Roh
Cynthia Chen
Joyce Guo
Kathryn Tso
Shirley Cao
Amy Liu
Joy Feng
"Yikes on Bikes"
```python
import pandas as pd
import numpy as np
import os
import string, json, csv
from facebook_df import get_facebook_df
from telegram_df import get_telegram_df
import heapq

# load the data
tele_df = get_telegram_df()
fb_df = get_facebook_df()

# create a word frequency dictionar #
# ignore 1000 most common words
# a_file = open("common_words.txt", "r")

# common_words = set()
# for line in a_file:
#     stripped_line = line.strip()
#     line_list = stripped_line.split()
#     common_words.update(line_list)
# a_file.close()

# freq_dict = {}
# for data in [tele_df.text, fb_df.content]:
#     for mess in data:
#         try:
#             mess = mess.translate(str.maketrans('\', '', string.punctuation))
#             words = mess.rsplit(" ")
#             for w in words:
#                 if w not in common_words:
#                     w = w.lower()
#                     try:
#                         freq_dict[w]['uses'] += 1
#                         freq_dict[w]['users']
#                     except:
#                         word_dict = {"uses": 0, "users": {}, "flag": 0}
#                         freq_dict[w] = 1
#         except:
#             # print('errored on message:', mess)
#             # continue

# open file for writing, "w" is writing
# w = csv.writer(open("word_freq_dict.csv", "w"))

# loop over dictionary keys and values
# for key, val in freq_dict.items():
#     # write every key and value to file
#     w.writerow([key, val])

# d = dict((k, v) for k, v in freq_dict.items() if v >= 500)
```
# print(d)

###

```python
def get_initials(name):
    if name == "":
        initials = "NA"
    else:
        try:
            name_list = name.split(' ')
            initials = ""
            for n in name_list:  # go through each name
                initials += n[0].upper()  # append the initial
        except:
            initials = "none"
    return initials
```

```python
base = "/Users/hannatuomi/Desktop/Message Data"
```

```python
os.chdir(base)
```

```python
possible = ["yikes on bikes", "meep", "beep ", "oh oh", ":\""]
```

```python
for p in possible:
    print('working on word:', p)
    word_df = pd.DataFrame()
    for i in [0, 1]:
        if i == 0:
            data = tele_df
            data['used'] = data['text'].str.contains(p)
            data['initials'] = data.apply(lambda row : get_initials(row['from']), axis = 1)
            # indexer = (data['text'].str.contains(p))
            cols = ['date', 'from', 'used', 'initials']
            user_name = 'from'
            dataset = "telegram"
        else:
            data = fb_df
            if p == "•\•":
                p = "\x00\x9F\x91\x80"
            data['used'] = data['content'].str.contains(p)
            data['initials'] = data.apply(lambda row : get_initials(row['sender_name']), axis = 1)
            # indexer = (data['content'].str.contains(p))
            cols = ['timestamp_ms', 'sender_name', 'used', 'initials']
            user_name = 'sender_name'
            dataset = "fb"
        data = data[data['used'] == True]
        data = data[cols]
        data.columns = ['date', 'from', 'used', 'initials']
        word_df = word_df.append([word_df, data])
```
users = word_df['from'].unique()

# get only top 5
if len(users) > 5:
    freq = []
    for user in users:
        d = word_df[word_df['from'] == user]
        freq.append(d.loc[:, 'used'].sum())
    indexes = [(i, x) for i, x in enumerate(freq)]
    keep_users = [users[i] for i in indexes]
    word_df = word_df.loc[word_df['from'].isin(keep_users)]

# get the updated users
users = word_df['from'].unique()
print('The users for ' + p + ' are: ', users)
word_df.to_csv('word_users/' + p + '_use.csv')
```python
import json, pandas as pd, csv
import numpy as np

# working with telegram message data #

def get_telegram_df():
    with open("result.json") as json_file:
        data = json.load(json_file)

        # reduce data down to the actual chat data
        chats = data["chats"]['list']

        # get the individual data points of the chat
        cols = chats[0]["messages"][0].keys()
        chat_deets = ["chat", cols]

        base_chat = chats[0]
        chat_name = base_chat["name"]
        mess = base_chat["messages"]
        tele_df = pd.DataFrame(mess)
        tele_df["chat_name"] = chat_name

        # take each chat and transform into csv
        for c in chats:
            if c["type"] != "saved_messages":  
                chat_name = c["name"]
                chat = c["messages"]
                chat_df = pd.DataFrame(chat)
                chat_df["chat_name"] = chat_name
                tele_df = tele_df.append(chat_df)

            tele_df = tele_df.drop(columns=['reply_to_message_id', 'photo', 'width', 'height', 'via_bot',
                                       'file', 'thumbnail', 'media_type', 'mime_type', 'duration_seconds',
                                       'forwarded_from',
                                       'self_destruct_period_seconds', 'contact_information',
                                       'sticker_emoji', 'performer',
                                       'discard_reason', 'location_information',
                                       'live_location_period_seconds', 'game_title',
                                       'game_description', 'game_link', 'game_message_id', 'score',
                                       'poll'])

            tele_df["date"] = pd.to_datetime(tele_df['date'])
            tele_df["date"] = tele_df["date"].astype('int64')

        print("Done Telegram Data")
        # tele_df.to_csv("telegram_df.csv")

    return tele_df
```
import json, pandas as pd, csv
import numpy as np
import os

# working with facebook message data #

def get_facebook_df():
    # assign directory
    archived = "/Users/hannatuomi/Desktop/Message Data/fb_messages/archived_threads"
    inbox = "/Users/hannatuomi/Desktop/Message Data/fb_messages/inbox"
    base = "/Users/hannatuomi/Desktop/Message Data"

    fb_df = pd.DataFrame()

    for direct in [archived, inbox]:
        # for direct in [archived]:
        print('entered loop')
        os.chdir(direct)
        for dir in os.listdir():
            try:
                os.chdir(direct + "/" + dir)

                for file in os.listdir():
                    if file.endswith("json"):
                        with open(file) as json_file:

                            data = json.load(json_file)

                            # get list of the people in the chats
                            people = []
                            for p in data["participants"]:
                                people.append(p["name"])

                            # only care about non-massive group chats
                            if len(people) < 12:
                                chats = data["messages"]
                                chat_df = pd.DataFrame(chats)
                                chat_df["chat_name"] = data["title"]
                                chat_df["thread_type"] = data["thread_type"]

                                # add chat members data
                                chat_df["members"] = np.NaN
                                chat_df["members"] = chat_df["members"].astype('object')
                                for row in range(len(chat_df)):
                                    chat_df.at[row, "members"] = people

                                fb_df = pd.concat([fb_df, chat_df])

                        # account for non-directories
                        except:
                            continue

                fb_df = fb_df.drop(columns=['is_unsent', 'reactions', 'photos', 'files', 'sticker', 'share', 'gifs', 'videos', 'call_duration', 'audio_files', 'ip', 'missed'])

                fb_df["timestamp_ms"] = pd.to_datetime(fb_df["timestamp_ms"], unit='ms')
fb_df['timestamp_ms'] = fb_df['timestamp_ms'].astype('int64')
fb_df = fb_df[fb_df['content'].notna()]

# attempt to throw out data that is not related to relevant people (TO DO)
# chatters = fb_df['sender_name'].unique()
# chatter_freq = {}
# for c in chatters:
#     chatter_freq[c] = len(fb_df[fb_df['sender_name'] == c])
# for i in range(len(fb_df)):
#     fb_df.loc[i,]['user_freq'] = chatter_freq[fb_df.loc[i,]['sender_name']] > 50

print("Done FB Data")
# os.chdir(base)
# fb_df.to_csv("facebook_df.csv")

return fb_df