Mark Gameng

CS 579

Project 1

For this project, I decided to use Twitter to make a follower and followee relationship using IIT's president, Raj Echambadi. His twitter is @rechambadi with 445 followers. This is not that much followers but due to Twitter's API rate limiting, going through all of his followers would take half a day. So, instead I decided to randomly choose 50 of his followers and get their following/followers for the network.

To get the Twitter followers and information, I used the Twitter API and Tweepy, a python library for that API. Essentially, I used my credentials such as my bearer token and then found a user's following and followers through the API. I then saved the "links" or edges, such as whether a user is following a person, or getting followed by a person. Here are some screenshots below to show the code that I used to get the data.

```
8 client = tweepy.Client(bearer_token=bearer_token)
```

```
1 # to get id then use get_users_followers
2 user = client.get_user(username = 'rechambadi').data
3 print('Name:', user.name)
4 print('Username:', user.username)
5 print('ID:', user.id)
```

```
Name: Raj Echambadi
Username: rechambadi
ID: 21366430
```

```
FOLLOWERS = []
next_tokens = []

followers = client.get_users_followers(id = user.id, max_results = 1000)
FOLLOWERS.extend(followers.data)
#print(followers.meta, type((followers.meta)))

# if more followers, need to request more
while ('next_token' in followers.meta and followers.meta['next_token']):
    next_tokens.append(followers.meta['next_token'])
    followers = client.get_users_followers(id = user.id, pagination_token = followers.meta['next_token'],
                                           max_results = 1000)
    #print(followers.meta, type((followers.meta)))
    FOLLOWERS.extend(followers.data)
    time.sleep(61)
```

```
# randomly choose 50 of followers to further expand
FOLLOWERS = random.sample(FOLLOWERS, 50)

EDGES = [] # [(follower, user), (follower1, user), (follower, user1) ...]

for f in FOLLOWERS:
    EDGES.append((f.username, user.username))

# to save, just incase errors
json.dump(EDGES, open('edges.json', 'w', encoding = 'utf8'), indent = 1)

# this gets all the followers of rechambadi done.
# now onto his followers
```

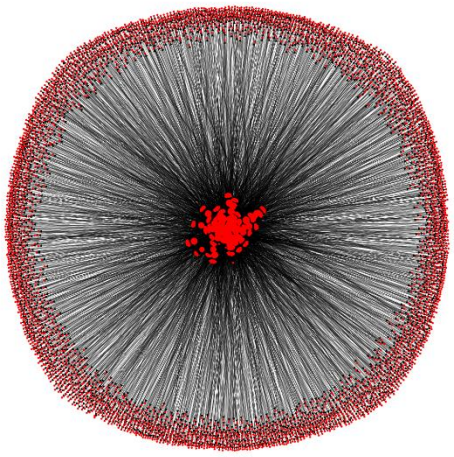
```
for f in FOLLOWERS:

    f_followers = client.get_users_followers(id = f.id, max_results = 100)
    time.sleep(61)
    if f_followers.data:
        for wers in f_followers.data:
            EDGES.append((wers.username, f.username))

    f_following = client.get_users_following(id = f.id, max_results = 100)
    time.sleep(61)
    if f_following.data:
        for wing in f_following.data:
            EDGES.append((f.username, wing.username))

    # to save, just incase errors
    json.dump(EDGES, open('edges.json', 'w', encoding = 'utf8'), indent = 1)

    total += 1
    #print(total, f.username, 'done')
```

I also saved all the edges in a json file so you can use it without having to use Twitter API since it requires a bearer token. Also, take note, that this took a few hours, due to rate limitations. Twitter API only allows 15 follower lookups in 15 minutes, so essentially, 1 follower lookup every minute. Here are some examples of the edges from @rechambadi.

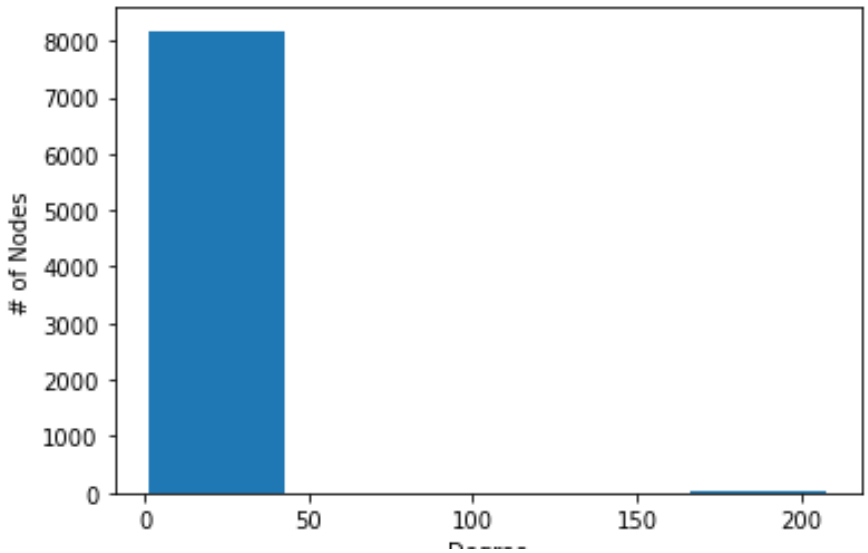
```
[('drcurtisodom', 'rechambadi'),
 ('IllinoisEMBA', 'rechambadi'),
 ('FeliciaLassk', 'rechambadi'),
 ('thejoeymak', 'rechambadi'),
 ('mdwiemer', 'rechambadi'),
 ('Jenyule', 'rechambadi'),
 ('carimclean', 'rechambadi'),
 ('Betsyabdallah', 'rechambadi'),
 ('sathisharucham', 'rechambadi'),
 ('priksp08', 'rechambadi'),
 ('Handyside', 'rechambadi'),
```

These edges or tuples are essentially (follower, user). So, in this case, @drcurstisodom and @Handyside are following @rechambadi. @drcurstisodom -> @rechambadi

At the end, I got **8226** total nodes and **9066** edges. I will clean this graph later in the paper to have a better visualization. But without any edits, and filters, this is what the default graph looks like. A higher resolution png file will be included in the submission as well.



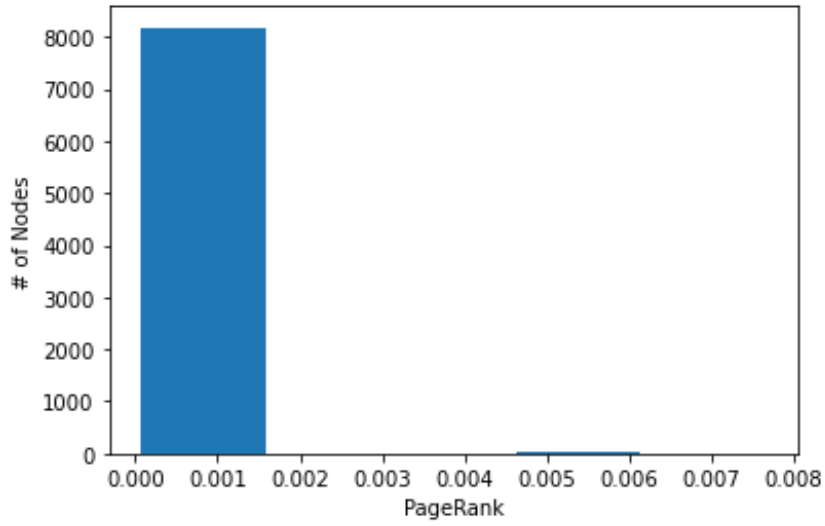
With this graph, the **degree distribution** is as such.



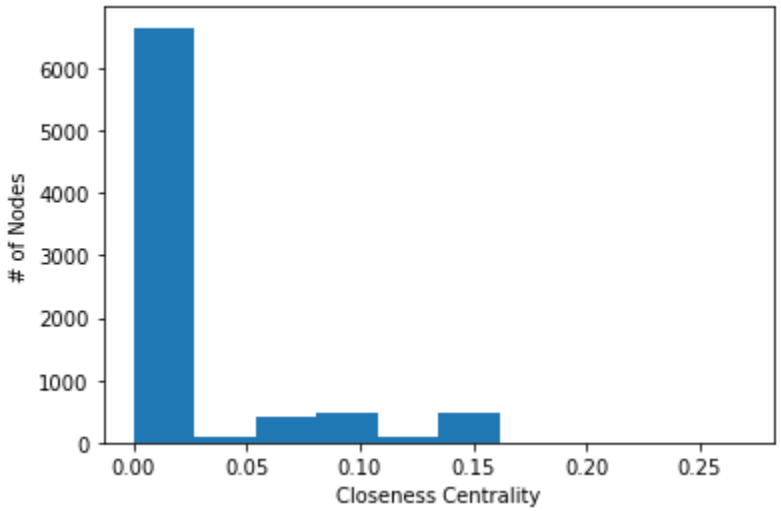
As you can see, most nodes, or users, have a degree of less than 25. Specifically, **7452** nodes have a degree of **1**, and **668** nodes have a degree of **2**. The average degree of this network is **2.2** and the max degree is **208** from NU_Business. This is interesting because that is Northeastern School of Business, and this network was built from IIT's President's followers. Here is the total degree distribution.

```
{200: 10, 56: 1, 202: 7, 133: 1, 201: 21, 97: 1, 203: 1, 81: 1, 185: 1, 186: 1, 111: 1, 90: 1, 1: 7452, 135: 1, 208: 1, 2:
668, 4: 9, 8: 1, 3: 45, 5: 1, 6: 1}
```

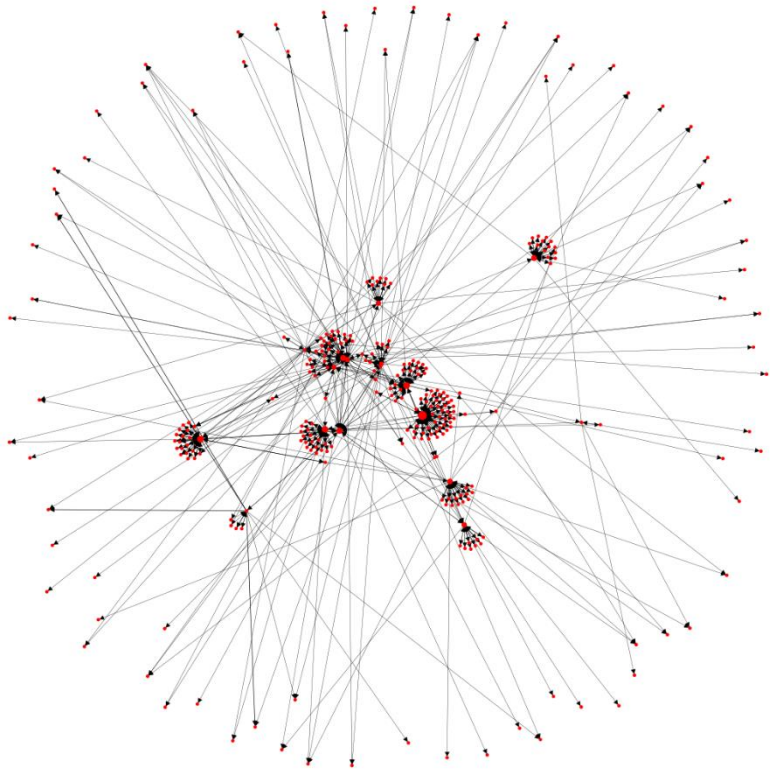
The next other thing I looked at was **PageRank**. The average was **0.0001** and the max is **0.007** from @LenM_Ops**.** Here is the histogram, which is very similar to the degree distribution histogram.



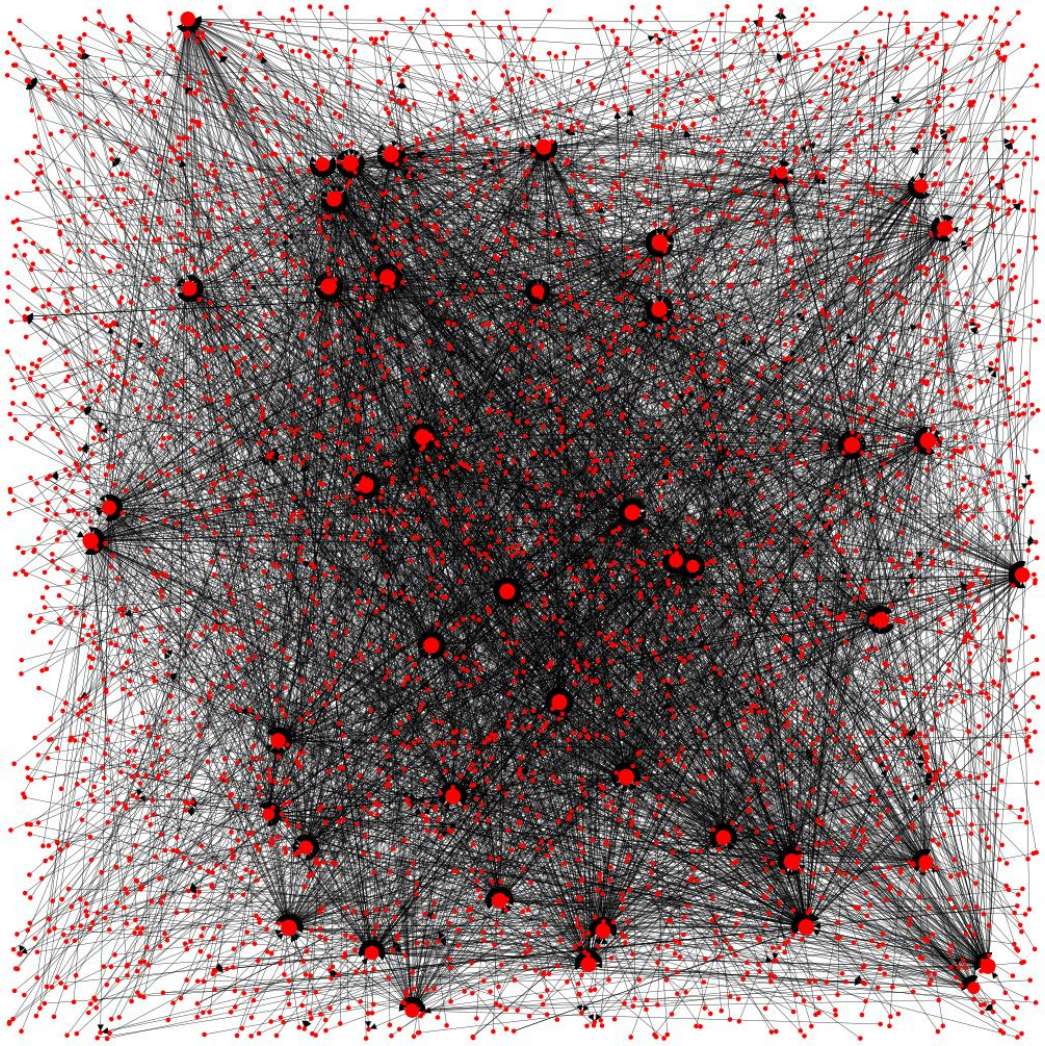
The next thing that I looked at was **closeness centrality**. The average was **0.021** and the max is **0.269** from @rechambadi which makes sense. Here is the histogram.



To clean up the graph, I tried various methods. One method was removing nodes with low closeness and a degree of 1. Which resulted in this which is much better than the previous one. This however only shows 287 nodes.

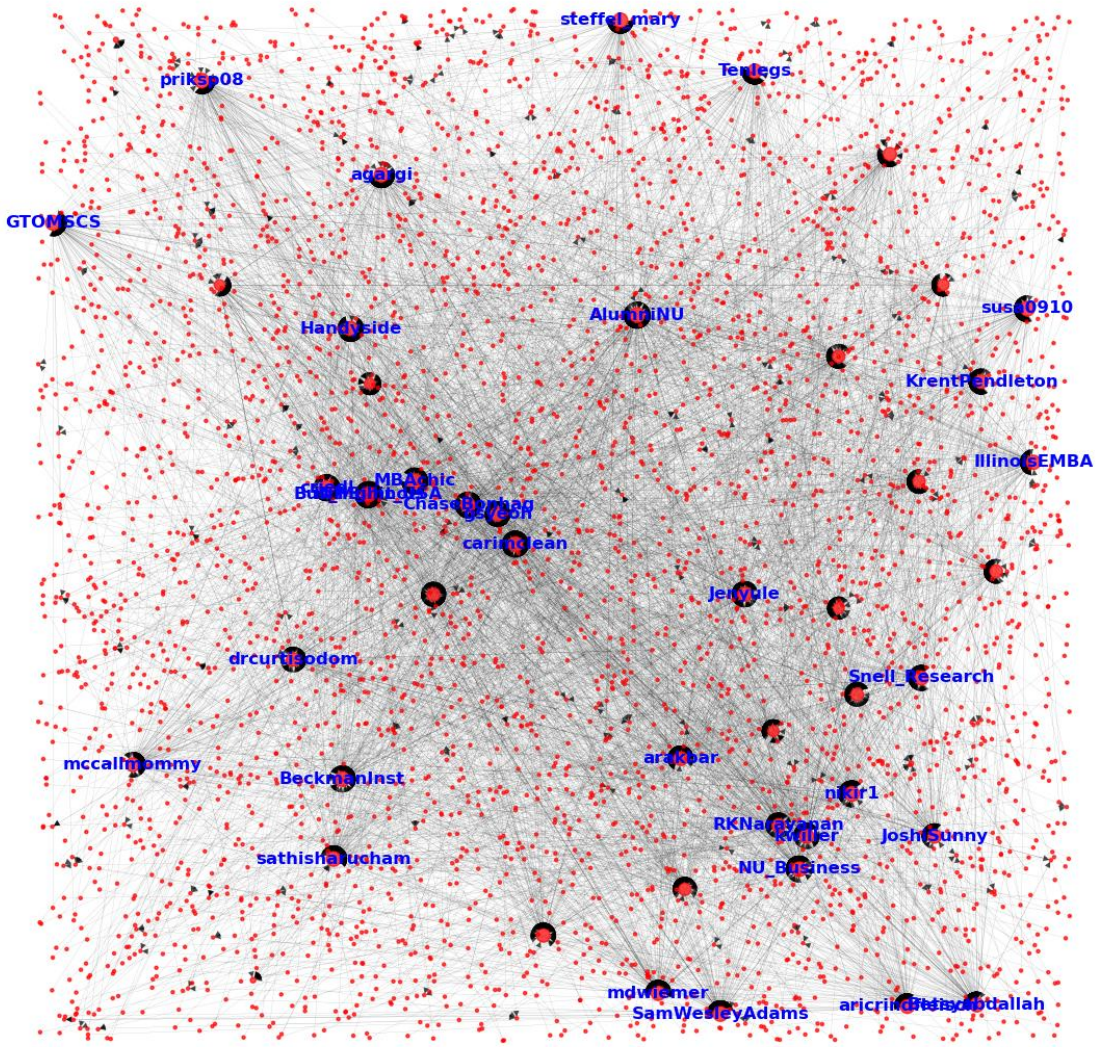


Another method was removing nodes with an in-degree of 1 which resulted in this. This one has a total of 3942 nodes.



I found this visualization to be better, while the layout is random, it is much easier to see the "important" nodes.

The graph below has less emphasis on the lines with added labels as well. The labeled nodes, or users, have in-degrees greater than 80 which means that those users have 80 or more followers in the network.


steffel_mary
Tenlegs
prikson08
agangi
GTOMSCS
AlumniNU
sus10910
HandySide
KrentPendleton
IllinoisEMBA
caringclean
JennyLe
drcurtisodom
Snell_Research
mccalmommy
araakbar
BeckmanInst
nikir1
RKNarayanan
kwiller
JoshSunny
sathishanucham
NU_Business
mdwiemer
SamWesleyAdams