Data visualization course

Laboratory work 7

Visualize social connections in your Instagram profile

Analysis of connections in social networks is a useful and effective tool for identifying the main specificity of human relations in social groups. These tools allow you to determine the quantitative and qualitative parameters of the formed connections.

One of the most popular social networks is Instagram. A special API is used to interact with the social network. It is with its help that you can get all the necessary information.

In most cases, you can install the Instagram API using the following command:

```
pip install instagrapi
```

An example of using the Instagram API is shown in Listing 1.

Remember that in order to log in, you need to open the letter you received in your mail with a 6-digit authorization code from Instagram.

This example available in google colab by link Open in Colab
https://colab.research.google.com/github/a-vodka/dv/blob/master/examples/labinstagraph.ipynb

Listing 1 – An example of using the Instagram API

```
import matplotlib.pyplot as plt
import networkx as nx
from instagrapi import Client
from instagrapi.exceptions import TwoFactorRequired
from instagrapi.types import UserShort

MAX_FOLLOWINGS_COUNT = 20  # Max number of followers
# Login to Instagram
instagram_client = Client()

# Set deley according to recomendation
instagram_client.delay_range = [1, 5]

USERNAME = input("input user name:")
PASSWORD = input("input user password:")

# check for login and password is not empty
assert USERNAME, 'LOGIN should be inputed'
assert PASSWORD, 'PASSWORD should be inputed'

# try-except check for two factor autorization
try:
    instagram_client.login(USERNAME, PASSWORD)
    print("Logged in successfully")
except TwoFactorRequired:
    print("Two-factor authentication required. Please disable it in your Instagram
settings.")
    raise

# Fetch our followers
my_followings = instagram_client.user_following(user_id=instagram_client.user_id,
amount=MAX_FOLLOWINGS_COUNT)

my_followings_names = [user.username for user in my_followings.values()]

G = nx.Graph()
G.add_node(instagram_client.username, label=instagram_client.username)

for following in my_followings.values():
    G.add_node(following.username, label=following.full_name)
    G.add_edge(instagram_client.username, following.username)

# Fetch followers of our followings
for person in my_followings.values():
    try:
        print(f'Processing following person: [{person.username}] followings...')
        following_followings = instagram_client.user_following(person.pk)
        for following in following_followings.values():
            if following.username in my_followings_names:
                G.add_node(following.username, label=following.full_name)
                G.add_edge(person.username, following.username)
    except Exception as e:
        print(f"Error fetching data for {person.username}: {e}")

print('Saving graph...')
# Save the graph in gexf format
nx.write_gexf(G, "InstaFriends.gexf")

# Visualize the graph
print("Drawing...")
nx.draw_spring(G, with_labels=True, font_weight='bold', font_size=5)
plt.savefig('InstaGraf.png', dpi=600)
plt.show()
```

The results of the program are presented in Figures 1 and 2.

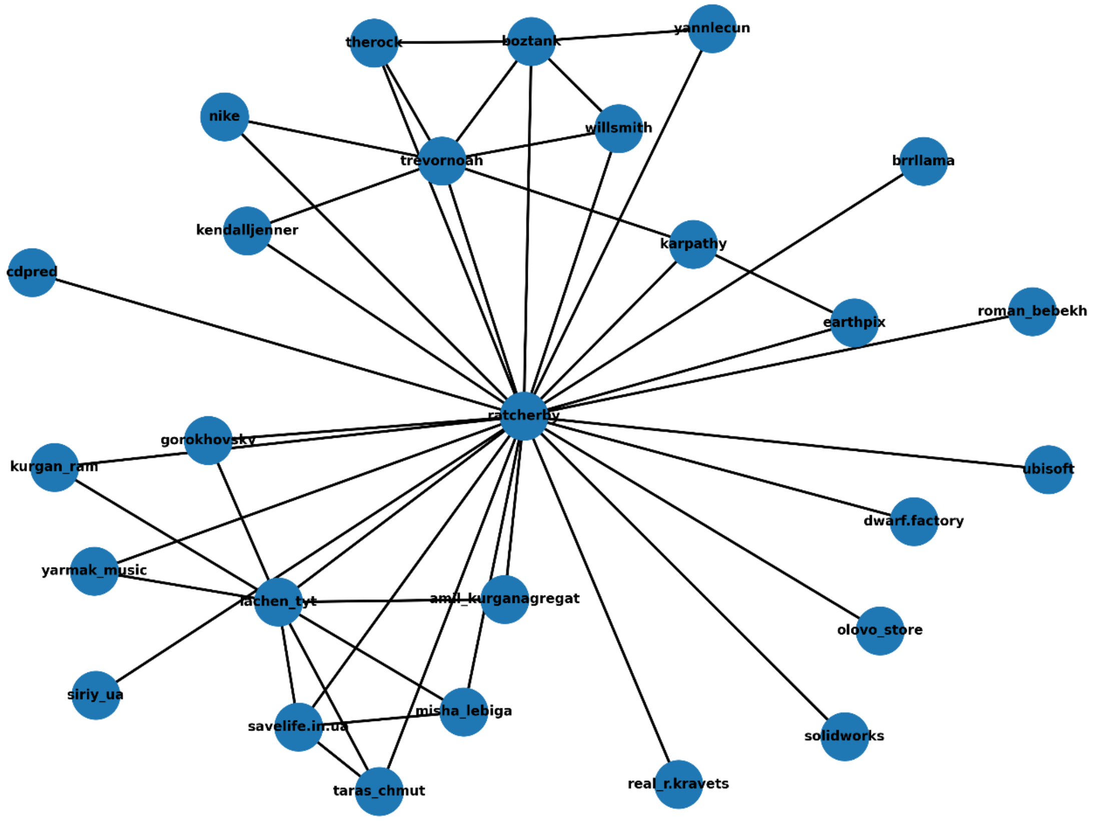

Figure 1 – Graph visualization result using networkX

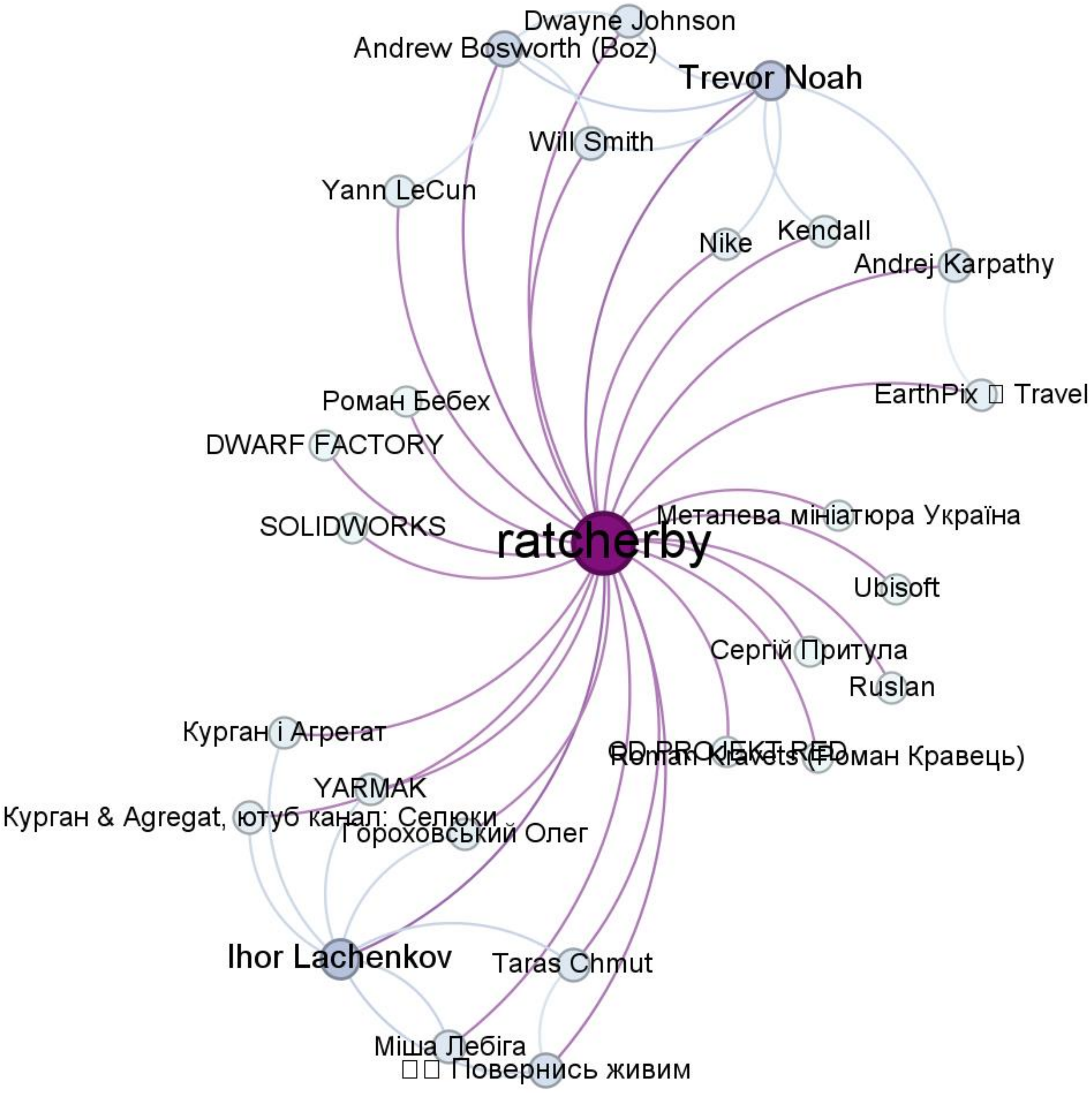

Figure 2 – Graph visualization result using gephi

Task

Build a graph of your followers on the social network Instagram. Visualize the graph using the networkx and gephi libraries. Also, determine the characteristics according to the variant using networkx or gephi:

1. Number of nodes in the graph.
2. Number of edges in the graph.
3. Average degree of the vertices.
4. Diameter of the graph.
5. Density of the graph.
6. Average clustering coefficient.
7. Average path length.
8. Modularity of the graph.
9. Ratio of the number of edges to the number of nodes.