Web Science Assessed Exercise

James Conway (2247492)

github.com/conwayjw97/Twitter-Analytics

1. Introduction

The software is capable of capturing tweets using a hybrid architecture of both Twitter's Streaming and REST API, the gathered tweets are then categorised through the use of K-Means clustering and saved to MongoDB. The software can output a range of statistics on the collected tweets, either on all the tweets as a whole or on the tweets in individual clusters. It can also construct interaction networks related to a variety of tweet aspects (hashtags, retweets, mentions) and graph and output the networks as well as their statistics.

The crawler was run for 2 minutes on Wednesday 18 March starting at 12:25. This is far less than the recommended crawling time of 1-2 hours however a smaller dataset had to be used in order to feasibly run the analytics functions, as my own PC lacks the processing power sufficient to perform grouping, analytics, and graph construction on large datasets in a feasible time. Despite the relatively small amount of data it was still enough to demonstrate the functioning of the software.

2. Data Crawl

The Tweepy library was used to perform all the crawling and data gathering tasks for the software. The software gathers tweets through the "collect_tweets.py" script, which takes streaming settings from the user as command line arguments.

```
james@james-P6689-MD61020:~/Uni/WS/Assessed Exercise$ python3 collect_tweets.py
Please run this program with arguments: collect_tweets.py <No_Power_Users> <Stream_Time> <Max_REST_Tweets>

<No_Power_Users>: Number of power users to be used in REST tweet crawling.
<Stream_Time>: Amount of time in seconds to perform 1% Stream crawling.
<Max_REST_Tweets>: Max number of tweets to try to retrieve for each REST tweet crawling request.
```

collect_tweets.py begins by scraping the current most trending keywords in the USA through tweepy, and then uses them to filter tweets gathered through the Streaming API. Trending American keywords are used to filter streamed tweets in order to have a linguistically homogenuous dataset which is more likely to contain tweets that come from a wide range of sources across the Twitter community.

Once Streaming has been run for the specified amount of time, or if the user chooses to preemptively interrupt the streaming process, the software will then find the most active users in the streamed tweets and gather tweets from these users through the REST API. Tweets were collected from the most active users so that the dataset could then contain a range of tweets from users that better represent the general Twitter population, as opposed to those who rarely use the service and may be considered less representative samples. More active users tend to use the platform in a greater variety of ways, such as retweeting, replying, and tweeting, as

opposed to only using the service to reply to tweets or to only write tweets, which provides little material to perform user interaction analysis on.

For both Streaming and REST crawling, the tweets are gathered in extended mode in order to collect the full tweet, the text is then parsed to remove URLs, new line symbols, and other broken symbols which are returned by Tweepy as “&”. Other tweet information which is gathered includes:

- Tweet ID
- Time posted
- User screen name
- Retweeted Tweet ID (if applicable)
- Retweeted Tweet text (if applicable)
- Retweeted user screen name (if applicable)
- Tweet ID being replied to (if applicable)
- Tweet user screen name being replied to (if applicable)
- User screen names mentioned in Tweet (if applicable)
- Hashtags present in Tweet (if applicable)

The gathered tweets are then saved to MongoDB, indexed by the Tweet ID before being grouped. This is so that if the software stops before grouping is finished, the tweets will still be saved and can be grouped later.

3. Tweet Grouping

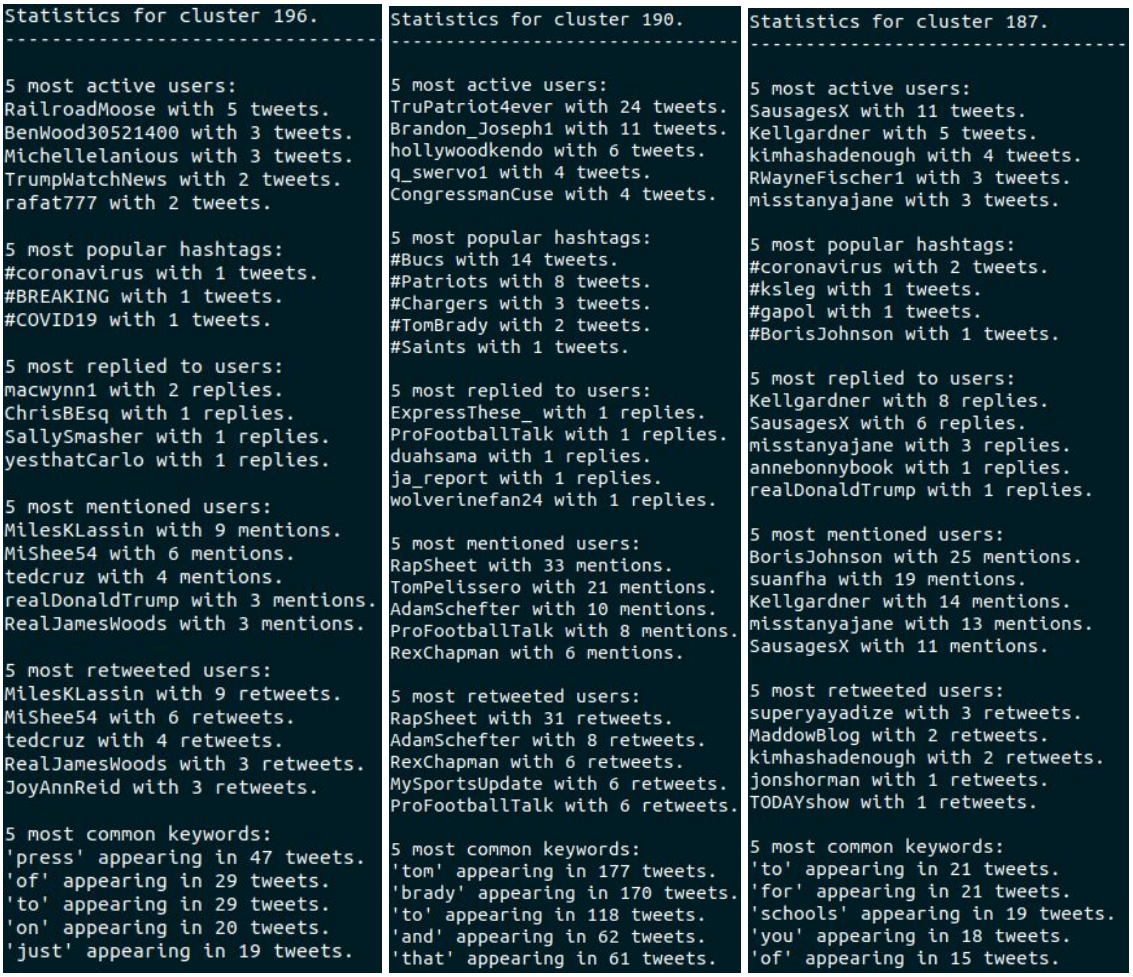
Tweets are grouped using K-Means clustering, which was done using functions from the SKLearn library, the code for clustering is present in “clustering.py”. The ideal cluster number was chosen to be 1 cluster for every 100 tweets as greater numbers of clusters would lead to similar keywords being divided into different clusters, and because the crawling process focuses mainly on collecting tweets related to particular trending keywords and thus the tweets are more likely to be relatively similar and require less clusters.

Here are some general statistics that were calculated by running “cluster_analytics.py”

Total Tweets	19772
Total Cluster	197
Average Cluster Size	100
Largest Cluster	109 with 5038 tweets

Smallest Cluster	162 with 2 tweets
5 Most Active Users	mwasaru3 with 151 tweets. anpaus4 with 114 tweets. SITAR70265187 with 102 tweets. LeungKW06 with 101 tweets. PepboaPersi with 101 tweets.
5 Most Popular Hashtags	#iHeartAwards with 402 tweets. #COVID19 with 360 tweets. #BoyWithLuv with 349 tweets. #BestMusicVideo with 327 tweets. #coronavirus with 277 tweets.
5 Most Replied to Users	y2ksiwon with 63 replies. realDonaldTrump with 52 replies. iainshet with 45 replies. thetank717 with 26 replies. 2kinasbichss with 25 replies.
5 Most Mentioned Users	BTS_twt with 1541 mentions. realDonaldTrump with 655 mentions. RealJamesWoods with 196 mentions. Oprah with 181 mentions. choi_bts2 with 167 mentions.
5 Most Retweeted Users	choi_bts2 with 160 retweets. realDonaldTrump with 145 retweets. jeremycyoung with 143 retweets. charts_k with 127 retweets. BTSChartDaily with 118 retweets.
5 Most Common Keywords	'to' appearing in 7912 tweets. 'and' appearing in 5324 tweets. 'of' appearing in 5256 tweets. 'is' appearing in 4673 tweets. 'in' appearing in 3995 tweets.

“cluster_analytics.py” can also provide statistics for individual clusters. There are too many clusters to show here and compare with the general data, so for a full analysis you can run the script yourself with the provided dataset. An example of some of the cluster analysis outputs however are:



4. User and Hashtag Information Organisation

User and hashtag information was used to construct interaction graphs using NetworkX. MatPlotLib and Gephi were also pused to output them to pdf files. The “networker.py” script contains the functions to construct interaction graphs for:

- Replies
- Mentions
- Retweets
- Hashtags

And can construct single graphs for all tweets or multiple ones for each cluster.

Most interaction graphs are made using directed graphs, where the direction implies the form of interaction.

Replies:

User replying -> User replied to

Mentions:

User mentioning -> User being mentioned

Retweets:

User retweeting -> User being retweeted

The only exception to this was hashtag interaction graphs, which are undirected graphs wherein hashtags which appear together in tweets are connected by edges.

The networks are constructed based on the users arguments passed to “network_analytics.py” which is explained in the next section.

5. Network Analysis

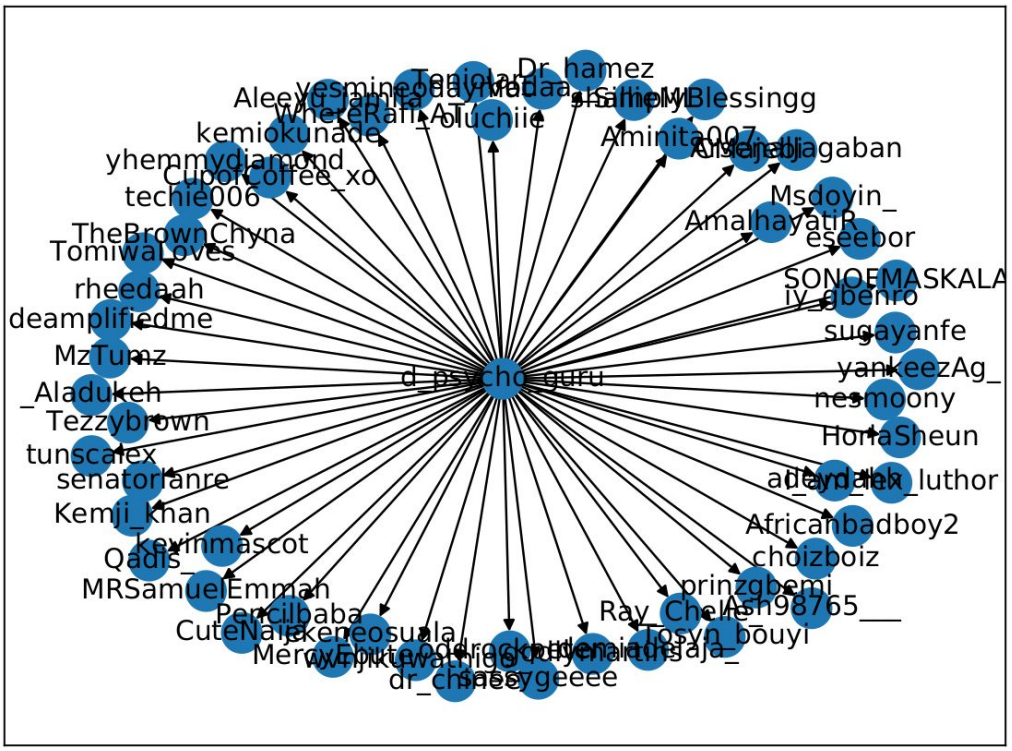
The software calls the network construction functions in “utils/networking.py” through the “network_analytics.py” script.

```
james@james-P6689-MD61020:~/Uni/WS/Assessed Exercise$ python3 network_analytics.py
Please run this program with arguments: network_analytics.py <Network_Type> <Save_Network> <Save_Graphs>

<Network_Type> choices:
1 - General Reply Interaction Graph
2 - Cluster Reply Interaction Graphs
3 - General Mention Interaction Graph
4 - Cluster Mention Interaction Graphs
5 - General Retweet Interaction Graph
6 - Cluster Retweet Interaction Graphs
7 - General Hashtag Co-occurence Graph
8 - Cluster Hashtag Co-occurence Graphs

<Save_Network> choices:
0 - Don't save networks
1 - Save networks as .pdf files in /graphs (WARNING: VERY TIME CONSUMING)

<Save_Graphs> choices:
0 - Don't save graphs
1 - Save graphs as .png files in /graphs
```

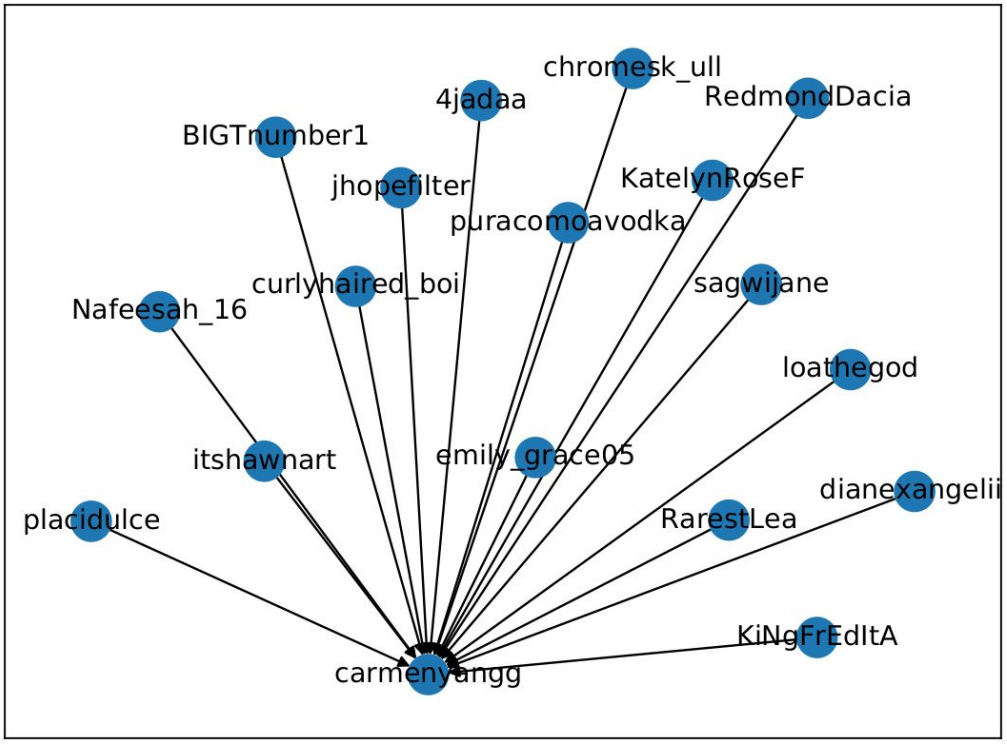
Based on the user’s choice, the script can construct the type of network chosen and then output gephi renderings of the network and/or save triad census results as bar charts. Both of these outputs will be saved to the graphs/ subdirectory.

Gephi graphs are very time consuming to construct and with my PC I was only able to construct small graphs for singular clusters in a feasible time. These network graphs could still however show interesting insights.

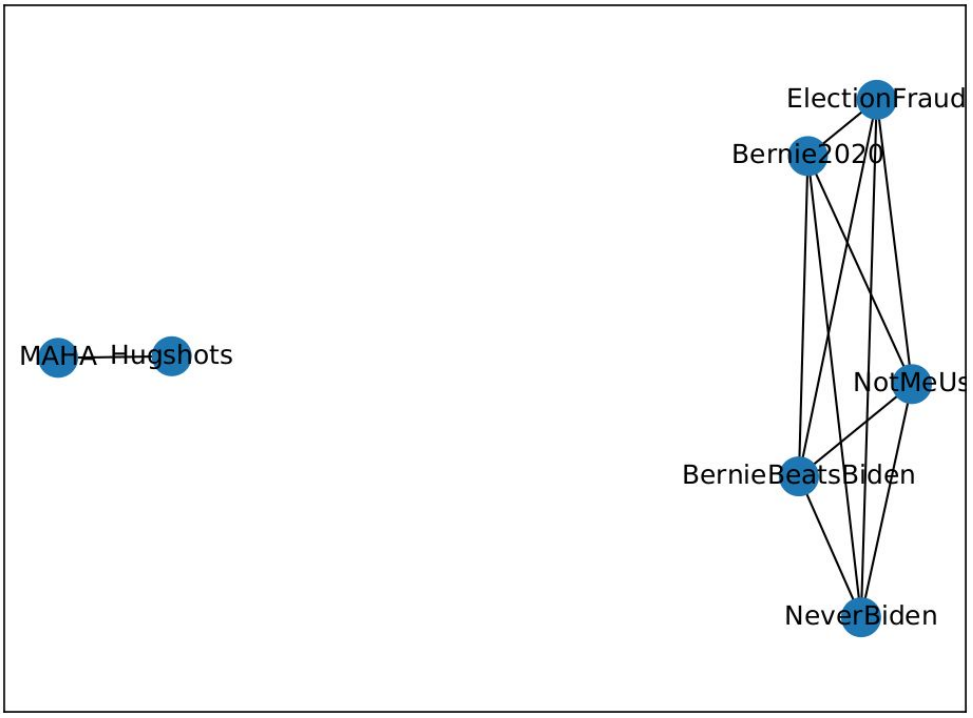
For example this mention graph shows a single user mentioning most other users in one cluster:



This reply graph shows a particular user as the main subject which many others seem to be replying to in a cluster:



This hashtag graph shows 5 hashtags which only appear together:



Other examples can be produced by running network_analytics.py with the provided dataset.

After network construction and outputting, the script calculates and outputs the number of ties and the triad census for the dataset / cluster. The triadic census was calculated for the directed graphs using NetworkX's "triadic_census" function while the undirected hashtag graph's triadic census was calculated using my own function.

The calculated statistics for the entire dataset are as follows:

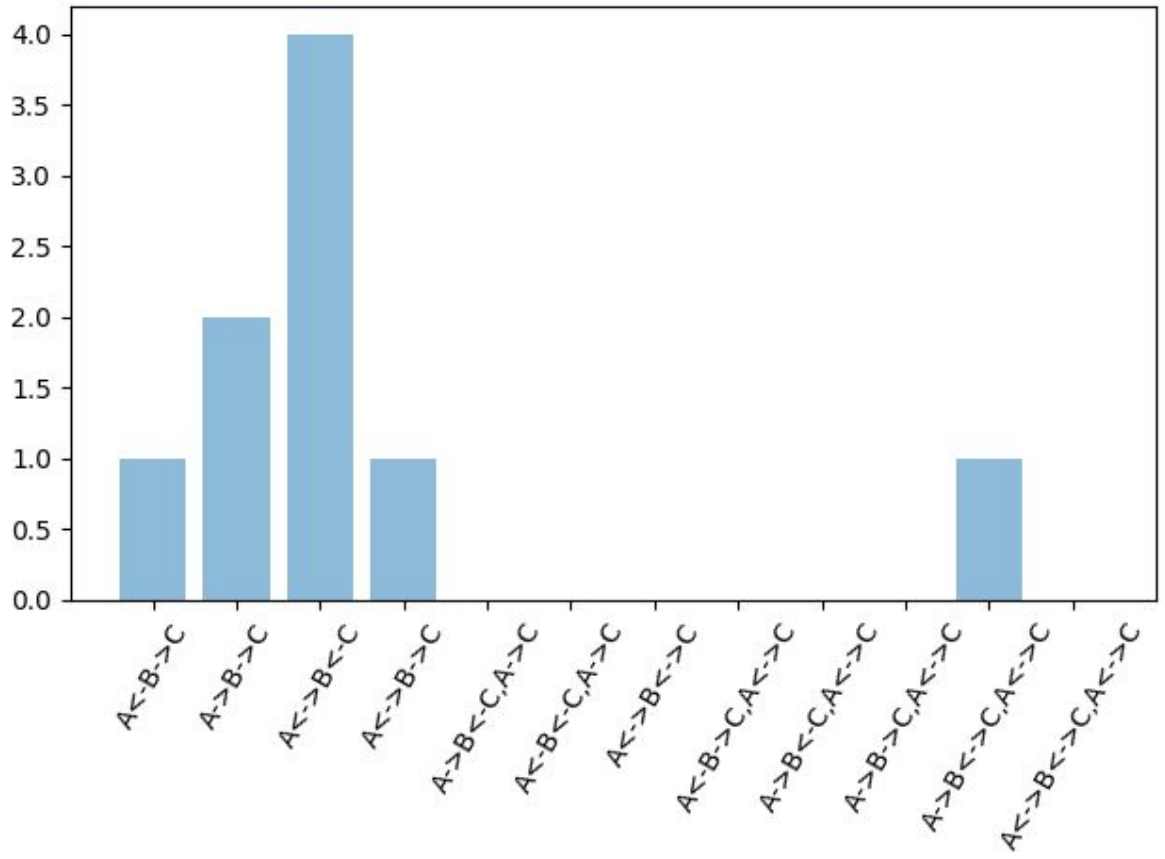
Ties	
Replies	6459
Mentions	19649
Retweets	12348
Hashtags	3056

Directed Graph Triad Census			
	Replies	Mentions	Retweets
A<-B->C	4464	420537	45670

A->B<-C	10250827	120158	269826
A->B->C	1672	27533	3319
A<->B<-C	498	1756	316
A<->B->C	1700	15404	107
A->B<-C,A->C	28	2644	24
A<-B<-C,A->C	2	0	0
A<->B<->C	316	524	2
A<-B->C,A<->C	1	414	1
A->B<-C,A<->C	15	2043	1
A->B->C,A<->C	4	10	0
A->B<->C,A<->C	13	134	0
A<->B<->C,A<->C	5	161	0

Undirected Hashtag Graph Triad Census	
A-B-C	9829
A-B-C, A-C	9476

The user can also choose to output the triad censuses to barcharts and saved in the “/graphs” subdirectory. There are too many triad censuses for each of the clusters to show here on the report but you can run the script to see them yourself with the provided data. An example reply network triad graph for one of the clusters is:



As can be seen, the bar charts give a better idea than the networks as to what kinds of interactions were prevalent in each cluster/overall. In the above chart for example most replies seem to be between two individuals, with a third also replying to one of them.