FINAL REPORT

11/30/2020

Vrinda, Inhee & Anirudh

Power of BTS ARMY for Social Change Envisaged by Twitter Network Analysis

Table of Contents

1 Twitter Data Crawling of #MatchMillion for 24 Hours

2 Setup MongoDB [Anirudh]

3 Retweet Network Structural Analysis [inhee]
- 3.0 Data Preparation
- 3.1 Construction & Characterization of the Network
- 3.2 Community Detection & Modulity

4 Retweet Network NLP-based Behavioral/Functional Analysis [inhee]
- 4.1 Word Embeddings of User's Bio by Word2Vec
- 4.2 PCA-assisted K-means Clustering for User's Bio in terms of Word2Vec

5 Dynamic Retweet Network (Evolution over 24 Hours) [Anirudh]
- 5.0 Data Preparation
- 5.1 Construction & Structural Analysis

6 Followers Network Structural Analysis [Optional]
- 6.0 Data Preparation
- 6.1 Construction & Characterization of the Network
- 6.2 Community Detection & Modulity

7 Followers Network NLP-based Behavioral/Functional Analysis [Optional]
- 7.0 Data Preparation
- 7.1 Word Embeddings of User's Bio by Word2Vec
- 7.2 K-means Clustering of User's Bio in terms of Word2Vec

8 Semantic Network Analysis on Original Tweets Network [Vrinda]

9 How to Find a Local Bridge Among Communities [Vrinda]

10 NLP Analysis on Original Tweets Network [Optional]

1 Twitter Data Crawling of #MatchMillion for 24 Hours

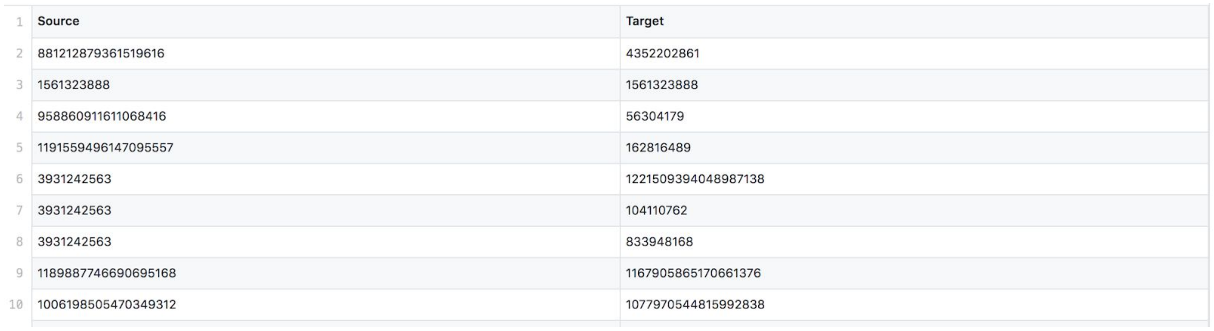
All three team members have applied to the Twitter API Developer Accounts, and have been granted the required credentials. We have scrapped the Twitter data for one hashtag #MatchAMillion for 24 hours time windows at every 15 minutes interval between 2020-06-07-2020-06-08 using python script.

2 Setup MongoDB [Anirudh]

3 Retweet Network Structural Analysis [inhee]

3.0 Data Preparation

To build a social network we prepared the "Edge" table data: connectivity between users (source and target) via retweeting will serve as directed Edge information. In total, we have 12,349 retweets shown below. We then import this edge data to the Gephi.

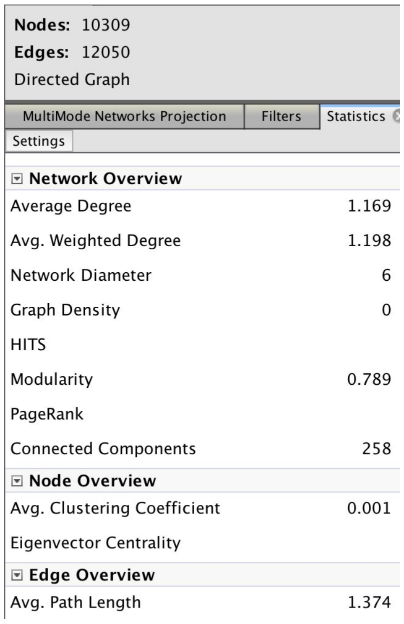


Source	Target
881212879361519616	4352202861
1561323888	1561323888
958860911611068416	56304179
1191559496147095557	162816489
3931242563	1221509394048987138
3931242563	104110762
3931242563	833948168
1189887746690695168	1167905865170661376
1006198505470349312	1077970544815992838

...

3.1 Construction & Characterization of the Network

- Number of nodes, edges and connected components.
- The diameter (longest shortest path).
- The five nodes with the highest clustering coefficients.
- The five nodes with highest betweenness centrality (node betweenness).
- Community detection



Nodes: 10309
Edges: 12050
Directed Graph

MultiMode Networks Projection | Filters | Statistics
Settings

Network Overview

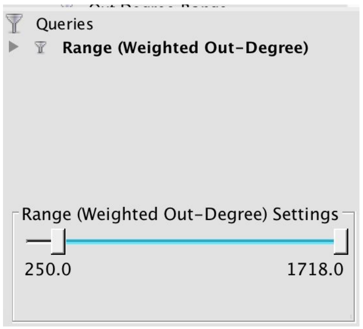
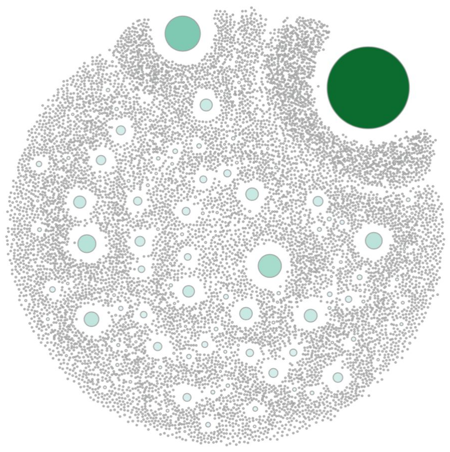
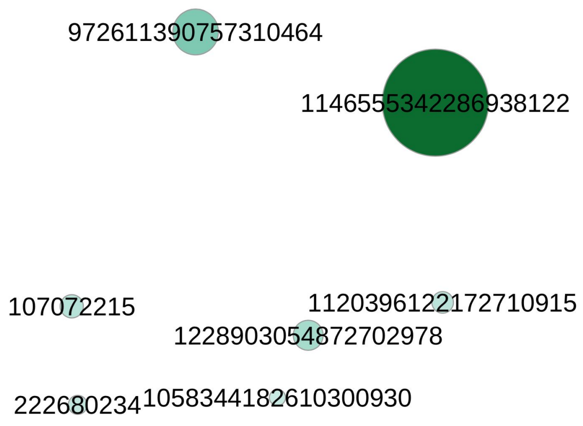
Average Degree	1.169
Avg. Weighted Degree	1.198
Network Diameter	6
Graph Density	0
HITS	
Modularity	0.789
PageRank	
Connected Components	258

Node Overview

Avg. Clustering Coefficient	0.001
Eigenvector Centrality	

Edge Overview

Avg. Path Length	1.374

Entire 24 Hour Retweet Network Node Size & Color : Out-degree (source user ID→ target user ID)	Entire 24 Hour Retweet Network Filtering the node with >= Out-degeree of 250  Queries **Range (Weighted Out-Degree)** Range (Weighted Out-Degree) Settings 250.0 1718.0
	 972611390757310464 1146555342286938122 107072215 1120396122172710915 1228903054872702978 222680234 1058344182610300930

Those users (1146555342286938122, 972611390757310464, 1120396122172710915107072215, 107072215, 1228903054872702978, 222680234, and 1058344182610300930) are original tweet message writers [sources]. They also have been retweeted by at least 200 other users [targets].

3.2 Community Detection & Modulity

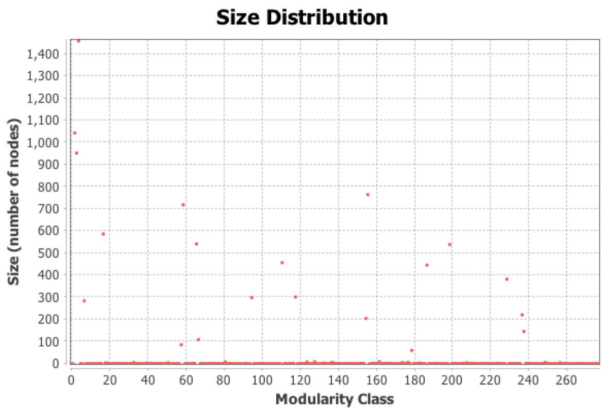
Modularity Report

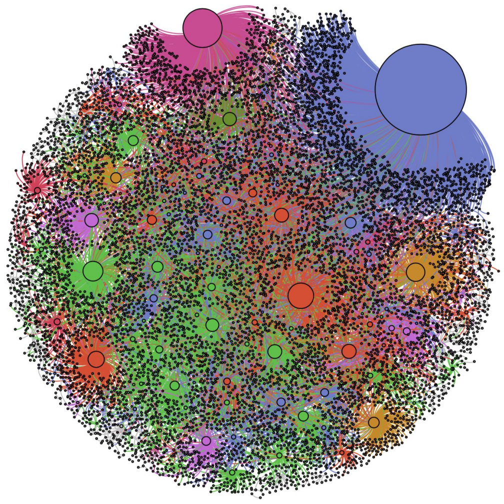
Parameters:

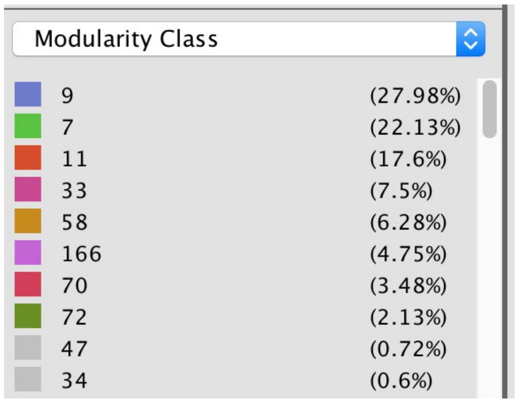
Randomize: On
Use edge weights: On
Resolution: 3.0

Results:

Modularity: 0.791
Modularity with resolution: 2.517
Number of Communities: 277







Modularity Class

🟦	9	(27.98%)
🟩	7	(22.13%)
🟥	11	(17.6%)
🟪	33	(7.5%)
🟧	58	(6.28%)
🟪	166	(4.75%)
🟥	70	(3.48%)
🟩	72	(2.13%)
⬜	47	(0.72%)
⬜	34	(0.6%)

We have experimented with different resolution numbers to adjust the resulting number of distinct communities. We found that using resolution of 3.0 resulted in around 10 communities as shown above. The above community detection figure represents distinct communities in different colors and its node side represents out-degree.

4 Retweet Network NLP-based Behavioral/Functional Analysis [inhee]

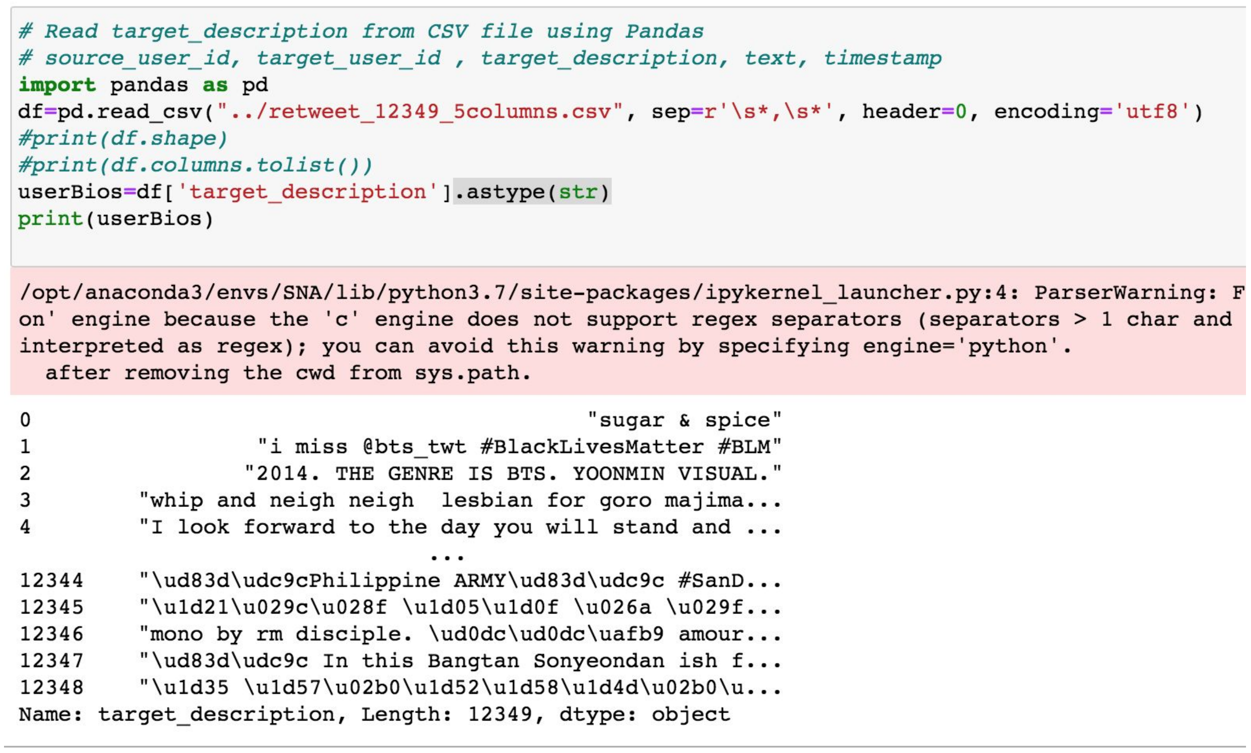
From the network-structural-based clusterings by community detection, we found top five clusters centered with a source user with >=200 out-degrees (i.e. original tweet writers and their IDs are 1146555342286938122, 972611390757310464, 1120396122172710915107072215, 107072215, 1228903054872702978, 222680234, and 1058344182610300930).

We would like to compare that structure-based clustering with a behavior-based clustering by using the twitter users information, especially their user description (aka bio).

4.1 Word Embeddings of User's Bio by Word2Vec

Basically, we will convert retweet user's description (https://github.com/TheChirpyWitch/BLMAndBTSArmy/blob/master/RetweetSNA/retweet_12349_5columns.csv) to numeric vectors using NLP embedding methods to prepare for the numeric data, so that we can further analyze them using Unsupervised Machine Learning techniques such as k-means clustering combined with Principal Component Analysis (PCA) in a reduced dimensional space.
Jupyter notebook is available in our github.
https://github.com/TheChirpyWitch/BLMAndBTSArmy/blob/master/RetweetSNA/NLP/Retweet_NLP.ipynb

1. Extract user's bio text

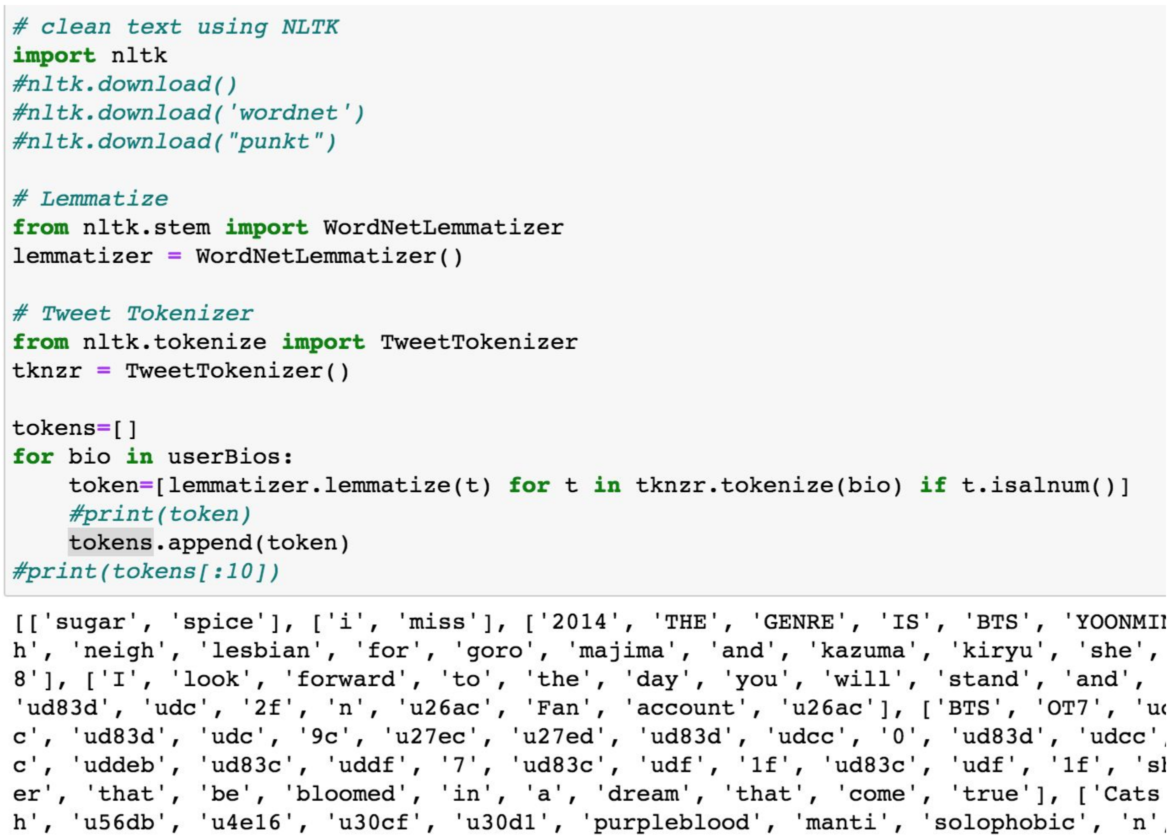


```
# Read target_description from CSV file using Pandas
# source_user_id, target_user_id , target_description, text, timestamp
import pandas as pd
df=pd.read_csv("../retweet_12349_5columns.csv", sep=r'\s*,\s*', header=0, encoding='utf8')
#print(df.shape)
#print(df.columns.tolist())
userBios=df['target_description'].astype(str)
print(userBios)
```

```
/opt/anaconda3/envs/SNA/lib/python3.7/site-packages/ipykernel_launcher.py:4: ParserWarning: F
on' engine because the 'c' engine does not support regex separators (separators > 1 char and
interpreted as regex); you can avoid this warning by specifying engine='python'.
  after removing the cwd from sys.path.
```

```
0                                           "sugar & spice"
1                        "i miss @bts_twt #BlackLivesMatter #BLM"
2                       "2014. THE GENRE IS BTS. YOONMIN VISUAL."
3               "whip and neigh neigh  lesbian for goro majima...
4               "I look forward to the day you will stand and ...
                                               ...
12344       "\ud83d\udc9cPhilippine ARMY\ud83d\udc9c #SanD...
12345       "\u1d21\u029c\u028f \u1d05\u1d0f \u026a \u029f...
12346       "mono by rm disciple. \ud0dc\ud0dc\uafb9 amour...
12347       "\ud83d\udc9c In this Bangtan Sonyeondan ish f...
12348       "\u1d35 \u1d57\u02b0\u1d52\u1d58\u1d4d\u02b0\u...
Name: target_description, Length: 12349, dtype: object
```

2. Lemmatize and Tokenize the user bio text

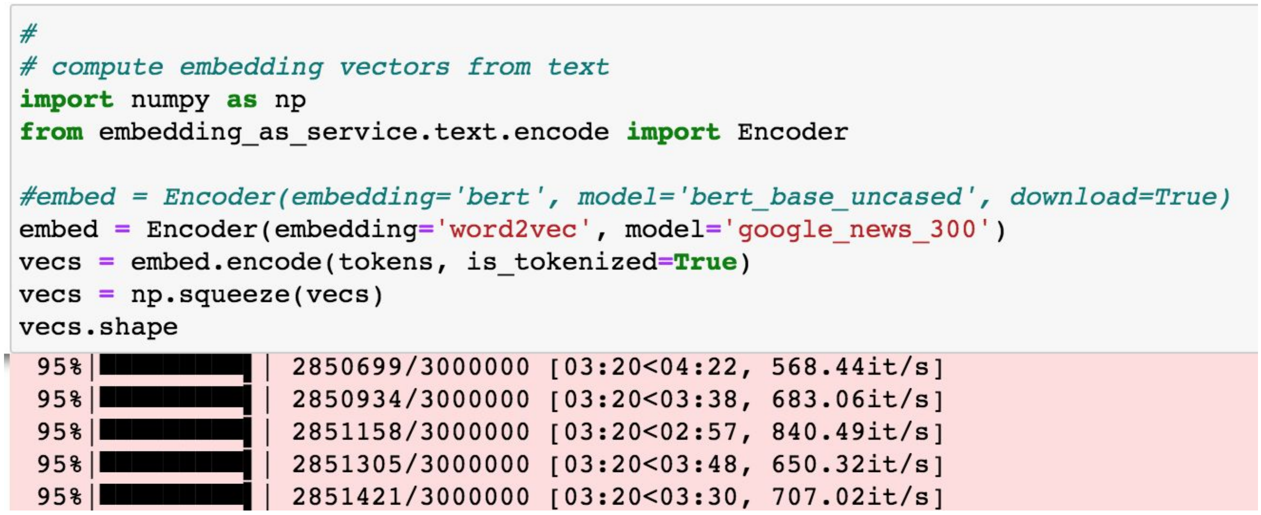


```
# clean text using NLTK
import nltk
#nltk.download()
#nltk.download('wordnet')
#nltk.download("punkt")

# Lemmatize
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

# Tweet Tokenizer
from nltk.tokenize import TweetTokenizer
tknzr = TweetTokenizer()

tokens=[]
for bio in userBios:
    token=[lemmatizer.lemmatize(t) for t in tknzr.tokenize(bio) if t.isalnum()]
    #print(token)
    tokens.append(token)
#print(tokens[:10])
```

```
[['sugar', 'spice'], ['i', 'miss'], ['2014', 'THE', 'GENRE', 'IS', 'BTS', 'YOONMI
h', 'neigh', 'lesbian', 'for', 'goro', 'majima', 'and', 'kazuma', 'kiryu', 'she',
8'], ['I', 'look', 'forward', 'to', 'the', 'day', 'you', 'will', 'stand', 'and',
'ud83d', 'udc', '2f', 'n', 'u26ac', 'Fan', 'account', 'u26ac'], ['BTS', 'OT7', 'u
c', 'ud83d', 'udc', '9c', 'u27ec', 'u27ed', 'ud83d', 'udcc', '0', 'ud83d', 'udcc',
c', 'uddeb', 'ud83c', 'uddf', '7', 'ud83c', 'udf', '1f', 'ud83c', 'udf', '1f', 's
er', 'that', 'be', 'bloomed', 'in', 'a', 'dream', 'that', 'come', 'true'], ['Cats
h', 'u56db', 'u4e16', 'u30cf', 'u30d1', 'purpleblood', 'manti', 'solophobic', 'n',
```

3. Convert text tokens into numerical vectors using text embeddings

To extract user's behavioral information from their bio in their twitter accounts, we utilized a word embedding method in NLP. There is a consolidated TF-hub for text-embeddings where pre-trained various NLP embedding models are available. The "Embeddings-As-Service" further wrapped the TF-hub models in a form of python API, so that we can just input our text, then can get the embedded numeric vectors without any training.



```
#
# compute embedding vectors from text
import numpy as np
from embedding_as_service.text.encode import Encoder

#embed = Encoder(embedding='bert', model='bert_base_uncased', download=True)
embed = Encoder(embedding='word2vec', model='google_news_300')
vecs = embed.encode(tokens, is_tokenized=True)
vecs = np.squeeze(vecs)
vecs.shape
```

```
 95%|██████████ | 2850699/3000000 [03:20<04:22, 568.44it/s]
 95%|██████████ | 2850934/3000000 [03:20<03:38, 683.06it/s]
 95%|██████████ | 2851158/3000000 [03:20<02:57, 840.49it/s]
 95%|██████████ | 2851305/3000000 [03:20<03:48, 650.32it/s]
 95%|██████████ | 2851421/3000000 [03:20<03:30, 707.02it/s]
```

```
import numpy as np
flatten_vecs = []
for v in vecs:
    flatten_vecs.append(v.flatten())

np_word2vec = np.array(flatten_vecs)
```

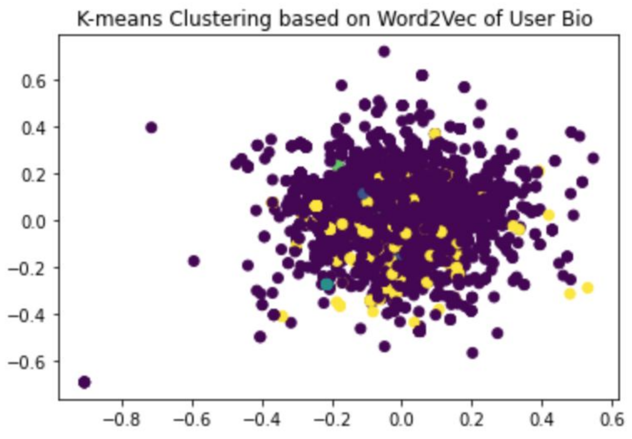
```
#print(np_word2vec[:100,0])
```

```
[-0.1875      -0.22558594  0.          0.05688477  0.07910156 -0.24414062
 -0.01281738  0.0324707   0.         -0.20507812  0.          0.08837891
  0.         -0.16210938  0.28125     0.          0.         -0.16992188
 -0.22558594  0.          0.          0.          0.          0.00215149
 -0.16992188  0.          0.03320312  0.         -0.10595703 -0.10595703
 -0.14648438 -0.22558594  0.28125     0.0324707   0.          0.
```

4.2 PCA-assisted K-means Clustering for User's Bio in terms of Word2Vec

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

X=np_word2vec

kmean = KMeans(5).fit(X)

plt.scatter(X[:, 0], X[:, 1], c=kmean.labels_.astype(float))
plt.title("K-means Clustering based on Word2Vec of User Bio ")


print(kmean.labels_)
```

```
[0 0 4 ... 0 0 0]
```



5 Dynamic Retweet Network (Evolution over 24 Hours) [Anirudh]

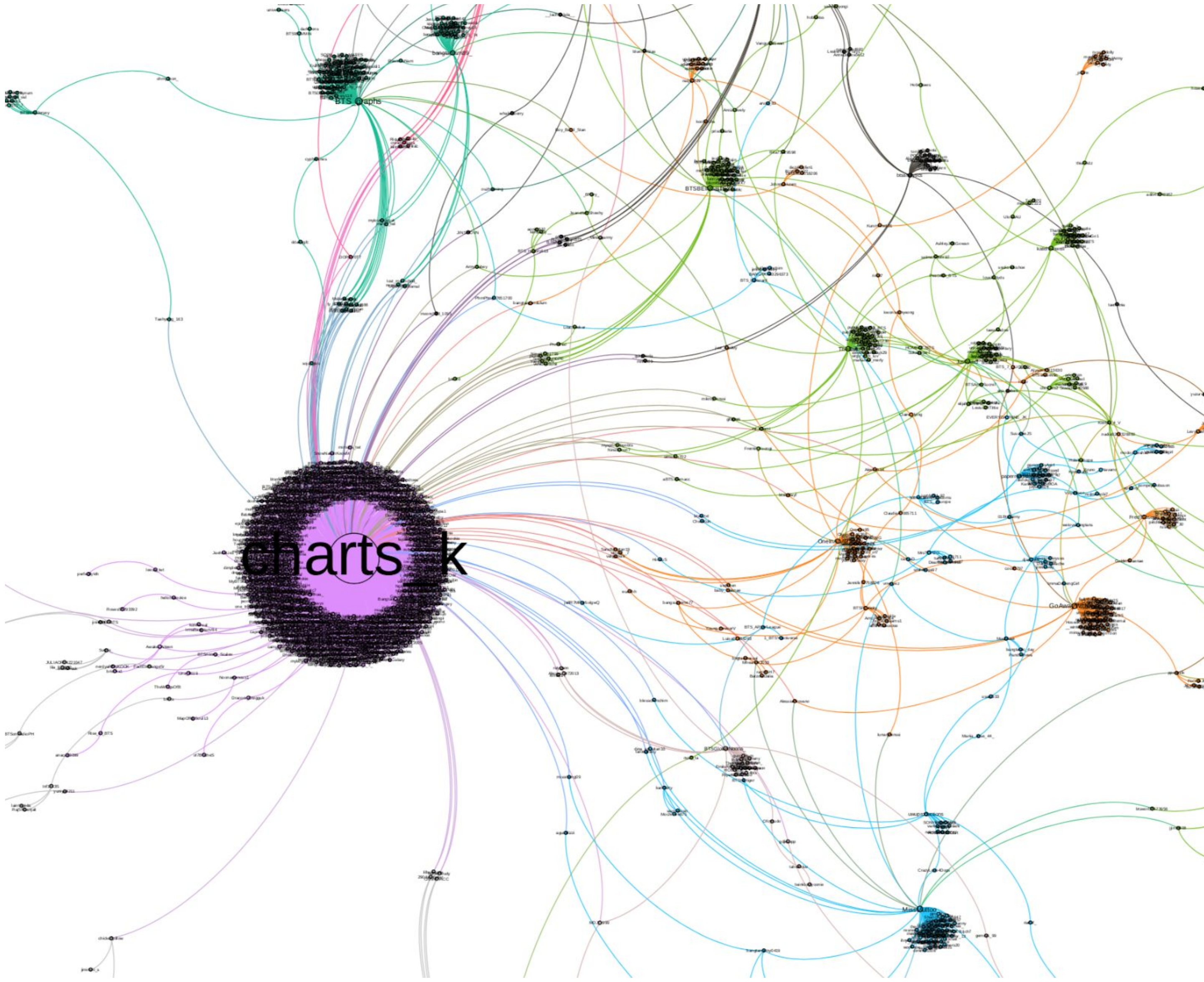
5.0 Data Preparation & Steps

1. Use python script to convert our collected entire twitter data in JSNL format (part1, part2) to GEXF (Graph Exchange XML Format) to use Dynamic/Temporal Network of Gephi.
   [test]
   - Input : JSONL_FILE = 'tweetsData.jsonl'
   - Output: GEXF_FILE = 'demo_network.gexf'

   [all 13,200 tweets]
   - Input: JSONL_FILE = 'twitterDataset_All_13200.json' part1, part2
   - Output: GEXF_FILE = 'retweetAll_network.gexf'
2. Import "gexf" from Gephi
3. Click "overview" button → "statistics" → compute structural statistics, especially "modularity" (resolution 3.0 or so to have ~10 major communities with > 1% population) → node colors partition by "modularity class" → node size partition by "out-degree" → "enable time" → shrink "time sliding bar" width (see temporal graph tutorial in detail)
4. May create snapshots (maybe every 4 hours or so)
5. Zoom in each snapshot and observe some "local bridge"?

5.1 Construction & Structural Analysis

- Snapshot on time interval 4:43PM-5:00
  Majority cluster is centered at user "charts_k".


charts_k
BTS_Graphs

PROPOSAL

10/13/2020

Vrinda, Inhee & Anirudh

Power of BTS ARMY for Social Change
Envisaged by Twitter Network Analysis

Motivation

In June 2020, the international sensation **BTS** (a group of 7 South Korean musicians) donated one million dollars for **Black Lives Matter**. Within 24 hours, their **fandom** called **ARMY** had mobilised via **Twitter** and matched BTS's one million dollar donation by using social media networks under the hashtags **#MatchAMillion , #Match1Million and #MatchTheMillion** as reported here. [1]

This collective effort sparked the idea for our project. If you are on Twitter, you would have come across the BTS ARMY at some point. Their mobilisation power for social causes [2] is unlike any other group on the **internet with multiple collectives hosting fundraisers for various causes - but we want back that claim with numbers.**

We also want to analyse how BTS ARMY's reach led to this unprecedented donation and how it inspired people who might have been living on the other side of the world to donate for a worthy cause. As pointed out in this article, [3] **ARMY** had help from other **fanbases** and people who were passionate about **Black Lives Matter Movement**. We want to try to see how these two clusters came together along with analysing some salient features of this **network using data scraped from Twitter.**

Proposed Method

To achieve the goal outlined in the above section, we will follow these seven steps:
1) Twitter data crawling;
2) Construction of a network;
3) Measurement of structural metrics of the network ;
4) Community detection to infer the behavior/function from the network structure;
5) Monitor community evolution by constructing a dynamic temporal network (our tweet data has 24-hour span); and
6) Evaluation of each detected community by constructing wordcloud (expected to see a distinct dominant word from each community).
7) Based on the findings from the steps above, we will conduct further measurement of influence and homophily.[7] Basically we would like to observe how the influence and homophily generate similarity in our twitter-based social network.

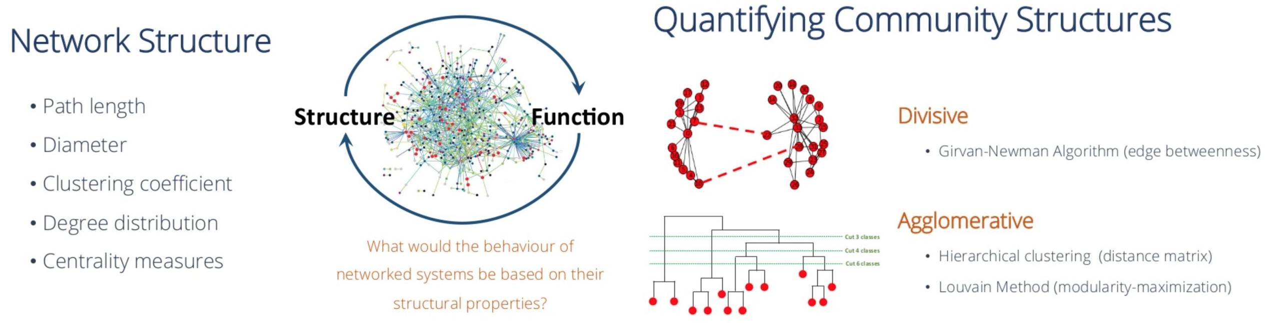
- **Creation of a social network**
  - Data Scraping from Twitter API
  - Setting nodes and edges to create a network using:
    a. Twitter's follower rules to obtain a directed network
    b. NLP-based phrase matching using word embeddings

All three team members have already applied to the Twitter API Developer Accounts [4], and have been granted the required credentials. We have started to scrap the Twitter data for one of hashtags. To build a social network we should prepare the two types of data tables:

(1) "Node" table data: twitter user IDs can serve as Node information; we also compute some NLP-based numerics of each user (such as TF-IDF, word embeddings) from their recent 20 tweets or their profile description to classify whether the user might belong to BTS fans or not.

(2) "Edge" table data: connectivity between users (source and target) via retweeting will serve as directed Edge information; also, connectivity between users sharing same hash-tags (either BTS or BLM) will also serve as undirected but weighted (counted by co-occurrence of different hash-tags) Edge information. Basically, we will convert Tweets text to numeric vectors using NLP Embedding Methods to prepare for the node attribution data.

- **Characterization of the network**
  - Number of nodes, edges and connected components.
  - The diameter (longest shortest path).
  - The five nodes with the highest clustering coefficients.
  - The five nodes with highest betweenness centrality (node betweenness).
  - Assortativity of the nodes.
  - Classification of nodes into BTS ARMY vs. not.
  - Community detection for BTS ARMY using:
    a. Node Similarity using NLP based word embeddings (ML-based clustering)
    b. Girvan-Newman algorithm (edge-betweenness-based)
    c. Louvain algorithm (modularity-maximization-based)
    d. Hierarchical clustering (distance-matrix-based)


Network Structure
• Path length
• Diameter
• Clustering coefficient
• Degree distribution
• Centrality measures
Structure
Function
What would the behaviour of networked systems be based on their structural properties?
Quantifying Community Structures
Divisive
• Girvan-Newman Algorithm (edge betweenness)
Agglomerative
• Hierarchical clustering (distance matrix)
• Louvain Method (modularity-maximization)
Cut 3 classes
Cut 4 classes
Cut 6 classes

On one hand, from the network structure-based metrics, we can detect communities by mainly three methods: 1) divisive, edge betweenness-based "Girvan-Newman" algorithm; 2) modularity-maximization-based “Louvain” algorithm; and 3) distance-matrix-based “Hierarchical” clustering.

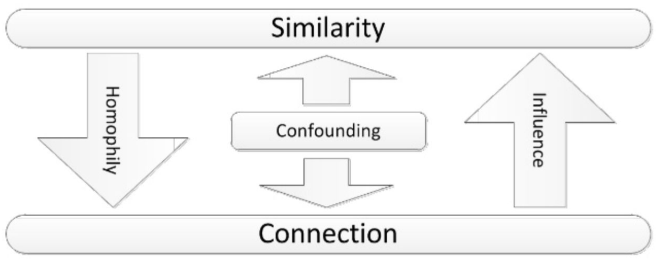
On the other hand, we can also detect communities using the ML-based, unsupervised “Clustering” method using the pre-computed node attributes such as word embedding vectors.

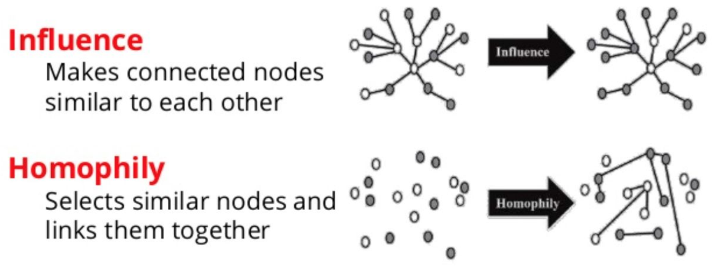
Major difference between **community detection** and **clustering** is that in community detection, individuals are connected to others via a network of links, whereas in clustering, data points are not embedded in a network. [7]

- **Visualization of the network**
  - A static visualisation of the networks using iGraph
  - Network evolution using temporal networks using Gelphi
  - Community evaluation using word clouds using iGraph

We will collect #MatchAMillion and#BlackLivesMatter hashtags for 24 hours time windows between 2020-06-07-2020-06-08. From the collected tweet data, we will extract retweeting connectivity in the form of three columns: source userID; target userID; time point (at every 15 minutes interval). We then create the dynamic (temporal) network using Gephi software. User's geographic location, if available, will be mapped onto the nodes and evolution of modularity (mapped onto the edges) will be observed over 24 hours. [8-10]

- **Report of results**
  - Collating a report with the observations from above.
  - Novelties in the project include:
    a. Identify key players by measuring influence:
        a) Within the fandom
        b) Outside the fandom
    b. Identifying ARMY accounts by measuring homophily
    c. Sentiment analysis of key ARMY accounts: During this 24 hour period, the fandom went through a lot of collective emotions, and we want to analyse those.




Influence
Makes connected nodes similar to each other
Influence
Homophily
Selects similar nodes and links them together
Homophily

As shown in figures above, “Influence” and “Homophily” are main driving social forces connecting individuals. By measuring influence and homophily from our constructed tweeter based network, we would expect to observe similarities between ARMY, also observe mutual influence between ARMY to BlackLivesMatter supporters for social awareness.

References

[1] BTS ARMY Matched The Group's $1 Million Black Lives Matter Donation, Proving The Positive Power Of Fandoms
https://www.forbes.com/sites/bryanrolli/2020/06/08/bts-army-black-lives-matter-1-million-donation/#4cdc05f56465
[2] BTS' ARMY are flipping conversations about stan culture, one good deed at a time
https://thefortyfive.com/opinion/bts-army-charity-work/
[3] How the South Korean band's fanbase – known as ARMY – raised over $1 million for the Black Lives Matter movement, mostly in just one day.
https://graphics.reuters.com/GLOBAL-RACE/BTS-FANS/nmopajgmxva/
[4] Get started with Twitter APIs and tools Apply for access
https://developer.twitter.com/en/apply-for-access
https://medium.com/swlh/extracting-tweets-using-twitter-premium-search-api-and-python-2d025144e8a4
[5] Embedding-as-Service : One-Stop Solution to encode sentence to fixed length vectors from various embedding techniques
https://pypi.org/project/embedding-as-service/
[6] Universal-Sentence-Encoder: Collection of universal sentence encoders trained on a variety of data.
https://tfhub.dev/google/collections/universal-sentence-encoder/1
[7] Zafarani, R., Abbasi, M., & Liu, H. (2014). Social Media Mining: An Introduction. Cambridge: Cambridge University Press. doi:10.1017/CBO9781139088510
[8] Mapping users (nodes) onto geographic location
https://gephi.org/plugins/#/plugin/geolayout-plugin
[9] Creating a simple dynamic network
https://seinecle.github.io/gephi-tutorials/generated-html/creating-a-simple-dynamic-network.html
[10] Converting a network with dates into a dynamic network
https://seinecle.github.io/gephi-tutorials/generated-html/converting-a-network-with-dates-into-dynamic.html