

confirmation_bias_vis

December 4, 2022

Create directories using the following command:

```
import os
```

```
if not os.path.exists("images"): os.mkdir("images")
```

TODO:

- VISUALIZE GROUPS AS DIFFERENT AXIS ON A 3D SCATTER PLOT IN PLOTLY
- NetworkX PLOTLY GRAPH VISUALIZATION

```
[1]: import os
import sys

import nltk
import tweepy
from dotenv import load_dotenv
import json
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
import matplotlib.pyplot as plt
from datetime import datetime
import re
import cv2
import numpy as np
import pandas as pd
import pickle
# NOTE: THE WORDCLOUD PACKAGE ISN'T WORKING FOR SOME VERSIONS OF PYTHON.
# LOCKING PYTHON VERSION TO 3.7 BECAUSE OF THAT.
from wordcloud import WordCloud
import spacy
import spacy_transformers
import torch
```

```
[2]: import ssl
def set_up_ssl():
    try:
        _create_unverified_https_context = ssl._create_unverified_context
    except AttributeError:
        pass
    else:
```

```
ssl._create_default_https_context = _create_unverified_https_context

set_up_ssl()
```

```
[3]: load_dotenv()
TWITTER_BEARER_TOKEN = os.getenv('TWITTER_BEARER_TOKEN')
TWITTER_API_KEY = os.getenv('TWITTER_API_KEY')
TWITTER_API_SECRET_KEY = os.getenv('TWITTER_API_SECRET_KEY')
TWITTER_ACCESS_TOKEN = os.getenv('TWITTER_ACCESS_TOKEN')
TWITTER_ACCESS_TOKEN_SECRET = os.getenv('TWITTER_ACCESS_TOKEN_SECRET')
```

```
[4]: print("Authenticating to Twitter...")

client = tweepy.Client(bearer_token=TWITTER_BEARER_TOKEN,
    ↪wait_on_rate_limit=True)
auth = tweepy.OAuthHandler(TWITTER_API_KEY, TWITTER_API_SECRET_KEY)
auth.set_access_token(TWITTER_ACCESS_TOKEN, TWITTER_ACCESS_TOKEN_SECRET)
api = tweepy.API(auth, wait_on_rate_limit=True)
```

Authenticating to Twitter..

```
[5]: twitter_russia_sources_rus_usernames = ["@1tvru_news", "@ru_rbc",
    "@er_novosti",
    "@rt_com",
    "@medvedevrussia", "@kremlinrussia",
    "@rentvchannel", "@vesti_news", "@kpuru"]

twitter_ukraine_sources_rus_usernames = ["@dmitry_gordon", "@SvobodaRadio",
    ↪"@euronewsru", "@FeyginMark4", "@tvrain",
    ↪"@teamnavalny"]

twitter_ukraine_sources_ukr_usernames = ["@HromadskeUA", "@tsnua", "@24tvua",
    ↪"@unian",
    "@radiosvoboda", "@5channel", "@EspressoTV"]

twitter_ukraine_sources_eng_usernames = ["@mschwartz", "@KyivIndependent",
    ↪"@KyivPost",
    "@mchancecnn", "@fpleitgenCNN", "@Kasparov63",
    "@ikhurshudyan", "@myroslavapetsa",
    "@langfittnpr", "@ElBeardsley", "@timkmak"]
```

<https://dev.to/twitterdev/a-comprehensive-guide-for-using-the-twitter-api-v2-using-tweepy-in-python-15d9>

```
[6]: # THIS CELL HITS TWITTER API

def get_user_id_from_username(username):
```

```
user = api.get_user(screen_name=username)
return user.id
```

```
[7]: russia_sources_rus = []
ukraine_sources_rus = []
ukraine_sources_ukr = []
ukraine_sources_eng = []
```

```
[8]: LOAD_CLUSTERS_DATA = True
```

```
[9]: if not LOAD_CLUSTERS_DATA:
    for username in twitter_russia_sources_rus_usernames:
        russia_sources_rus.append((username,
        ↪get_user_id_from_username(username)))

    for username in twitter_ukraine_sources_rus_usernames:
        ukraine_sources_rus.append((username,
        ↪get_user_id_from_username(username)))

    for username in twitter_ukraine_sources_ukr_usernames:
        ukraine_sources_ukr.append((username,
        ↪get_user_id_from_username(username)))

    for username in twitter_ukraine_sources_eng_usernames:
        ukraine_sources_eng.append((username,
        ↪get_user_id_from_username(username)))
```

```
[10]: if not LOAD_CLUSTERS_DATA:
    russia_sources_rus_pickled = pickle.dumps(russia_sources_rus)
    ukraine_sources_rus_pickled = pickle.dumps(ukraine_sources_rus)
    ukraine_sources_ukr_pickled = pickle.dumps(ukraine_sources_ukr)
    ukraine_sources_eng_pickled = pickle.dumps(ukraine_sources_eng)
```

```
[11]: CLUSTERS_SERIALIZATION_DIR = "data_clusters/"
```

```
[12]: if not LOAD_CLUSTERS_DATA:
    print("Writing pickled data to a file...")

    with open(CLUSTERS_SERIALIZATION_DIR + 'russia_sources_rus_pickled.pickle',
    ↪'wb') as f:
        f.write(russia_sources_rus_pickled)

    with open(CLUSTERS_SERIALIZATION_DIR + 'ukraine_sources_rus_pickled.
    ↪pickle', 'wb') as f:
        f.write(ukraine_sources_rus_pickled)
```

```

    with open(CLUSTERS_SERIALIZATION_DIR + 'ukraine_sources_ukr_pickled.
↳pickle', 'wb') as f:
        f.write(ukraine_sources_ukr_pickled)

    with open(CLUSTERS_SERIALIZATION_DIR + 'ukraine_sources_eng_pickled.
↳pickle','wb') as f:
        f.write(ukraine_sources_eng_pickled)

```

```

[13]: if LOAD_CLUSTERS_DATA:
    with open(CLUSTERS_SERIALIZATION_DIR + 'russia_sources_rus_pickled.pickle',\
        'rb') as f:
        russia_sources_rus = pickle.load(f)

    with open(CLUSTERS_SERIALIZATION_DIR + 'ukraine_sources_rus_pickled' \
        '.pickle', 'rb') as f:
        ukraine_sources_rus = pickle.load(f)

    with open(CLUSTERS_SERIALIZATION_DIR + 'ukraine_sources_ukr_pickled' \
        '.pickle', 'rb') as f:
        ukraine_sources_ukr = pickle.load(f)

    with open(CLUSTERS_SERIALIZATION_DIR + 'ukraine_sources_eng_pickled' \
        '.pickle', 'rb') as f:
        ukraine_sources_eng = pickle.load(f)

```

```

[14]: # THIS CELL HITS TWITTER API

def get_user_followers(user_name, user_id, num_pages=1, num_followers=10):
    followers = []
    # User rate limit (User context): 15 requests per 15-minute window per each_
↳authenticated user
    # limit - Maximum number of requests to make to the API
    # max_results : The maximum number of results to be returned per page. This_
↳can be a number between 1 and the 1000. By default, each page will return_
↳100 results.
    # I.E. 15 000 followers can be returned in 15 minutes
    followers_paginator = tweepy.Paginator(client.get_users_followers, id =
user_id, max_results = num_followers, limit = num_pages).flatten()
    for follower in followers_paginator:
        followers.append(follower)
    return (user_name, user_id), followers

```

```

[15]: # THIS CELL HITS TWITTER API

def get_user_follower_count(user_id):
    # fetching the user
    user = api.get_user(user_id = user_id)

```

```
return user.followers_count
```

```
[16]: rus_cluster_followers = []
      ukr_eng_cluster_followers = []
      ukr_rus_cluster_followers = []
      ukr_ukr_cluster_followers = []
```

```
[17]: # PROBLEM: pickling followers causes recursion depth exceeded problem
      # SOLUTION: process the followers data right away and write the result in csv
      # format
```

```
LOAD_FOLLOWERS = False
FOLLOWERS_DIR = "followers/"
```

```
[18]: # PRODUCTION CODE
      # get 105 000 followers per cluster: 1:45 min per cluster, 105 requests

      '''
      if not LOAD_FOLLOWERS:
          for cluster in russia_sources_rus:
              rus_cluster_followers.append(get_user_followers(cluster[0], cluster[1]))

          for cluster in ukraine_sources_rus:
              ukr_eng_cluster_followers.append(get_user_followers(cluster[0],
              ↪cluster[1]))

          for cluster in ukraine_sources_ukr:
              ukr_rus_cluster_followers.append(get_user_followers(cluster[0],
              ↪cluster[1]))

          for cluster in ukraine_sources_eng:
              ukr_ukr_cluster_followers.append(get_user_followers(cluster[0],
              ↪cluster[1]))
      '''
```

```
[18]: '\nif not LOAD_FOLLOWERS:\n    for cluster in russia_sources_rus:\n
      rus_cluster_followers.append(get_user_followers(cluster[0], cluster[1]))\n\n
      for cluster in ukraine_sources_rus:\n
      ukr_eng_cluster_followers.append(get_user_followers(cluster[0], cluster[1]))\n\n
      for cluster in ukraine_sources_ukr:\n
      ukr_rus_cluster_followers.append(get_user_followers(cluster[0], cluster[1]))\n\n
      for cluster in ukraine_sources_eng:\n
      ukr_ukr_cluster_followers.append(get_user_followers(cluster[0], cluster[1]))\n'
```

```
[19]: # THIS CODE HITS TWITTER API
      # TEST CODE: 50 followers per cluster, 2 cluster per each
      if not LOAD_FOLLOWERS:
```

```

    for i in range(0, 2):
        rus_cluster_followers.
↪append(get_user_followers(russia_sources_rus[i][0],
↪                        russia_sources_rus[i][1]))

    for i in range(0, 2):
        ukr_eng_cluster_followers.
↪append(get_user_followers(ukraine_sources_rus[i][0],
↪                        ukraine_sources_rus[i][1]))

```

```
[20]: type(rus_cluster_followers)
```

```
[20]: list
```

```
[21]: if not LOAD_FOLLOWERS:
        rus_cluster_followers_pickled = pickle.dumps(rus_cluster_followers)
        ukr_eng_cluster_followers_pickled = pickle.dumps(ukr_eng_cluster_followers)
        ukr_rus_cluster_pickled = pickle.dumps(ukr_rus_cluster_followers)
        ukr_ukr_cluster_pickled = pickle.dumps(ukr_ukr_cluster_followers)

```

```
[22]: if not LOAD_FOLLOWERS:
        print("Writing pickled data to a file...")

        with open(FOLLOWERS_DIR + 'rus_cluster_followers.pickle', 'wb') as f:
            f.write(rus_cluster_followers_pickled)

        with open(FOLLOWERS_DIR + 'ukr_eng_cluster_followers.pickle', 'wb') as f:
            f.write(ukr_eng_cluster_followers_pickled)

        with open(FOLLOWERS_DIR + 'ukr_rus_cluster_followers.pickle', 'wb') as f:
            f.write(ukr_rus_cluster_pickled)

        with open(FOLLOWERS_DIR + 'ukr_ukr_cluster_followers.pickle', 'wb') as f:
            f.write(ukr_ukr_cluster_pickled)

```

Writing pickled data to a file...

```
[23]: # NOTE: causes recursion depth exceeded problem

if LOAD_FOLLOWERS:

    with open(FOLLOWERS_DIR + 'rus_cluster_followers.pickle', \
              'rb') as f:
        rus_cluster_followers = pickle.load(f)

    with open(FOLLOWERS_DIR + 'ukr_eng_cluster_followers' \
              '.pickle', 'rb') as f:

```

```

ukr_eng_cluster_followers = pickle.load(f)

with open(FOLLOWERS_DIR + 'ukr_rus_cluster_followers' \
          '.pickle', 'rb') as f:
    ukr_rus_cluster_followers = pickle.load(f)

with open(FOLLOWERS_DIR + 'ukr_ukr_cluster_followers' \
          '.pickle', 'rb') as f:
    ukr_ukr_cluster_followers = pickle.load(f)

```

[24]: `print(ukr_eng_cluster_followers)`

```

[ (('@dmitry_gordon', 1334400780), [<User id=1599295496643919872 name=
username=01094895832>, <User id=1598318795512336387 name=CasualReader
username=_Casual_Reader_>, <User id=1532500969371598862 name=Natasha Milman
username=milman_natasha>, <User id=1552729759725010946 name=Marina Andrushchenko
username=Marina35162778>, <User id=1577327477675941890 name=Ihor
username=Ihor_Sh_UA>, <User id=1598848661542379520 name=
username=Denis87078157>, <User id=1582089127419428866 name=Iryna
username=Iryna0903>, <User id=1599278632165425152 name=
username=Evgenj06891204>, <User id=2634282462 name=
username=SvetycikS>, <User id=1480953831634747400 name=Vitautas
username=VitautasAlushta>]], (('@SvobodaRadio', 47562921), [<User
id=790171261846581253 name=Marcel -Emile YAO username=marcel_emile>, <User
id=1599304219714109447 name=
username=Viktor41660377>, <User
id=1599303977031581696 name=Pasko Alex username=PaskoAlex2>, <User id=3055603197
name=Anna koma username=KomaAnna>, <User id=1599301658797916160 name=
username=Ramil19631404>, <User id=1599301440131973121 name=
username=gurev691969>, <User id=1522864464491692035 name=
username=KatorginaG>, <User id=1599302061627154433 name=
username=SohruhHajdarov>, <User id=1598318795512336387 name=CasualReader
username=_Casual_Reader_>, <User id=1599298105186344961 name=Elena Boyko
username=boooykot>]])]

```

[25]: `'''`
`print(type(followers[('@minregion_ua', 3333475643)][0]))`
`print(followers[('@minregion_ua', 3333475643)][0].name)`
`print(followers[('@minregion_ua', 3333475643)][0].id)`
`'''`
`type(rus_cluster_followers[0])`

[25]: tuple

[26]: `CLUSTER_IDX = 0`
`FOLLOWER_IDX = 1`
`rus_cluster = {}`

```
ukr_eng_cluster = {}
ukr_rus_cluster = {}
ukr_ukr_cluster = {}
```

```
[27]: def cluster_to_dict(cluster_list):

    cluster_dfs = {}

    for cluster in cluster_list:
        follower_names = [follower.name for follower in cluster[FOLLOWER_IDX]]
        follower_ids = [follower.id for follower in cluster[FOLLOWER_IDX]]
        followers_data = {
            'username': follower_names,
            'user_id': follower_ids
        }
        cluster_dfs[cluster[CLUSTER_IDX]] = pd.DataFrame(followers_data)

    return cluster_dfs
```

```
[28]: rus_cluster = cluster_to_dict(rus_cluster_followers)
ukr_eng_cluster = cluster_to_dict(ukr_eng_cluster_followers)
ukr_rus_cluster = cluster_to_dict(ukr_rus_cluster_followers)
ukr_ukr_cluster = cluster_to_dict(ukr_ukr_cluster_followers)
```

```
[29]: # this function converts a list of cluster centers and their followers to csv
# files
CLUSTERS_DIR = "clusters/"

def clusters_to_files(clusters_df):
    for cluster_center, cluster_followers in clusters_df.items():
        cluster_csv = cluster_followers.to_csv()
        with open(CLUSTERS_DIR + cluster_center[CLUSTER_IDX][1:] + '_' +
↪str(cluster_center[FOLLOWER_IDX]) + '.csv',
            'w') as f:
            f.write(cluster_csv)
```

```
[30]: LOAD_CLUSTERS = True
```

```
[31]: if not LOAD_CLUSTERS:
    clusters_to_files(rus_cluster)
    clusters_to_files(ukr_eng_cluster)
```

```
[32]: russia_sources_rus
```

```
[32]: [('@itvru_news', 160881696),
      ('@ru_rbc', 269770723),
      ('@er_novosti', 152984812),
```



```
( '@rt_com', 64643056),
( '@medvedevrussia', 153812887),
( '@kremlinrussia', 158650448),
( '@rentvchannel', 200036850),
( '@vesti_news', 72525490),
( '@kpru', 40807205)]
```

```
[33]: if LOAD_CLUSTERS:
    print("Loading data for rus_cluster")
    rus_cluster = {}
    for cluster in russia_sources_rus:
        path_to_file = CLUSTERS_DIR + cluster[0][1:] + '_' + str(cluster[1]) + \
            '.csv'
        if os.path.exists(path_to_file):
            cluster_followers = pd.read_csv(path_to_file,
                                             usecols=['username', 'user_id'],
                                             dtype={'user_id': int})
            rus_cluster[cluster] = cluster_followers

    print("Loading data for ukr_eng_cluster")
    ukr_eng_cluster = {}
    for cluster in ukraine_sources_eng:
        path_to_file = CLUSTERS_DIR + cluster[0][1:] + '_' + str(cluster[1]) + \
            '.csv'
        if os.path.exists(path_to_file):
            cluster_followers = pd.read_csv(path_to_file,
                                             usecols=['username', 'user_id'],
                                             dtype={'user_id': int})
            ukr_eng_cluster[cluster] = cluster_followers

    print("Loading data for the ukr_rus_cluster")
    ukr_rus_cluster = {}
    for cluster in ukraine_sources_rus:
        path_to_file = CLUSTERS_DIR + cluster[0][1:] + '_' + str(cluster[1]) + \
            '.csv'
        if os.path.exists(path_to_file):
            cluster_followers = pd.read_csv(path_to_file,
                                             usecols=['username', 'user_id'],
                                             dtype={'user_id': int})
            ukr_rus_cluster[cluster] = cluster_followers

    print("Loading data for the ukr_ukr_cluster")
    ukr_ukr_cluster = {}
    for cluster in ukraine_sources_ukr:
        path_to_file = CLUSTERS_DIR + cluster[0][1:] + '_' + str(cluster[1]) + \
            '.csv'
        if os.path.exists(path_to_file):
```

```

cluster_followers = pd.read_csv(path_to_file,
                                usecols=['username', 'user_id'],
                                dtype={'user_id': int})
ukr_ukr_cluster[cluster] = cluster_followers

```

Loading data for rus_cluster
 Loading data for ukr_eng_cluster
 Loading data for the ukr_rus_cluster
 Loading data for the ukr_ukr_cluster

```
[34]: rus_cluster
```

```

[34]: {('@1tvru_news',
      160881696):
      username      user_id
0      jose filipe      19927296
1      lechantdesvoyelles      2258325268
2      SPEED NYC      2924392191
3      valera karaman      1599152454398099459
4      ivanrussia      1599147506654380036
..      ...
995  Zayniddin Xusniddino'v      1595878050276773889
996      NEOSHA NEOSHA      1542440663698100226
997      #Ziyod05      1595848999302479872
998      1595867951399600128
999      kas      1332061843

[1000 rows x 2 columns],
('@ru_rbc',
 269770723):
      username      user_id
0      1599163293746302976
1      Pletunoff      878179754804957184
2      Mila      1599157008330096640
3      Set Mortensen      1599155538507567104
4      sweetdreamslizz      1599154015195086848
..      ...
995      Tofig Davudov      1597577365365858307
996  Natallia Chodosowska      1597572648267030528
997      1597574448831516672
998      Timur Isakov      1597571656968445954
999      1597566730188062720

[1000 rows x 2 columns]}

```

Visualizing Connections as a Graph

```
[35]: ukr_rus_cluster
```

```
[35]: {('@dmitry_gordon',
      1334400780):
      username      user_id
0      Kryp4ek  1476613048018382850
1      1599163293746302976
2      1599155737766281218
3      stieroglif      492418187
4      1599150982625218560
..      ...
994      osokinnnnnnnn  1531585954607161347
995      tterewew      926760353878253568
996      Artur Kazarian  1595900167667990530
997      1576624669863591938
998      801410400

[999 rows x 2 columns],
('@SvobodaRadio',
 47562921):
      username      user_id
0      Denis  1599152874797371393
1      Žabba Blue  1599160775054135305
2      Andrius      983390142
3      1599160703394480128
4      Vennikov Elena  1528909651936481282
..      ...
995      Pečka  1593594577474174976
996      Roman Gabisonia  1597975081472458755
997      AntonnY195  1151908574697861120
998      62536997
999      1597201905238818818

[1000 rows x 2 columns]}
```

```
[36]: import networkx as nx
```

```
[37]: print("Creating a DataFrame containing the complete graph")

GROUP_ID_MAP = {
    "rus_cluster": 0,
    "ukr_eng_cluster": 1,
    "ukr_rus_cluster": 2,
    "ukr_ukr_cluster": 3
}

global_graph_pd = pd.DataFrame(columns=['username', 'user_id',
                                       'cluster_name', 'cluster_id',
                                       'cluster_follow_count', 'group_id'])
```

Creating a DataFrame containing the complete graph

```
[38]: def add_cluster_to_global_graph(cluster, group_name, graph_pd):
    curr_cluster_df = pd.DataFrame()
    for cluster_center, cluster_followers in cluster.items():
        curr_cluster_df = cluster_followers.copy(deep=True)
        curr_cluster_df.insert(2, "cluster_name", cluster_center[0])
        curr_cluster_df.insert(3, "cluster_id", int(cluster_center[1]))
        curr_cluster_df.insert(4, "cluster_follow_count",
                               get_user_follower_count(cluster_center[1]),
                               True)
        curr_cluster_df.insert(5, "group_id", GROUP_ID_MAP[group_name])
    return pd.concat([graph_pd, curr_cluster_df])
```

```
[39]: def save_image(img, img_name):
    IMG_DIR = "visualizations/"
    plt.imsave(IMG_DIR + img_name, img)
```

```
[40]: global_graph_pd = global_graph_pd.iloc[0:0]
global_graph_pd = add_cluster_to_global_graph(rus_cluster, "rus_cluster",
↪global_graph_pd)
global_graph_pd = add_cluster_to_global_graph(ukr_rus_cluster,
↪"ukr_rus_cluster",
                                         global_graph_pd)
```

```
[41]: print(global_graph_pd.size)
print(set(global_graph_pd['cluster_name']))
global_graph_pd.head()
print(global_graph_pd.size)
```

```
12000
{'@SvobodaRadio', '@ru_rbc'}
12000
```

```
[ ]: # TODO: VISUALIZE GROUPS AS DIFFERENT AXIS ON A 3D SCATTER PLOT IN PLOTLY
```

```
[55]: global_graph_pd = global_graph_pd.astype({
    "username" : str,
    "user_id" : int,
    "cluster_name" : str,
    "cluster_id" : int,
    "cluster_follow_count" : int,
    "group_id" : int
})
global_graph_pd_columns = list(global_graph_pd.columns)
print(global_graph_pd.dtypes)
print(global_graph_pd_columns)
print(global_graph_pd.head())
```

```

username          object
user_id           int64
cluster_name      object
cluster_id        int64
cluster_follow_count int64
group_id          int64
dtype: object
['username', 'user_id', 'cluster_name', 'cluster_id', 'cluster_follow_count',
'group_id']

```

	username	user_id	cluster_name	cluster_id \
0	1599163293746302976	@ru_rbc	269770723	
1	Pletunoff	878179754804957184	@ru_rbc	269770723
2	Mila	1599157008330096640	@ru_rbc	269770723
3	Set Mortensen	1599155538507567104	@ru_rbc	269770723
4	sweetdreamslizz	1599154015195086848	@ru_rbc	269770723

	cluster_follow_count	group_id
0	535342	0
1	535342	0
2	535342	0
3	535342	0
4	535342	0

```

[56]: print("Constructing NetworkX graph")

# what if you store all attributes as edge attributes?
G = nx.from_pandas_edgelist(global_graph_pd, source='username',
                           target='cluster_name', edge_attr =_
                           ↪global_graph_pd.columns,
                           create_using=nx.DiGraph())

pos = nx.spring_layout(G)

```

Constructing NetworkX graph

```

[58]: print("Visualizing a small subset of connections...")

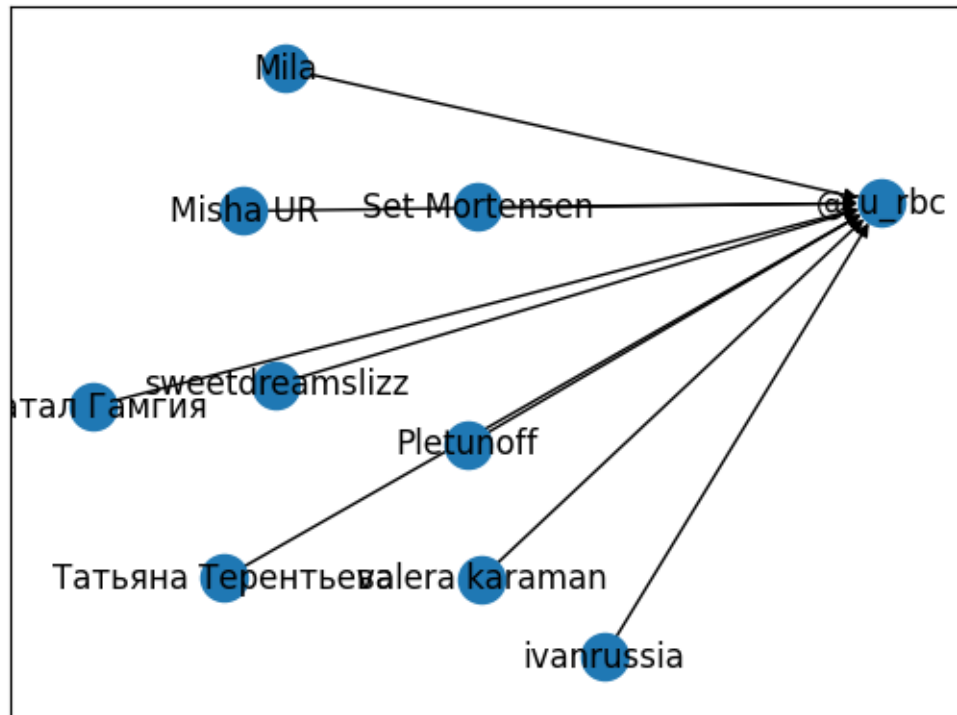
subgraph = G.subgraph(list(G.nodes)[:10])

subgraph_pos = nx.spring_layout(subgraph)

nx.draw_networkx(subgraph, subgraph_pos)

```

Visualizing a small subset of connections...



Network Analysis Questions:

1. How many connections are overlapping within each given group. (e.g. how many people following SWJ also follow NYT)
2. How many overlaps are there in between clusters from different groups?
3. Do nodes cluster into tightly connected groups?

Network Visualization Questions

3. Visualize the connections in pretty way
4. Visualize Cluster sizes
5. Visualize groups by color coding them

Sentiment Analysis Questions

1. Word Cloud: what are people within each group discussing (Use entity recognition)?
2. Tag Cloud: who are people within each group discussing?
3. What sentiment do mentions within each groups have?
4. What is a general Twitter sentiment on this topic (Can use Ukraine dataset)

Visualization Techniques that can be leveraged

1. Network Visualization
2. Coloring nodes

3. Coloring Connections
4. Size of the nodes
5. Shape of the nodes

```
[59]: def get_edge_attributes(graph, attr_list):
    KEY = 0
    VALUE = 1
    edge_attrs = {}
    for attr in attr_list:
        edges_attribute = nx.get_edge_attributes(graph, attr)
        for edge_attr in edges_attribute.items():
            if edge_attr[KEY] in edge_attrs.keys():
                edge_attr_values = edge_attrs[edge_attr[KEY]]
                edge_attr_values.append(edge_attr[VALUE])
                edge_attrs[edge_attr[KEY]] = edge_attr_values
            else:
                edge_attrs[edge_attr[KEY]] = [edge_attr[VALUE]]
    return edge_attrs

edge_attrs = get_edge_attributes(subgraph, global_graph_pd_columns)

print(edge_attrs)
```

```
{('      ', '@ru_rbc'): ['      ', 1599163293746302976, '@ru_rbc',
269770723, 535342, 0], ('Set Mortensen', '@ru_rbc'): ['Set Mortensen',
1599155538507567104, '@ru_rbc', 269770723, 535342, 0], ('Misha UR', '@ru_rbc'):
['Misha UR', 1599145502104330240, '@ru_rbc', 269770723, 535342, 0], ('valera
karaman', '@ru_rbc'): ['valera karaman', 1599152454398099459, '@ru_rbc',
269770723, 535342, 0], ('ivanrussia', '@ru_rbc'): ['ivanrussia',
1599147506654380036, '@ru_rbc', 269770723, 535342, 0], ('      ',
'@ru_rbc'): ['      ', 1599150059827896322, '@ru_rbc', 269770723,
535342, 0], ('Pletunoff', '@ru_rbc'): ['Pletunoff', 878179754804957184,
'@ru_rbc', 269770723, 535342, 0], ('sweetdreamslizz', '@ru_rbc'):
['sweetdreamslizz', 1599154015195086848, '@ru_rbc', 269770723, 535342, 0],
('Mila', '@ru_rbc'): ['Mila', 1599157008330096640, '@ru_rbc', 269770723, 535342,
0]}
```

```
[60]: print("Running analysis on the network...")

# https://subscription.packtpub.com/book/big-data-and-business-intelligence/
# ↪9781789955316/7

G.edges
```

Running analysis on the network...

```

[60]: OutEdgeView([('      ', '@ru_rbc'), ('Pletunoff', '@ru_rbc'), ('Mila',
'@ru_rbc'), ('Set Mortensen', '@ru_rbc'), ('sweetdreamslizz', '@ru_rbc'),
('valera karaman', '@ru_rbc'), ('Misha UR', '@ru_rbc'), ('      ',
'@ru_rbc'), ('ivanrussia', '@ru_rbc'), ('      ', '@ru_rbc'), ('Kostya',
'@ru_rbc'), ('CasualReader', '@ru_rbc'), ('      ', '@ru_rbc'),
('      ', '@SvobodaRadio'), ('      ', '@ru_rbc'), ('Alex',
'@ru_rbc'), ('Alex', '@SvobodaRadio'), ('      ', '@ru_rbc'), ('      ',
'@ru_rbc'), ('      ', '@SvobodaRadio'), ('Gulia', '@ru_rbc'), ('Surush',
'@ru_rbc'), ('Almaz Makhmut', '@ru_rbc'), ('I      ', '@ru_rbc'), ('I      ',
'@SvobodaRadio'), ('Zayniddin Hasanov', '@ru_rbc'), ('Zayniddin Hasanov',
'@SvobodaRadio'), ('      ', '@ru_rbc'), ('Viliana Titova',
'@ru_rbc'), ('      ', '@ru_rbc'), ('Marina S.', '@ru_rbc'), ('Inter Studio',
'@ru_rbc'), ('      ', '@ru_rbc'), ('Sherzod Akhmedov', '@ru_rbc'),
('      ', '@ru_rbc'), ('      ', '@SvobodaRadio'),
('Aleksandr', '@ru_rbc'), ('Aleksandr', '@SvobodaRadio'), ('      ', '@ru_rbc'),
('      ', '@SvobodaRadio'), ('      ', '@ru_rbc'), ('Dmitrii Smirnov',
'@ru_rbc'), ('      ', '@ru_rbc'), ('WA', '@ru_rbc'), ('      ',
'@ru_rbc'), ('      ', '@ru_rbc'), ('Ronin Barnes', '@ru_rbc'),
('      ', '@ru_rbc'), ('      ', '@ru_rbc'), ('      ',
'@SvobodaRadio'), ('john roger', '@ru_rbc'), ('Yashar Ismaylov', '@ru_rbc'),
('      ', '@ru_rbc'), ('samsara', '@ru_rbc'), ('Ru.by', '@ru_rbc'),
('Dmitry Bazekin', '@ru_rbc'), ('Zaira', '@ru_rbc'), ('Galina', '@ru_rbc'),
('Galina', '@SvobodaRadio'), ('      ', '@ru_rbc'), ('      ',
'@SvobodaRadio'), ('Babek Al-Khalili', '@ru_rbc'), ('Vladimir', '@ru_rbc'),
('Vladimir', '@SvobodaRadio'), ('ETHEREUM_BOSS', '@ru_rbc'), ('ETHEREUM_BOSS',
'@SvobodaRadio'), ('      ', '@ru_rbc'), ('Rodion', '@ru_rbc'),
('      ', '@ru_rbc'), ('      ', '@ru_rbc'), ('      ',
'@ru_rbc'), ('Garnik Avagyan', '@ru_rbc'), ('Garnik Avagyan', '@SvobodaRadio'),
('      ', '@ru_rbc'), ('Vit Xak', '@ru_rbc'), ('      ',
'@ru_rbc'), ('      ', '@ru_rbc'), ('      ', '@ru_rbc'), ('Lari Sergiu',
'@ru_rbc'), ('Ralyube', '@ru_rbc'), ('      ', '@ru_rbc'), ('      ',
'@SvobodaRadio'), ('      ', '@ru_rbc'), ('Danny Moore',
'@ru_rbc'), ('Hassan Riaz', '@ru_rbc'), ('      ', '@ru_rbc'),
('Gulliver007', '@ru_rbc'), ('Elyor Elyor', '@ru_rbc'), ('      ',
'@ru_rbc'), ('      ', '@ru_rbc'), ('IVAN LOBYNICHEV', '@ru_rbc'), ('      ',
'@ru_rbc'), ('Haichuan gen', '@ru_rbc'), ('N.V.', '@ru_rbc'),
('dionisovich', '@ru_rbc'), ('Balloon Brothers', '@ru_rbc'), ('Qizzbo',
'@ru_rbc'), ('      ', '@ru_rbc'), ('      ', '@ru_rbc'), ('      ',
'@ru_rbc'), ('      ', '@ru_rbc'), ('      ', '@ru_rbc'),
('      ', '@SvobodaRadio'), ('Alexander Dageron', '@ru_rbc'), ('      ',
'@ru_rbc'), ('A      ', '@ru_rbc'), ('      ',
'@ru_rbc'), ('      ', '@SvobodaRadio'), ('      ', '@ru_rbc'),
('      ', '@ru_rbc'), ('mutavali', '@ru_rbc'), ('Bethany Kimmet',
'@ru_rbc'), ('Jazzman603', '@ru_rbc'), ('      ', '@ru_rbc'), ('      ',
'@SvobodaRadio'), ('rt', '@ru_rbc'), ('Anton Baukov', '@ru_rbc'), ('Anton
Baukov', '@SvobodaRadio'), ('Genadiy Babitski', '@ru_rbc'), ('Genadiy Babitski',
'@SvobodaRadio'), ('Slava Pivnyuk', '@ru_rbc'), ('      ', '@ru_rbc'),

```


('Parker', '@ru_rbc'), ('one second', '@ru_rbc'), ('jek dingo', '@ru_rbc'),
 ('Akmal Juraev', '@ru_rbc'), ('lil lulil', '@ru_rbc'), ('lil lulil',
 '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'),
 ('', '@SvobodaRadio'), ('', '@ru_rbc'),
 ('memento7mori#.hft (,)', '@ru_rbc'), ('nan', '@ru_rbc'), ('',
 '@ru_rbc'), ('Svetlana Ionova', '@ru_rbc'), ('ANDRIIPILLOT', '@ru_rbc'),
 ('', '@ru_rbc'), ('', '@ru_rbc'), ('Maxer Warzone Mobile',
 '@ru_rbc'), ('Frederick Taxo', '@ru_rbc'), ('', '@ru_rbc'), ('',
 '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'),
 ('', '@ru_rbc'), ('Ilya Romanov', '@ru_rbc'), ('',
 '@ru_rbc'), ('', '@SvobodaRadio'), ('', '@ru_rbc'), ('
 ', '@ru_rbc'), ('Fanil Valirov', '@ru_rbc'), ('',
 '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('
 ', '@ru_rbc'), ('', '@SvobodaRadio'), ('
 ', '@ru_rbc'), ('', '@ru_rbc'), ('Cyberayo', '@ru_rbc'),
 ('Cyberayo', '@SvobodaRadio'), ('', '@ru_rbc'), ('
 ', '@ru_rbc'), ('', '@ru_rbc'), ('',
 '@ru_rbc'), ('Micah Spruill', '@ru_rbc'), ('Philipp Korneev', '@ru_rbc'),
 ('', '@ru_rbc'), ('YaN KalasH Kalash "LE RETOUR !"', '@ru_rbc'),
 ('Ruslan Almanov', '@ru_rbc'), ('', '@ru_rbc'), ('',
 '@ru_rbc'), ('Roman Tarabanov', '@ru_rbc'), ('Oksana', '@ru_rbc'), ('
 ', '@ru_rbc'), ('', '@SvobodaRadio'), ('Moise Kisonia',
 '@ru_rbc'), ('Roman Ko', '@ru_rbc'), ('Roman Ko', '@SvobodaRadio'), ('
 ', '@ru_rbc'), ('Visher', '@ru_rbc'), ('ABetkhemyan', '@ru_rbc'),
 ('Mirjalol', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'),
 ('NikKom', '@ru_rbc'), ('Cindy Miller', '@ru_rbc'), ('Pishumm2', '@ru_rbc'),
 ('', '@ru_rbc'), ('', '@SvobodaRadio'),
 ('Iskhakov Bulat', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'),
 ('Mike', '@ru_rbc'), ('Mike', '@SvobodaRadio'), ('',
 '@ru_rbc'), ('', '@ru_rbc'), ('The Spellsinger', '@ru_rbc'), ('',
 '@ru_rbc'), ('Mephistopheles', '@ru_rbc'), ('Muhammad P', '@ru_rbc'), ('Alena',
 '@ru_rbc'), ('', '@ru_rbc'), ('crouzz', '@ru_rbc'), ('',
 '@ru_rbc'), ('', '@SvobodaRadio'), ('Shaarani Lawan', '@ru_rbc'), ('Hamidjon
 Akbarif', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Katarina
 Maran', '@ru_rbc'), ('borzoi57', '@ru_rbc'), ('Alexey Gordeyko', '@ru_rbc'),
 ('Ivan Litvinov', '@ru_rbc'), ('', '@ru_rbc'),
 ('dasigu3(,)', '@ru_rbc'), ('Aleksandr S', '@ru_rbc'), ('
 ', '@ru_rbc'), ('', '@ru_rbc'), ('garjou56(,)',
 '@ru_rbc'), ('Vladimir Tsoy', '@ru_rbc'), ('', '@ru_rbc'), ('
 ', '@SvobodaRadio'), ('CarrieRobin', '@ru_rbc'), ('JasperMaggie', '@ru_rbc'),
 ('', '@ru_rbc'), ('', '@ru_rbc'), ('
 ', '@ru_rbc'), ('METE AYMAYA', '@ru_rbc'), ('No Zombie', '@ru_rbc'), ('No
 Zombie', '@SvobodaRadio'), ('', '@ru_rbc'), ('Ahmadov
 Shodmon', '@ru_rbc'), ('Ahmadov Shodmon', '@SvobodaRadio'), ('Alice',
 '@ru_rbc'), ('', '@ru_rbc'), ('',
 '@SvobodaRadio'), ('Natal'ya Kudryavtseva', '@ru_rbc'), ('Natal'ya
 Kudryavtseva', '@SvobodaRadio'), ('', '@ru_rbc'), ('Akmaral

Batalova', '@ru_rbc'), ('', '@ru_rbc'), ('lotte rocio', '@ru_rbc'), ('Jim Munden', '@ru_rbc'), ('Abdulwahid Afzaly', '@ru_rbc'), ('lanyanting', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('', '@ru_rbc'), ('Aziz', '@ru_rbc'), ('Aziz', '@SvobodaRadio'), ('Léonidas', '@ru_rbc'), ('Michael Ryzhichenko', '@ru_rbc'), ('Tatyana Letvitskaya', '@ru_rbc'), ('Tatyana Letvitskaya', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Dadojon Bakosov', '@ru_rbc'), ('Yury Meshkov', '@ru_rbc'), ('Maryna', '@ru_rbc'), ('Maryna', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('d.flawyou', '@ru_rbc'), ('Natali', '@ru_rbc'), ('Natali', '@SvobodaRadio'), ('Celeste Iannacone', '@ru_rbc'), ('bngrkzstr', '@ru_rbc'), ('Svetlana Karpekina', '@ru_rbc'), ('', '@ru_rbc'), ('Laurie Schneider', '@ru_rbc'), ('Dr. Shore | iPhotoStreet.com', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Igor', '@ru_rbc'), ('Igor', '@SvobodaRadio'), ('thechosenone', '@ru_rbc'), ('Aphelios', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Olga', '@ru_rbc'), ('Olga', '@SvobodaRadio'), ('', '@ru_rbc'), ('mustafeahmedmohamud@gmail.com', '@ru_rbc'), ('IreneSt', '@ru_rbc'), ('Ena Meyer', '@ru_rbc'), ('Rasim Sk', '@ru_rbc'), ('Inoagentiy', '@ru_rbc'), ('Uliana', '@ru_rbc'), ('Eddie', '@ru_rbc'), ('Arsenii Shcherba', '@ru_rbc'), ('', '@ru_rbc'), ('Juri Kusnir', '@ru_rbc'), ('dont_smoke_here_please', '@ru_rbc'), ('Waldemar', '@ru_rbc'), ('Richard Kara', '@ru_rbc'), ('', '@ru_rbc'), ('Muhammad Anvarov', '@ru_rbc'), ('Muhammad Anvarov', '@SvobodaRadio'), ('Sergij Lazebnyk', '@ru_rbc'), ('Askhabull', '@ru_rbc'), ('Askhabull', '@SvobodaRadio'), ('Ilham Ahmad', '@ru_rbc'), ('Godella Petty', '@ru_rbc'), ('Heather Noland', '@ru_rbc'), ('Gintautas Noreika', '@ru_rbc'), ('Elmirhesenli', '@ru_rbc'), ('Elmirhesenli', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('NIROLAI', '@ru_rbc'), ('Cars', '@ru_rbc'), ('', '@ru_rbc'), ('Serous Pavel', '@ru_rbc'), ('', '@ru_rbc'), ('Andrey Makarov', '@ru_rbc'), ('Dasa Tusco', '@ru_rbc'), ('Disappear', '@ru_rbc'), ('Ivan', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('NatalyaS.', '@ru_rbc'), ('Adrian', '@ru_rbc'), ('', '@ru_rbc'), ('Assel', '@ru_rbc'), ('Assel', '@SvobodaRadio'), ('Andrey', '@ru_rbc'), ('Andrey', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Jabrail', '@ru_rbc'), ('Lera', '@ru_rbc'), ('sergio Buskets', '@ru_rbc'), ('Mi Tan', '@ru_rbc'), ('', '@ru_rbc'), ('Elena', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Eldar', '@ru_rbc'), ('', '@ru_rbc'), ('History', '@ru_rbc'), ('Elnara safronova', '@ru_rbc'), ('REKCIK FORTNITE', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('alex go', '@ru_rbc'), ('alex go', '@SvobodaRadio'), ('Iacubenco Vitalii', '@ru_rbc'), ('Iacubenco Vitalii', '@SvobodaRadio'), ('A Ab', '@ru_rbc'), ('shohjahon abdujalilov', '@ru_rbc'), ('G.O.A.T.', '@ru_rbc'), ('Sabyr Pernebay', '@ru_rbc'), ('', '@ru_rbc'), ('Tina Cummins', '@ru_rbc'), ('Komiljon Akhrorov', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Parvin ismatov', '@ru_rbc'), ('Parvin ismatov', '@SvobodaRadio'), ('', '@ru_rbc'), ('Chenxing', '@ru_rbc'),

(
'@ru_rbc'), ('@SvobodaRadio'), ('Igor Kch',
'@ru_rbc'), ('Igor Kch', '@SvobodaRadio'), ('Evelina Khmelevska', '@ru_rbc'),
(
'@ru_rbc'), ('@SvobodaRadio'), (
'@ru_rbc'), ('@ru_rbc'), ('Andrii', '@ru_rbc'), ('Igorrr',
'@ru_rbc'), ('anderson', '@ru_rbc'), ('Katiusha', '@ru_rbc'), (
'@ru_rbc'), ('@ru_rbc'), ('Dear God', '@ru_rbc'), ('Dear God',
'@SvobodaRadio'), ('Liudmila', '@ru_rbc'), ('zp', '@ru_rbc'), ('Rev lur',
'@ru_rbc'), ('Bahadır İnang Özkan', '@ru_rbc'), ('Maximus General', '@ru_rbc'),
(
'@ru_rbc'), ('@SvobodaRadio'), ('Elaman Toktoshev', '@ru_rbc'),
(
'@ru_rbc'), ('Iuliia', '@ru_rbc'), ('Iuliia',
'@SvobodaRadio'), ('Anastasiia', '@ru_rbc'), ('Susan Necochea', '@ru_rbc'),
(
'@ru_rbc'), ('Gari Siradekia', '@ru_rbc'), ('Diyor Pubg', '@ru_rbc'),
(
'@ru_rbc'), ('@SvobodaRadio'), (
'@ru_rbc'),
(
'@ru_rbc'), ('martin', '@ru_rbc'), ('martin',
'@SvobodaRadio'), (
'@ru_rbc'), (
'@ru_rbc'), ('@SvobodaRadio'), ('ninza', '@ru_rbc'), ('ninza', '@SvobodaRadio'), (
'@ru_rbc'), ('Jury Badulin', '@ru_rbc'), ('Jury Badulin',
'@SvobodaRadio'), (
'@ru_rbc'), ('Denis Novikov', '@ru_rbc'),
(
'@ru_rbc'), (
'@ru_rbc'), ('@SvobodaRadio'), (
'@ru_rbc'), ('Anne Johnson',
'@ru_rbc'), ('Avgii', '@ru_rbc'), (
'@ru_rbc'), (
'@ru_rbc'), ('@SvobodaRadio'), ('Anna', '@ru_rbc'), ('Anna', '@SvobodaRadio'), ('NYSOV',
'@ru_rbc'), (
'@ru_rbc'), ('Mykeys_123', '@ru_rbc'), ('Ira',
'@ru_rbc'), ('Vitalie', '@ru_rbc'), ('Vitalie', '@SvobodaRadio'), ('Nurlan',
'@ru_rbc'), (
'@ru_rbc'), ('Jen Jones', '@ru_rbc'), (
'@ru_rbc'), (
'@ru_rbc'), ('@SvobodaRadio'), ('Mama V',
'@ru_rbc'), ('Maximov Artem', '@ru_rbc'), ('Sergey Spak', '@ru_rbc'), (
'@ru_rbc'), ('Murad', '@ru_rbc'), ('Umida Usmonova', '@ru_rbc'), ('Rima
Sloutskaia', '@ru_rbc'), ('Rima Sloutskaia', '@SvobodaRadio'), ('Teddy Zamora',
'@ru_rbc'), ('Liberty Crew', '@ru_rbc'), ('Liberty Crew', '@SvobodaRadio'),
(
'@ru_rbc'), ('WilliamAstrid', '@ru_rbc'), ('Sergey
Krasnobaev', '@ru_rbc'), ('Sergey Krasnobaev', '@SvobodaRadio'), (
'@ru_rbc'), (
'@ru_rbc'), ('@SvobodaRadio'), (
'@ru_rbc'), (
'@ru_rbc'), ('@SvobodaRadio'), (
'@ru_rbc'), ('C', '@ru_rbc'), ('Baxtiyor Abdurasulov', '@ru_rbc'),
(
'@ru_rbc'), ('David Ekanger', '@ru_rbc'),
(
'@ru_rbc'), ('Idibek Abdiev', '@SvobodaRadio'), ('Azizbek',
'@ru_rbc'), ('Azizbek', '@SvobodaRadio'), (
'@ru_rbc'),
(
'@ru_rbc'), ('iljukis', '@ru_rbc'),
(
'@ru_rbc'), ('Militant Robots', '@ru_rbc'), ('Militant
Robots', '@SvobodaRadio'), ('Kkkkk Kkkkkk', '@ru_rbc'), (
'@ru_rbc'), ('Michail (L3,)', '@ru_rbc'), (
'@ru_rbc'),
(
'@ru_rbc'), ('@SvobodaRadio'), ('Andrey Vasilyev', '@ru_rbc'), ('SOUR
Adams', '@ru_rbc'), (
'@ru_rbc'), (
'@ru_rbc'), ('Nata Sara', '@ru_rbc'), (
'@ru_rbc'),
(
'@ru_rbc'), ('Operation Witchcraft', '@SvobodaRadio'),
(
'@ru_rbc'), ('d', '@ru_rbc'), (
'@ru_rbc'), (
'@ru_rbc'),

'@SvobodaRadio'), ('', '@ru_rbc'), ('Naila Rustamova', '@ru_rbc'), ('Naila Rustamova', '@SvobodaRadio'), ('Atabekian Lucine', '@ru_rbc'), ('Eduard Kovalenko', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Ilya', '@ru_rbc'), ('Vertex', '@ru_rbc'), ('Vertex', '@SvobodaRadio'), ('corsica', '@ru_rbc'), ('', '@ru_rbc'), ('Jurijs Ivanovs', '@ru_rbc'), ('', '@ru_rbc'), ('Veriko', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('@SvobodaRadio'), ('Vladimir Sapego', '@ru_rbc'), ('Nata', '@ru_rbc'), ('Nata', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Mira Kutilina', '@ru_rbc'), ('Ruslan Kurchev', '@ru_rbc'), ('zaebashka', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('A.Eskin', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Rz Rz', '@ru_rbc'), ('', '@ru_rbc'), ('Mygame', '@ru_rbc'), ('ELSANA TUJUH', '@ru_rbc'), ('Yerevan Armenia', '@ru_rbc'), ('', '@ru_rbc'), ('Muhammad', '@ru_rbc'), ('Muhammad', '@SvobodaRadio'), ('Natalya', '@ru_rbc'), ('Firuz Shukurov', '@ru_rbc'), ('Gintaras Miežanskas', '@ru_rbc'), ('shujo', '@ru_rbc'), ('shujo', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('red door', '@ru_rbc'), ('red door', '@SvobodaRadio'), ('Nosirov Dilshodxan', '@ru_rbc'), ('Ivan Shmelev', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Aquarius13271', '@ru_rbc'), ('Alimehdi Memmedov', '@ru_rbc'), ('', '@ru_rbc'), ('Onione Tamiko', '@ru_rbc'), ('Galina Grigoryan', '@ru_rbc'), ('Ekaterina', '@ru_rbc'), ('Ekaterina', '@SvobodaRadio'), ('600022', '@ru_rbc'), ('', '@ru_rbc'), ('ixdaya', '@ru_rbc'), ('', '@ru_rbc'), ('Franciscosantos', '@ru_rbc'), ('Franciscosantos', '@SvobodaRadio'), ('Luccas Viana nunes', '@ru_rbc'), ('Daniel Ward', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Grecal'o', '@ru_rbc'), ('Grecal'o', '@SvobodaRadio'), ('Samandar Mirzabaev', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Arthur Vali', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Maxim Kazhyna', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Fayzullo', '@ru_rbc'), ('a.pogodin.xo', '@ru_rbc'), ('Karmela Hodos', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Denis', '@ru_rbc'), ('Denis', '@SvobodaRadio'), ('', '@ru_rbc'), ('Angry Cat', '@ru_rbc'), ('Angry Cat', '@SvobodaRadio'), ('', '@ru_rbc'), ('mordosovyura', '@ru_rbc'), ('Lina G', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Klara Bek', '@ru_rbc'), ('Klara Bek', '@SvobodaRadio'), ('', '@ru_rbc'), ('Mansurali jamoliddinov', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Avazbek', '@ru_rbc'), ('Avazbek', '@SvobodaRadio'), ('', '@ru_rbc'), ('arkel karkel', '@ru_rbc'), ('Lisa Joseph', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Juia

(' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@SvobodaRadio'), ('Victor Armishev', '@ru_rbc'), (' o ', '@ru_rbc'), ('Angs One', '@ru_rbc'), ('Danielle Rose', '@ru_rbc'), ('Sebastian', '@ru_rbc'), ('Masha', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), ('Eduard Kolesnikov', '@ru_rbc'), ('Eduard Kolesnikov', '@SvobodaRadio'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@SvobodaRadio'), ('Pohjola', '@ru_rbc'), ('Pohjola', '@SvobodaRadio'), ('Alizamin', '@ru_rbc'), ('Alizamin', '@SvobodaRadio'), ('Agamergen Isangulyyew', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), ('resmon', '@ru_rbc'), ('lazybuzzy', '@ru_rbc'), (' ', '@ru_rbc'), ('Begzod', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@SvobodaRadio'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@SvobodaRadio'), ('Denis Vilman', '@ru_rbc'), ('khlud', '@ru_rbc'), (' ', '@ru_rbc'), ('Aziz Khaydarov', '@ru_rbc'), ('Da Phantom Shtr', '@ru_rbc'), ('Da Phantom Shtr', '@SvobodaRadio'), (' ', '@ru_rbc'), ('Janelle Stephens', '@ru_rbc'), ('Arvydas', '@ru_rbc'), ('Arvydas', '@SvobodaRadio'), (' . . _ ', '@ru_rbc'), ('ruppinwin Odessa', '@ru_rbc'), ('Alina _Emoili', '@ru_rbc'), ('Alina _Emoili', '@SvobodaRadio'), ('ERIMBET', '@ru_rbc'), ('ERIMBET', '@SvobodaRadio'), ('Oxana', '@ru_rbc'), ('Oxana', '@SvobodaRadio'), ('Ismoil', '@ru_rbc'), (' ', '@ru_rbc'), ('Nursultan Baisakalov', '@ru_rbc'), ('Jez', '@ru_rbc'), ('Nadin', '@ru_rbc'), ('TK', '@ru_rbc'), (' ', '@ru_rbc'), ('Vvhale', '@ru_rbc'), (' ', '@ru_rbc'), ('Ciupaga', '@ru_rbc'), ('Zebo Xaydarova', '@ru_rbc'), ('Zebo Xaydarova', '@SvobodaRadio'), ('antapo', '@ru_rbc'), ('Sergey63', '@ru_rbc'), ('Al', '@ru_rbc'), ('Victoria Johnson', '@ru_rbc'), ('Lev', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), ('Kristy Smith', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@SvobodaRadio'), ('waymarkt', '@ru_rbc'), ('waymarkt', '@SvobodaRadio'), ('Abduganikhodja Bakhtiyorkhadjaev', '@ru_rbc'), ('Ranago', '@ru_rbc'), ('Mendsaikhan Purevkhaidav', '@ru_rbc'), ('Mendsaikhan Purevkhaidav', '@SvobodaRadio'), ('Bobir Akbarov', '@ru_rbc'), ('Aizhan Mambetalieva', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), ('Renato Calore', '@ru_rbc'), ('barbara', '@ru_rbc'), ('Emir Sarvi', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), ('Soffee', '@ru_rbc'), ('Zura Batiashvili', '@ru_rbc'), (' ', '@ru_rbc'), ('Rus Tam77', '@ru_rbc'), ('Rus Tam77', '@SvobodaRadio'), (' ', '@ru_rbc'), ('hacrimao', '@ru_rbc'), (' ', '@ru_rbc'), ('Svetlana Babinica', '@ru_rbc'), ('Rinata', '@ru_rbc'), ('Nadezhda Paul', '@ru_rbc'), (' ', '@ru_rbc'), ('alessandro michelozz Z Z Z', '@ru_rbc'), ('The freemason', '@ru_rbc'), ('VitAtlas', '@ru_rbc'), ('Guntars Harisovs', '@ru_rbc'), (' ', '@ru_rbc'), ('TLS CARGO', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), (' ', '@ru_rbc'), ('Pussey Crasher ', '@ru_rbc'), ('Viktor Larionov', '@ru_rbc'), ('Pasha Kim', '@ru_rbc'), (' ', '@ru_rbc'), ('grgbltv', '@ru_rbc'), (' ', '@ru_rbc'), ('cyclowalk', '@ru_rbc'), ('evgenii markelov', '@ru_rbc'), (' ', '@ru_rbc'),

'@ru_rbc'), ('lia', '@ru_rbc'), ('lido1942', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('la la la', '@ru_rbc'), ('Daniil', '@ru_rbc'), ('Dinorah Jackson', '@ru_rbc'), ('No Face No Name', '@ru_rbc'), ('20 22', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('Sherali Nabijonov', '@ru_rbc'), ('', '@ru_rbc'), ('PanzerKampfWagenVITigerHI', '@ru_rbc'), ('Viktors Dukalskis', '@ru_rbc'), ('sncdtyu8(,)', '@ru_rbc'), ('Igor Lihatsky', '@ru_rbc'), ('Maxim Vartanov', '@ru_rbc'), ('Juri Zotov', '@ru_rbc'), ('k007ig', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Vladimir Bibishev', '@ru_rbc'), ('Ramazi Giorgadze', '@ru_rbc'), ('', '@ru_rbc'), ('Arigato', '@ru_rbc'), ('Otabek', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Olga Prudnikova', '@ru_rbc'), ('Toni Pham', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Ottoscarceni', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('BATTHEAD', '@ru_rbc'), ('', '@ru_rbc'), ('Alex Punch', '@ru_rbc'), ('Artur Akopian', '@ru_rbc'), ('Shakhzod Akhrorov', '@ru_rbc'), ('Victoria Belgique', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Karen', '@ru_rbc'), ('Vasiliy Kalyuzhnyy', '@ru_rbc'), ('Robert Still', '@ru_rbc'), ('', '@ru_rbc'), ('Radoslav', '@ru_rbc'), ('Larisa Ushkanova', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Alex Sarkisov', '@ru_rbc'), ('Abdussalam Qayumov', '@ru_rbc'), ('jr.rafo', '@ru_rbc'), ('Julia Gilko/NOwar', '@ru_rbc'), ('', '@ru_rbc'), ('Crizis', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Kevin', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Julia Kurbatova', '@ru_rbc'), ('', '@ru_rbc'), ('Nadavonnizeque', '@ru_rbc'), ('dibley Ivanov', '@ru_rbc'), ('VönHell', '@ru_rbc'), ('Cenimunolusacen', '@ru_rbc'), ('', '@ru_rbc'), ('Daniel', '@ru_rbc'), ('MingwangSuodeTelusi', '@ru_rbc'), ('', '@ru_rbc'), ('Cudeezereratolo', '@ru_rbc'), ('Nodorinemid', '@ru_rbc'), ('Alex loukitch', '@ru_rbc'), ('Enoheusella', '@ru_rbc'), ('Alexander', '@ru_rbc'), ('Alexander', '@SvobodaRadio'), ('Hanna Grininger', '@ru_rbc'), ('ZloyDed', '@ru_rbc'), ('nehapesasaiston', '@ru_rbc'), ('firuzi Kintsurashvili', '@ru_rbc'), ('Willem van S.', '@ru_rbc'), ('Mobil. Centre', '@ru_rbc'), ('maladetidox', '@ru_rbc'), ('Miron Markus', '@ru_rbc'), ('Vincenza', '@ru_rbc'), ('Alexey Deryaga', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Niko Putra', '@ru_rbc'), ('lanuhematonnell', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('DonPadloKoksobar', '@ru_rbc'), ('Asgar', '@ru_rbc'), ('', '@ru_rbc'), ('Alexander Kim', '@ru_rbc'), ('Omowimozij', '@ru_rbc'), ('Hieininen', '@ru_rbc'), ('erefothanodu', '@ru_rbc'), ('Ratanitut', '@ru_rbc'), ('', '@ru_rbc'), ('Shireddin aslanov', '@ru_rbc'), ('', '@ru_rbc'), ('Nuhonotunuther', '@ru_rbc'), ('', '@ru_rbc'), ('hemalepoma', '@ru_rbc'), ('tedananalob', '@ru_rbc'), ('', '@ru_rbc'), ('Aigul Xab', '@ru_rbc'), ('Tefijorina', '@ru_rbc'), ('Potapova

Natalia', '@ru_rbc'), ('', '? ', '@ru_rbc'), ('', '@ru_rbc'), ('WorldWatcher', '@ru_rbc'), ('hitorunonetorow', '@ru_rbc'), ('Sairan Jarboldu', '@ru_rbc'), ('Returohalir', '@ru_rbc'), ('Tehapupesefetac', '@ru_rbc'), ('tesendedagog', '@ru_rbc'), ('', '@ru_rbc'), ('Wehandewem', '@ru_rbc'), ('Inna Konnova', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Adl', '@ru_rbc'), ('Cryptoexplorer0711', '@ru_rbc'), ('Nicole Ruiz', '@ru_rbc'), ('', '@ru_rbc'), ('Aleksy Ermolaev', '@ru_rbc'), ('nenoppiteth', '@ru_rbc'), ('Hohatetoululucu', '@ru_rbc'), ('Mikhail', '@ru_rbc'), ('ZARINA', '@ru_rbc'), ('', '@ru_rbc'), ('deremozowoono', '@ru_rbc'), ('Drew', '@ru_rbc'), ('Rasbora', '@ru_rbc'), ('lolope', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Nickolas Sarkisyan', '@ru_rbc'), ('()', '@ru_rbc'), ('Natalia Zarechneva', '@ru_rbc'), ('Helen Mikhailovskaya', '@ru_rbc'), ('Sad E', '@ru_rbc'), ('', '@ru_rbc'), ('Merab M', '@ru_rbc'), ('Utounubeisaf', '@ru_rbc'), ('nowanofaiqiduw', '@ru_rbc'), ('motyA', '@ru_rbc'), ('rehiopejeanuni', '@ru_rbc'), ('', '@ru_rbc'), ('Stan', '@ru_rbc'), ('', '@ru_rbc'), ('rustam', '@ru_rbc'), ('Vera Smirnova', '@ru_rbc'), ('', '@ru_rbc'), ('Vasiliy Bichkov', '@ru_rbc'), ('maks kaverin', '@ru_rbc'), ('kirill krivoshapkin', '@ru_rbc'), ('', '@ru_rbc'), ('IXTIYOR ODILOV', '@ru_rbc'), ('Knox Knowledge', '@ru_rbc'), ('Consulat Onorific', '@ru_rbc'), ('ELN DEV', '@ru_rbc'), ('Max', '@ru_rbc'), ('Sophia Clarence', '@ru_rbc'), ('', '@ru_rbc'), ('Munchie', '@ru_rbc'), ('Tetiana', '@ru_rbc'), ('', '@ru_rbc'), ('Dominum', '@ru_rbc'), ('love', '@ru_rbc'), ('Gg Ol', '@ru_rbc'), ('', '@ru_rbc'), ('Kostik228 (,)', '@ru_rbc'), ('Olev Kotka', '@ru_rbc'), ('Sergey', '@ru_rbc'), ('Sergey', '@SvobodaRadio'), ('maaax____', '@ru_rbc'), ('Mirelle', '@ru_rbc'), ('I.Wallau', '@ru_rbc'), ('Kevin Douglas', '@ru_rbc'), ('iOS' , '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Savin Serg', '@ru_rbc'), ('DecimalCrown', '@ru_rbc'), ('', '@ru_rbc'), ('Su', '@ru_rbc'), ('', '@ru_rbc'), ('molnia18 games', '@ru_rbc'), ('', '@ru_rbc'), ('danny costa', '@ru_rbc'), ('Aston Astonzoda', '@ru_rbc'), ('', '@ru_rbc'), ('Kote Gelashvili', '@ru_rbc'), ('Dmitriy Kalashnikov', '@ru_rbc'), ('Natasha Touzovskaia', '@ru_rbc'), ('028 ', '@ru_rbc'), ('', '@ru_rbc'), ('Dmitri Sm', '@ru_rbc'), ('', '@ru_rbc'), ('Gennadii', '@ru_rbc'), ('ConorMoore', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Laky', '@ru_rbc'), ('Nikolay Kaplin', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('VAIN', '@ru_rbc'), ('Shadow Hide', '@ru_rbc'), ('vasas István', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('(SEA)', '@ru_rbc'), ('Munteanu Jasmin', '@ru_rbc'), ('', '@ru_rbc'), ('Kulturamultura', '@ru_rbc'), ('Phantom 47', '@ru_rbc'), ('', '@ru_rbc'), ('', '@SvobodaRadio'), ('surald', '@ru_rbc'), ('Lucas Daniel Olivera', '@ru_rbc'), ('', '@ru_rbc'), ('Dmitriy Dumkin', '@ru_rbc'), ('Arih', '@ru_rbc'), ('cc', '@ru_rbc'), ('Nikolai Grigoriev', '@ru_rbc'), ('Liliya', '@ru_rbc'), ('

', '@ru_rbc'), ('', '@ru_rbc'), ('Kurban Mamedov', '@ru_rbc'), ('Harvey Dent', '@ru_rbc'), ('', '@ru_rbc'), ('Frank William Abagnale', '@ru_rbc'), ('Dumitru Rusu', '@ru_rbc'), ('', '@ru_rbc'), ('Muana', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('', '@ru_rbc'), ('Muhammed Riza', '@ru_rbc'), ('Tom Kerrigan', '@ru_rbc'), ('Stanislav', '@ru_rbc'), ('Stanislav', '@SvobodaRadio'), ('', '@ru_rbc'), ('lolkek', '@ru_rbc'), ('Tofig Davudov', '@ru_rbc'), ('Natallia Chodosowska', '@ru_rbc'), ('', '@ru_rbc'), ('Timur Isakov', '@ru_rbc'), ('', '@ru_rbc'), ('Žabba Blue', '@SvobodaRadio'), ('Andrius', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Vennikov Elena', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Alex Levin', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Giorgi Koba', '@SvobodaRadio'), ('Dooron Kaldarov', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Roman Green', '@SvobodaRadio'), ('Leqso', '@SvobodaRadio'), ('Tvoy Razum "(,)"', '@SvobodaRadio'), ('your', '@SvobodaRadio'), ('Nikolya 0311', '@SvobodaRadio'), ('Vera Grigoryeva', '@SvobodaRadio'), ('Nandy', '@SvobodaRadio'), ('Viltarin', '@SvobodaRadio'), ('Viacheslav', '@SvobodaRadio'), ('Grigori Draci', '@SvobodaRadio'), ('tolga yetil', '@SvobodaRadio'), ('Crypto', '@SvobodaRadio'), ('Ruslan', '@SvobodaRadio'), ('Tatiana Berhin', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('DesemberWiin00', '@SvobodaRadio'), ('Scorpion', '@SvobodaRadio'), ('Pavel', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Yury Mironenlo', '@SvobodaRadio'), ('jalmar pavlovec', '@SvobodaRadio'), ('Ordures à Bennes', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Geoger', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Sherzod Khamidullaev', '@SvobodaRadio'), ('nuonlithuania', '@SvobodaRadio'), ('(Yasha)', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Clarita Zartman', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Terloyeva', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Kats Yanina', '@SvobodaRadio'), ('leonroy', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Kolya Osten-Backen', '@SvobodaRadio'), ('littlerobbergirl', '@SvobodaRadio'), ('Mutribsho', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Ukrainian man', '@SvobodaRadio'), ('Muxammadjon Abdullaev', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Mohamed El alfy', '@SvobodaRadio'), ('Sarv', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Hein Ivar', '@SvobodaRadio'), ('gintoxs', '@SvobodaRadio'), ('sunwukong', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Andrey Blinov', '@SvobodaRadio'), ('', '@SvobodaRadio'),

'@SvobodaRadio'), ('Cherious', '@SvobodaRadio'), ('Armineh Grigorian',
 '@SvobodaRadio'), ('Vladimir Tartas uk', '@SvobodaRadio'), ('Maria',
 '@SvobodaRadio'), ('Provinsjules', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Farxod Norov', '@SvobodaRadio'), (' Sarah || \$TTC',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('maxsh',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Olcia Ostroushko', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('tranquiI', '@SvobodaRadio'), ('NoziroFF N',
 '@SvobodaRadio'), ('BAHAGIA ITU INDAH', '@SvobodaRadio'), ('Boris Burshtein',
 '@SvobodaRadio'), ('Boris.korg88', '@SvobodaRadio'), ('Mabel Rodriguez',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Eduard Martirosyan',
 '@SvobodaRadio'), ('Mihael Katsev', '@SvobodaRadio'), ('DAVID Beleg',
 '@SvobodaRadio'), ('aleksandra', '@SvobodaRadio'), ('Family nft',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Yamach Pubgm', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Yevgeny Khodos', '@SvobodaRadio'), ('Bakdaulet Talgat',
 '@SvobodaRadio'), ('Vyacheslav Serebryakov', '@SvobodaRadio'), ('Iftikhor
 Shraliev', '@SvobodaRadio'), ('Dmytro Ivanov', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Nikolay
 Grigorievich', '@SvobodaRadio'), ('Siti Munadiroh', '@SvobodaRadio'), ('Claudia
 Boyd', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('emil ibraev',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Alexandru Mihalachi', '@SvobodaRadio'), ('Zahir Esedov',
 '@SvobodaRadio'), ('Bobir Akhmedjanov', '@SvobodaRadio'), ('Volodumur Grisyuk',
 '@SvobodaRadio'), (' ----- ', '@SvobodaRadio'), ('Generation Y',
 '@SvobodaRadio'), ('A Riazanski', '@SvobodaRadio'), ('Şamsi Akhmed',
 '@SvobodaRadio'), ('Rus', '@SvobodaRadio'), ('AMIR ALI', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('MoniR HoseN', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Sergei Safin', '@SvobodaRadio'), ('VITALY EL',
 '@SvobodaRadio'), ('Donda', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Aslan Akshin', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Rasulov_04',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('History of the
 conspiracy /', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('LRBN', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Sasha
 NHK', '@SvobodaRadio'), ('Alla Danik', '@SvobodaRadio'), ('galina Sylima',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('stepan',
 '@SvobodaRadio'), ('Roman Juve', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Larrf', '@SvobodaRadio'), (' Username',
 '@SvobodaRadio'), ('Maryaka', '@SvobodaRadio'), ('Nurbek', '@SvobodaRadio'),
 (' , , ', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('Larissa Pitkenina', '@SvobodaRadio'), ('Muhammad
 54', '@SvobodaRadio'), ('Petro Protsiv', '@SvobodaRadio'), ('Jamshed',
 '@SvobodaRadio'), (' ! ', '@SvobodaRadio'), ('Abu Halas',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Valerii',

'@SvobodaRadio'), ('ion', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('Khatuna Mamisbedashvili', '@SvobodaRadio'), ('Genadi Kravets',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Mkrtich Mkrtchyan',
 '@SvobodaRadio'), ('evgen', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Murodjon Eshboyev',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('maruf',
 '@SvobodaRadio'), ('Renatas Katilius', '@SvobodaRadio'), ('Azamat Bilalov',
 '@SvobodaRadio'), ('Natalie', '@SvobodaRadio'), ('Datu Man', '@SvobodaRadio'),
 ('Hokim Valiev', '@SvobodaRadio'), ('alpaca1900', '@SvobodaRadio'), ('steven
 mazurskih', '@SvobodaRadio'), ('Evgeny', '@SvobodaRadio'), ('Fade Asaf',
 '@SvobodaRadio'), ('mishka', '@SvobodaRadio'), ('Anatoliy Buzdalin',
 '@SvobodaRadio'), ('Romero Shelby', '@SvobodaRadio'), ('Hoji Bekov',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Nunadievis', '@SvobodaRadio'), ('Dmitrij Krahn',
 '@SvobodaRadio'), ('Ainura Ermeeva', '@SvobodaRadio'), ('ida sveinhaus',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('King Lion',
 '@SvobodaRadio'), ('Ilyas Sarsenbaev', '@SvobodaRadio'), ('Ximin | ARG',
 '@SvobodaRadio'), ('Ivan Karasov', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('luuuuul 2', '@SvobodaRadio'), ('Samuel Felipe',
 '@SvobodaRadio'), ('Peach', '@SvobodaRadio'), ('Abdukodir Abdusattarov',
 '@SvobodaRadio'), ('cezar nichiforac', '@SvobodaRadio'), ('ulia lanna',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Ong Chong Yee
 (ichi1996.crypto)', '@SvobodaRadio'), ('Andrei Timof v', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('morik', '@SvobodaRadio'), ('olga mark', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Galinochka', '@SvobodaRadio'), ('Robert', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Pashtet_coins',
 '@SvobodaRadio'), ('Jonsnow', '@SvobodaRadio'), ('Nurdaulet Zharylkapov',
 '@SvobodaRadio'), ('Yana', '@SvobodaRadio'), ('Hasitha Leanage',
 '@SvobodaRadio'), ('Curt B.', '@SvobodaRadio'), ('Anastasia Nathans',
 '@SvobodaRadio'), ('Pirat', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Saidbek Abduraimov', '@SvobodaRadio'), ('D.V.Zion',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('DD', '@SvobodaRadio'),
 ('Moscow Apartments', '@SvobodaRadio'), ('Bakyt Tursaliyev', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Ellen
 Muniz', '@SvobodaRadio'), ('Irons', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Iulian Andriuta', '@SvobodaRadio'), ('Anton Belkin',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Jerry Rodriguez',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('John', '@SvobodaRadio'), ('Ilya Osterman', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('Boris Nedov', '@SvobodaRadio'),
 ('Vladimir Slavinsky', '@SvobodaRadio'), ('Johnny', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Aleksei Aleksei', '@SvobodaRadio'), ('

', '@SvobodaRadio'), ('Special Operative Rizzo', '@SvobodaRadio'),
 (' ', '@SvobodaRadio'), ('timer.ge', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Serge Ovcharenko', '@SvobodaRadio'), ('Manus
 Aeon', '@SvobodaRadio'), ('petrovich', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Nataliia', '@SvobodaRadio'), ('Katusiime', '@SvobodaRadio'),
 ('Bob67', '@SvobodaRadio'), ('Gala', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Denis Vladimirovich', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Neuron Shop', '@SvobodaRadio'), ('Irina Sokolova',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Ololosh Ololoew',
 '@SvobodaRadio'), ('SHYRIK NIKOLAEV', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Markian Kuzmowycz ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Vasil Zoll', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Kalicha Alikulova', '@SvobodaRadio'), ('giorgi',
 '@SvobodaRadio'), ('Sevara Sevara', '@SvobodaRadio'), ('Rimvydas Bajorinas',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('maRti', '@SvobodaRadio'),
 ('Kris Muntz', '@SvobodaRadio'), (' ', '@SvobodaRadio'),
 ('Alex&Alis', '@SvobodaRadio'), ('Daler', '@SvobodaRadio'), ('volodymir',
 '@SvobodaRadio'), ('Iurasco Sergiu', '@SvobodaRadio'), ('Alexey Vasilyev',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('SHAHBOZ RAKHMATOV', '@SvobodaRadio'), ('leftist drainer
 \U0001faa9', '@SvobodaRadio'), ('erno demjen', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Sino Khisainov',
 '@SvobodaRadio'), ('Alexandrbar', '@SvobodaRadio'), ('JF', '@SvobodaRadio'),
 ('Leszek Nowak', '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Andriy
 Lazorka', '@SvobodaRadio'), ('Fayzullo Dadajonov', '@SvobodaRadio'), ('Marufjon
 Toxtasinov', '@SvobodaRadio'), ('Ravshanbek Abdukadirov', '@SvobodaRadio'),
 (' ', '@SvobodaRadio'), ('Andreu Semenukandreu', '@SvobodaRadio'),
 ('QiZiQM ? ', '@SvobodaRadio'), ('Eugene Hashchuk', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Monica Miller', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Aniemigos \u200d ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Vladimir Marchukov', '@SvobodaRadio'), (' Xiaome',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Mishsqx',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ', '@SvobodaRadio'),
 ('killPut\$404\$n.NotFound', '@SvobodaRadio'), ('Sahil', '@SvobodaRadio'),
 (' ', '@SvobodaRadio'), ('Hutu', '@SvobodaRadio'), ('MariaSky',
 '@SvobodaRadio'), ('pavel kotorobai', '@SvobodaRadio'), ('Mason Harvey',
 '@SvobodaRadio'), ('Xamidullo Ulmasov', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('28 1997 .', '@SvobodaRadio'), ('mubariz',
 '@SvobodaRadio'), ('zlatusik', '@SvobodaRadio'), ('944994411', '@SvobodaRadio'),
 ('kickbox', '@SvobodaRadio'), ('Andrey IMN', '@SvobodaRadio'), ('ngochuyen0612',
 '@SvobodaRadio'), ('Ylsaint Wenson', '@SvobodaRadio'), ('Alecks Galley',
 '@SvobodaRadio'), ('vaska', '@SvobodaRadio'), ('Oleksandr Gladyshko',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Emomali', '@SvobodaRadio'), ('Malekula', '@SvobodaRadio'),

('Ingvaridze', '@SvobodaRadio'), ('Prakash rathore', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), (' ', '@SvobodaRadio'),
 (' ', '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Renat',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Jekabs Kleins',
 '@SvobodaRadio'), ('Ratundra production', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Aigera Kalymzhanova', '@SvobodaRadio'),
 ('Anatol', '@SvobodaRadio'), ('KUZICH', '@SvobodaRadio'), ('Dumitru Guma',
 '@SvobodaRadio'), ('Amiran', '@SvobodaRadio'), ('Tetiana Kovalenko',
 '@SvobodaRadio'), ('Lats', '@SvobodaRadio'), ('martin lakosil',
 '@SvobodaRadio'), ('Temo', '@SvobodaRadio'), ('Vakhobov', '@SvobodaRadio'),
 ('Arda Oktay', '@SvobodaRadio'), ('Serdar Eymirov', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Tina Rocha', '@SvobodaRadio'), ('Leslie
 Abney', '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('komron nemonov', '@SvobodaRadio'), ('kate.net',
 '@SvobodaRadio'), ('Nickolas Foxwell', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('IM Sevostianov Pavel', '@SvobodaRadio'), ('Doctor Sadiq
 khan', '@SvobodaRadio'), ('nadia zozulia', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Roma EXP', '@SvobodaRadio'), ('MOna Ibrahim',
 '@SvobodaRadio'), ('Elena Rozhkova', '@SvobodaRadio'), ('Doniallia Webb',
 '@SvobodaRadio'), ('Artūrs Vanceris', '@SvobodaRadio'), ('arina',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Mwc',
 '@SvobodaRadio'), ('muhibullo holov', '@SvobodaRadio'), ('Eduard Gasumjants',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Violetta',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('ash', '@SvobodaRadio'),
 ('Karaganda Mapper', '@SvobodaRadio'), (' ', '@SvobodaRadio'),
 ('Azimbek', '@SvobodaRadio'), ('...', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Saba gochilaidze', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Ahmadjon Tohirzoda', '@SvobodaRadio'), ('souS_teRiyaki',
 '@SvobodaRadio'), ('Andey', '@SvobodaRadio'), ('Alexandr', '@SvobodaRadio'),
 (' ', '@SvobodaRadio'), ('Rambutan', '@SvobodaRadio'), (' C.A. Tim
 Gravett ', '@SvobodaRadio'), ('Aki Scirpion', '@SvobodaRadio'),
 ('((3 3))', '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Alex
 S', '@SvobodaRadio'), (' ', '@SvobodaRadio'), (' ',
 '@SvobodaRadio'), ('Nadinator94', '@SvobodaRadio'), ('Maxmud Nazarov',
 '@SvobodaRadio'), ('Qodirhon Botirjanov', '@SvobodaRadio'), ('Bismallah
 Khoshdil', '@SvobodaRadio'), ('Valensia', '@SvobodaRadio'), ('legio ner',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Vad', '@SvobodaRadio'), ('Ihor
 Humeniuk', '@SvobodaRadio'), ('Andriy Misko', '@SvobodaRadio'), ('
 - ', '@SvobodaRadio'), ('Bory Bory', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('R. Wladimir S.',
 '@SvobodaRadio'), ('Vladimir Gerasimov', '@SvobodaRadio'), ('Ivan Ivanov',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Max Arbuzov',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('TEMREZ',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Daniel M.',
 '@SvobodaRadio'), (' ', '@SvobodaRadio'), ('Vladymyr Rabenok',

'@SvobodaRadio'), ('Mr Dan', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('!', '@SvobodaRadio'),
 ('sethlon', '@SvobodaRadio'), ('Khurshedjon Rustamov', '@SvobodaRadio'),
 ('Sunnat', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Oksana Syrvarovska', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Dr. Joshua', '@SvobodaRadio'), ('Max Tanaka',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Alexander Geiro',
 '@SvobodaRadio'), ('Mukhammadali', '@SvobodaRadio'), ('Slava Nort',
 '@SvobodaRadio'), ('macpawspace', '@SvobodaRadio'), ('Razmo Karapetyan',
 '@SvobodaRadio'), ('iii ooo', '@SvobodaRadio'), ('Yashar emlak',
 '@SvobodaRadio'), ('Malu Ficent', '@SvobodaRadio'), ('Peter Pantke',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Artush Ghazaryan',
 '@SvobodaRadio'), ('No name', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Tatjana', '@SvobodaRadio'), ('Oleg Busmanov',
 '@SvobodaRadio'), ('Ramaz Gvarishvili', '@SvobodaRadio'), ('Leo Moses',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('2010',
 '@SvobodaRadio'), ('Makszz77', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('Umarov Daler', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('ANASJON
 NEGMATOV', '@SvobodaRadio'), ('Pr.Farnsworth', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('bebbe123', '@SvobodaRadio'), ('Gttggtt',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Erica Zavala',
 '@SvobodaRadio'), ('Misty Brown', '@SvobodaRadio'), ('Andre_ja Pokorná',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('New User', '@SvobodaRadio'),
 ('.', '@SvobodaRadio'), ('Shuhrat Kamilov', '@SvobodaRadio'), ('CumDowns',
 '@SvobodaRadio'), ('Juan Manuel Arbeloa Cidoncha', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Z ff', '@SvobodaRadio'),
 ('Kross - Gen', '@SvobodaRadio'), ('Michele Jones', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Umid Aytbayev', '@SvobodaRadio'), ('Eugene',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('sabrina', '@SvobodaRadio'), ('Alla Ostapchuk',
 '@SvobodaRadio'), ('valijon2008', '@SvobodaRadio'), ('sany samy',
 '@SvobodaRadio'), ('Vlad Sheiko', '@SvobodaRadio'), ('TerraEternity',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Locs',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Rashid Huja', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Joni', '@SvobodaRadio'), ('Nicole Adams', '@SvobodaRadio'),
 ('Tatyana Nikitchenko', '@SvobodaRadio'), ('Azam Apple', '@SvobodaRadio'),
 ('SETR0', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('bradstplug',
 '@SvobodaRadio'), ('Clover Frostwaker', '@SvobodaRadio'), ('Pejskov Dmitrij S.',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('jonas', '@SvobodaRadio'), ('Stefan', '@SvobodaRadio'),
 ('Linda Wilson', '@SvobodaRadio'), ('Evgeniy', '@SvobodaRadio'), ('
 ', '@SvobodaRadio'), ('Andreas', '@SvobodaRadio'), ('Yuliia
 Zemliakova', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Irina

Aleksandrova', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Lamro Arthur', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Ventsi', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('IrinaYu', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Miron', '@SvobodaRadio'), ('nina klyukina', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Dmitry', '@SvobodaRadio'), ('RAHMATJON', '@SvobodaRadio'), ('Svitlana Onufrii', '@SvobodaRadio'), ('Korkem Zhumashkina', '@SvobodaRadio'), ('Lauren Holmes', '@SvobodaRadio'), ('Evgeniy Makar', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Stephany1', '@SvobodaRadio'), ('Megan', '@SvobodaRadio'), ('Robert Nolton', '@SvobodaRadio'), ('sharifjon rahimov', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('@@@@@', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Zoya Kinstler', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Tojiddin Sirojov', '@SvobodaRadio'), ('Danya', '@SvobodaRadio'), ('© @', '@SvobodaRadio'), ('NickZ', '@SvobodaRadio'), ('LIC HERNANDO CARDENAS', '@SvobodaRadio'), ('esmaeil', '@SvobodaRadio'), ('Crocodile Gennadi', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Desh.tjk', '@SvobodaRadio'), ('Sarah Bates', '@SvobodaRadio'), ('Leopold', '@SvobodaRadio'), ('Mykola Svydran` (Forsher)', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Omitanofitoubef', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Anna Movsesyan', '@SvobodaRadio'), ('Mirzoev Muhammad', '@SvobodaRadio'), ('Den', '@SvobodaRadio'), ('Fedy', '@SvobodaRadio'), ('BENDERA STEPAN', '@SvobodaRadio'), ('Sukhumi Denzel', '@SvobodaRadio'), ('Djub', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Regina Hill', '@SvobodaRadio'), ('Andriy', '@SvobodaRadio'), ('IrB', '@SvobodaRadio'), ('JulBari', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('oldmammut', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('019asvx', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('rem san', '@SvobodaRadio'), ('Dima', '@SvobodaRadio'), ('gg', '@SvobodaRadio'), ('Kira Orions', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Nurullo Ergashev', '@SvobodaRadio'), ('Ihar Haro', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Marko.M', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Aruan Shaukenov', '@SvobodaRadio'), ('Muhammad Ali', '@SvobodaRadio'), ('Yan Sivashcko', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Lochinbek Allamuratov', '@SvobodaRadio'), ('Gefaestus', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Gleb Sokolov', '@SvobodaRadio'), ('Wildfire Cat', '@SvobodaRadio'), ('honto no neko', '@SvobodaRadio'), ('Andrejs Borovkovs', '@SvobodaRadio'), ('Biloljon Qodirv', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Conea Valeriu', '@SvobodaRadio'), ('Mahir Eyvazov', '@SvobodaRadio'), ('', '@SvobodaRadio'),

'@SvobodaRadio'), ('John Lackland', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('AkvaservisGL', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Xander Paul', '@SvobodaRadio'), ('Slava', '@SvobodaRadio'),
 ('Dmitry Zypravlev', '@SvobodaRadio'), ('Adil', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('LK\U0001f90d\U0001f90d', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('bob', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Rafu Kygyrov',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('valeria smith',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Samir Musoev', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Gbulanova', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('Timporello', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Viktor
 Zhornyak', '@SvobodaRadio'), ('Awayfromputin', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Eugene Donnik', '@SvobodaRadio'), ('Sumay Behzodzoda',
 '@SvobodaRadio'), ('firdavs', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('Wiljams', '@SvobodaRadio'), ('Tall Guy', '@SvobodaRadio'), ('taratata',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Don Hamster de
 Shawshank (aka)', '@SvobodaRadio'), ('ibeaded.com',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Serhii',
 '@SvobodaRadio'), ('Ashlee Cummings', '@SvobodaRadio'), ('Keny G',
 '@SvobodaRadio'), ('lok1zz | 61', '@SvobodaRadio'), ('Odessa Ukraine Dnipro',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Leyla', '@SvobodaRadio'), ('Nicole Cottrill',
 '@SvobodaRadio'), ('prostozashel', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Sandy Alicandro', '@SvobodaRadio'), ('Vahit Sultanov',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Roman Kudriashov', '@SvobodaRadio'), ('Issayev Erkin',
 '@SvobodaRadio'), ('Dawn Cox', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('Ibraghimov.i', '@SvobodaRadio'), ('Storm Rain',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Ruzanna Dadyan',
 '@SvobodaRadio'), ('Xinhao Fortunata', '@SvobodaRadio'), ('lukas zvezdovas',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('nikolai boyadjiev',
 '@SvobodaRadio'), ('Emil Buba', '@SvobodaRadio'), ('Azbilirkishi',
 '@SvobodaRadio'), ('Alex Merlin', '@SvobodaRadio'), ('Ángel', '@SvobodaRadio'),
 ('Denis Klimov', '@SvobodaRadio'), ('nikicak', '@SvobodaRadio'), ('An',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Arcrun martirosyan',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('neonilla',
 '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Kairat Sultanbek',
 '@SvobodaRadio'), ('CIRCLE', '@SvobodaRadio'), ('SOHIL.4249', '@SvobodaRadio'),
 ('daria', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('',
 '@SvobodaRadio'), ('asel', '@SvobodaRadio'), ('', '@SvobodaRadio'),
 ('', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Madina
 Karimdodova', '@SvobodaRadio'), ('lexander', '@SvobodaRadio'), ('Vladimir


```
Zigarevich', '@SvobodaRadio'), ('ART', '@SvobodaRadio'), ('yo1q1v',
'SvobodaRadio'), ('Hashmatullah saadat', '@SvobodaRadio'), ('Arman',
'SvobodaRadio'), ('', '@SvobodaRadio'), ('',
'SvobodaRadio'), ('szur', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('
', '@SvobodaRadio'), ('umbet seksenbayev', '@SvobodaRadio'),
('Fiodor logvinenko', '@SvobodaRadio'), ('Muhammadnabi', '@SvobodaRadio'),
('VTKuts', '@SvobodaRadio'), ('Alex Mishkin', '@SvobodaRadio'), ('N',
'SvobodaRadio'), ('Vytautas.N', '@SvobodaRadio'), ('mate xeladze',
'SvobodaRadio'), ('Vitalik Ryabokozhushniy', '@SvobodaRadio'), ('
', '@SvobodaRadio'), ('', '@SvobodaRadio'), ('Aisha Usenova',
'SvobodaRadio'), ('', '@SvobodaRadio'), ('Zulqaynar
Hudayqulov', '@SvobodaRadio'), ('Nicolae Mereuta', '@SvobodaRadio'), ('bultik',
'SvobodaRadio'), ('', '@SvobodaRadio'), ('Vladimir Fedorenko',
'SvobodaRadio'), ('NightBirdSava Women', '@SvobodaRadio'), ('',
'SvobodaRadio'), ('', '@SvobodaRadio'), ('Zzzz Mmmm',
'SvobodaRadio'), ('Augustin Zelenka', '@SvobodaRadio'), (' «speeller»
', '@SvobodaRadio'), ('', '@SvobodaRadio'),
('', '@SvobodaRadio'), ('Arif Lbrahimi', '@SvobodaRadio'), ('Natalija
Gavrilova', '@SvobodaRadio'), ('Malik', '@SvobodaRadio'), ('Nikolay Churlyae',
'SvobodaRadio'), ('nigora.rustamovna.99@mail.ru', '@SvobodaRadio'), ('Luda',
'SvobodaRadio'), ('Azizjon Pirmukhamedov', '@SvobodaRadio'), ('Ieva',
'SvobodaRadio'), ('Roma Dovganchuk', '@SvobodaRadio'), ('',
'SvobodaRadio'), ('Angela Wallace', '@SvobodaRadio'), ('kirichenko.k',
'SvobodaRadio'), ('marina shteynberg', '@SvobodaRadio'), ('',
'SvobodaRadio'), ('Aziz Timp', '@SvobodaRadio'), ('',
'SvobodaRadio'), ('mirfuad@mail.ru', '@SvobodaRadio'), ('serega',
'SvobodaRadio'), ('', '@SvobodaRadio'), ('Pečka',
'SvobodaRadio'), ('Roman Gabisonia', '@SvobodaRadio'), ('Antonny195',
'SvobodaRadio'), ('', '@SvobodaRadio'), ('',
'SvobodaRadio'))]
```

```
[61]: subgraph.edges
```

```
[61]: OutEdgeView([(' ', '@ru_rbc'), ('Set Mortensen', '@ru_rbc'), ('Misha
UR', '@ru_rbc'), ('valera karaman', '@ru_rbc'), ('ivanrussia', '@ru_rbc'),
(' ', '@ru_rbc'), ('Pletunoff', '@ru_rbc'), ('sweetdreamslizz',
 '@ru_rbc'), ('Mila', '@ru_rbc')])
```

```
[62]: # probability some bad values are in dataframe that have incorrect type
# TODO: CREATE COMMUNITIES BY HAND, YOU KNOW WHAT THEY ARE

import networkx.algorithms.community as nxcom

# identifying communities within the network
twitter_communities = sorted(nxcom.greedy_modularity_communities(G,
                                                                    weight=None),
                             key=len,
```


'vasas István', ' ', 'IreneSt', ' ', 'Viktor
 Larionov', 'Cyberayo', 'Andrian Grati', ' ', 'Ena Meyer', 'Alexey
 Deryaga', 'Shireddin aslanov', ' ', 'iOS', ' ', 'Jazzman603', ' ',
 ' ', 'Cindy Miller', 'bngkrkzstr', ' ', ' ', 'JasperMaggie',
 'Zura Batiashvili', 'lia', ' ', ' ', 'deremozowoono', 'Michail
 (L3,)', ' ', 'Vvhale', 'Natalia Zarechneva', '20 22',
 'Gintautas Noreika', 'Mila', ' ', ' ', 'My name is yangzuo', 'corsica', ' ',
 'Ranago', 'Jury Badulin', 'Andrey Vasilyev', 'Arvydas', 'Natasha Touzovskaia',
 'Denis Novikov', ' ', 'Timur Isakov', 'Willem van S.', ' ',
 ' ', ' ', 'Dmitriy Kalashnikov', ' ', ' ', ' ',
 'Klara Bek', 'Alexander Kim', 'Anna', 'Pishumm2', ' ', ' ',
 ' ', 'Hieininen', 'Kristy Smith', ' ', ' ', 'Adrian', ' ',
 ' ', ' ', 'resmon', ' ', ' ', 'Ciupaga', 'ELSANA
 TUJUH', 'Svetlana Karpekina', 'Aigul Xab', 'Komiljon Akhrorov', 'Alena',
 'Jokhongir', ' ', ' ', ' ', ' ', 'Rodion', 'maaax_----', ' ',
 ' ', 'Baxtiyor Abdurasulov', ' ', ' ', ' ',
 'Jabrail', 'Hassan Riaz', 'Lisa Joseph', 'dasigu3(,)', 'surald', ' ',
 ' ', 'Karina', ' ', ' ', 'Abdussalam Qayumov', ' ', ' ', ' ',
 ' ', 'A Pao Tráng', 'Sairan Jarboldu', ' ', ' ', '... _',
 'CarrieRobin', 'Munteanu Jasmin', ' ', '900', ' ', ' ', 'Victor
 Armishev', ' ', ' ', ' ', ' ', 'Akmal Juraev', ' ',
 ' ', ' ', ' ', ' ', ' ', 'Vasiliiy
 Kalyuzhnyy', 'Bahadır İnanc Özkan', 'Tina Cummins', ' ', ' ', 'Victoria
 Johnson', ' ', ' ', ' ', ' ', ' ', 'Natali', 'Liberty
 Crew', 'Danny Moore', ' ', ' ', 'martin', 'Knox Knowledge', 'john
 roger', 'CasualReader', ' ', ' ', ' ', ' ', ' ', ' ',
 ' ', ' ', ' ', ' ', 'Angry Cat', ' ', ' ',
 'VönHell', ' ', ' ', 'kirill krivoschapkin', 'Bobir Akbarov',
 ' ', ' ', ' ', ' ', 'G.O.A.T.', ' ', ' ', 'Yashar
 Ismaylov', 'rt', ' ', ' ', ' ', 'Aston Astonzoda', 'Azizbek',
 'Ismoil', 'sergio Buskets', 'Ilham Ahmad', 'Fanil Valirov', 'Surush', ' ', ' ',
 ' ', ' ', ' ', ' ', 'Moise Kisonia', ' ', ' ', 'Alex', ' ',
 ' ', ' ', ' ', ' ', 'Vladimir Sapego', 'Wehandewem',
 'thechosenone', ' ', ' ', ' ', ' ', 'Maryna', ' ', ' ',
 'maks kaverin', 'Avgii', 'Avazbek', 'Romashka Petrushka', 'Toni Pham',
 ' ', ' ', 'Pussey Crasher', ' ', 'Karmela Hodos', 'Karen', ' ',
 ' ', 'ivanrussia', 'Godella Petty', ' ', ' ', 'Anh Dũng',
 'CallMe', 'Stanislav Antilevsky', ' ', ' ', ' ', ' ',
 ' ', ' ', 'Frederick Taxo', ' ', 'Cars', ' ', ' ',
 'Robert Still', 'ocrxx', 'Gulia', ' ', ' ', 'Naila Rustamova', ' ',
 ' ', 'Abdulwahid Afzaly', ' ', ' ', ' ', 'Rev lur',
 'Danielle Rose', 'Vera Smirnova', 'memento7mori#.hft (,)', ' ', ' ',
 ' ', ' ', ' ', ' ', 'Maxim Kazhyna', 'Lina G',
 'tedananalob', ' ', ' ', ' ', ' ', 'Sergey63', ' ', ' ',
 ' ', ' ', 'Aziz', 'garjou56(,)', 'Dominum', 'Denis Vilman',
 'Laky', 'Enoheusella', 'motyA', 'Olga', ' ', 'Alsapteka"', ' ', ' ', 'Muhammed
 Riza', ' ', ' ', 'I.Wallau', 'Rasim Sk', 'Sergij Lazebnyk', 'nan', 'Galina',
 'Umida Usmonova', ' ', ' ', 'METE AYMAYA', 'Omowimozij', ' ', ' ', 'Andrej

Tiganov', 'Tartar', ' ', ' ', 'C', 'daria mosolova', ' ',
 ' ', 'nehapesasaiston', ' ', ' ', 'DecimalCrown', 'Adl', 'ruppinwin
 Odessa', ' ', 'Su', ' ', ' ', ' ', ' ', 'Kiki_and_Buba', 'A.Eskin', 'David
 Ekanger', 'Stanislav', 'A ', ' ', 'Nata Sara', 'Oksana', 'red door',
 'crouzz', 'Tom Kerrigan', 'Aleksey Ermolaev', 'borzoi57', 'Gulliver007', 'Lari
 Sergiu', 'Mygame', 'Vasiliy Bichkov', ' ', ' ', 'Nikolay Kaplin',
 ' ', ' ', ' ', 'dibley Ivanov', ' ', ' ', ' ', ' ',
 'nagita debora', 'grgbltv', 'Dumitru Rusu', ' ', ' ', 'Laurie
 Schneider', 'IVAN LOBYNICHEV', 'lazybuzzy', 'I ', ' ', ' ', 'Mama V',
 ' ', ' ', ' ', 'Mansurali jamoliddinov', ' ', ' ',
 ' ', 'The freemason', 'Tofig Davudov', 'Qizzbo', ' ', ' ', 'TLS CARGO',
 'evgenii markelov', ' ', ' ', ' ', ' ',
 ' ', 'Nuhonotunuther', 'Ahmadov Shodmon', ' ', ' ', ' ',
 ' ', ' ', 'Svetlana Babinica', 'WilliamAstrid ', 'Disappear', 'Almaz
 Makhmut', ' ', ' ', ' ', 'Akmaral Batalova', 'Alexey Gordeyko',
 ' ', ' ', 'Anton', ' ', ' ', ' ', 'Katy Mrock', ' ',
 ' ', 'Alice', ' ', ' ', 'ConorMoore', 'Gg Ol', ' ', ' ',
 ' ', 'Aphelios', ' ', ' ', ' ', ' ', 'Jez', 'Mikhail',
 'Muhammad P', 'Dr. Shore | iPhotoStreet.com', 'Mirelle', 'nowanofaiqiduw',
 'Andrey Gornushechkin', 'one second', ' ', ' ', 'Daniil', 'Lev', ' ',
 ' ', ' ', 'Crizis', ' ', ' ', 'Sergey Krasnobaev', ' ',
 ' ', ' ', ' ', ' ', 'Sergey', 'Dmitriy Dumkin',
 ' ', ' ', ' ', ' ', ' ',
 ' ', 'XOJAXMAD', 'Peggy Quimby', 'Mike', 'Igor Kch', ' ', ' ', 'Arigato',
 'sweetdreamslizz', ' ', ' ', 'Da Phantom Shtr', ' ',
 'Michael Ryzhichenko', 'khlud', 'Operation Witchcraft', 'Alex Sarkisov', ' ',
 ' ', ' ', 'Franciscosantos', 'Vitalie', 'barbara', 'No Face No
 Name', ' ', ' ', ' ', ' ', ' ', 'VAIN', 'Mendsaikhan
 Purevkhaidav', ' ', ' ', 'Mephistopheles', 'Juia Sevostyanova', ' ',
 'Galina Grigoryan', 'Léonidas', ' ', ' ', 'Asgar', 'Denis',
 'Kulturamultura', 'molnia18 games', ' ', ' ', 'ZloyDed', 'Dildora',
 'Rasbora', 'erefothanodu', ' ', ' ', 'REKCIK FORTNITE', ' ',
 ' ', 'Nursultan Baisakalov', 'Rima Sloutskaia', 'Aleksandr', ' ',
 'Sherali Nabijonov', 'Hanna Grininger', 'A Ab', ' ', 'Dmitry Bazekin',
 'Elmirhesenli', 'Kurban Mamedov', 'Mi Tan', 'Mobil. Centre', ' ',
 'mustafeahmedmohamud@gmail.com', ' ', ' ', '600022', ' ',
 'firuzi Kintsurashvili', 'Hohatetoululucu', 'Inter Studio', 'Oxana', ' ',
 ' ', 'Alimehdi Memmedov', ' ', ' ', 'Elnara safronova',
 'Natallia Chodosowska', ' ', ' ', 'Utounubeisaf', 'Haichuan gen', ' ',
 ' ', ' ', 'Abduganikhodja Bakhtiyorkhadjaev', 'Gintaras
 Miežanskas', 'Sergey Spak', 'ixdaya', ' ', ' ', 'albert', 'Set
 Mortensen', 'Militant Robots', 'Vladimir', 'Eldar', ' ',
 'Tefijorina', ' ', ' ', ' ', ' ', ' ',
 'Parvin ismatov', 'Firuz Shukurov', ' ', ' ', 'Parker', 'Dmitrii Smirnov', 'Olev
 Kotka', 'Hamidjon Akbarif', ' ', ' ', 'Jurijs Ivanovs', 'jr.rafo',
 'Muslim Muxtarov', ' ', ' ', 'Lucas Daniel Olivera', 'Olga Prudnikova', 'Helen
 Mikhailovskaya', ' ', ' ', 'Serous Pavel', ' ', ' ', 'Renato Calore',
 'Nadin', 'Andrey Makarov', 'Irina', 'dont_smoke_here_please', 'Iacubenco

Vitalii', 'Shaarani Lawan', 'Daniel', 'Sabyr Pernebay', ' ',
 'Aleksandr S', 'Fayzullo', 'samsara', 'Artem Gritsenko', 'Ivan Shmelev',
 'Kostya', ' ', 'Rinata', 'Diyor Pubg', ' ', 'KIRA',
 'YaN KalasH Kalash "LE RETOUR !"', ' ', 'Assel', 'ELN DEV', 'Murad',
 ' ', ' ', 'Ivan Litvinov', 'Yury Meshkov', 'Kote
 Gelashvili', 'Soffee', ' ', 'Rus Tam77', 'Jasur Valiev', 'Svetlana
 Ionova', ' ', ' ', 'Alexander Dageron', '
 ', ' ', 'Bethany Kimmet', 'danny costa', 'Chenxing',
 'Askhabull', 'Ronin Barnes', 'Uliana', ' ', 'Janelle Stephens', 'Olia',
 ' ', 'Iskhakov Bulat', 'Ira', 'IXTIYOR ODILOV', ' ', '
 ', 'Katiusha', ' ', 'Kostik228 (,)', 'Alizamin', 'Shadow Hide',
 ' ', 'Dasa Tusco', ' ', 'Tisya 99', '
 ', 'la la la', ' ', 'Celeste Iannacone', 'Marina S.', '
 ', 'Sapphire cat', 'Ralyube', 'Heather Noland', ' ',
 ' ', 'Anastasiia', 'Babek Al-Khalili', 'Nikolai Grigoriev', 'Agamergen
 Isangulyyew'}), frozenset({' ', 'Zzzz Mmmm', 'Markian
 Kuzmowycz ', 'nikolai boyadjiev', ' ', ' ', '
 'Galinochka', 'Artush Ghazaryan', 'jalmar pavlovec', 'Mkrtich Mkrtchyan',
 ' ', 'muhibullo holov', ' ', 'Kats Yanina',
 ' ', 'SOHIL.4249', ' ', 'Linda Wilson', 'Sevara
 Sevara', 'Maxmud Nazarov', 'Don Hamster de Shawshank (aka)',
 'Andrejs Borovkovs', 'Evgeniy Makar', ' ', ' ', 'Xamidullo
 Ulmasov', 'Malu Ficent', ' ', 'Daler', ' ', '
 ', ' ', 'Denis Klimov', 'Sumay Behzodzoda', 'Mutribsho',
 'Sarah || \$TTC', 'Nicole Cottrill', ' ', 'Mishsqx', ' ',
 ' ', ' ', ' ', 'Mwc', 'mishka',
 'sunwukong', ' ', ' ', 'littlerobbergirl', '
 ', 'Sherzod Khamidullaev', 'Gefaestus', 'Kolya Osten-Backen', 'Alex&Alis',
 ' ', 'LRBN', 'Iurasco Sergiu', ' ', ' ', '
 'nina klyukina', 'Datu Man', 'kirichenko.k', 'Gbulanova', 'Malik', 'Yevgeny
 Khodos', ' ', 'mate xeladze', 'History of the conspiracy /
 ', 'Şamsi Akhmed', 'Nadinator94', ' ', 'Ravshanbek
 Abdukadirov', 'Alecks Galley', ' ', 'leftist drainer \U0001faa9',
 ' ', '...', 'KUZICH', ' ', 'galina Sylima', 'asel',
 ' ', 'Azam Apple', 'Ruslan', 'Robert Nolton', ' ',
 ' ', ' ', 'LIC HERNANDO CARDENAS', 'Max Tanaka', 'Madina
 Karimdodova', 'N', ' ', 'maRti', 'Tvoy Razum "(,)"', ' ',
 'Renat', 'Alex S', ' ', ' ', 'Evgeny', ' ', 'CumDowns', '
 ', 'Andriy Lazorka', 'Mukhammadali', ' ', 'szur',
 'Mabel Rodriguez', ' ', 'Korkem Zhumashkina', 'Nurbek', 'DesemberWiin00',
 'Timporello', ' ', ' ', 'Roman Kudriashov', ' ',
 'Roman Juve', ' ', 'Ilyas Sarsenbaev', 'Sukhumi Denzel',
 'Renatas Katilius', ' ', 'Regina Hill', 'Sasha NHK', 'Boris
 Burshtein', 'AkvaservisGL', 'macpawspace', ' ', 'Pečka', 'DD',
 'morik', 'Aslan Akshin', 'Leslie Abney', 'firdavs', 'Denis Vladimirovich',
 'Vytautas.N', 'Erica Zavala', 'Rafu Kygyrov', 'zlatusik', 'Awayfromputin',
 'Amiran', 'Monica Miller', 'Qodirhon Botirjanov', ' ',
 ' ', 'Olcia Ostroushko', ' ', 'VITALY EL',

', 'Sunnat', 'Samuel
 Felipe', 'erno demjen', '
 ', '2010', 'Zahir Esedov', 'Roman Green', '
 ', 'Vad', 'Alla Danik',
 'Boris.korg88', 'steven mazurskih', '@SvobodaRadio', '
 'Ainura Ermeeva', 'legio ner', 'LK\U0001f90d\U0001f90d', 'Leopold', 'NoziroFF
 N', 'Artūrs Vanceris', 'luuuuul 2', 'Farxod Norov', '
 'Jonsnow', 'Yashar emlak', 'ANASJON NEGMATOV', 'nikicak',
 'Adil', 'Lauren Holmes', '
 'killPut\$404\$n.NotFound', 'Andriy Misko', 'Abu Halas', 'Curt B.', '
 ', 'nuonlithuania', 'sharifjon rahimov', 'Arda Oktay', '
 ', 'evgen', '
 ', 'Žabba Blue', 'daria', 'Ratundra production', 'Pr.Farnsworth',
 'ibeaded.com', 'Alex Levin', 'Tetiana Kovalenko',
 ', 'Grigori Draci', 'prostozashel', 'Muxammadjon
 Abdullaev', 'Neuron Shop', 'nigora.rustamovna.99@mail.ru',
 'Johnny', 'martin lakosil', 'Ruzanna Dadyan', 'Crocodile Gennadi', '
 ', 'Samir Musoev', 'Khatuna Mamisbedashvili',
 'Hutu', 'Eugene Hashchuk', 'Luda', '#####', '
 ', 'Nurullo Ergashev', 'Arcrun
 martirosyan', 'Evgeniy', '
 ', 'Alex Mishkin', 'nadia zozulia', 'Rus', '
 ', 'Giorgi Koba', '
 'Ventsi', 'Hasitha Leanage', 'Nikolay Grigorievich', 'taratata',
 ', 'Eugene', 'Scorpion', 'Viacheslav',
 'Aniemigos \u200d', 'Sandy Alicandro', 'Valensia', 'Aruan
 Shaukenov', 'Andrius', 'Nurdaulet Zharylkapov', 'Nicole Adams',
 ', 'Yana', 'tolga
 yetil', 'Ellen Muniz', 'Alexander Geiro', '
 'Petro Protsiv', 'Larissa Pitkenina', 'Hein Ivar', '
 ', 'Wildfire Cat', 'King Lion', '
 'Kalicha Alikulova', '
 ', 'Hoji Bekov', 'Rambutan', 'iii
 ooo', 'Alexey Vasilyev', 'leonroy', 'SHAHBOZ RAKHMATOV',
 'Anatoliy Buzdalin', 'BAHAGIA ITU INDAH', 'Ingvaridze',
 'mubariz', '
 ', 'Ololosh Ololoew', 'Clarita Zartman', 'Dumitru Guma', 'MoniR
 HoseN', 'Vladimir Gerasimov', 'SHYRIK NIKOLAEV', 'John Lackland', '
 ', 'Alex Merlin', 'IrinaYu', 'yo1q1v', 'Andreu
 Semenukandreu', 'Andrey Blinov', '
 ', 'Leo Moses', 'Zoya Kinstler', 'Slava Nort', 'Arif Lbrahimi', '
 ', 'alpaca1900', 'Dmitry',
 'Misty Brown', 'Anastasia Nathans', 'Antonny195', '
 ', 'Peach', 'Terloyeva', 'lukas zvezdovas', '
 ', 'Vlad Sheiko', 'tranquiI', 'sethlon', 'Leszek Nowak', '
 ', 'C.A. Tim
 Gravett', 'kickbox', 'Tall Guy', 'gintoxs', 'petrovich', '
 ', 'New User', 'Ordures à Bennes', 'Vladimir Marchukov', 'Ivan

Ivanov', 'Mohamed El alfy', 'sany samy', 'vaska', ' ', 'Tatjana',
 'Nikolay Churlyayev', 'Fade Asaf', 'bob', 'Geoger', 'bebbe123', 'stepan', '
 ', 'Andreas', 'Bobir Akhmedjanov', 'Anna Movsesyan', 'Vladimir
 Tartas uk', ' ', ' ', ' ', ' ', 'Desh.tjk',
 ' ', 'timer.ge', ' ', '!', 'Nickolas
 Foxwell', 'CIRCLE', ' ', ' ', 'Dmitry Zypravlev',
 'Vladimir Fedorenko', 'Yamach Pubgm', 'Ibraghimov.i', ' ',
 'Arman', 'Ilya Osterman', ' ', ' ', 'Armineh Grigorian', '
 ', ' ', 'Odessa Ukraine Dnipro', 'Andey', '
 ', ' ', 'oldmammut', ' ', 'R. Wladimir S.',
 'esmaeil', 'Storm Rain', ' ', ' ', ' ', ' ',
 ' ', 'Romero Shelby', 'Vasil Zoll', ' ', ' ', 'Leyla', 'Augustin
 Zelenka', 'Vladimir Zigarevich', ' ', ' ', 'Megan', ' ', 'Max
 Arbuzov', 'Omitanofitoubef', 'Valerii', 'Vladymyr Rabenok', 'Family nft',
 ' ', ' ', 'pavel kotorobai', 'Dooron Kaldarov', 'TEMREZ', '
 ', 'Bory Bory', 'Ivan Karasov', 'Eduard Gasumjants', 'Azbilirkishi',
 'Nandy', 'Serdar Eymirov', 'Miron', ' ', 'Den', 'Juan Manuel Arbeloa
 Cidoncha', 'olga mark', ' ', ' ', 'Emil Buba', 'Leqso', 'Gleb
 Sokolov', 'Vahit Sultanov', 'Aleksei Aleksei', ' ', ' ',
 ' ', 'VTKuts', 'Elena Rozhkova', 'Dawn Cox', 'Irina Sokolova', 'Larrf',
 'Razmo Karapetyan', 'komron nemonov', ' ', ' ', 'Jerry Rodriguez', 'Ahmadjon
 Tohirzoda', 'Anatol', ' ', ' ', ' ', 'Gttgtt', '
 ', 'NightBirdSava Women', ' ', ' ', ' ', 'Andrey IMN',
 ' ', 'Doniallia Webb', 'Dmytro Ivanov', 'Makszz77', 'Natalie', 'Roma
 Dovganchuk', 'Nunadievis', 'Sergei Safin', ' ', ' ', 'IM
 Sevostianov Pavel', 'JulBari', 'SETRO', 'Dmitrij Krahn', 'Robert', 'Fiodor
 logvinenko', 'Manus Aeon', ' ', '«speeller»', 'QiZiQM?', 'MariaSky',
 ' ', 'Claudia Boyd', 'Kairat Sultanbek', 'Muhammad 54', ' ', ' ', 'Umid
 Aytbayev', ' ', ' ', ' ', 'Azizjon Pirmukhamedov', 'Ong
 Chong Yee (ichi1996.crypto)', 'gg', 'Nataliia', ' ', ' ', 'Iulian
 Andriuta', 'Alexandru Mihalachi', 'Zulqaynar Hudayqulov', 'giorgi', '
 ', 'Tatiana Berhin', 'Bakyt Tursaliev', 'Pirat', 'Aki Scirpion', '
 ', ' ', 'No name', 'Karaganda Mapper', 'Oksana Syrvarovska',
 ' ', 'A Riazanski', 'Doctor Sadiq khan', 'kate.net', 'Xander Paul',
 ' ', ' ', ' ', 'Moscow Apartments', ' ', ' ',
 ' ', ' ', 'D.V.Zion', 'Roman Gabisonia', 'souS_teRiyaki', '
 ', ' ', ' ', 'honto no neko', 'Ieva', ' ', ' ', ' ',
 'Serge Ovcharenko', 'Irina Aleksandrova', 'Shuhrat Kamilov', 'Abdukodir
 Abdusattarov', 'Azamat Bilalov', ' ', ' ', 'Marko.M', 'Lamro Arthur', '
 ', ' ', ' ', 'Andrei Timof v', ' ', ' ',
 ' ', ' ', 'IrB', ' ', ' ', ' ', 'Ashlee
 Cummings', 'Ramaz Gvarishvili', ' ', ' ', ' ', ' ', 'Clover
 Frostwaker', 'aleksandra', ' ', ' ', ' ', ' ', 'TerraEternity',
 ' ', ' ', ' ', ' ', ' ', ' ',
 ' ', 'Vitalik Ryabokozhushniy', ' ', ' ', 'Malekula', ' ', ' ',
 'Maria', 'Mykola Svydran` (Forsher)', 'Lats', 'Alla Ostapchuk', ' ', ' ',
 ' ', ' ', ' ', ' ', ' ', ' ', 'Biloljon Qodirv',
 ' ', ' ', ' ', ' ', ' ', ' ', 'Natalija Gavrilova', 'mirfuad@mail.ru',

'serega', ' ', 'Kross - Gen', ' ', 'Ukrainian man',
'Rimvydas Bajorinas', 'Muhammad Ali', 'Special Operative Rizzo', 'emil ibraev',
'lok1zz | 61', 'Marufjon Toxtasinov', 'Mahir Eyvazov', 'Murodjon Eshboyev',
'Conea Valeriu', 'Sahil', 'Serhii', 'Bakdaulet Talgat', ' ---- ', '
', 'MOna Ibrahim', 'Sarah Bates', 'ion', 'Mason Harvey', '
', 'Generation Y', ' ', 'Svitlana Onufrii', 'valijon2008',
', ' ', 'Aigera Kalymzhanova', 'Jamshed', ' ' (Yasha)',
'volodymir', ' ', 'Umarov Daler', ' ', 'Lochinbek
Allamuratov', 'Azimbek', 'An', 'Gala', 'BENDERA STEPAN', 'Ylsaint Wenson',
', ' ', 'Yury Mironenlo', 'Eduard Martirosyan', ' ', '
', ' ', 'Aisha Usenova', ' ', 'Ihor Humeniuk',
'Anton Belkin', ' ', 'Tojiddin Sirojov', 'Genadi Kravets',
', ' ', 'Saba gochilaidze', ' ', 'Keny G', ' ', '
', ' ', 'Cherious', 'maruf', 'Mihael Katsev', '
', ' ', 'Vennikov Elena', ' ', 'Tatyana
Nikitchenko', ' ', ' ', 'Viltarin', 'Oleg Busmanov',
'Provinsjules', ' ' Username', 'Hokim Valiev', 'Pejskov Dmitriy S.',
'Andre_ja Pokorná ', 'Iftikhor Shraliev', 'Mirzoev Muhammad', 'RAHMATJON',
'valeria smith', 'Bob67', 'Sino Khisainov', 'Aziz Timp', 'lexander', '
', ' ', 'bultik', 'Katusiime', 'Saidbek Abduraimov', '
Z ff', 'jonas', 'Daniel M.', 'Irons', ' ', 'Hashmatullah
saadat', 'Khurshedjon Rustamov', ' ', 'Nicolae Mereuta', '
', 'ART', ' ', 'Violetta', 'Fedy', 'Angela Wallace', '
', 'ida sveinhaug', 'Vakhobov', ' ', ' ', 'Slava',
', ' ', '(((3 3))', 'sabrina', ' ' Xiaome', '
', 'Maryaka', 'JF', 'Jekabs Kleins', 'Dr. Joshua', 'Rasulov_04',
'Stephany1', 'Fayzullo Dadajonov', ' ', ' ', '
'Stefan', '019asvx', ' ', ' ', ' ', '
', ' ', ' ', ' ', 'Temo',
'Emomali', 'Kira Orions', 'Dima', 'bradstplug', ' ', 'Roma EXP', 'Djub',
'944994411', 'cezar nichiforac', 'AMIR ALI', 'Pavel', 'Vladimir Slavinsky',
', ' ', 'Boris Nedov', 'Danya', ' ', '
', ' ', ' ', 'DAVID Beleg', 'Tina Rocha',
'Volodumur Grisyuk', 'Ángel ', 'ulia lanna', ' ', 'Oleksandr
Gladyshko', ' ', 'Wiljams', ' ', 'Mr Dan', 'maxsh', ' ', '
', ' ', 'Siti Munadiroh', ' ' © @', 'umbet seksenbayev',
', ' ', ' ', 'Andriy', 'John', ' ', '
', ' ', ' ', 'Yan Sivashcko', 'Alexandr', 'Issayev
Erkin', ' ', ' ', ' ', 'Alexandrbar', ' ', 'Nikolya
0311', 'Donda', '28 1997 .', 'Yuliia Zemliakova', 'Sarv', '
', 'Locs', 'ngochuyen0612', ' ', ' ', 'Xinhao
Fortunata', ' ', 'Viktor Zhorniyak', 'Peter Pantke',
', ' ', 'Vera Grigoryeva', 'Eugene Donnik', 'ash', ' ', '
', ' ', ' ', 'Kris Muntz', 'Pashtet_coins', 'Ximin |
ARG', ' ', 'Bismallah Khoshdil', 'Muhammadnabi', ' ', '
'Ihar Haro', 'Crypto', 'Prakash rathore', 'marina shteynberg', 'NickZ',
', ' ', 'your', ' - ', 'Vyacheslav Serebryakov', '
', 'arina', 'Rashid Huja'}}]]

```
[64]: print("Visualizing Identified Communities")

def set_node_community(G, communities):
    '''Add community to node attributes'''
    for c, v_c in enumerate(communities):
        for v in v_c:
            # Add 1 to save 0 for external edges
            G.nodes[v]['community'] = c + 1

def set_edge_community(G):
    '''Find internal edges and add their community to their attributes'''
    for v, w, in G.edges:
        if G.nodes[v]['community'] == G.nodes[w]['community']:
            # Internal edge, mark with community
            G.edges[v, w]['community'] = G.nodes[v]['community']
        else:
            # External edge, mark as 0
            G.edges[v, w]['community'] = 0
```

Visualizing Identified Communities

```
[65]: def get_color(i, r_off=1, g_off=1, b_off=1):
    r0, g0, b0 = 0, 0, 0
    n = 16
    low, high = 0.1, 0.9
    span = high - low
    r = low + span * (((i + r_off) * 3) % n) / (n - 1)
    g = low + span * (((i + g_off) * 5) % n) / (n - 1)
    b = low + span * (((i + b_off) * 7) % n) / (n - 1)
    return (r, g, b)
```

```
[66]: # Set node and edge communities
set_node_community(G, twitter_communities)
set_edge_community(G)

# Set community color for nodes
node_color = [
    get_color(G.nodes[v]['community'])
    for v in G.nodes]

# Set community color for internal edges
external = [
    (v, w) for v, w in G.edges
    if G.edges[v, w]['community'] == 0]

internal = [
    (v, w) for v, w in G.edges
```

```

        if G.edges[v, w]['community'] > 0]

internal_color = [
    get_color(G.edges[e]['community'])
    for e in internal]

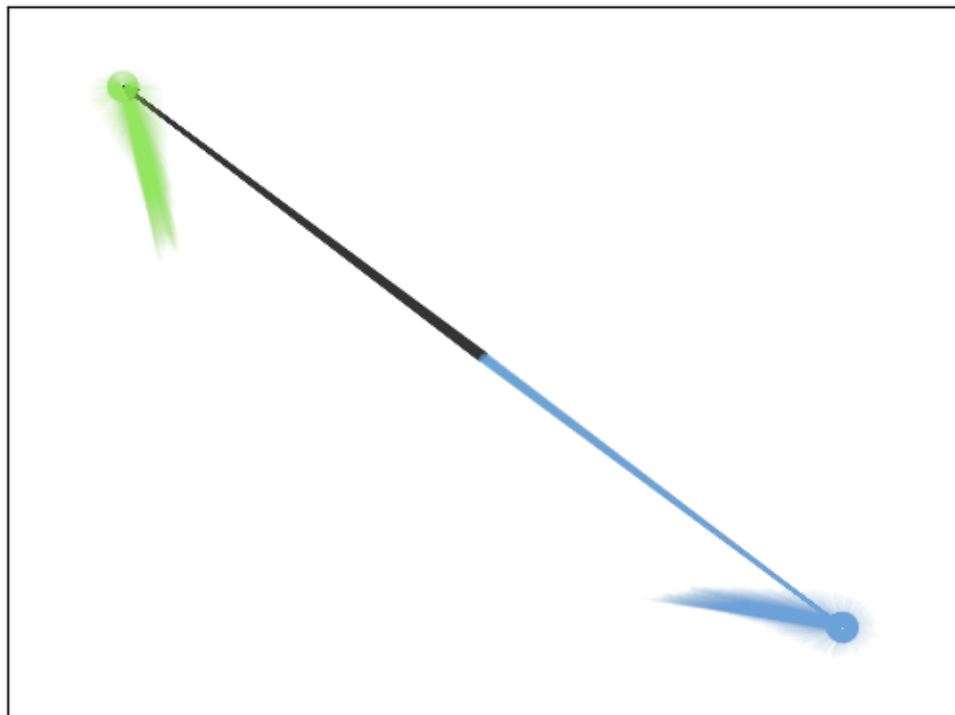
```

```

[67]: # Draw external edges
nx.draw_networkx(
    G, pos=pos, node_size=0,
    edgelist=external, edge_color="#333333",
    alpha=0.2, with_labels=False)

# Draw internal edges
nx.draw_networkx(
    G, pos=pos, node_size=0,
    edgelist=internal, edge_color=internal_color,
    alpha=0.05, with_labels=False)

```



```

[68]: all_clusters = []

all_clusters.extend(rus_cluster)
all_clusters.extend(ukr_eng_cluster)
all_clusters.extend(ukr_rus_cluster)

```

```

all_clusters.extend(ukr_ukr_cluster)

all_clusters_dict = {}

for cluster in all_clusters:
    if str(cluster[0]) in G.nodes:
        all_clusters_dict[str(cluster[0])] = str(cluster[0])

```

[69]: *# TODO: SAVE AS PDF*

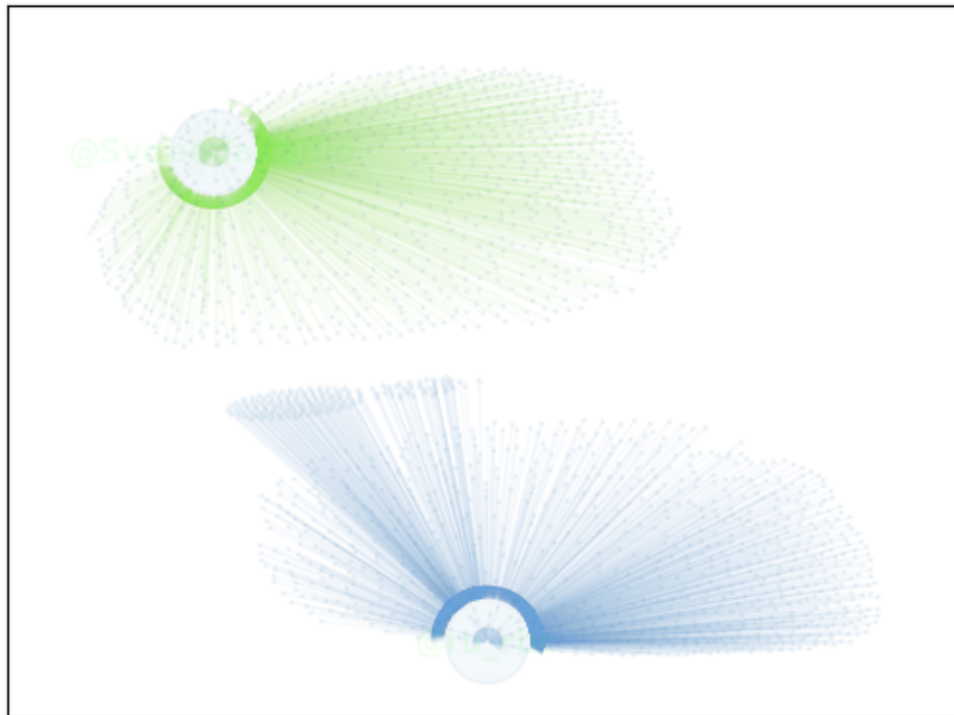
```

pos = nx.spring_layout(G, k=0.15)

G_node_degrees = dict(G.degree)

# Draw internal edges
nx.draw_networkx(
    G, pos=pos, node_size=[v * 1 for v in G_node_degrees.values()],
    edgelist=internal, edge_color=internal_color,
    alpha=0.05, with_labels=True, label='Group Follower Clusters',
    labels=all_clusters_dict, font_color='#00ff00', font_weight='bold')

```



```

[104]: def sort_graph_by_deg_desc(graph):
        return sorted(G.degree, key=lambda x: x[1], reverse=True)

```

```
[105]: G_deg_sorted = sort_graph_by_deg_desc(G)
```

```
[106]: print("Performing connectivity analysis on a graph")

node_connections = nx.all_pairs_node_connectivity(subgraph)
```

Performing connectivity analysis on a graph

```
[107]: def get_shared_nodes_within_group(node_connections, group_members):
        shared_connections = []
        for username, connections_dict in node_connections.items():
            for group_member in group_members:
                if group_member in connections_dict.keys() and
↪connections_dict[group_member] == 1:
                    shared_connections.append(username)
        return shared_connections
```

```
[108]: get_shared_nodes_within_group(node_connections, rus_cluster)
```

```
[108]: []
```

```
[109]: def get_shared_nodes_between_groups(node_connections, cluster_groups):
        shared_connections = {}
        for username, connections_dict in node_connections.items():
            for group_name, group_members in cluster_groups.items():
                for group_member in group_members:
                    if group_member in connections_dict.keys() and
↪connections_dict[group_member] == 1:
                        if group_name not in shared_connections.keys():
                            shared_connections.append(username)
                        else:
                            shared_connections[group_name] =
↪shared_connections[group_name].append(username)
        return shared_connections
```

```
[110]: TWITTER_CLUSTER_GROUPS = {
        "rus_cluster": russia_sources_rus,
        "ukr_eng_cluster": ukraine_sources_rus,
        "ukr_rus_cluster": ukraine_sources_ukr,
        "ukr_ukr_cluster": ukraine_sources_eng
    }
```

```
[111]: shared_connections_between_groups = get_shared_nodes_between_groups\
(node_connections, TWITTER_CLUSTER_GROUPS)
```

```
[112]: shared_connections_between_groups
```

[112]: {}

```
[113]: '''
import matplotlib.pyplot as plt

f, ax = plt.subplots(figsize=(10, 10))
plt.style.use('ggplot')
nodes = nx.draw_networkx_nodes(G, pos,
                              alpha=0.8)
nodes.set_edgecolor('k')
nx.draw_networkx_labels(G, pos, font_size=8)
nx.draw_networkx_edges(G, pos, width=1.0, alpha=0.2)
'''
```

```
[113]: "\nimport matplotlib.pyplot as plt\n\nf, ax = plt.subplots(figsize=(10,\n10))\nplt.style.use('ggplot')\nnodes = nx.draw_networkx_nodes(G, pos,\nalpha=0.8)\nnodes.set_edgecolor('k')\nnx.draw_networkx_labels(G, pos,\nfont_size=8)\nnx.draw_networkx_edges(G, pos, width=1.0, alpha=0.2)\n"
```

```
[114]: # TODO: analyze social network graph using NetworkX
# TODO: perhaps the network should be cleaned of low degree connected users
# for visualization purposes
# TODO: run the graph analytics on the original network w/o visualizing it
```

```
[115]: # TODO: node size should correspond to the number of followers the node has
# TODO: node color should correspond to the node group
```

Visualizing the Network using Neo4j

Analyzing group sentiments

```
[116]: # THIS CELL HITS TWITTER API

# NOTE: THIS FUNCTION PULLS THE TWEETS MENTIONING A PARTICULAR USER, NOT FROM
# A PARTICULAR USER

def get_user_tweet_mentions(username, num_tweets, entities=None):

    search_query = username + " OR "

    if entities is not None:
        # include all possible candidate names in a query
        for entity in entities:
            search_query += "entity:" + "'" + entity + "'" + " OR "

    # remove the last OR statement
    search_query = search_query[:-3]
```

```

search_query += "-is:retweet"

tweets = api.search_tweets(q = search_query, count = num_tweets,
                           tweet_mode="extended")

return tweets

```

```

[117]: # THIS CELL HITS TWITTER API

def get_user_tweets(user_id, num_tweets):

    # https://docs.tweepy.org/en/stable/client.html#tweepy.Client.
    ↪get_users_tweets
    # https://docs.tweepy.org/en/stable/expansions_and_fields.
    ↪html#expansions-parameter
    # https://dev.to/twitterdev/
    ↪a-comprehensive-guide-for-using-the-twitter-api-v2-using-tweepy-in-python-15d9

    tweets = client.get_users_tweets(id = user_id, max_results = num_tweets,
                                     exclude = ['retweets'], expansions='entities.
    ↪mentions.username')

    return tweets

```

```

[118]: # Sentiment in relation to a different group

# Tweets word Cloud

# Tweets HashTags Word Cloud

# Tweets events mapping? (Maybe)

```

```

[119]: # https://www.caida.org/catalog/software/walrus/

```

```

[120]: print("Pulling a sample of tweets for all clusters")

NUM_TWEETS = 5
USERNAME_IDX = 0
USER_ID_IDX = 1

cluster_tweets = {}

for cluster in all_clusters:
    cluster_tweets[cluster] = get_user_tweets(cluster[USER_ID_IDX], NUM_TWEETS)

```

Pulling a sample of tweets for all clusters

```
[121]: cluster_tweets
```

```
[121]: {('@itvru_news',  
    160881696): Response(data=[<Tweet id=1599396930378817536 text='  
        .           .           «           »  
            ,           -  
                -60: https://t.co/TLZGXMbk9L  
https://t.co/zytZG09Uof'>, <Tweet id=1599392876462424064 text='  
    ,           ,  
        ,           Politico.  
    ,  
        .   https://t.co/yQQUEYFWTk https://t.co/UP96iochry'>, <Tweet  
id=1599388669516783616 text='  
        .           -  
            -  
                https://t.co/iU4SpJNc60  
https://t.co/KDixdtxgbq'>, <Tweet id=1599381575929860097 text='  
        .           .  
            «       ».           ,  
                .  
        : https://t.co/Mj8JzsW2HC https://t.co/Trf02Ja8oh'>, <Tweet  
id=1599352240585965569 text='  
        .           .  
            .   https://t.co/QSpzV3IdNm  
https://t.co/MgfG6akIlp'>], includes={}, errors=[], meta={'result_count': 5,  
'newest_id': '1599396930378817536', 'oldest_id': '1599352240585965569',  
'next_token': '7140dibdnow9c7btw424ch4yoh0i9o0o58jo1ap46cd09'}),  
( '@ru_rbc',  
    269770723): Response(data=[<Tweet id=1599452244251918338 text='  
        .           ,  
            \n\nhttps://t.co/u2JkYcq88i'>, <Tweet id=1599444692449611777  
text='  
        .           ,  
            15  
            \n\nhttps://t.co/NsJoR5nMFa'>, <Tweet id=1599440896856150021 text='  
        ,  
            ,           \n\nhttps://t.co/tVyAZLLM0o'>,  
<Tweet id=1599438290649743362 text='           «       » (  
        )           ,           ,  
            \n\nhttps://t.co/xuiZBQsht8'>, <Tweet id=1599433932101111811  
text='           2  
        .           .           ,  
            -  
            \n\nhttps://t.co/JeId6PZE dh'>], includes={}, errors=[],  
meta={'next_token': '7140dibdnow9c7btw424ch59dvd2hfk2659y98o9agchd',  
'result count': 5, 'newest_id': '1599452244251918338', 'oldest_id':
```



```

'1599433932101111811'})),
('@dmitry_gordon',
1334400780): Response(data=[<Tweet id=1599460833712480256 text='
,
. 100 . - . !
https://t.co/2W9HvhvraK'>, <Tweet id=1599453054532669440 text='
, 57- , 3
.
https://t.co/t5pyJqRIBK'>, <Tweet id=1599444325829484545 text='\"#
\n# # # # # #
#stoprussia # # # #
# \nhttps://t.co/M1sktBFiUE'>, <Tweet id=1599435980292771840 text='#
, , , \nhttps://t.co/TFszxS78HD'>,
<Tweet id=1599424206063603712 text='# :
, - \n# # #
# # # #stoprussia # #
# # # \nhttps://t.co/KZrVsukbE6'>], includes={}, errors=[],
meta={'result_count': 5, 'newest_id': '1599460833712480256', 'oldest_id':
'1599424206063603712', 'next_token':
'7140dibdnow9c7btw424ch59ccy4l0l1uw22vdb6i0al2q'}),
('@SvobodaRadio',
47562921): Response(data=[<Tweet id=1599454260978384898 text='
.
\" \nhttps://t.co/V8SFQFjHDu'>, <Tweet id=1599442175376310273 text='\"
, , , -
, - ,
\". \n
\nhttps://t.co/oSke9wcrln'>, <Tweet id=1599417434779570178
text='
.
\nhttps://t.co/BaSSJiB39N'>, <Tweet id=1599412583554949120
text=' - +
- 2 , 2%
.
\nhttps://t.co/oWI71n6GXL'>, <Tweet id=1599393426025631744 text='\"
,
.
\nhttps://t.co/9WN5Mn3K3U'>], includes={}, errors=[],
meta={'next_token': '7140dibdnow9c7btw424ch4yujpwad528fupz1v8xh1ro',
'result_count': 5, 'newest_id': '1599454260978384898', 'oldest_id':
'1599393426025631744'}}})}

```

```
[122]: def convert_tweets_to_json(raw_tweets_dict):
        json_tweets = []
        for username, tweets in raw_tweets_dict.items():
            for tweet in tweets:
                json_tweet_str = json.dumps(tweet.text)
                json_tweet = json.loads(json_tweet_str)
                json_tweets.append(json_tweet)
        return json_tweets
```

```
[123]: tweet_status = api.get_status(1598709448129662977)

        print(tweet_status._json)
```

```
{'created_at': 'Fri Dec 02 16:03:29 +0000 2022', 'id': 1598709448129662977,
'id_str': '1598709448129662977', 'text': '
        "
        - "# " #
... https://t.co/ta77zbvZLc', 'truncated': True, 'entities': {'hashtags':
[{'text': ' ', 'indices': [88, 95]}, {'text': ' ', 'indices': [103,
110]}]}, 'symbols': [], 'user_mentions': [], 'urls': [{'url':
'https://t.co/ta77zbvZLc', 'expanded_url':
'https://twitter.com/i/web/status/1598709448129662977', 'display_url':
'twitter.com/i/web/status/1...', 'indices': [117, 140]}]}, 'source': '<a
href="https://mobile.twitter.com" rel="nofollow">Twitter Web App</a>',
'in_reply_to_status_id': None, 'in_reply_to_status_id_str': None,
'in_reply_to_user_id': None, 'in_reply_to_user_id_str': None,
'in_reply_to_screen_name': None, 'user': {'id': 1334400780, 'id_str':
'1334400780', 'name': ' ', 'screen_name': 'dmitry_gordon',
'location': ' ', 'description': ' ',
,
- @Gordonuacom.
https://t.co/Xt1COHTG44...', 'url': 'https://t.co/va3Q3oZcxT', 'entities': {'url':
{'urls': [{'url': 'https://t.co/va3Q3oZcxT', 'expanded_url':
'https://gordonua.com/', 'display_url': 'gordonua.com', 'indices': [0, 23]}]},
'description': {'urls': [{'url': 'https://t.co/Xt1COHTG44', 'expanded_url':
'http://t.me/dmytrogordon_o', 'display_url': 't.me/dmytrogordon_o', 'indices':
[87, 110]}]}}, 'protected': False, 'followers_count': 155132, 'friends_count':
304, 'listed_count': 771, 'created_at': 'Sun Apr 07 15:50:59 +0000 2013',
'favourites_count': 444, 'utc_offset': None, 'time_zone': None, 'geo_enabled':
False, 'verified': True, 'statuses_count': 168564, 'lang': None,
'contributors_enabled': False, 'is_translator': False, 'is_translation_enabled':
False, 'profile_background_color': '131516', 'profile_background_image_url':
'http://abs.twimg.com/images/themes/theme14/bg.gif',
'profile_background_image_url_https':
'https://abs.twimg.com/images/themes/theme14/bg.gif', 'profile_background_tile':
True, 'profile_image_url': 'http://pbs.twimg.com/profile_images/3714778768/Odd3d
f4ba1c8a979022cb50fdea8c5c5_normal.jpeg', 'profile_image_url_https': 'https://pb
s.twimg.com/profile_images/3714778768/Odd3df4ba1c8a979022cb50fdea8c5c5_normal.jp
eg', 'profile_banner_url':
```

```
[124]: cluster_tweets
```

15

```

\n\nhttps://t.co/NsJoR5nMFa'>, <Tweet id=1599440896856150021 text='
,
, \n\nhttps://t.co/tVyAZLlM0o'>,
<Tweet id=1599438290649743362 text=' « » (
)
,
\n\nhttps://t.co/xuiZBQsht8'>, <Tweet id=1599433932101111811
text='
2
.
-
\n\nhttps://t.co/JeId6PZEdh'>], includes={}, errors=[],
meta={'next_token': '7140dibdnow9c7btw424ch59dvd2hfk2659y98o9agchd',
'result_count': 5, 'newest_id': '1599452244251918338', 'oldest_id':
'1599433932101111811'}),
('@dmitry_gordon',
1334400780): Response(data=[<Tweet id=1599460833712480256 text=' :
,
100 - !
https://t.co/2W9HvhvraK'>, <Tweet id=1599453054532669440 text='
57- , 3
.
https://t.co/t5pyJqRIBK'>, <Tweet id=1599444325829484545 text='\"#
\"
\n# # # # #
#stoprussia # # #
# \nhttps://t.co/M1sktBFiUE'>, <Tweet id=1599435980292771840 text='#
' \"
\nhttps://t.co/TFszxS78HD'>,
<Tweet id=1599424206063603712 text='# :
,
\n# # #
# # # #stoprussia # #
# # # \nhttps://t.co/KZrVsukbE6'>], includes={}, errors=[],
meta={'result_count': 5, 'newest_id': '1599460833712480256', 'oldest_id':
'1599424206063603712', 'next_token':
'7140dibdnow9c7btw424ch59ccy4l0l1uw22vdb6i0al2q'}),
('@SvobodaRadio',
47562921): Response(data=[<Tweet id=1599454260978384898 text='
.
\"
\nhttps://t.co/V8SFQFjHDu'>, <Tweet id=1599442175376310273 text='\"
,
, -
,
. \n
\nhttps://t.co/oSke9wcrln'>, <Tweet id=1599417434779570178
text='
.
\nhttps://t.co/BaSSJiB39N'>, <Tweet id=1599412583554949120
text=' - +

```

```

- 2 , 2%
.\nhttps://t.co/oWI71n6GXL'>, <Tweet id=1599393426025631744 text='
",
.\n
,
.\n
.\nhttps://t.co/9WN5Mn3K3U'>], includes={}, errors=[],
meta={'next_token': '7140dibdnow9c7btw424ch4yujpwad528fupz1v8xh1ro',
'result_count': 5, 'newest_id': '1599454260978384898', 'oldest_id':
'1599393426025631744'}}}

```

```

[125]: TWEETS_DIR = 'tweets/'

def tweets_to_df(raw_tweets_dict):
    tweets_df = pd.DataFrame(
        {
            'Cluster Username' : pd.Series(dtype='str'),
            'Cluster ID' : pd.Series(dtype='int'),
            'Tweet Text' : pd.Series(dtype='str')
        }
    )
    for username, tweets in raw_tweets_dict.items():
        for tweet in tweets.data:
            new_df_row = {
                'Cluster Username' : username[0],
                'Cluster ID' : username[1],
                'Tweet Text' : str(tweet)
            }
            tweets_df = tweets_df.append(new_df_row, ignore_index=True)
    return tweets_df

def tweets_df_to_csv(tweets_df):
    tweets_df.to_csv(TWEETS_DIR + 'tweets.csv', index=False)
    return

```

```

[126]: tweets_df = tweets_to_df(cluster_tweets)
        tweets_df_to_csv(tweets_df)

```

Processing Collected Tweets Before Visualizing them

```

[127]: print("Use this function to clean the tweet's body")

def clean_tweet(tweet_body):
    # remove @ mentions from the tweet
    # text = re.sub(r'@[A-Za-z0-9]+', '', tweet_body)
    # remove the hashtags from tweets
    # text = re.sub(r'#', '', text)

```

```

# remove retweet
text = re.sub(r'RT[\s]+', '', tweet_body)
# remove hyperlinks
text = re.sub(r'https?:\/\/\S+', '', text)
return text

```

Use this function to clean the tweet's body

```

[128]: import string

def remove_ua_stopwords(tweet_body):
    stopwords_ua = pd.read_csv("stopwords_ua.txt", header=None,
    ↪names=['stopwords'])
    stop_words_ua = list(stopwords_ua.stopwords)
    text = "".join([word for word in tweet_body if word not in string.
    ↪punctuation])
    text = text.lower()
    tokens = re.split('\W+', text)
    text = [word for word in tokens if word not in stop_words_ua]
    return text

```

```

[129]: # this fucntion doesn't support Ukrainian

from nltk.corpus import stopwords

print("Supported langauges are: ")
print(stopwords.fileids())

def remove_stopwords(tweet_body, lang='english'):
    stop_words = set(stopwords.words(lang))
    word_tokens = nltk.word_tokenize(tweet_body)
    filtered_sentence = [w for w in word_tokens if not w.lower() in stop_words]
    filtered_sentence = ' '.join(filtered_sentence)
    return filtered_sentence

```

Supported langauges are:

```

['arabic', 'azerbaijani', 'basque', 'bengali', 'catalan', 'chinese', 'danish',
'dutch', 'english', 'finnish', 'french', 'german', 'greek', 'hebrew',
'hinglish', 'hungarian', 'indonesian', 'italian', 'kazakh', 'nepali',
'norwegian', 'portuguese', 'romanian', 'russian', 'slovene', 'spanish',
'swedish', 'tajik', 'turkish']

```

```

[130]: txt = "
    ↪ . [49] : « , , »
    ↪ , 1856 « »."

txt = remove_ua_stopwords(txt)

```

[illegible]

« » ,
[13] [14] .
- , [15] [16] . XVII .
[16] .

Visualizing Tweet Content using Word Cloud

[illegible]

[illegible]

#' ,
| | ,
' , | ,
' , | ,
' , | ,
' , | ,
| | ,
| | ,
| , | ,
| | ,
| , | ,
| | ,
| , | ,
| | ,
| | ,
| | ,
| | ,
| , | ,
| | ,
| | ,
| | ,
| , | ,
| | ,
| ~ ~ | ,
| | ,
| ~ ~ | ,
| . | ,
| | ,
| | ,
| | ,
| . | ,
| | ,
' , | ,
| : | , | ,
| | ,
| | ,
| | ,
| - | ,
| | ,
| | ,
' , | ,

[illegible]

[illegible]

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 104

```
[134]: # NOTE: wordcloud per cluster, not per group
def create_wordcloud(tweets_df_tokens, cluster_name):
    tweet_word_cloud = WordCloud(random_state=21,
                                   max_font_size=119).generate(' '.
    ↪join(tweets_df_tokens[cluster_name]))
    plt.figure(figsize=(20,10))
    plt.imshow(tweet_word_cloud, interpolation='bilinear')
    plt.axis('off')
    plt.show()
    return
```

```
[135]: create_wordcloud(tweets_df_tokens, '@SvobodaRadio')
```


Requirement already satisfied: numpy>=1.15.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (1.21.6)

Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (2.0.8)

Requirement already satisfied: srsly<3.0.0,>=2.4.3 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (2.4.5)

Requirement already satisfied: cymem<2.1.0,>=2.0.2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (2.0.7)

Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (4.64.1)

Requirement already satisfied: setuptools in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (60.2.0)

Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (1.0.3)

Requirement already satisfied: thinc<8.2.0,>=8.1.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (8.1.5)

Requirement already satisfied: typing-extensions<4.2.0,>=3.7.4 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (4.1.1)

Requirement already satisfied: typer<0.8.0,>=0.3.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (0.7.0)

Requirement already satisfied: requests<3.0.0,>=2.13.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (2.28.1)

Requirement already satisfied: pathy>=0.3.5 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (0.10.0)

Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.10 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (3.0.10)

Requirement already satisfied: jinja2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (3.1.2)

Requirement already satisfied: preshed<3.1.0,>=3.0.2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (3.0.8)

Requirement already satisfied: packaging>=20.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (21.3)

Requirement already satisfied: zipp>=0.5 in ./venv/lib/python3.7/site-packages (from catalogue<2.1.0,>=2.0.6->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (3.11.0)

Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in ./venv/lib/python3.7/site-packages (from packaging>=20.0->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (3.0.9)

Requirement already satisfied: smart-open<6.0.0,>=5.2.1 in ./venv/lib/python3.7/site-packages (from pathy>=0.3.5->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (5.2.1)

Requirement already satisfied: urllib3<1.27,>=1.21.1 in

```

./venv/lib/python3.7/site-packages (from
requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (1.26.13)
Requirement already satisfied: idna<4,>=2.5 in ./venv/lib/python3.7/site-
packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->en-core-web-
sm==3.4.1) (3.4)
Requirement already satisfied: certifi>=2017.4.17 in ./venv/lib/python3.7/site-
packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->en-core-web-
sm==3.4.1) (2022.9.24)
Requirement already satisfied: charset-normalizer<3,>=2 in
./venv/lib/python3.7/site-packages (from
requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (2.1.1)
Requirement already satisfied: confection<1.0.0,>=0.0.1 in
./venv/lib/python3.7/site-packages (from
thinc<8.2.0,>=8.1.0->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (0.0.3)
Requirement already satisfied: blis<0.8.0,>=0.7.8 in ./venv/lib/python3.7/site-
packages (from thinc<8.2.0,>=8.1.0->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1)
(0.7.9)
Requirement already satisfied: click<9.0.0,>=7.1.1 in ./venv/lib/python3.7/site-
packages (from typer<0.8.0,>=0.3.0->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1)
(8.1.3)
Requirement already satisfied: MarkupSafe>=2.0 in ./venv/lib/python3.7/site-
packages (from jinja2->spacy<3.5.0,>=3.4.0->en-core-web-sm==3.4.1) (2.1.1)
Requirement already satisfied: importlib-metadata in ./venv/lib/python3.7/site-
packages (from
click<9.0.0,>=7.1.1->typer<0.8.0,>=0.3.0->spacy<3.5.0,>=3.4.0->en-core-web-
sm==3.4.1) (5.1.0)
WARNING: You are using pip version 21.3.1; however, version 22.3.1 is
available.

You should consider upgrading via the '/Users/dennisfenchenko/NYU-
Fall-2022/info_vis/confirmation-bias-vis/venv/bin/python3 -m pip install
--upgrade pip' command.
Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')

```

```
[138]: !python3 -m spacy download uk_core_news_sm
```

```
nlp_ukr = spacy.load('uk_core_news_sm')
```

```

Collecting uk-core-news-sm==3.4.0
  Downloading https://github.com/explosion/spacy-
models/releases/download/uk_core_news_sm-3.4.0/uk_core_news_sm-3.4.0-py3-none-
any.whl (14.9 MB)
    | 14.9 MB 4.0 MB/s
Requirement already satisfied: pymorphy2-dicts-uk in
./venv/lib/python3.7/site-packages (from uk-core-news-sm==3.4.0)
(2.4.1.1.1460299261)

```

Requirement already satisfied: spacy<3.5.0,>=3.4.0 in ./venv/lib/python3.7/site-packages (from uk-core-news-sm==3.4.0) (3.4.3)

Requirement already satisfied: pymorphy2>=0.9 in ./venv/lib/python3.7/site-packages (from uk-core-news-sm==3.4.0) (0.9.1)

Requirement already satisfied: dawg-python>=0.7.1 in ./venv/lib/python3.7/site-packages (from pymorphy2>=0.9->uk-core-news-sm==3.4.0) (0.7.2)

Requirement already satisfied: docopt>=0.6 in ./venv/lib/python3.7/site-packages (from pymorphy2>=0.9->uk-core-news-sm==3.4.0) (0.6.2)

Requirement already satisfied: pymorphy2-dicts-ru<3.0,>=2.4 in ./venv/lib/python3.7/site-packages (from pymorphy2>=0.9->uk-core-news-sm==3.4.0) (2.4.417127.4579844)

Requirement already satisfied: srsly<3.0.0,>=2.4.3 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2.4.5)

Requirement already satisfied: pydantic!=1.8,!1.8.1,<1.11.0,>=1.7.4 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (1.10.2)

Requirement already satisfied: setuptools in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (60.2.0)

Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (1.0.9)

Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (4.64.1)

Requirement already satisfied: typer<0.8.0,>=0.3.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (0.7.0)

Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2.0.8)

Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.3.0)

Requirement already satisfied: packaging>=20.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (21.3)

Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.10 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.0.10)

Requirement already satisfied: cymem<2.1.0,>=2.0.2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2.0.7)

Requirement already satisfied: requests<3.0.0,>=2.13.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2.28.1)

Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (1.0.3)

Requirement already satisfied: jinja2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.1.2)

Requirement already satisfied: wasabi<1.1.0,>=0.9.1 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-

sm==3.4.0) (0.10.1)

Requirement already satisfied: thinc<8.2.0,>=8.1.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (8.1.5)

Requirement already satisfied: preshed<3.1.0,>=3.0.2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.0.8)

Requirement already satisfied: numpy>=1.15.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (1.21.6)

Requirement already satisfied: pathy>=0.3.5 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (0.10.0)

Requirement already satisfied: typing-extensions<4.2.0,>=3.7.4 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (4.1.1)

Requirement already satisfied: zipp>=0.5 in ./venv/lib/python3.7/site-packages (from catalogue<2.1.0,>=2.0.6->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.11.0)

Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in ./venv/lib/python3.7/site-packages (from packaging>=20.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.0.9)

Requirement already satisfied: smart-open<6.0.0,>=5.2.1 in ./venv/lib/python3.7/site-packages (from pathy>=0.3.5->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (5.2.1)

Requirement already satisfied: charset-normalizer<3,>=2 in ./venv/lib/python3.7/site-packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2.1.1)

Requirement already satisfied: idna<4,>=2.5 in ./venv/lib/python3.7/site-packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (3.4)

Requirement already satisfied: urllib3<1.27,>=1.21.1 in ./venv/lib/python3.7/site-packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (1.26.13)

Requirement already satisfied: certifi>=2017.4.17 in ./venv/lib/python3.7/site-packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2022.9.24)

Requirement already satisfied: confection<1.0.0,>=0.0.1 in ./venv/lib/python3.7/site-packages (from thinc<8.2.0,>=8.1.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (0.0.3)

Requirement already satisfied: blis<0.8.0,>=0.7.8 in ./venv/lib/python3.7/site-packages (from thinc<8.2.0,>=8.1.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (0.7.9)

Requirement already satisfied: click<9.0.0,>=7.1.1 in ./venv/lib/python3.7/site-packages (from typer<0.8.0,>=0.3.0->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (8.1.3)

Requirement already satisfied: MarkupSafe>=2.0 in ./venv/lib/python3.7/site-packages (from jinja2->spacy<3.5.0,>=3.4.0->uk-core-news-sm==3.4.0) (2.1.1)

Requirement already satisfied: importlib-metadata in ./venv/lib/python3.7/site-packages (from click<9.0.0,>=7.1.1->typer<0.8.0,>=0.3.0->spacy<3.5.0,>=3.4.0->uk-core-news-

sm==3.4.0) (5.1.0)

WARNING: You are using pip version 21.3.1; however, version 22.3.1 is available.

You should consider upgrading via the '/Users/dennisfenchenko/NYU-Fall-2022/info_vis/confirmation-bias-vis/venv/bin/python3 -m pip install --upgrade pip' command.

Download and installation successful

You can now load the package via `spacy.load('uk_core_news_sm')`

```
[139]: !python3 -m spacy download ru_core_news_sm
```

```
nlp_rus = spacy.load('ru_core_news_sm')
```

Collecting ru-core-news-sm==3.4.0

Downloading https://github.com/explosion/spacy-models/releases/download/ru_core_news_sm-3.4.0/ru_core_news_sm-3.4.0-py3-none-any.whl (15.3 MB)

| 15.3 MB 8.6 MB/s

Requirement already satisfied: pymorphy2>=0.9 in

./venv/lib/python3.7/site-packages (from ru-core-news-sm==3.4.0) (0.9.1)

Requirement already satisfied: spacy<3.5.0,>=3.4.0 in ./venv/lib/python3.7/site-packages (from ru-core-news-sm==3.4.0) (3.4.3)

Requirement already satisfied: dawg-python>=0.7.1 in ./venv/lib/python3.7/site-packages (from pymorphy2>=0.9->ru-core-news-sm==3.4.0) (0.7.2)

Requirement already satisfied: pymorphy2-dicts-ru<3.0,>=2.4 in ./venv/lib/python3.7/site-packages (from pymorphy2>=0.9->ru-core-news-sm==3.4.0) (2.4.417127.4579844)

Requirement already satisfied: docopt>=0.6 in ./venv/lib/python3.7/site-packages (from pymorphy2>=0.9->ru-core-news-sm==3.4.0) (0.6.2)

Requirement already satisfied: pathy>=0.3.5 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (0.10.0)

Requirement already satisfied: thinc<8.2.0,>=8.1.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (8.1.5)

Requirement already satisfied: srsly<3.0.0,>=2.4.3 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2.4.5)

Requirement already satisfied: jinja2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (3.1.2)

Requirement already satisfied: numpy>=1.15.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (1.21.6)

Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (4.64.1)

Requirement already satisfied: cymem<2.1.0,>=2.0.2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2.0.7)

Requirement already satisfied: preshed<3.1.0,>=3.0.2 in ./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (3.0.8)

Requirement already satisfied: pydantic!=1.8,!1.8.1,<1.11.0,>=1.7.4 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (1.10.2)

Requirement already satisfied: requests<3.0.0,>=2.13.0 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2.28.1)

Requirement already satisfied: typing-extensions<4.2.0,>=3.7.4 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (4.1.1)

Requirement already satisfied: setuptools in ./venv/lib/python3.7/site-packages
(from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (60.2.0)

Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.10 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (3.0.10)

Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (3.3.0)

Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (1.0.9)

Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (1.0.3)

Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2.0.8)

Requirement already satisfied: packaging>=20.0 in ./venv/lib/python3.7/site-packages
(from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (21.3)

Requirement already satisfied: typer<0.8.0,>=0.3.0 in ./venv/lib/python3.7/site-packages
(from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (0.7.0)

Requirement already satisfied: wasabi<1.1.0,>=0.9.1 in
./venv/lib/python3.7/site-packages (from spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (0.10.1)

Requirement already satisfied: zipp>=0.5 in ./venv/lib/python3.7/site-packages
(from catalogue<2.1.0,>=2.0.6->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0)
(3.11.0)

Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
./venv/lib/python3.7/site-packages (from
packaging>=20.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (3.0.9)

Requirement already satisfied: smart-open<6.0.0,>=5.2.1 in
./venv/lib/python3.7/site-packages (from pathy>=0.3.5->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (5.2.1)

Requirement already satisfied: urllib3<1.27,>=1.21.1 in
./venv/lib/python3.7/site-packages (from
requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (1.26.13)

Requirement already satisfied: charset-normalizer<3,>=2 in
./venv/lib/python3.7/site-packages (from
requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2.1.1)

Requirement already satisfied: certifi>=2017.4.17 in ./venv/lib/python3.7/site-packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2022.9.24)

Requirement already satisfied: idna<4,>=2.5 in ./venv/lib/python3.7/site-packages (from requests<3.0.0,>=2.13.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (3.4)

Requirement already satisfied: blis<0.8.0,>=0.7.8 in ./venv/lib/python3.7/site-packages (from thinc<8.2.0,>=8.1.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (0.7.9)

Requirement already satisfied: confection<1.0.0,>=0.0.1 in ./venv/lib/python3.7/site-packages (from thinc<8.2.0,>=8.1.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (0.0.3)

Requirement already satisfied: click<9.0.0,>=7.1.1 in ./venv/lib/python3.7/site-packages (from typer<0.8.0,>=0.3.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (8.1.3)

Requirement already satisfied: MarkupSafe>=2.0 in ./venv/lib/python3.7/site-packages (from jinja2->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (2.1.1)

Requirement already satisfied: importlib-metadata in ./venv/lib/python3.7/site-packages (from click<9.0.0,>=7.1.1->typer<0.8.0,>=0.3.0->spacy<3.5.0,>=3.4.0->ru-core-news-sm==3.4.0) (5.1.0)

WARNING: You are using pip version 21.3.1; however, version 22.3.1 is available.

You should consider upgrading via the '/Users/dennisfenchenko/NYU-Fall-2022/info_vis/confirmation-bias-vis/venv/bin/python3 -m pip install --upgrade pip' command.

Download and installation successful

You can now load the package via spacy.load('ru_core_news_sm')

[140]: *# Spacy has vocabulary for English, Ukrainian and Russian languages*

```
text = ("          ,          #          "
        "          # ,          # # # # #          "
        "\u2192#          #stoprussia # #          # ")

doc = nlp_rus(text)

# Find named entities, phrases and concepts
for entity in doc.ents:
    print(entity.text)
    print(entity.text, entity.label_)
```

PER

PER

LOC

ORG

```
[141]: print("Performing sentiment analysis on tweets")

analyzer = SentimentIntensityAnalyzer()

analyzer.polarity_scores("                ,                !!!")
```

Performing sentiment analysis on tweets

```
[141]: {'neg': 0.0, 'neu': 0.281, 'pos': 0.719, 'compound': 0.882}
```

```
[142]: analyzer.polarity_scores("                .                .□

↪                .                .                □
↪                https://t.co/xUgOmV4iWN https://t.co/hN8uLeHwzu")
```

```
[142]: {'neg': 0.234, 'neu': 0.766, 'pos': 0.0, 'compound': -0.9042}
```

```
[168]: tweets_df.head()
```

```
[168]:   Cluster Username  Cluster ID \
0      @1tvru_news  160881696
1      @1tvru_news  160881696
2      @1tvru_news  160881696
3      @1tvru_news  160881696
4      @1tvru_news  160881696
```

				Tweet Text
0	-50	
1		,	...	
2		
3			...	
4			...	

Performing Entity and Sentiment Analysis

```
[158]: def augment_entity_to_df(tweets_df, lang='english'):

    new_df = tweets_df.copy(deep=True)

    entities_col = []

    for _, row in tweets_df.iterrows():
        entities = []
        tweet_text = row['Tweet Text']
```

```

tweet_text = clean_tweet(tweet_text)
if lang == 'english':
    nlp_doc = nlp_eng(tweet_text)
elif lang == 'russian':
    nlp_doc = nlp_rus(tweet_text)
elif lang == 'ukrainian':
    nlp_doc = nlp_ukr(text)
else: raise Exception('Language not supported.')

# NOTE: enteteties can be filtered based on the entity. e.g. only
# people
for entity in nlp_doc.ents:
    entities.append(entity.text)

entities = list(set(entities))

entities_col.append(entities)

new_df['Entities'] = entities_col

return new_df

```

```

[159]: new_df = augment_entity_to_df(tweets_df, lang='russian')
print(new_df)

```

	Cluster Username	Cluster ID \
0	@itvru_news	160881696
1	@itvru_news	160881696
2	@itvru_news	160881696
3	@itvru_news	160881696
4	@itvru_news	160881696
5	@ru_rbc	269770723
6	@ru_rbc	269770723
7	@ru_rbc	269770723
8	@ru_rbc	269770723
9	@ru_rbc	269770723
10	@dmitry_gordon	1334400780
11	@dmitry_gordon	1334400780
12	@dmitry_gordon	1334400780
13	@dmitry_gordon	1334400780
14	@dmitry_gordon	1334400780
15	@SvobodaRadio	47562921
16	@SvobodaRadio	47562921
17	@SvobodaRadio	47562921
18	@SvobodaRadio	47562921
19	@SvobodaRadio	47562921

			Tweet Text \
0	-50
1		,	...
2	
3			...
4			...
5			...
6			...
7		,	...
8	«	» ()...
9			...
10	:	,	...
11		,	...
12	"#	"	...
13	#	,	...
14	#	:	...
15			...
16	"	,	...
17			...
18	-	+	...
19	"	,	...

		Entities
0	[, ,]
1	[, , Politico, , ...
2		[,]
3	[, , , ,]
4	[, , , ...
5		[, , ,]
6	[, , ...
7		[]
8		[, , ,]
9		[,]
10		[,]
11	[, , , ,]
12		[, ,]
13	[, , , , , ...
14		[, , , ,]
15	[, , ...
16		[, , ,]
17		[]
18		[+,]
19		[, , ,]

```
[160]: def augment_sent_to_df(tweets_df, lang='english'):
```

```
new_df = tweets_df.copy(deep=True)
```

```

analyzer = SentimentIntensityAnalyzer()

neg_sent = []
pos_sent = []

for _, row in tweets_df.iterrows():
    tweet_text = row['Tweet Text']
    tweet_text = clean_tweet(tweet_text)
    tweet_sentiment_scores = analyzer.polarity_scores(tweet_text)
    neg_sent.append(tweet_sentiment_scores['neg'])
    pos_sent.append(tweet_sentiment_scores['pos'])

new_df['Neg'] = neg_sent
new_df['Pos'] = pos_sent

return new_df

```

```
[169]: global_graph_pd.head()
```

```
[169]:
```

	username	user_id	cluster_name	cluster_id	\
0		1599163293746302976	@ru_rbc	269770723	
1	Pletunoff	878179754804957184	@ru_rbc	269770723	
2	Mila	1599157008330096640	@ru_rbc	269770723	
3	Set Mortensen	1599155538507567104	@ru_rbc	269770723	
4	sweetdreamslizz	1599154015195086848	@ru_rbc	269770723	

	cluster_follow_count	group_id
0	535342	0
1	535342	0
2	535342	0
3	535342	0
4	535342	0

```
[187]: def augment_num_followers_to_df(tweets_df, global_df):
    new_df = tweets_df.copy(deep=True)

    new_df.rename(columns = {'Cluster ID': 'cluster_id'}, inplace=True)

    new_df = pd.merge(new_df, global_df[['cluster_id',
                                         'cluster_follow_count']],
                      on='cluster_id', how='right')

    return new_df

```

```
[161]: new_df = augment_sent_to_df(new_df, lang='russian')
print(new_df)
```

	Cluster Username	Cluster ID \
0	@itvru_news	160881696
1	@itvru_news	160881696
2	@itvru_news	160881696
3	@itvru_news	160881696
4	@itvru_news	160881696
5	@ru_rbc	269770723
6	@ru_rbc	269770723
7	@ru_rbc	269770723
8	@ru_rbc	269770723
9	@ru_rbc	269770723
10	@dmitry_gordon	1334400780
11	@dmitry_gordon	1334400780
12	@dmitry_gordon	1334400780
13	@dmitry_gordon	1334400780
14	@dmitry_gordon	1334400780
15	@SvobodaRadio	47562921
16	@SvobodaRadio	47562921
17	@SvobodaRadio	47562921
18	@SvobodaRadio	47562921
19	@SvobodaRadio	47562921

	Tweet Text \
0	-50
1	, ...
2
3	...
4	...
5	...
6	...
7	, ...
8	« » ()...
9	...
10	: , ...
11	, ...
12	"# " ...
13	# , , ...
14	# : ...
15	...
16	" , ...
17	...
18	- + ...
19	" , ...

	Entities	Neg	Pos
0	[, ,] 0.000 0.000		
1	[, , Politico, , ... 0.289 0.115		
2	[,] 0.062 0.062		


```

3  [ , , , , ] 0.079 0.000
4  [ , , , , ... 0.059 0.000
5  [ , , , , ] 0.062 0.161
6  [ , , , , ... 0.000 0.126
7  [ , , , , ] 0.311 0.000
8  [ , , , , ] 0.135 0.000
9  [ , , , , ] 0.053 0.000
10 [ , , , , ] 0.141 0.180
11 [ , , , , ] 0.193 0.000
12 [ , , , , ] 0.333 0.000
13 [ , , , , , ... 0.263 0.000
14 [ , , , , , ] 0.303 0.000
15 [ , , , , , ... 0.197 0.180
16 [ , , , , , ] 0.349 0.000
17 [ , , , , , ] 0.176 0.000
18 [ +, , , , ] 0.127 0.057
19 [ , , , , , ] 0.195 0.096

```

```
[188]: print("Augmenting the number of cluster followers to the DataFrame")

tweets_df = augment_num_followers_to_df(new_df, global_graph_pd)
```

Augmenting the number of cluster followers to the DataFrame

```
[190]: print(tweets_df.head())
```

```

Cluster Username  cluster_id \
0      @ru_rbc    269770723
1      @ru_rbc    269770723
2      @ru_rbc    269770723
3      @ru_rbc    269770723
4      @ru_rbc    269770723

Tweet Text \
0      ...
1      ...
2      ,
3      « » ( )...
4      ...

Entities  Neg  Pos \
0  [ , , , , ] 0.062 0.161
1  [ , , , , ... 0.000 0.126
2  [ , , , , ] 0.311 0.000
3  [ , , , , ] 0.135 0.000
4  [ , , , , ] 0.053 0.000

cluster_follow_count

```

```

0          535342
1          535342
2          535342
3          535342
4          535342

```

Creating Sentiment Visualizations in Plotly

```
[162]: import plotly.express as px
```

```
[163]: df = px.data.iris()
fig = px.scatter(df, x="sepal_width", y="sepal_length", color="species",
                 size='petal_length', hover_data=['petal_width'])
fig.show()
```

```
[164]: print(df.head())
```

	sepal_length	sepal_width	petal_length	petal_width	species	species_id
0	5.1	3.5	1.4	0.2	setosa	1
1	4.9	3.0	1.4	0.2	setosa	1
2	4.7	3.2	1.3	0.2	setosa	1
3	4.6	3.1	1.5	0.2	setosa	1
4	5.0	3.6	1.4	0.2	setosa	1

```
[192]: tweets_df.head()
```

```
[192]: Cluster Username cluster_id \
0      @ru_rbc  269770723
1      @ru_rbc  269770723
2      @ru_rbc  269770723
3      @ru_rbc  269770723
4      @ru_rbc  269770723
```

```

                                Tweet Text \
0                                     ...
1                                     ...
2                                     ...
3      «  » (                        )...
4                                     ...

```

```

                                Entities  Neg  Pos \
0      [ , , , ] 0.062 0.161
1  [ , , ... 0.000 0.126
2                                     [] 0.311 0.000
3      [ , , , ] 0.135 0.000
4      [ , ] 0.053 0.000

```

```
cluster_follow_count
```

```

0          535342
1          535342
2          535342
3          535342
4          535342

```

```

[195]: tweet_sent_fig = px.scatter(tweets_df, x="Pos", y="Neg",
                                   color="Cluster Username",
                                   size="cluster_follow_count",
                                   hover_data=['Entities'])

tweet_sent_fig.show()

```

```

[211]: FIGURES_DIR = 'figures/'

def write_figures_to_file(fig, fig_name, file_format):
    fig.write_image(FIGURES_DIR + fig_name + '.' + file_format)
    return

```

```

[212]: ! pip install -U kaleido

```

Requirement already satisfied: kaleido in ./venv/lib/python3.7/site-packages (0.2.1)

WARNING: You are using pip version 21.3.1; however, version 22.3.1 is available.

You should consider upgrading via the '/Users/dennisfenchenko/NYU-Fall-2022/info_vis/confirmation-bias-vis/venv/bin/python -m pip install --upgrade pip' command.

```

[213]: write_figures_to_file(tweet_sent_fig, 'tweet_sent', 'pdf')

```

```

-----
ValueError                                Traceback (most recent call last)
/var/folders/4l/kfc4gh5d1jn6zkrmv51vvbyw0000gn/T/ipykernel_92412/67443788.py in
↳<module>
----> 1 write_figures_to_file(tweet_sent_fig, 'tweet_sent', 'pdf')

/var/folders/4l/kfc4gh5d1jn6zkrmv51vvbyw0000gn/T/ipykernel_92412/2786137543.py
↳in write_figures_to_file(fig, fig_name, file_format)
      2
      3 def write_figures_to_file(fig, fig_name, file_format):
----> 4     fig.write_image(FIGURES_DIR + fig_name + '.' + file_format)
      5     return

```

```

~/NYU-Fall-2022/info_vis/confirmation-bias-vis/venv/lib/python3.7/site-packages
↳plotly/basedatatypes.py in write_image(self, *args, **kwargs)
    3827         import plotly.io as pio
    3828
-> 3829         return pio.write_image(self, *args, **kwargs)
    3830
    3831     # Static helpers

~/NYU-Fall-2022/info_vis/confirmation-bias-vis/venv/lib/python3.7/site-packages
↳plotly/io/_kaleido.py in write_image(fig, file, format, scale, width, height,
↳validate, engine)
    272         height=height,
    273         validate=validate,
--> 274         engine=engine,
    275     )
    276

~/NYU-Fall-2022/info_vis/confirmation-bias-vis/venv/lib/python3.7/site-packages
↳plotly/io/_kaleido.py in to_image(fig, format, width, height, scale, validate
↳engine)
    136 which can be installed using pip:
    137     $ pip install -U kaleido
--> 138 """
    139     )
    140

ValueError:
Image export using the "kaleido" engine requires the kaleido package,
which can be installed using pip:
    $ pip install -U kaleido

```

Visualizing Network Graph using NetworkX and Bokeh

<https://melaniewalsh.github.io/Intro-Cultural-Analytics/06-Network-Analysis/02-Making-Network-Viz-with-Bokeh.html>

```
[280]: # creating a copy of the original network G
```

```
G_vis = G.copy()
```

```
[281]: from bokeh.io import output_notebook, show, save
from bokeh.models import Range1d, Circle, ColumnDataSource, MultiLine,
↳EdgesAndLinkedNodes, NodesAndLinkedEdges
from bokeh.plotting import figure
from bokeh.plotting import from_networkx
from bokeh.palettes import Blues8, Reds8, Purples8, Oranges8, Viridis8,
↳Spectral8
```

```

from bokeh.transform import linear_cmap
from networkx.algorithms import community

```

```

[282]: modularity_class = dict(zip(global_graph_pd.username,
                                global_graph_pd.group_id))

modularity_color = {}

for username, group_id in modularity_class.items():
    modularity_color[username] = Spectral8[group_id]

```

```

[283]: # Add modularity class and color as attributes from the network above
cluster_name_follower_count_dict = dict(zip(global_graph_pd.cluster_name,
                                             global_graph_pd.
↳cluster_follow_count))

nx.set_node_attributes(G_vis, modularity_class, 'modularity_class')
nx.set_node_attributes(G_vis, modularity_color, 'modularity_color')
# nx.set_node_attributes(G_vis, name='cluster_name_follower_count_dict',
#                        values=cluster_name_follower_count_dict)

```

```

[294]: # make clusters bigger in size

node_sizes = {}

for node in G_vis.nodes:
    if node in all_clusters_dict.keys():
        node_sizes[node] = 20
    else:
        node_sizes[node] = 5

```

```

[295]: nx.set_node_attributes(G_vis, name='node_size',
                             values=node_sizes)

```

```

[299]: from bokeh.models import EdgesAndLinkedNodes, NodesAndLinkedEdges

#Choose colors for node and edge highlighting
node_highlight_color = 'white'
edge_highlight_color = 'black'

#Choose attributes from G network to size and color by - setting manual size (e.
↳g. 10) or color (e.g. 'skyblue') also allowed

size_by_this_attribute = 'node_size'
color_by_this_attribute = 'modularity_color'

#Pick a color palette - Blues8, Reds8, Purples8, Oranges8, Viridis8

```

```

color_palette = Blues8

#Choose a title!
title = 'Twitter Social Media Network'

#Establish which categories will appear when hovering over each node
HOVER_TOOLTIPS = [
    ("Username", "@index"),
    ("Follower count", "@cluster_name_follower_count_dict"),
    ("Cluster Class", "@modularity_class"),
    ("Cluster Color", "$color[swatch]:modularity_color"),
]

#Create a plot - set dimensions, toolbar, and title
plot = figure(tooltips = HOVER_TOOLTIPS,
              tools="pan,wheel_zoom,save,reset", active_scroll='wheel_zoom',
              x_range=Range1d(-10.1, 10.1), y_range=Range1d(-10.1, 10.1),
              title=title)

#Create a network graph object
# https://networkx.github.io/documentation/networkx-1.9/reference/generated/
# networkx.drawing.layout.spring\_layout.html
network_graph = from_networkx(G_vis, nx.spring_layout, scale=10, center=(0,0),
                              k=0.2)

#Set node sizes and colors according to node degree (color as category from
# attribute)
network_graph.node_renderer.glyph = Circle(size=size_by_this_attribute,
# fill_color=color_by_this_attribute)
# network_graph.node_renderer.glyph = Circle(fill_color=color_by_this_attribute)

#Set node highlight colors
# network_graph.node_renderer.hover_glyph = Circle(size=size_by_this_attribute,
# fill_color=node_highlight_color, line_width=2)
network_graph.node_renderer.hover_glyph =
# Circle(fill_color=node_highlight_color, line_width=2)

# network_graph.node_renderer.selection_glyph =
# Circle(size=size_by_this_attribute, fill_color=node_highlight_color,
# line_width=2)
network_graph.node_renderer.selection_glyph =
# Circle(fill_color=node_highlight_color, line_width=2)

#Set edge opacity and width
network_graph.edge_renderer.glyph = MultiLine(line_alpha=0.5, line_width=1,
                                              line_color='grey')

```

```

#Set edge highlight colors
network_graph.edge_renderer.selection_glyph = □
    ↪ MultiLine(line_color=edge_highlight_color, line_width=2)
network_graph.edge_renderer.hover_glyph = □
    ↪ MultiLine(line_color=edge_highlight_color, line_width=2)

#Highlight nodes and edges
network_graph.selection_policy = NodesAndLinkedEdges()
network_graph.inspection_policy = NodesAndLinkedEdges()

plot.renderers.append(network_graph)

show(plot)

```

```
[300]: save(plot, filename=f"{title}.html")
```

```
[300]: '/Users/dennisfenchenko/NYU-Fall-2022/info_vis/confirmation-bias-vis/Twitter
Social Media Network.html'
```

```
[ ]:
```