Application Framework for Network Data Analysis on Twitter

Muhammad Salik Nadeem
Faculty of Science
Ontario Tech University (UOIT)
Oshawa, Ontario, Canada
muhammadsalik.nadeem@uoit.ca

ABSTRACT

We have created a python based framework for searching Twitter using a keyword query which can be either a Hashtag or a URL. The program returns all tweets containing that query along with a lot of meta data. Our algorithm then creates a network of all the users involved in sending that tweet and creates a directed graph with users who tweeted that query as nodes and edge direction showing a "followed by" relationship (i.e. the direction in which information flows on Twitter).

Once we construct the graph, we run various network analysis algorithms on it such as degree distribution, betweenness and closeness centrality, community detection and average clustering among others. We also collect available location data and time stamps to provide a complete picture of the spread of that query or 'meme' over time.

CCS CONCEPTS

• Networks~Social media networks • Networks~Online social • Information systems~Wrappers (data mining) • Information systems~Data mining

KEYWORDS

Twitter, online social networks, information propagation, data mining, reciprocity, retweet, information diffusion, network analysis.

1. INTRODUCTION

Twitter is a very popular social media platform. It is unique in its micro-blogging style service which makes it a platform for people to share important pieces of information with a larger community of followers. Over the past few years it has gain a lot of media attention as most celebrities and politicians as well release statements on the platform making it the center of different news media stories. Twitter users send and receive messages known as "tweets". These tweets as shown in figure 1, are generally text based post of up to 280 characters in length. They may also contain multimedia content such as images, GIFs, videos and audios as well as URLs. Tweets are delivered to users on a subscription based process whereby if user A is following user B then user A will receive tweets from user B. We represent this as an edge in our system as E (B -> A). This is because the information is flowing from B to A or in other words B is followed by A. If B is not following A, then B will not receive A's tweets. Because of this subscription based model, our graph then becomes a directed graph with nodes representing users and edges representing this "followed by" relationship of information flow. This relationship is different from other social media platforms such as Facebook, MySpace or Snapchat in that following someone doesn't require reciprocity. User A can follow user B, but user B doesn't have to follow user A back.

Figure 1. Example of a tweet with user mentions, a hashtag, a URL and a multimedia image

Tweets as shown in Figure 1, often contain Hashtags and URLs which are reused and retweeted by other users. These Hashtags are very critical for analyzing the flow of a particular piece of information over the social media. They spread in a similar fashion as 'memes' as mentioned by Leskovec et al [1]. Our application framework is designed in particular to search such Hashtags and similarly named entities over a period of time collecting key metadata and user network information to allow for detailed quantitative study of how particular memes spread over the social media platform.

The reason why Twitter is an idea social media platform for doing such network analysis study is because of a number of factors. Firstly, it has an explicitly defined social media structure (the follower-following subscription model). Secondly, information on Twitter naturally spreads with hashtag associations to certain pieces of stories. Which makes tracking the flow of that particular information in the social network easy. We don't need to spend time applying complex topic modelling algorithms or searching for named entities or looking for similar phrase clusters to categorize tweets taking about similar topics.

These hashtags also then become very integral for people to find out more news and information related to particular stories, event or news. Certain hashtags if tweeted enough times in a short span of time start to 'trend'. These top trends are shown on a users' home screen. This feature actually makes the top trending tweets even more 'sticky' as more and more people start talking about it.

Our framework will allow for users to collect the required data to perform detailed analysis on large sets of data and discover interesting behaviors. The system also performs key network analysis algorithms to gain better insights into the underlying network of users.

2. RELATED WORK

There have been many studies done in the past on Twitter. It has gained popularity in the past decade for its role in the spread of information over social networks. There are many research papers focusing on different aspects of the social network. One can also find paid software frameworks for conducting analysis on Twitter for commercial purposes. There are also a number of open source implementations available on github to start using the Twitter API for accessing tweets. We will discuss both research and developmental work done in this area.

2.1. RESEARCH WORK

Due to the social network structure and public nature of Twitter, it has gained a lot of attention in the past from various researchers in the social network domain.

Some have studied the topology of the follower-following subscription model [2], [3]. Some have also conducted sentiment analysis on tweets based on query terms [4]. Meeyoung et al [5] have also studied how influence works on Twitter using parameters such as in-degree, retweets and mentions. Some have also studied accessing the credibility of tweets by learning features and propagation patterns of real and fake tweets [6].

Haewoon et al [2] have done exhaustive data mining on twitter collecting all user data and tweets to analyze the structure and dynamics of the social networking platform. Their results show that for a large majority of people, the network shows power law distribution on the number of followers/following apart from the most popular users which have a skewed number of followers and following dynamic. They follow orders of magnitude less than the number of users following them. Their research also shows Twitter follows the 'small world' paradigm. Where most users are separated by 4 degrees on the platform.

Takeshi et al did work on the real-time aspect of twitter [7], using the platform to detect earthquakes and pin point its location. They investigate the real-time interaction of events such as earthquakes in Twitter and propose an algorithm to monitor tweets and to detect a target event. To detect a target event, they devise a classifier of tweets based on features such as the keywords in a tweet, the number of words, and their context. Subsequently, they produce a probabilistic spatiotemporal model for the target event that can find the center and the trajectory of the event location. They consider each Twitter user as a sensor, and set a problem to detect an event based on sensory observations. Location estimation methods such as Kalman filtering and particle filtering are used to estimate the locations of events.

2.2. IMPLEMENTATION WORK

Twitter has developed and published their own API framework which can be used to get all manner of public data form twitter [8]. Many various wrappers have been created for this API in languages such as Python, PHP, Java, JavaScript, etc. which allows developers to write their own code based on what type of data they want and in what format [9]. However, there is no free open source analysis tool available to easily fetch data from twitter, organize and clean it, perform network and graphic algorithms on it and finally to visualize these results in a meaningful way. There are various small snippet codes available on GitHub to connect to the Twitter API to get tweets but nothing in terms of developing an actual network analysis toolbox.

This is where our work comes in. We are creating an open source Python framework to scrape data from Twitter. The code will be available on Github [10] for users to download and start collecting data from Twitter. Our framework not only collects the data, it also cleans it, organizes it and also performs some visualization work in the form of network graphs, scatter plots over time and location maps, etc.

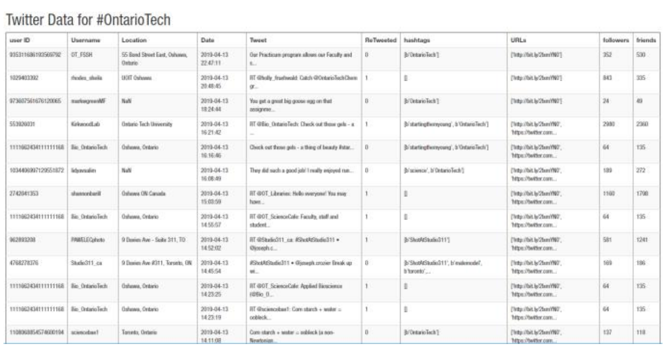
Another key aspect about our work is that users can perform analysis at a granular level. Most research work done focuses on the larger network and global trends. Our framework will allow users to perform localized analysis on a per tweet/Hashtag level.

3. DATA GATHERING

There are varies data mining and cleaning processes involved in making the data ready for detailed analysis work. Below we mention the APIs, Libraries and the programing environment used to gather all the required data for the framework.

3.1. Twitter API

Twitter offers a comprehensive Application programming Interface (API) that is easy to crawl and collect data from [8]. The challenge is when we gather data from the free version [11] of the API which imposes strict usage and rate limitations. One key limitation is being restricted to having access to only the past 7 days of Twitter data. Which essentially makes it really hard to crawl a hashtag from its point of origin if it was first started earlier than 7 days. This can be overcome if we keep on collecting the data every day, giving us larger datasets to work with. Another big restriction is on the amount of requests we can make over time. Twitter imposes a wait time of 15 minutes after a certain number of requests made by the API, which again makes it very difficult to collect large amounts of data as collecting it all will take a lot of time. We have to make several API calls in order to collect all the relevant tweets data. First, we need to run a query for a specific term such as a hashtag. We get the complete tweets data in a stream from that query which we clean up and fetch only the key important data from it such as the Users name tweeting, the tweet content, location of the user, followers and following numbers, all the hashtags and URLs mentioned in the tweet, etc. This requires making N API calls where N = total number of tweets fetched. Figure 2 shows a snapshot of the data fetched for "#OntarioTech", showing all the key data points gathers for the gathered tweets.



Twitter Data for #OntarioTech

user ID	Username	Location	Date	Tweet	ReTweeted	hashtags	URLs	followers	friends
935311686193569792	OT_FSSH	55 Bond Street East, Oshawa, Ontario	2019-04-13 22:47:11	Our Practicum program allows our Faculty and s...	0	[b'OntarioTech']	['http://bit.ly/2bmYN0']	352	530
1029403392	rhodes_sheila	UOIT Oshawa	2019-04-13 20:48:45	RT @holly_fruehwald: Catch @OntarioTechChem gr...	1	[]	['http://bit.ly/2bmYN0']	843	335
973607561676120065	markegreenMF	NaN	2019-04-13 18:24:44	You get a great big goose egg on that assignme...	0	[b'OntarioTech']	['http://bit.ly/2bmYN0']	24	49
553926031	KirkwoodLab	Ontario Tech University	2019-04-13 16:21:42	RT @Bio_OntarioTech: Check out these gels - a ...	1	[b'startingthemyoung', b'OntarioTech']	['http://bit.ly/2bmYN0', 'https://twitter.com...	2980	2360
1111662434111111168	Bio_OntarioTech	Oshawa, Ontario	2019-04-13 16:16:46	Check out these gels - a thing of beauty #star...	0	[b'startingthemyoung', b'OntarioTech']	['http://bit.ly/2bmYN0', 'https://twitter.com...	64	135
1034406997129551872	ladyavalin	NaN	2019-04-13 16:08:49	They did such a good job! I really enjoyed run...	0	[b'science', b'OntarioTech']	['http://bit.ly/2bmYN0', 'https://twitter.com...	189	272
2742041353	shannonbarill	Oshawa ON Canada	2019-04-13 15:03:59	RT @OT_Libraries: Hello everyone! You may have...	1	[]	['http://bit.ly/2bmYN0', 'https://twitter.com...	1160	1798
1111662434111111168	Bio_OntarioTech	Oshawa, Ontario	2019-04-13 14:55:57	RT @OT_ScienceCafe: Faculty, staff and student...	1	[]	['http://bit.ly/2bmYN0', 'https://twitter.com...	64	135
962893208	PAWELECphoto	9 Davies Ave - Suite 311, TO	2019-04-13 14:52:02	RT @Studio311_ca: #ShotAtStudio311 • @joseph.c...	1	[b'ShotAtStudio311']	['http://bit.ly/2bmYN0', 'https://twitter.com...	581	1241
4768278376	Studio311_ca	9 Davies Ave #311, Toronto, ON	2019-04-13 14:45:54	#ShotAtStudio311 • @joseph.crozier Break up wi...	0	[b'ShotAtStudio311', b'malemodel', b'toronto', ...	['http://bit.ly/2bmYN0', 'https://twitter.com...	169	186
1111662434111111168	Bio_OntarioTech	Oshawa, Ontario	2019-04-13 14:23:25	RT @OT_ScienceCafe: Applied Bioscience (@Bio_O...	1	[]	['http://bit.ly/2bmYN0', 'https://twitter.com...	64	135
1111662434111111168	Bio_OntarioTech	Oshawa, Ontario	2019-04-13 14:23:19	RT @sciencebae1: Corn starch + water = oobleck...	1	[]	['http://bit.ly/2bmYN0', 'https://twitter.com...	64	135
1108068854574600194	sciencebae1	Toronto, Ontario	2019-04-13 14:11:08	Corn starch + water = oobleck (a non-Newtonian...	0	[b'OntarioTech']	['http://bit.ly/2bmYN0', 'https://twitter.com...	137	118

Figure 2. Snapshot of the data gathered for "#OntarioTech", showing all the key data points our framework is collecting.

For all these tweets we find M number of unique users (there are times when users have made multiple tweets on a hashtag). For all these M users we make ${}^{M}C_{2}$ API calls. For each pair of users, we find if they have a follower, following relationship. This query helps us in forming a directed edge graph network.

3.2. Geolocation

Another key piece of data we are gathering is location data of users. For this we are using a Python library called Geopy [12]. This library has a free service available called Nominatim [13], which is an open street map API, used to convert a given text of location address into actual geographical coordinates. This library is free but applies a 1 request per second per user limitation. Which again requires a lot of time for large number of user locations. We store the latitude and longitude data of each user in our database.

The reason we have to perform a separate location API call is because Twitter allows users to put in location information as a string, without limiting them to actual location data. This in turn creates a problem for data analysis as users occasionally put in fake or made up locations. Our geolocation code takes in all the users' location information and tries to convert them into longitude and latitude data when possible. We use these coordinates to map out the users on a world map, showing where the users are tweeting that specific hashtag from.

4. IMPLEMENTATION

The entire code has been written in a Python 3.7 [14] development environment using Anaconda package manager [15]. The code has been pushed to Github [16], where anyone can download the code and start gathering their own data and start performing analysis on Twitter.

We used Tweepy [17], a Python wrapper for the Twitter API to gather the initial tweets data. Our code searches for a query term on the Twitter network, which returns a list of JSON objects of matching tweets with that query term. For network analysis purposes we recommend searching for Hashtags, although the code can be used to search for URLs, Names, Places, phrases and other random terms as well. The JSON object contains a lot of information regarding the tweet and the user. We collect the text of the tweet, all the hashtags and URLs mentioned in it. The user's ID and followers count and location data. In addition to the tweets data, we then perform another API request on all the user IDs collected to find follower relationships between the users tweeting query term. This gives us a directed graph with users tweeting about a particular term as nodes and the directed edges between them representing the direction of the flow of information.

After gathering all the data using Tweepy, we store it in CSV files for easy access and storage. The CSV files are then loaded into the Data Analysis script which cleans the data, performs location update using Geopy and sets up the graphs for users using NetworkX [18]. We create a directed graph based on the "followed by" relationship. The arrow indicates the flow of information. We also use Plotly [19] to plot various graphs, line and scatter plots for our results. Figure 3 shows a graph of user connections for the hashtag query #OntarioTech.

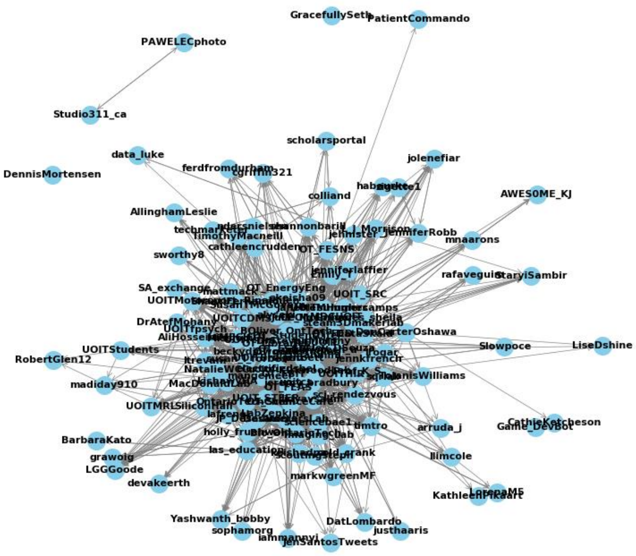


Figure 3. Graph for the users who tweeted using the hashtag #OntarioTech during a 7-day period. For this graph the Average Clustering = 0.5928

5. ANALYSIS

Once we have all our data gathered and have constructed the network graph G, we have access to a host of tools to perform analysis on the data.

5.1. Linear Degree Distribution

For the given network of users, we plot their total degree, in degree (following) and out degree (followed by). Figure 4 shows the plot of the graph G obtained of users who used the hashtag #OntarioTech.

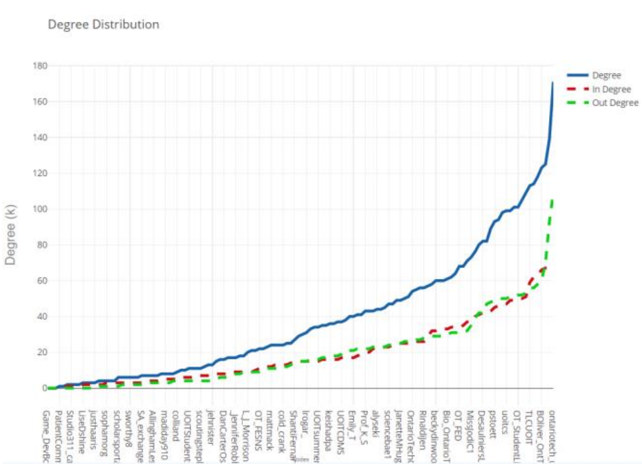


Figure 4. Linear Degree distribution of nodes in G, showing total degree, in degree and out degree. The plot shows OntarioTech_U as the node with the highest total degree and highest in degree

5.2. Average Clustering

Compute the average clustering coefficient for the graph G. The clustering coefficient for the graph is the average,

$$C = \frac{1}{n}\sum_{u \in G} c_u$$

where c_u is the clustering,

$$c_u = \frac{1}{deg^{tot}(u)(deg^{tot}(u)-1) - 2deg^{\leftrightarrow}(u)} T(u)$$

where T(u) is the number of directed triangles through node u, degtot(u) is the sum of in degree and out degree of u and deg↔(u) is the reciprocal degree of u.

5.3. Average Degree Connectivity

The average degree connectivity is the average nearest neighbor degree of nodes with degree k. For a given node i,

$$k^{w}_{nn,i} = \frac{1}{s_i}\sum_{j \in N(i)} w_{ij} k_j$$

where s_i is the weighted degree of node i, w_{ij} is the weight of the edge that links i and j, and N(i) are the neighbors of node i. Figure 4 shows the plot of the average degree connectivity against degree k. This algorithm is also called the "K nearest neighbors".

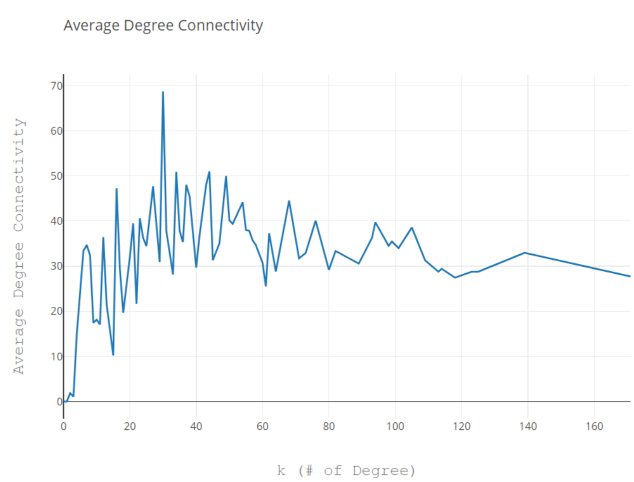


Figure 5. Plot of the "K Nearest Neighbors" against k number of degree for users who tweeted with the hashtag #OntarioTech

5.4. Betweenness Centrality

Betweenness centrality of a node v is the sum of the fraction of all-pairs shortest paths that pass through v

$$c_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$

where V is the set of nodes, σ(s,t) is the number of shortest (s,t)-paths, and σ(s,t|v) is the number of those paths passing through some node v other than s,t. If s=t, σ(s,t)=1, and if v∈s,t, σ(s,t|v)=0. Figure 5 shows the Betweenness Centrality of the users who tweeted with the hashtag #OntarioTech.

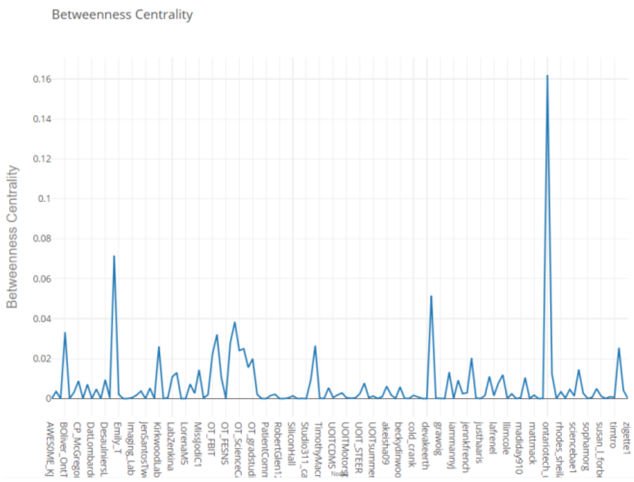


Figure 6. Betweenness Centrality calculation for the users who tweeted with the hashtag #OntarioTech. This shows @OntarioTech_U as the most in between user.

5.5. Closeness Centrality

Closeness centrality of a node u is the reciprocal of the average shortest path distance to u over all n-1 reachable nodes.

$$C(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)}$$

where d(v, u) is the shortest-path distance between v and u, and n is the number of nodes that can reach u. Notice that the closeness distance function computes the incoming distance to u for directed graphs. Figure 7 shows the closeness centrality for the in degree of the graph of users who tweeted with the hashtag #OntarioTech

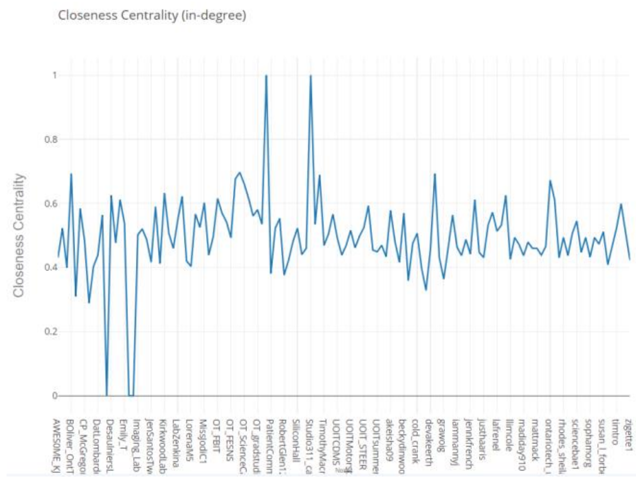


Figure 7. Closeness Centrality for the in degree of the graph of users who tweeted with the hashtag #OntarioTech

5.5.2 Closeness Centrality (Wasserman and Faust)

Wasserman and Faust [21] propose an improved formula for graphs with more than one connected component. The result is "a ratio of the fraction of actors in the group who are reachable, to the average distance" from the reachable actors. Nodes from small components receive a smaller closeness value. Letting N denote the number of nodes in the graph,

$$C_{WF}(u) = \frac{n-1}{N-1} \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)}$$

Figure 8 shows the results of this Wasserman-Faust closeness centrality measure.

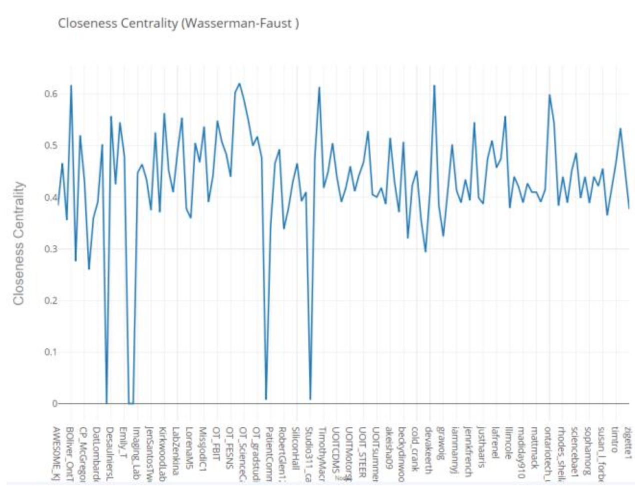


Figure 8. Closeness Centrality based on the Wasserman-Faust algorithm for the in degree of the graph of users who tweeted with the hashtag #OntarioTech

5.6. Community Detection

For community detection we use the Louvain Heuristic [20]. The method is a greedy optimization method that attempts to optimize the "modularity" of a partition of the network. Modularity is defined as:

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\partial(c_i, c_j)$$

Where A_{ij} represents the edge weight between nodes i and j; k_i and k_j are the sum of the weights of the edges attached to nodes i and j, respectively; 2m is the sum of all of the edge weights in the graph; c_i and c_j are the communities of the nodes; and ∂ is a simple kroneker delta function. Figure 9 shows the result of applying this community detection algorithm on the undirected graph of the users who tweeted the hashtag #OntarioTech.

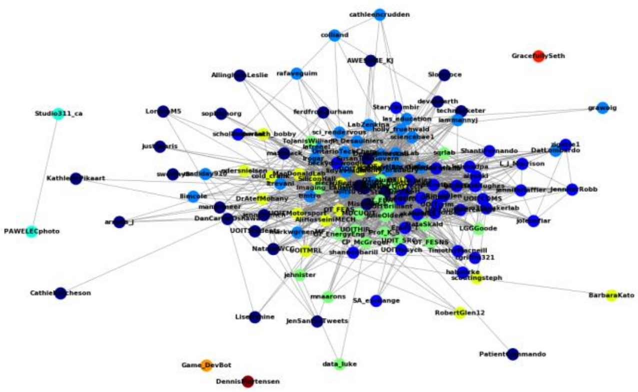


Figure 9. Community Detection algorithm, based on modularity optimization on the users who tweeted the hashtag #OntarioTech

5.7. Location Data

For all the users we ran a check on their geographical coordinates. Twitter allows users to enter an address text string for location. This requires an additional cleaning step to get geographical coordinates from the address text. We are using the open source Nominatim framework to convert readable address text to location coordinates. Figure 10 shows the location data which was fetched for the users who tweeted with the hashtag #OntarioTech.

Figure 10. Location map for the users who tweeted with the hashtag #OntarioTech

5.8. Degree Distribution Plots

For each hashtag analysis we also plot degree distribution plots for the users' network behind it. We plot the frequency of degree, cumulative frequency and complementary cumulative frequency distribution for both in-degrees and out-degrees of the graph. Figure 11 shows the plots for the users' network of people who tweeted with the hashtag #OntarioTech.

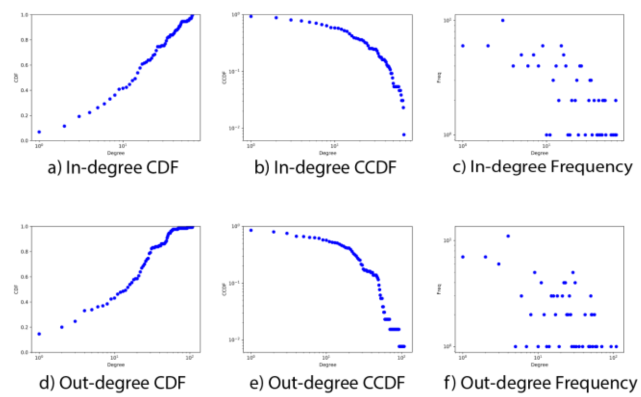


Figure 11. Plots of Frequency, CFD and CCDF against degrees for in-degrees and out-degrees of the network of users who tweeted with hashtag #OntarioTech

For the case of #OntarioTech, our degree distribution doesn't follow the power law distribution. This is probably because this is the new name of the University and is primarily being promoted by a small connected group of people, which deviates from a real world model.

6. FUTURE WORK

There are many directions which we can take to extend this work. The framework can be made into a proper Python library, making it even more accessible and usable for network analysis work. More features can be added such as showing simulations of the flow of information or hashtags over a network overtime like an infection network. This can be achieved using all the existing data collected using the code base and plugging that data in a JavaScript library for graphs and simulations.

We can also try answering questions such as: How are people connected on Twitter? Do people with more followers influence more people? How does information flow through retweets? What is a typical topology of a network on Twitter? How different hashtags correlated with one another? And so on. We can answer many such questions once we have access to sufficient data and have constructed the underlying network of users.

We can also conduct a detailed study of external influence on various hashtags that trend over the network. In particular, how different "News Cycle" stories evolve overtime on the social media platform. Similar to the work on meme tracking done by Leskovec et al [1].

7. CONCLUSION

In this project we have created a Python framework for collecting key network data from Twitter and run network analysis algorithms. The code can query any term, returning all matched tweets. We collect key data points from the tweets text and meta data, which we use to construct users network data. With the network graph we implement various network analysis algorithms such as degree distribution, closeness and betweenness centrality, clustering and community detection.

Using this repository of code, users can start performing detailed analysis on Twitter at a very granular level, analyzing individual Hashtag propagation on an daily scale, perform network analysis on the users tweeting those hashtags and map out location data of the users.

ACKNOWLEDGMENTS

We would like to thank Twitter Inc. for providing an API to their service. The contributors to Tweepy, the Python wrapper for the Twitter API. Geopy for providing an API to Nominatim Open Street Maps. Plotly and NetworkX Python library contributors for providing comprehensive charts and plotting and detailed network creation and analysis tools. Finally, AmirAli Salehi-Abari for his guidance and knowledge.

REFERENCES

[1] J. Leskovec, L. Backstrom, J. Kleinberg (2009). *Meme-tracking and the Dynamics of the News Cycle*. ACM.

[2] Haewoon Kwak, Changhyun Lee, Hosung Park, Sue Moon (2010). *What is Twitter, a social network or a news media?* WWW '10 Proceedings of the 19th international conference on World wide web Pages 591-600

[3] Akshay Java, Xiaodan Song, Tim Finin, Belle Tseng (2007). *Why we twitter: understanding microblogging usage and communities*. Proceedings of the 9th WebKDD.

[4] Alec Go, Richa Bhayani, Lei Huang (2009). *Twitter sentiment classification using distant supervision.* Processing. 150.

[5] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, Krishna P. Gummadi (2010). *Measuring User Influence in Twitter: The Million Follower Fallacy*. AAAI Publications.

[6] Carlos Castillo, Marcelo Mendoza, Barbara Poblete (2011). *Information Credibility on Twitter*. Fourth International AAAI Conference.

[7] Takeshi Sakaki, Makoto Okazaki, Yutaka Matsuo (2010). *Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors*. WWW '10 Proceedings of the 19th international conference on World wide web Pages 851-860

[8] Twitter API documentation: https://developer.twitter.com/en/docs.html

[9] Twitter API wrappers and libraries: https://developer.twitter.com/en/docs/developer-utilities/twitter-libraries.html

[10] Personal repository code for the project: https://github.com/saliknadeem/Twitter-Network-Analysis

[11] Twitter API playment plans: https://developer.twitter.com/en/pricing.html

[12] Geopy library: https://geopy.readthedocs.io/en/stable/

[13] Nominatim Open Streep Map Framework: https://nominatim.openstreetmap.org/

[14] Python programming language: https://www.python.org/

[15] Anaconda Package Manager for Python: https://www.anaconda.com/

[16] Github - The world's leading software development platform: https://github.com/

[17] Tweepy – Python Wrapper for Twitter API: http://www.tweepy.org/

[18] NetworkX - Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks: https://networkx.github.io/

[19] Plotly - develops online data analytics and visualization tools: https://plot.ly/

[20] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, Etienne Lefebvre (2008). *Fast unfolding of communities in large networks*. IOP Publishing Ltd

[21] pg. 201 of Wasserman, S. and Faust, K., Social Network Analysis: Methods and Applications, 1994, Cambridge University Press.