Web Science (H) COMPSCI4077

Network based Social Media Analytics

Scott Brown - 2305539B

Sourcecode
https://github.com/BlueLinks/WebScience-WebCrawler

Data
https://github.com/BlueLinks/WebScience-WebCrawler/blob/master/all_tweets.json

March 12, 2020

1 Introduction

a.

For my analytics I have written 5 python scripts.

`get_tweet_data.py` - This script uses lists 'keywords' and 'users' to filter a stream of twitter data and insert the raw tweet json into a mongoDB collection `TwitterStream.all_tweets`. To use this script your twitter api access keys must be contained in a file `keys.txt`. Each key must be placed on a newline with no extra characters in the order `consumer_key` > `consumer_secret` >`access_token` and `access_token_secret` with `consumer_key` being on the top line and `access_token_secret` being on the bottom line.

`cluster_text.py` - Simple script to find all tweets in a collection and cluster all the text in the '`text`' field into k categories and print the top 10 words used

`cluster.py` - Similar to the `cluster_text.py` script but instead of just the text this script extracts usernames and hashtags. To build this and the `cluster_text.py` scripts I used this stack overflow question as a reference `https://stackoverflow.com/questions/27889873/clustering-text-documents-using-scikit-learn-kmeans-in-python`

`user_interactions.py` - This is the most complex of the three and uses the networkx and matplot libraries to draw the user interaction graphs. Lines '`63`', '`65`' , '`67`', and '`69`' can be commented in and out to control which interactions are graphed.

```
# Add Mentions
interactions.add((row["user_mentions_id"], row["user_mentions_screen_name"]))
# Add Retweets
interactions.add((row["retweeted_id"], row["retweeted_screen_name"]))
# Add replies
interactions.add((row["in_reply_to_user_id"], row["in_reply_to_screen_name"]))
# Add quotes
interactions.add((row["quoted_id"], row["quoted_screen_name"]))
```

I used this medium article `https://medium.com/future-vision/visualizing-twitter-interactions-with-networkx-a391da` as a reference while building this script

`get_hashtags.py` - This script is used to get the hashtag information from the `TwitterStream.all_tweets` collection in a format that is readable by `https://www.cortext.net` to produce graphs for the hashtag data.

b.

Data was collected for the initial 1% stream on Tuesday the 3rd of March at 3pm and data collected using the more specific rest probes was collected on Monday the 9th of March at 6pm. To collect as much data as possible for the initial stream, generic keywords found at `https://techland.time.com/2009/06/08/the-500-most-frequently-used-words-on-twitter/` were used

2 Data crawl

a. 1% of Data

I used the tweepy library to construct my twitter crawler and pymongo to connect to the local mongoDB database

```
def on_data(self, data):
    if (time.time() - self.start_time) < self.limit:
        # Load the Tweet into the variable "tweet"
        tweet = json.loads(data)
        try:
            all_collection.insert_one(tweet)
            return True
        except:
            # This is a duplicate
            self.duplicates += 1
            print("Duplicates so far: %d" % self.duplicates)
    else:
```

```
        # Times Up
        return False
```

This code is called whenever the stream receives a tweet. When initialising the listener there is an option to specify a time limit in minutes (defaults to 1hr). The full raw json is inserted into the collection `TwitterStream.all_tweets` and a try/except structure is used to avoid duplicates and track the number of them which is printed at the end of execution. From the first initial stream I identified that there was a noticeable trend with the current presidential election in the US

US	736	Trump	1569	Bernie	1270	Biden	1542

b. Enhancing the crawler

After deciding on looking at the presidential election race I decided to use keywords

```
keywords = ['trump','biden','republican','democrat',"bernie",'bernie sanders','sanders','joe',
'donald','presidential election','@realDonaldTrump','#PresidentialRace2020','#PresidentialPrimary',
'@BernieSanders','Bill','@JoeBiden','@SenSanders','#2020election']
```

and users @realDonaldTrump, @BernieSanders, and @JoeBiden

```
users = ['25073877','216776631', '939091']
```

I configured the stream as follows

```
stream.filter(follow=users, track=keywords, languages=["en"],is_async=True)
```

3 Grouping of tweets

a. Method for grouping

For the grouping of the tweets I wrote a python script '`cluster.py`' to load all the data into a dictionary containing usernames and the full text of the tweets and used the sklearn library to vectorise and cluster the dictionary. I chose to cluster into 8 groups as that gave a wide spread of information and use the cluster centres to identify key records and because I used KMeans, all the clusters are of equal size.

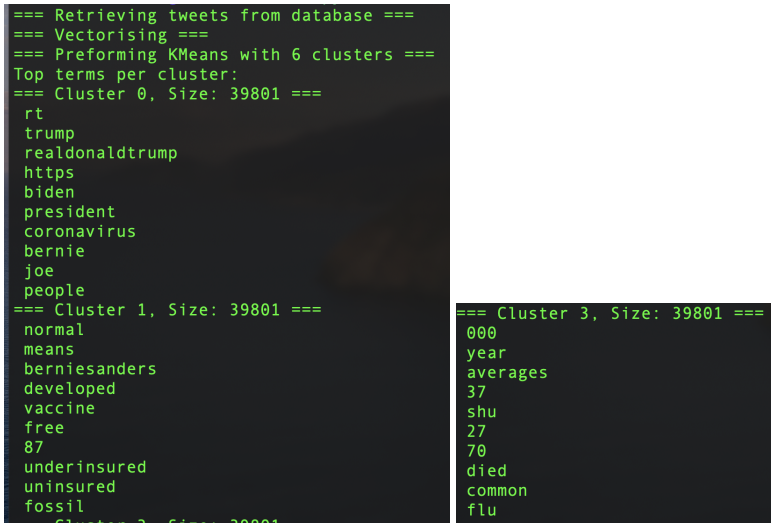
b. Method for Username and Hashtag identification

To indentify key usernames and hashtags from groups I used regular expressions to extract usernames and hashtags and then used a collections.Counter object to count occurrences.
For example:

```
=== Preforming KMeans with 10 clusters ===
        === Cluster 0, Size: 357538 ===
=== Usernames ===
[('@bananaphone001', 3), ('@donnamellon1', 2), ('@MARINES2_2_', 2), ('@HeARTofGod99', 2), ('@sdh07141976', 2)]
=== Hashtags ===
[('#CompTIA', 1), ('#Certifications', 1), ('#training', 1), ('#instruction', 1), ('#SecurityPlus', 1)]
        === Cluster 1, Size: 357538 ===
=== Usernames ===
[('@MeedgeKnowsBest', 5), ('@fairside231', 1), ('@blysx', 1), ('@zzzrazia', 1), ('@JoeyxRoss', 1)]
no hashtags
        === Cluster 2, Size: 357538 ===
=== Usernames ===
[('@MeedgeKnowsBest', 6), ('@VoxVorago', 1), ('@BernieSanders', 1), ('@zzzrazia', 1), ('@JoeyxRoss', 1)]
no hashtags
        === Cluster 3, Size: 357538 ===
=== Usernames ===
[('@MeedgeKnowsBest', 6), ('@PRINTSVONJANE', 1), ('@zzzrazia', 1), ('@JoeyxRoss', 1), ('@Medstudntopines', 1)]
=== Hashtags ===
[('#bolshevism', 1), ('#Ukraine', 1)]
```

Text grouping

I also wrote another script called `cluster_text.py` to cluster the text alone and print out the top 10 terms, this however strips the @ and # symbols.



I choose to only use a sample of 50,000 tweet as it can take a long time to cluster using KMeans, I used 6 clusters as that gave some good clusters on the different candidates as well as one on the coronavirus. Again because I used KMeans, all the clusters were of equal sizes.

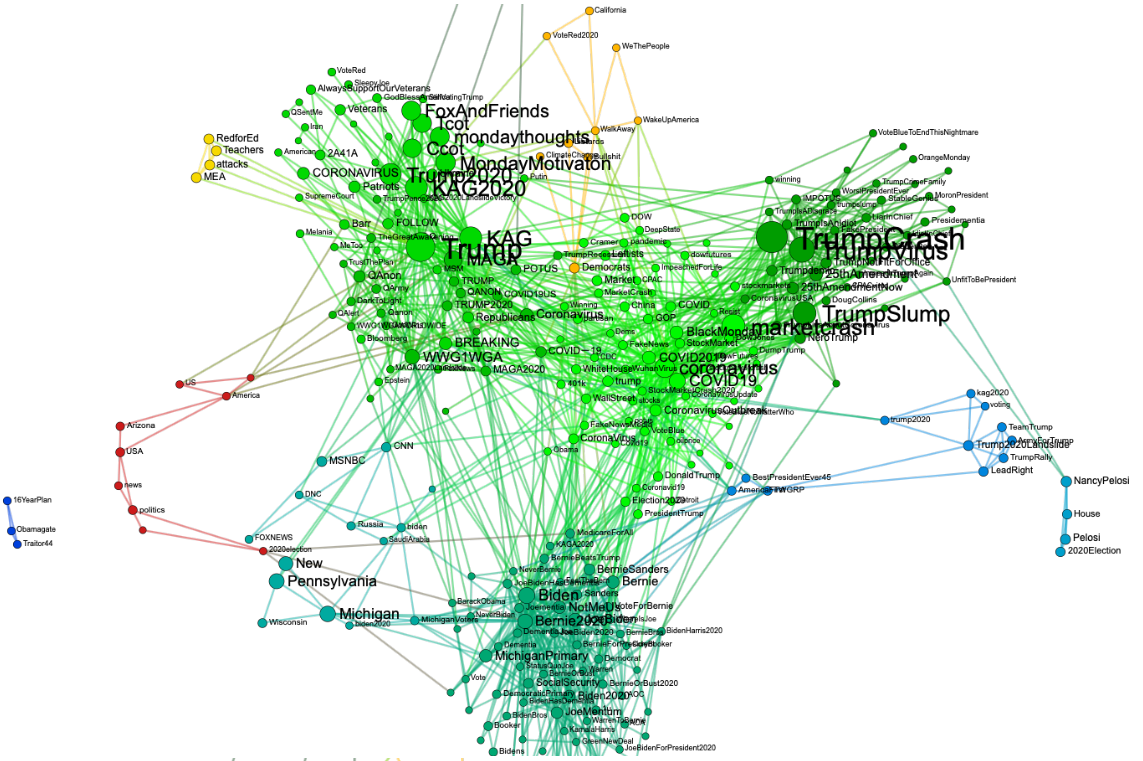
4 Methods for constructing user interaction graphs

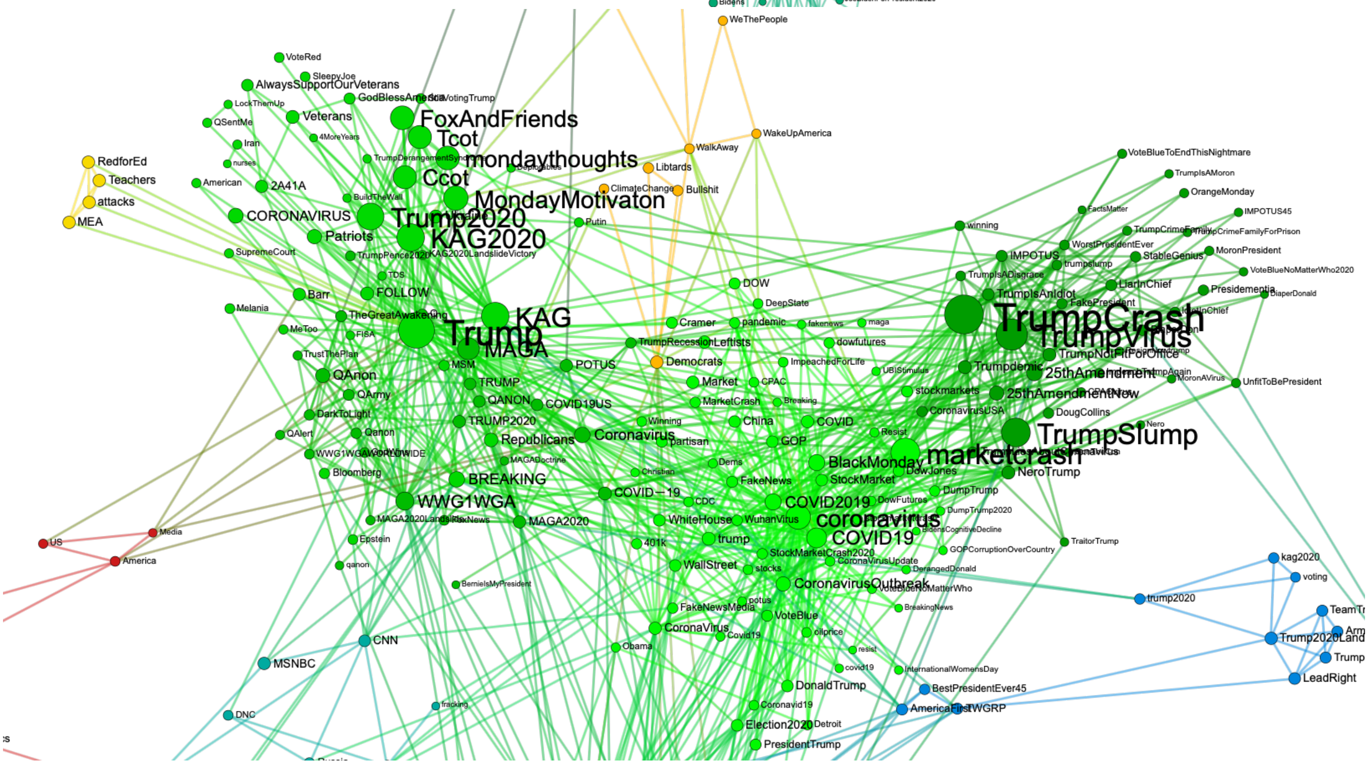
a. Mentions, Replies, Retweets and Quote Tweets

Capturing and organising user interactions such as mentions, replies, retweets, and quote tweets is done by the `user_interactions.py` script. I had a problem with indexing the nested json format of the mongoDB database so the script normalises the data taken in through pymongo (This can take some time!) before storing in a pandas dataframe. The initial dataframe is parsed to retrieve relevant information which is stored in another dataframe which is itself parsed using the `get_interactions` function which contains lines which can be commented in and out to control which interactions are recorded. After the interactions are recorded, the networkx library is used to construct the graph with the nodes being users and the edges being interactions. I used a sample size of 10,000 tweets for each graph as the runtime increases exponentially for drawing the graphs, especially the mentions graph.

b. Hashtag co occurrence

To capture hashtag co occurrence information I used a website `https://www.cortext.net` to generate maps of hashtags commonly used together by multiple users. In order to get the data in a format that was readable by the website I constructed a script `get_hashtags.py` that would read in a raw tweet json collection from mongoDB and output a csv file`hashtags.csv` containing one column filled with all hashtags from the entities filed of a tweet separated by spaces.


FoxAndFriends
Tcot
mondaythoughts
Ccot
MondayMotivaton
Trump2020
KAG2020
KAG
Trump
MAGA
TrumpCrash
TrumpVirus
TrumpSlump
marketcrash
coronavirus
COVID19
Biden
Bernie
Pennsylvania
Michigan
New
MSNBC
CNN
NancyPelosi
House
Pelosi
2020Election
California
WeThePeople
RedforEd
Teachers
attacks
MEA

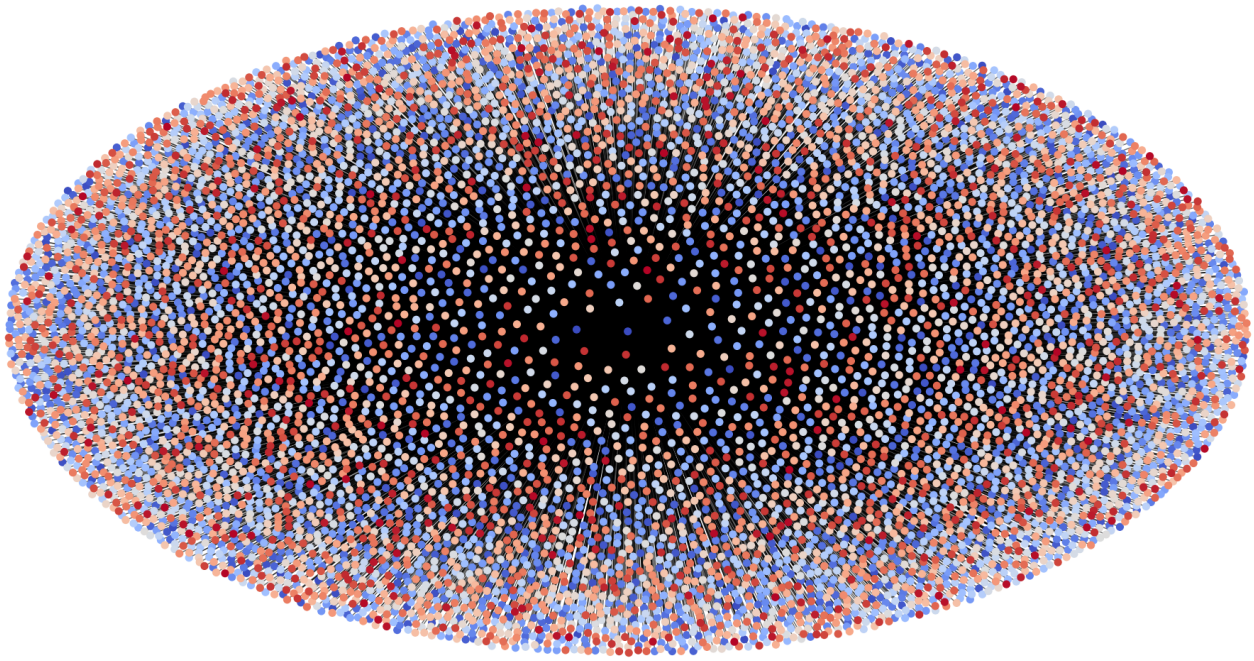

FoxAndFriends
Tcot
mondaythoughts
Ccot
MondayMotivaton
Trump2020
KAG2020
KAG
Trump
MAGA
TrumpCrash
TrumpVirus
TrumpSlump
marketcrash
coronavirus
COVID19
COVID2019
CORONAVIRUS
Patriots
WWG1WGA
BREAKING
Republicans
Coronavirus
BlackMonday
CoronavirusOutbreak
25thAmendment
25thAmendmentNow
WeThePeople
WalkAway
WakeUpAmerica
RedforEd
Teachers
attacks
MEA
MSNBC
CNN
America
US
LeadRight
BestPresidentEver45
trump2020
kag2020

Here are screen shots from the hashtag graphs that show that the most popular hashtags used were in relation to Donald Trump. The graph seems to be similar in shape to that of the quotes and replies graphs seen below which leads me to suspect that the epicentres of the tries and triads in the graphs relate to a similar subject.

5 Network analysis information

Mentions

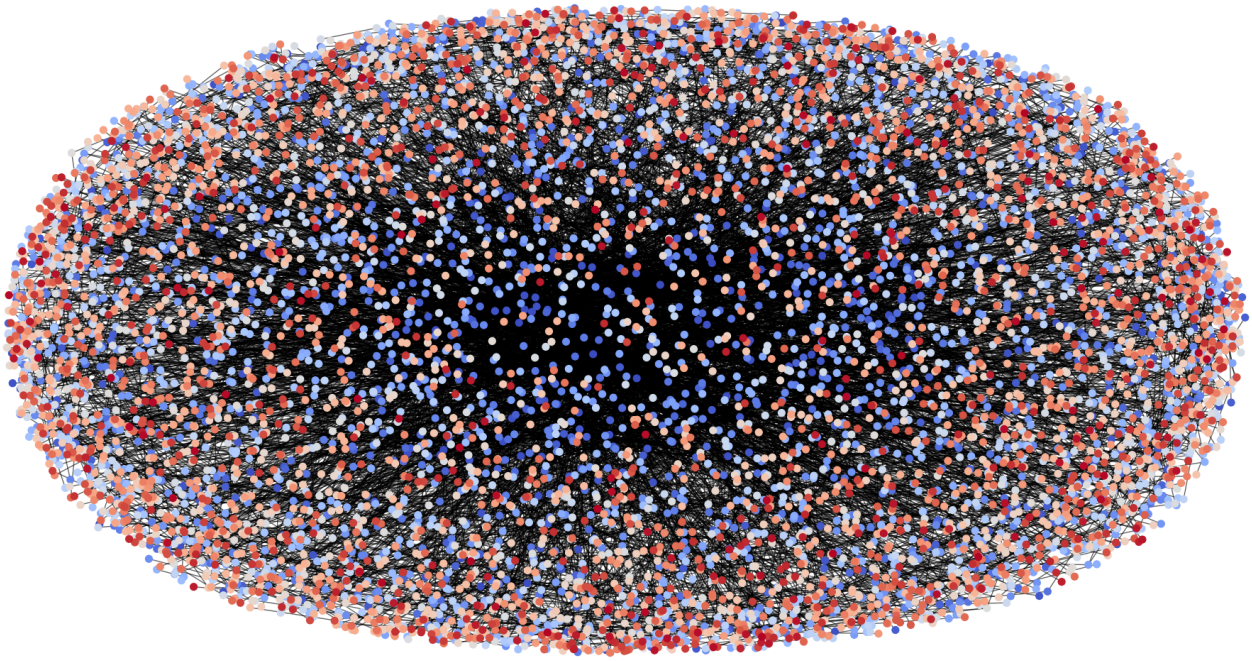
For the mentions graph with 7992 nodes and 7991 edges the average degree of the nodes was 2 yet the most frequent was 1.



As can be seen from the graph it is clear that there is a central user being mentioned the most.

Retweets

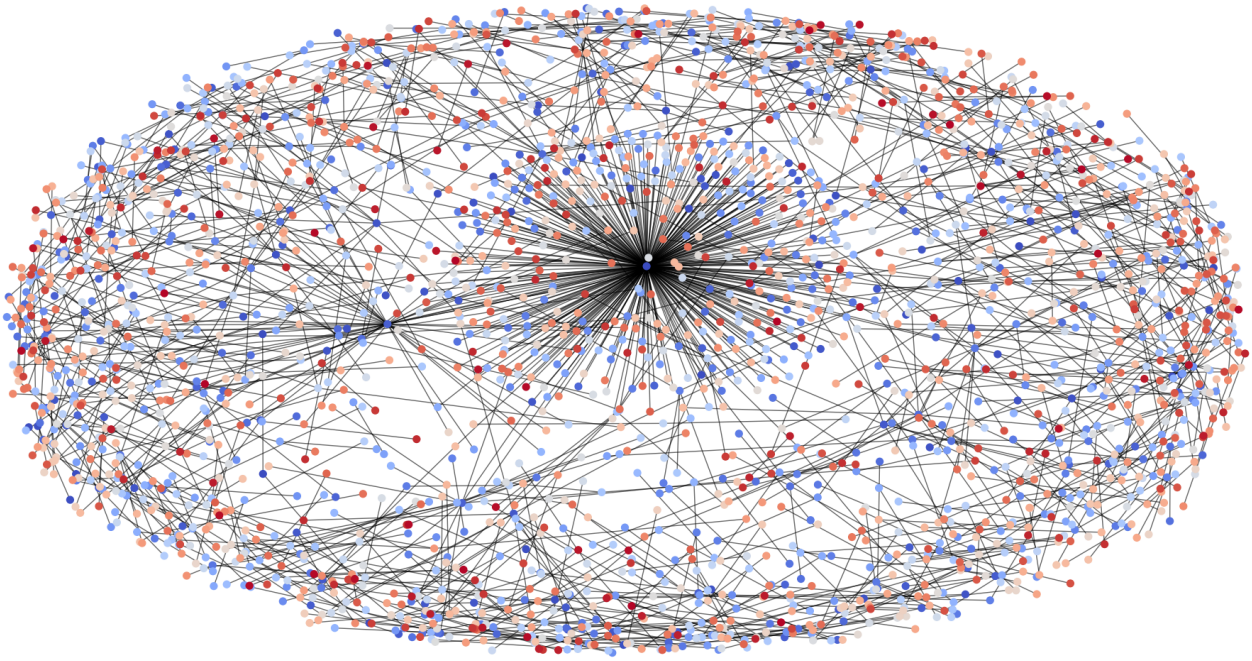
For the Retweets graph with 7872 nodes and 7008 edges the average degree of the nodes was 1.8 the most frequent was 1.



This graph shows that there are many tweets being retweeted but in my data collection most are only retweeted by one user as the most frequent degree is 1

Replies

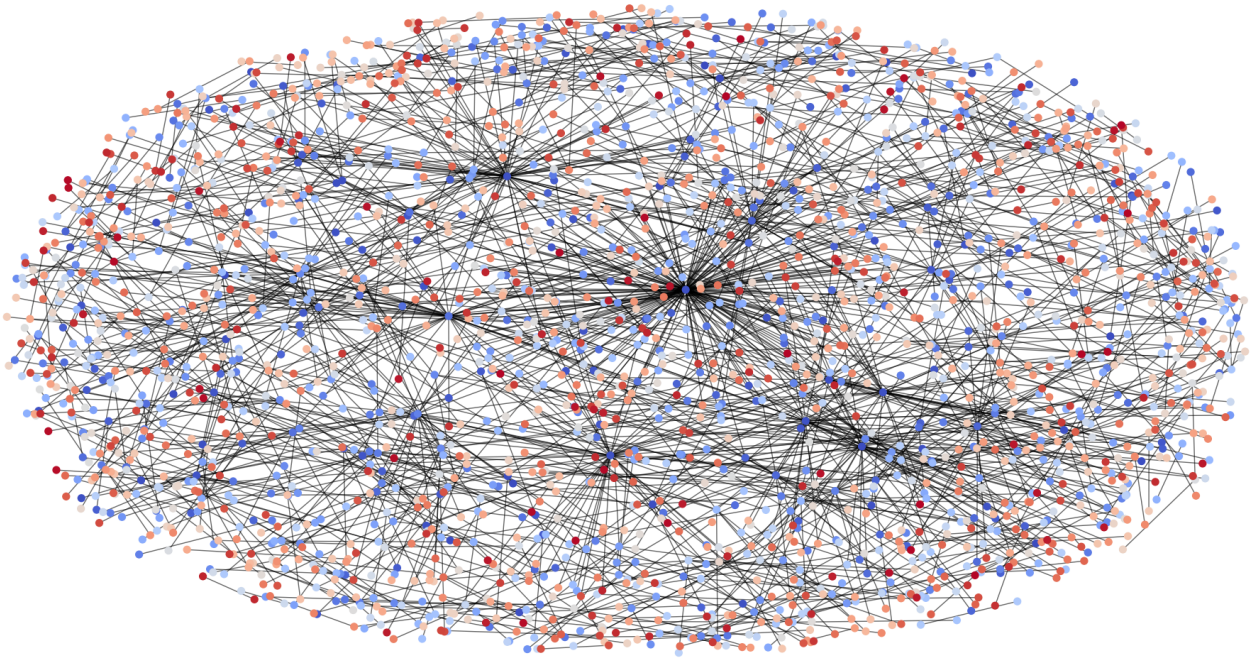
For the Replies graph with 2484 nodes and 1591 edges the average degree of the nodes was 1.3 the most frequent was 1.



It can be seen from the graph that there is a clear tweet in the centre that was replied to the most.

Quotes

For the Quotes graph with 2340 nodes and 1769 edges the average degree of the nodes was 1.5 the most frequent was 1.



As can be seen from the graph few clear tweets that were quoted the most.

Sample Data

Sample data is provided in the `all_tweets.json` file and must be imported into a local mongoDB database `TwitterStream` and collection `all_tweets` for the scripts of function correctly. Alternately new data can be collected by the `get_twitter_data.py` script.