

What did they say?

Network analysis of Twitter quotes @JoeBiden and @realDonaldTrump during 2020 United States presidential election (second debate)

by Natchanun Sanitdee

LDA-H312 Method Course in Digital Humanities II: Applying Network Analysis to Humanities

Faculty of Arts, University of Helsinki

January, 2023

1. Introduction

Since the US plays a crucial role in global politics, the presidential elections have been closely monitored. One of the ways to inspect the elections is through social media. Twitter is among the most popular social media used to examine public opinion because of the availability of datasets. It is interesting to explore and compare what the US netizens say about the Republican and Democratic presidential nominees for president in the 2020 US election (second debate) on Twitter, as well as the characteristics of users based on their ID names. The dataset used in this study is derived from Tweetsets. The dataset is analyzed under the framework of Social Network Analysis (SNA) (Moreno, Jacob 1934) and Discourse Analysis.

The results show that @realDonaldTrump network is more extensive and has more active tweeters. For text analysis, it is divided into two parts: tweet texts and usernames, as well as the unique words in each community. The content in @realDonaldTrump network also contains more criticisms and sarcastic messages. The tweets in the network are also more diverse in terms of people who are mentioned, though the usernames in the network are less meaningful and interpretable than @JoeBiden network.

2. Datasets

The tweet IDs are retrieved from TweetSets. TweetSets website provides Twitter datasets for research and archiving created by George Washington University. The tweet IDs used in this study are found under the topic of “2020 United States Presidential Election: Second Debate”. The tweet collection lasted during 21-24 October, 2020-10-21. A total Tweet counts are 4,832,700. All tweet types: Original, Quote, Retweet, and Reply are selected. We ended up with .txt files with 420,222 tweet IDs for @realDonaldTrump and 198,025 tweet IDs for @JoeBiden. We download Tweet ID files and use the programme Hydrator (version 0.3.0) to pull more information like the user IDs of those whose mentions, quotes, and replies include @realDonaldTrump or @JoeBiden. We save .csv files and upload them to the R programme for further analysis.

3. Methods

From the full table, `user_id` and `quote_id` are selected and put into a new edge list table. Blank cells are filtered out, and the two columns are grouped into weight. The table is turned into a directed `'tbl_graph'`. Since most tweets only mention @realDonaldTrump and @JoeBiden only once, to enhance the clarity of the network, only the pairs with the weight more than 1 are taken into account. In other words, the network will concern only the users who mention @realDonaldTrump and @JoeBiden in their tweets more than once. Finally, with this graph, the global metric and node-level statistics are calculated.

Then, a community detection algorithm is used to zoom into the clusters of users and tweets with the top 10 highest degree. The tweet texts, as well as usernames, are analyzed based on discourse analysis framework to see if there are common characteristics and how tweets and usernames related to @realDonaldTrump and @JoeBiden are different.

4. Results

First, we install packages and load in necessary libraries.

```
library(tidygraph)
library(tidyverse)
library(ggraph)
library(igraph)
library(ggraph)
install.packages("tidytext")
library(tidytext)
tinytex::install_tinytex()
```

4.1. Edge-level Metrics (Global Metrics)

After the hydrated tweet files of @JoeBiden and @realDonaldTrump are uploaded, the tables of tweets are created for each in R. Then, from the tweet files, the edge list files are made. The edge list files will allow us to calculate global metric statistics, generate a `tbl_graph` table, generate a node list, and also draw a network diagram at a later stage.

```
joebiden = read_csv("../my-work/@JoeBiden.csv", col_types = c("user_id"="character", "id"="character", "r
edge_list_joe = joebiden %>%
  #Save the object in a new table called edge_list_joe.
  filter(!is.na(quote_id)) %>%
  #Filter out the N/A value in quote_id column.
  group_by(user_id, quote_id) %>%
  #Group user IDs and quote IDs using group_by function. The two columns act as source and target which
  filter(!is.na(user_id)) %>%
  #Filter out the N/A value in user_id column.
  tally(name = 'weight') %>%
  #Count user IDs and quote IDs using tally function.
  filter(weight>1) %>%
  #Use filter function to only keep the edges that have more than one degree because most tweeters in t
  arrange(desc(weight))
  #Arrange the table based on the value of 'weight' from high to low.
```

Next, we repeat the same edge list creation processes for @realDonaldTrump.

```

donaldtrump = read_csv("../my-work/@realDonaldTrump.csv", col_types = c("user_id"="character","id"="cha

edge_list_trump = donaldtrump %>%
  filter(!is.na(quote_id)) %>%
  group_by(user_id, quote_id) %>%
  filter(!is.na(user_id)) %>%
  tally(name = 'weight') %>%
  filter(weight>1) %>%
  arrange(desc(weight))

```

We, then, turn the network objects in to a directed ‘tbl_graph’ named ‘g1’ for @JoeBiden and ‘g2’ for @realDonaldTrump. The details of the graph tables including a number of nodes and edges are displayed below.

```

g1 = edge_list_joe %>%
  as_tbl_graph(directed = TRUE)

g1

```

```

## # A tbl_graph: 378 nodes and 227 edges
## #
## # A rooted forest with 151 trees
## #
## # Node Data: 378 x 1 (active)
##   name
##   <chr>
## 1 43646823
## 2 1269258105235308551
## 3 1299457248243089409
## 4 1203737149759201280
## 5 1190025969244889088
## 6 1164661144386248704
## # ... with 372 more rows
## #
## # Edge Data: 227 x 3
##   from to weight
##   <int> <int> <int>
## 1     1   204     57
## 2     2   205     29
## 3     3   206     15
## # ... with 224 more rows

```

```

g2 = edge_list_trump %>%
  as_tbl_graph(directed = TRUE)

g2

```

```

## # A tbl_graph: 626 nodes and 403 edges
## #
## # A directed acyclic simple graph with 224 components
## #
## # Node Data: 626 x 1 (active)

```

```
##   name
##   <chr>
## 1 43646823
## 2 1216876347198324736
## 3 27129936
## 4 1275891762855317507
## 5 475549688
## 6 1042752898130751490
## # ... with 620 more rows
## #
## # Edge Data: 403 x 3
##   from   to weight
##   <int> <int> <int>
## 1     1    362    29
## 2     2    363    25
## 3     3    363    24
## # ... with 400 more rows
```

The finished network for @JoeBiden has 378 nodes and 227 edges, and 626 nodes and 403 edges for @realDonaldTrump network. With the same filter (degree of more than one) @realDonaldTrump network has almost two times more nodes and edges than @JoeBiden network.

With the graph objects 'g1' and 'g2', we can calculate global metric statistics: network density, transitivity, and average path length. The density of @JoeBiden network is saved as an object called 'd1', global clustering in 'c1', and average path length in 'p1', and 'd2' 'c2', and 'p2' for @realDonaldTrump network respectively.

The global metric figures of @JoeBiden and @realDonaldTrump are put in a table using tibble() function and saved as the object called 'global_stats'.

```
d1 = g1 %>% igraph::graph.density()
c1 = g1 %>% transitivity(type = 'global')
p1 = g1 %>% igraph::average.path.length()

d2 = g2 %>% igraph::graph.density()
c2 = g2 %>% transitivity(type = 'global')
p2 = g2 %>% igraph::average.path.length()

global_stats = tibble(Global_Metrics = c('Density', 'Global Clustering', 'Average Path Length'), Joe_Biden = d1, Donald_Trump = d2, Joe_Biden = c1, Donald_Trump = c2, Joe_Biden = p1, Donald_Trump = p2)

global_stats
```

```
## # A tibble: 3 x 3
##   Global_Metrics      Joe_Biden Donald_Trump
##   <chr>              <dbl>      <dbl>
## 1 Density            0.00159      0.00103
## 2 Global Clustering    0          0
## 3 Average Path Length  3.04       3.43
```

The global metric table demonstrates that @JoeBiden network is denser than @realDonaldTrump (global density of 0.00159 and 0.00103 respectively) and @JoeBiden network also has a shorter average path length (3.04 and 3.43 respectively).

4.2. Node-level Metrics

Next, we generate a table of nodes that will be used to calculate node level statistics (degree, betweenness, and eigenvector centrality). Below are the node tables for @JoeBiden and @realDonaldTrump networks respectively.

```
nodes_table_joe = g1 %>%
  mutate(degree = centrality_degree(mode = 'all')) %>%
  #Create new columns in the network nodes using mutate() function. Change the degree method from 'in'
  mutate(between = centrality_betweenness(directed = F)) %>%
  #Change the betweenness centrality measurement to 'undirected'.
  mutate(eigen = centrality_eigen(weights = NULL, directed = F)) %>%
  #Add the weight column to the eigenvector centrality measurement.
  as_tibble() %>%
  #Turn it from a network object into a regular dataframe, using as_tibble(), and arrange the display order
  arrange(desc(degree))

nodes_table_joe
```

```
## # A tibble: 378 x 4
##   name          degree between eigen
##   <chr>         <dbl>   <dbl> <dbl>
## 1 1319420241227714560      14      91      1
## 2 992890858339905537       9      52      0
## 3 279270981              6      26      0
## 4 1319474222306045954       6      15      0
## 5 1319419711046799360       5      18      0
## 6 1319420772469727232       4       6      0
## 7 3035730489              3       8      0
## 8 881813910327672832       3       3      0
## 9 1319466818231218179       3       3      0
## 10 1319486308046176256      3      19      0
## # ... with 368 more rows
```

```
nodes_table_trump = g2 %>%
  mutate(degree = centrality_degree(mode = 'all')) %>%
  mutate(between = centrality_betweenness(directed = F)) %>%
  mutate(eigen = centrality_eigen(weights = NULL, directed = F)) %>%
  as_tibble() %>%
  arrange(desc(degree))

nodes_table_trump
```

```
## # A tibble: 626 x 4
##   name          degree between eigen
##   <chr>         <dbl>   <dbl> <dbl>
## 1 1319499984358805504      22    1442 1e+0
## 2 1319497720244162560      18     901 1.48e- 2
## 3 1319467519820922881      12      66 2.29e-16
## 4 1225497537341198339      10      75 1.19e-16
## 5 1298384504512557056       7      48 1.27e-16
## 6 1319486308046176256       7      21 8.96e-17
## 7 1319893049858805760       7      21 8.86e-17
```

```
## 8 1319473316189569024      6      295 6.47e- 2
## 9 1319495657632206848      6     1115 6.27e- 2
## 10 1318991474948775937      5       10 6.18e-17
## # ... with 616 more rows
```

The tables show that during the debate period, both networks have only a small amount of active Tweeters while most Tweeters are minimally engaged with only one tweet.

According to degree centrality measure, @realDonaldTrump network has more ‘heavy’ tweeters. The top five tweeters have 22, 18, 12, 10, and 5 @realDonaldTrump quotes compared with 14, 9, 6, 6, and 5 in @JoeBiden network.

According to betweenness centrality measure, Tweeters in @realDonaldTrump network are more expansive meaning Tweeters in the network quote @realDonaldTrump from one another’s tweet in a chain-like pattern.

Eigenvector centrality shows that a higher number of heavy’ Tweeters in @realDonaldTrump network tend to quote from each other.

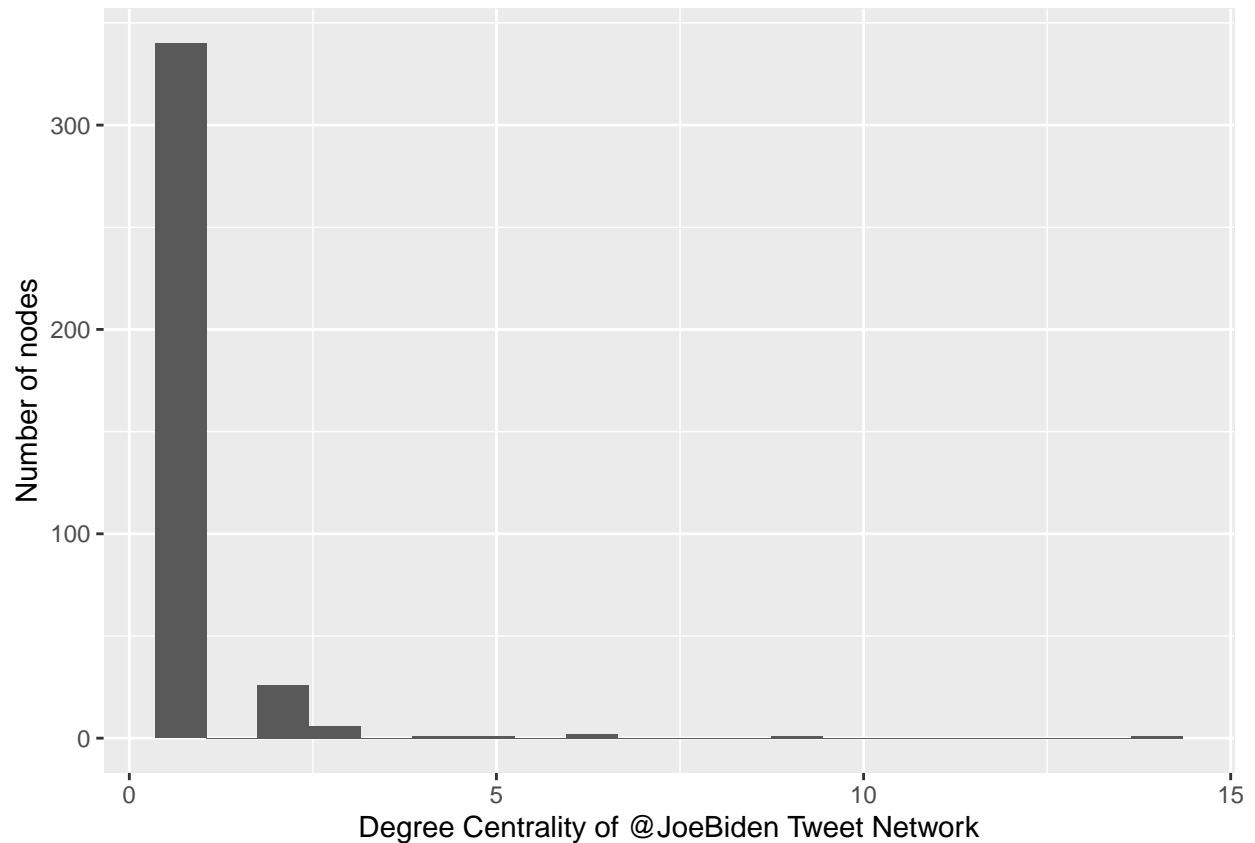
We can see each metric in a descending order, as well as its histogram distribution in the following section.

```
g1 %>% mutate(degree = centrality_degree(mode = 'all')) %>%
  #Create a column that contains the calculation of degree centrality value.
  as_tibble() %>%
  #Turn the object into a table with as_tibble() function, and arrange the display order by degree centrality
  arrange(desc(degree))
```

4.2.1. Degree Centrality of @JoeBiden Network

```
## # A tibble: 378 x 2
##   name          degree
##   <chr>         <dbl>
## 1 1319420241227714560      14
## 2 992890858339905537       9
## 3 279270981                6
## 4 1319474222306045954       6
## 5 1319419711046799360       5
## 6 1319420772469727232       4
## 7 3035730489               3
## 8 881813910327672832        3
## 9 1319466818231218179        3
## 10 1319486308046176256        3
## # ... with 368 more rows
```

```
nodes_table_joe %>%
  ggplot() +
  #Draw a chart using ggplot() function.
  geom_histogram(aes(x = degree), binwidth = 0.7) +
  #Choose to display a histogram of the values (degree) with the specified binwidth and name x and y axis
  labs(x = 'Degree Centrality of @JoeBiden Tweet Network', y = 'Number of nodes')
```



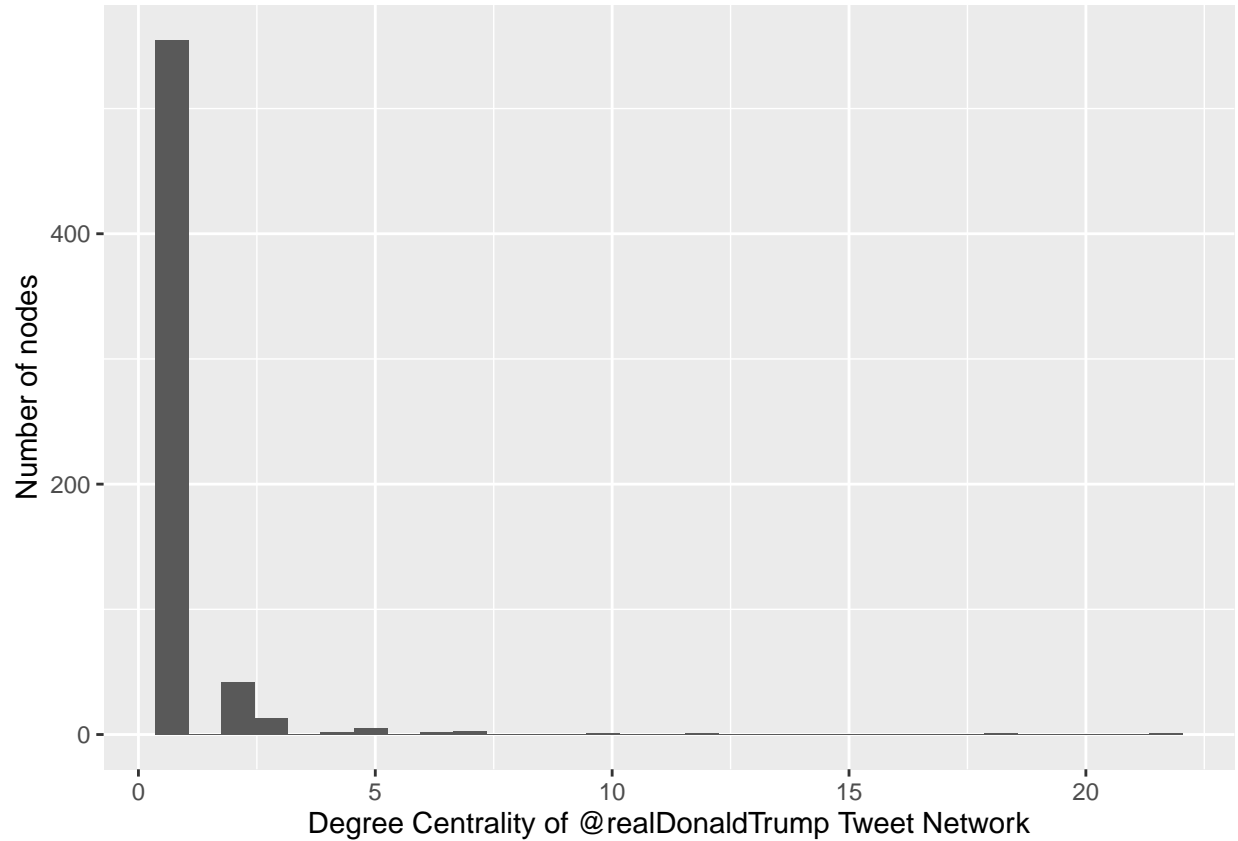
We repeat the same node metric calculation and histogram plotting processes for @realDonaldTrump network and other values.

```
g2 %>% mutate(degree = centrality_degree(mode = 'all')) %>%
  as_tibble() %>%
  arrange(desc(degree))
```

4.2.2. Degree Centrality of @realDonaldTrump Network

```
## # A tibble: 626 x 2
##   name          degree
##   <chr>         <dbl>
## 1 1319499984358805504    22
## 2 1319497720244162560    18
## 3 1319467519820922881    12
## 4 1225497537341198339    10
## 5 1298384504512557056     7
## 6 1319486308046176256     7
## 7 1319893049858805760     7
## 8 1319473316189569024     6
## 9 1319495657632206848     6
## 10 1318991474948775937     5
## # ... with 616 more rows
```

```
nodes_table_trump %>%
  ggplot() +
  geom_histogram(aes(x = degree), binwidth = 0.7) +
  labs(x = 'Degree Centrality of @realDonaldTrump Tweet Network', y = 'Number of nodes')
```



```
g1 %>%
  mutate(between = centrality_betweenness(weights = weight, directed = F)) %>%
  as_tibble() %>%
  arrange(desc(between))
```

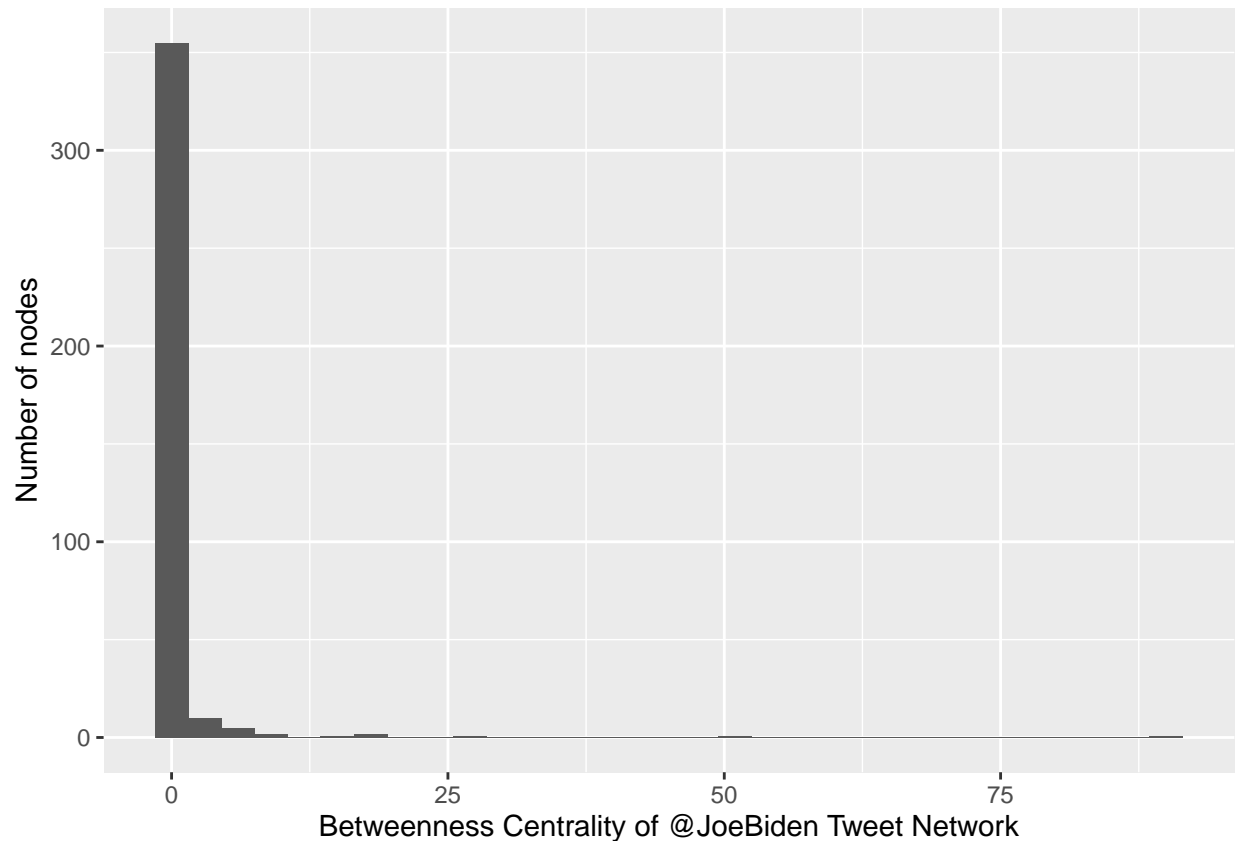
4.2.3. Betweenness Centrality of @JoeBiden Network

```
## # A tibble: 378 x 2
##   name                between
##   <chr>              <dbl>
## 1 1319420241227714560    91
## 2 992890858339905537    52
## 3 279270981             26
## 4 1319486308046176256    19
## 5 1319419711046799360    18
## 6 1319474222306045954    15
## 7 33981571              10
```



```
## 8 3035730489      8
## 9 1319461482028040192 7
## 10 1319495379805753345 7
## # ... with 368 more rows
```

```
nodes_table_joe %>%
  ggplot() +
  geom_histogram(aes(x = between), binwidth = 3) +
  labs(x = 'Betweenness Centrality of @JoeBiden Tweet Network', y = 'Number of nodes')
```



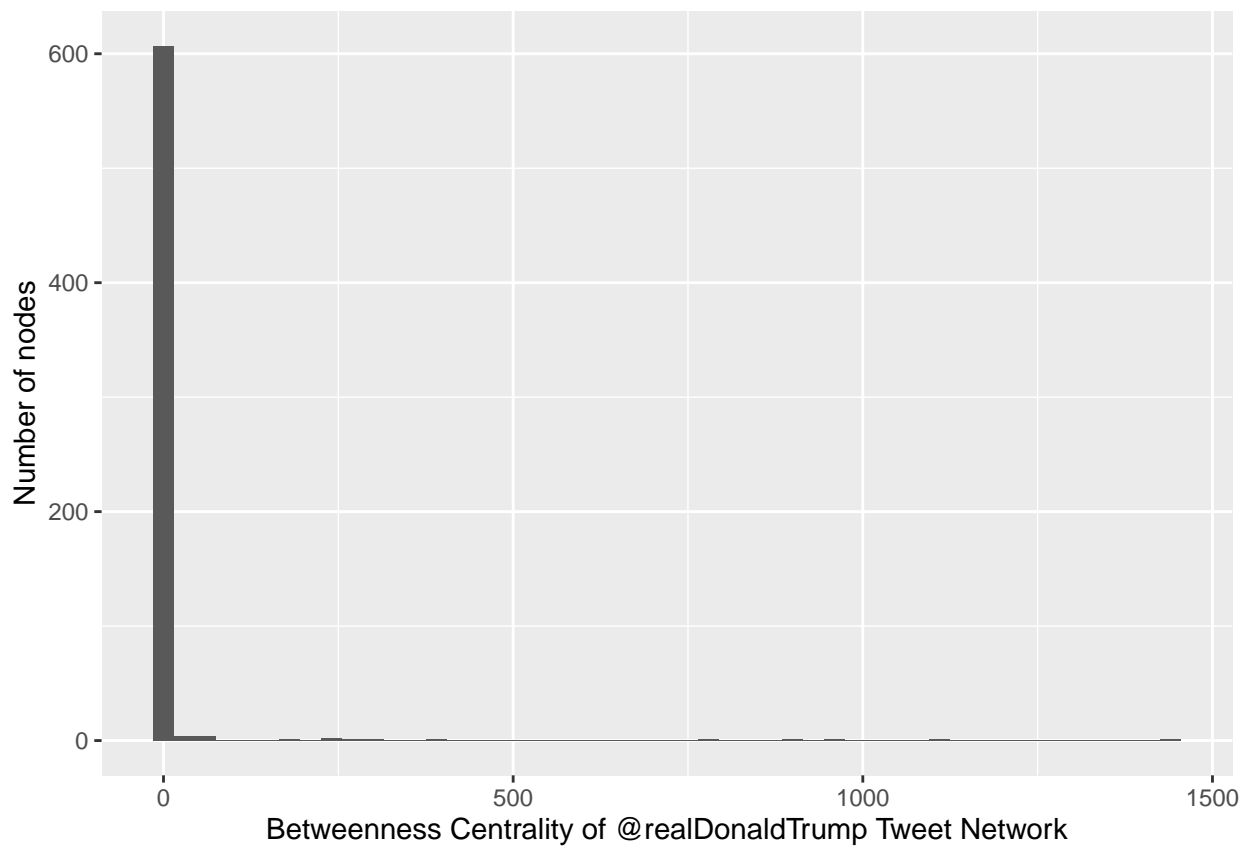
```
g2 %>%
  mutate(between = centrality_betweenness(weights = weight, directed = F)) %>%
  as_tibble() %>%
  arrange(desc(between))
```

4.2.4. Betweenness Centrality of @realDonaldTrump Network

```
## # A tibble: 626 x 2
##   name      between
##   <chr>      <dbl>
## 1 1319499984358805504 1442
## 2 1319495657632206848 1115
```

```
## 3 291751879          952
## 4 1319497720244162560 901
## 5 60709069           792
## 6 1369341750         391
## 7 1319473316189569024 295
## 8 902538000755806208 285
## 9 1319482275021623297 238
## 10 1194097105826172928 232
## # ... with 616 more rows
```

```
nodes_table_trump %>%
  ggplot() +
  geom_histogram(aes(x = between), binwidth = 30) +
  labs(x = 'Betweenness Centrality of @realDonaldTrump Tweet Network', y = 'Number of nodes')
```



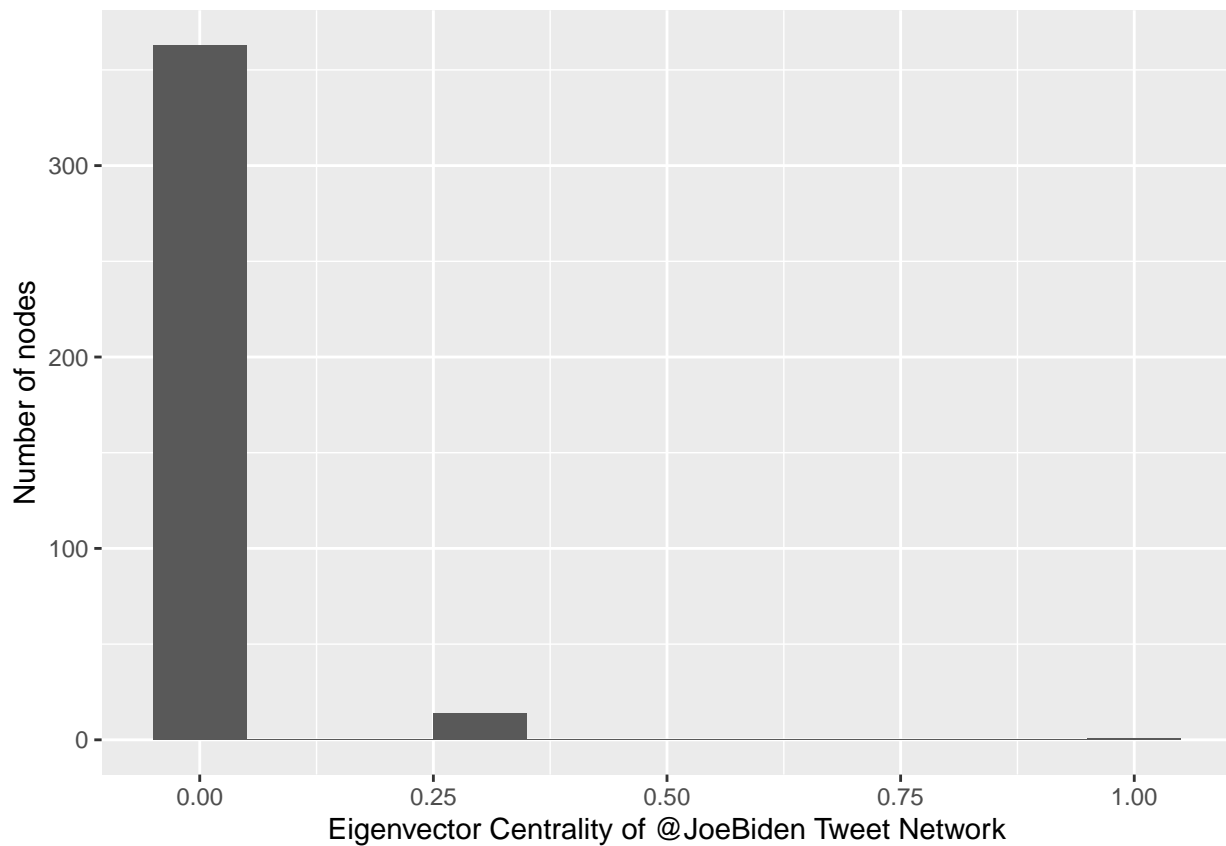
```
g1 %>%
  mutate(eigen = centrality_eigen(weights = weight, directed = F)) %>%
  as_tibble() %>%
  arrange(desc(eigen))
```

4.2.5. Eigenvector Centrality of @JoeBiden Network

```
## # A tibble: 378 x 2
```

```
##      name                eigen
##      <chr>                <dbl>
## 1 1318255281801973760 1     e+ 0
## 2 43646823              1     e+ 0
## 3 1269258105235308551 2.35e-17
## 4 1319685650245844992 2.04e-17
## 5 1299457248243089409 0
## 6 1203737149759201280 0
## 7 1190025969244889088 0
## 8 1164661144386248704 0
## 9 1290303723030433793 0
## 10 217864228           0
## # ... with 368 more rows
```

```
nodes_table_joe %>%
  ggplot() +
  geom_histogram(aes(x = eigen), binwidth = 0.1) +
  labs(x = 'Eigenvector Centrality of @JoeBiden Tweet Network', y = 'Number of nodes')
```

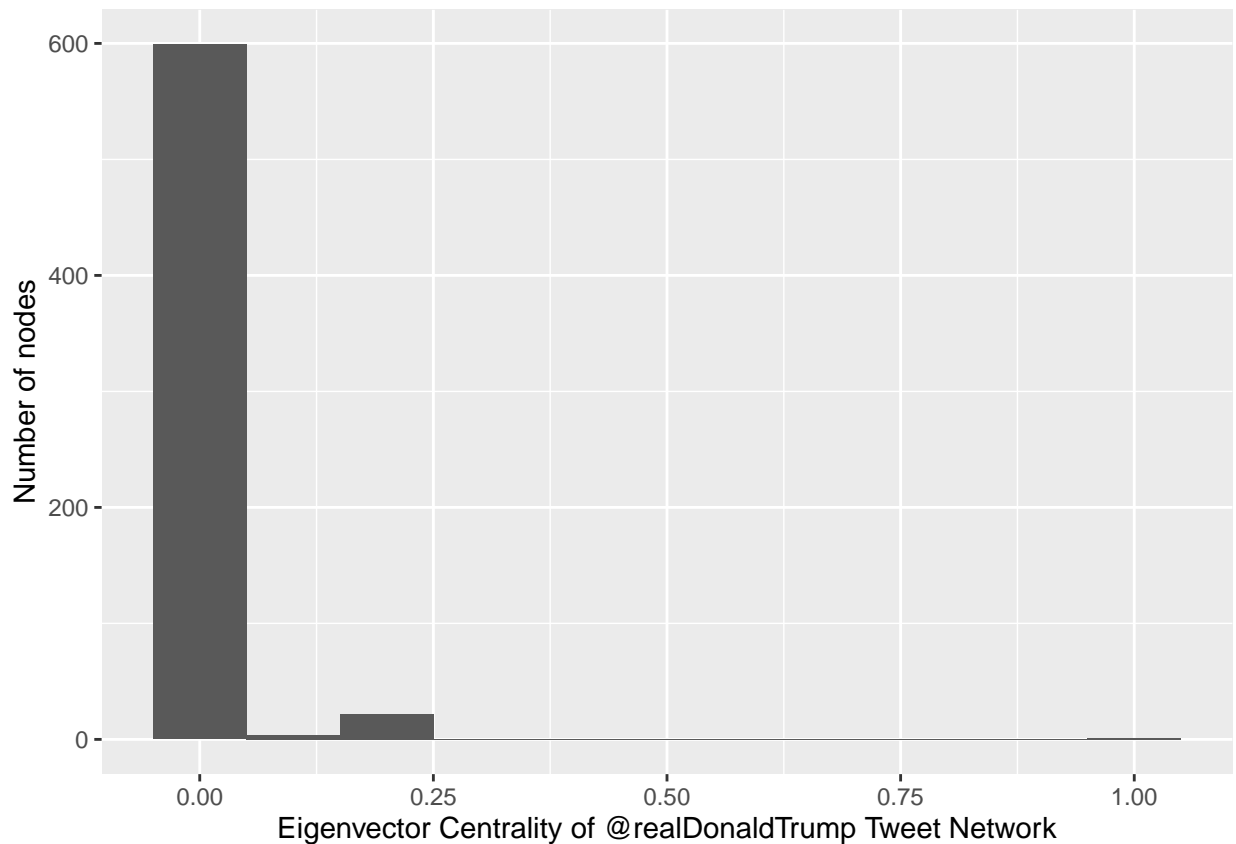


```
g2 %>%
  mutate(eigen = centrality_eigen(weights = weight, directed = F)) %>%
  as_tibble() %>%
  arrange(desc(eigen))
```

4.2.6. Eigenvector Centrality of @realDonaldTrump Network

```
## # A tibble: 626 x 2
##   name                eigen
##   <chr>              <dbl>
## 1 1319486308046176256 1     e+ 0
## 2 1216876347198324736 6.05e- 1
## 3 27129936            5.81e- 1
## 4 1275891762855317507 5.33e- 1
## 5 24750907            7.26e- 2
## 6 305366693          4.84e- 2
## 7 1129222592915791872 4.84e- 2
## 8 2310427903          4.84e- 2
## 9 475549688          4.07e-17
## 10 1302981841100722180 1.25e-17
## # ... with 616 more rows
```

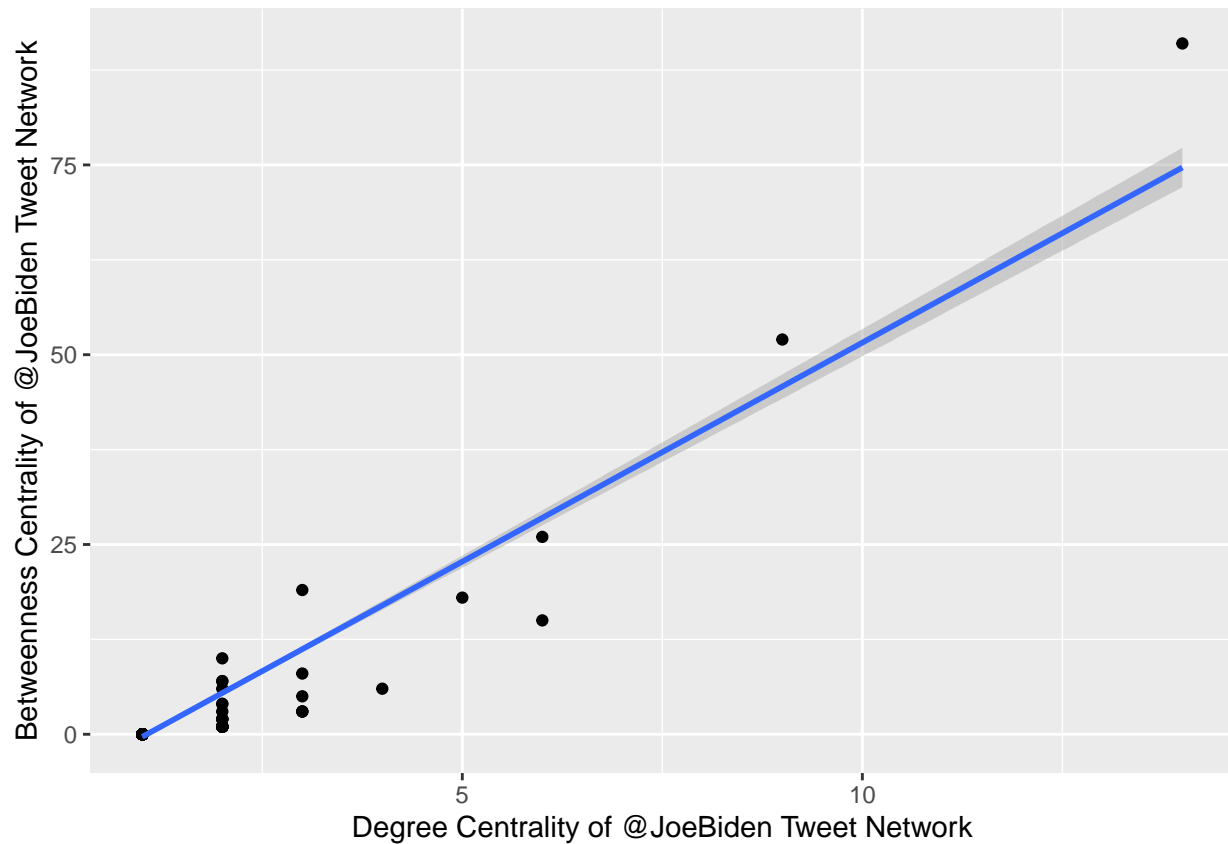
```
nodes_table_trump %>%
  ggplot() +
  geom_histogram(aes(x = eigen), binwidth = 0.1) +
  labs(x = 'Eigenvector Centrality of @realDonaldTrump Tweet Network', y = 'Number of nodes')
```



4.2.7. Correlation between Degree and Betweenness Centrality We can examine if Tweeters who quote @JoeBiden and @realDonaldTrump often also often quote the same message in a chain manner by

plotting the degree centrality metric against the betweenness centrality metric for all the nodes in the graph and calculating the correlation value.

```
nodes_table_joe %>%
  ggplot() +
    #Draw a chart using ggplot() function.
    geom_point(aes(x = degree, y = between)) +
    #Choose to display the values in scatterplots with geom_point() function, and assign degree centrality
    geom_smooth(aes(x = degree, y = between), method='lm') +
    #Specify method='lm' to fit linear models to the data frame, and name x and y axis with labs(x='...', y = ...)
    labs(x = 'Degree Centrality of @JoeBiden Tweet Network', y = 'Betweenness Centrality of @JoeBiden Tweet Network')
```

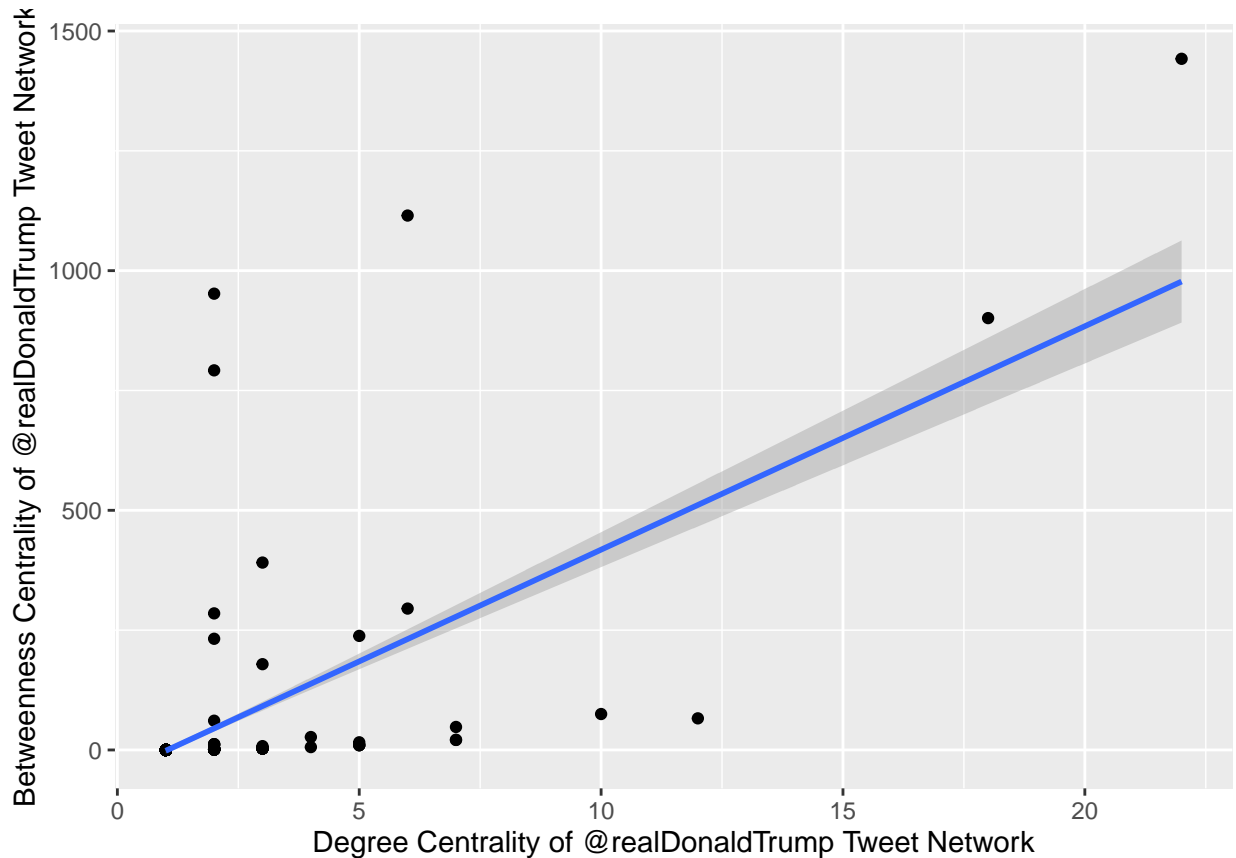


```
#Calculate correlation of the two values in the chart using cor() function below.
cor(nodes_table_joe$degree, nodes_table_joe$between)
```

```
## [1] 0.9450611
```

The graph above shows a strong correlation between degree and betweenness centrality of @JoeBiden network (correlation value of 0.94) meaning the nodes with a low degree also have a low betweenness centrality values and vice-versa. In this case, since the plots are clustered on the lower left of the graph, it means a majority of Tweeters tweet a message with the mention @JoeBiden only a few times, and they rarely quote the same tweet.

```
nodes_table_trump %>%
  ggplot() +
  geom_point(aes(x = degree, y = between)) +
  geom_smooth(aes(x = degree, y = between), method='lm') +
  labs(x = 'Degree Centrality of @realDonaldTrump Tweet Network', y = 'Betweenness Centrality of @realDonaldTrump Tweet Network')
```



```
cor(nodes_table_trump$degree, nodes_table_trump$between)
```

```
## [1] 0.6639921
```

The nodes in @realDonaldTrump network are more scattered resulting in a less correlation value (0.66) between degree and betweenness centrality. There are many nodes with low degree centrality but high betweenness centrality value. Those are the Tweeters who tweet not so often with the mention @realDonaldTrump but tend to quote the same tweet from another Tweeter, and this continues in a chain-like pattern.

4.3. Network Diagrams

With the graph objects 'g1' and 'g2' created in the previous sections, we can now draw network diagrams for @JoeBiden and @realDonaldTrump. After drawing a network, tweet texts and usernames in the top communities will be analyzed. Since the information in the dataframes is IDs, they need to be joined back to the original tweet files to get tweet texts and usernames.

4.3.1. Look up Tables First, we make two types of ‘lookup tables’; one to get a username for each ID and another to get tweet texts. The information is saved in new dataframes called `all_users` for usernames and `table_of_tweets` for tweet texts.

```
#From the original files, select user IDs, screen names, and user description and put it in a new data .

#Use distinct() function to take only one ID because people can change their usernames.

#Repeat the previous steps for @realDonaldTrump.

all_users_joe = joe Biden %>%
  select(user_id, user_screen_name, user_description) %>%
  distinct(user_id, .keep_all = TRUE)

all_users_trump = donal trump %>%
  select(user_id, user_screen_name, user_description) %>%
  distinct(user_id, .keep_all = TRUE)

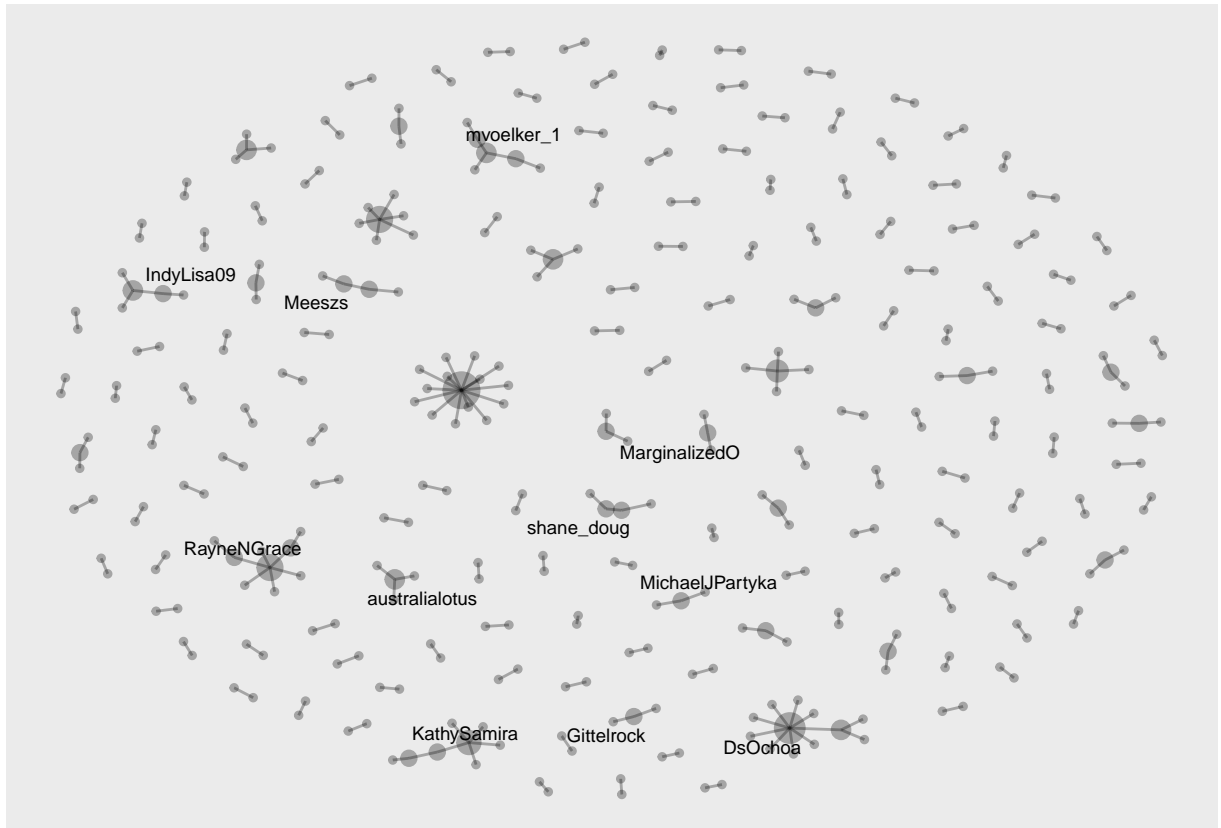
#For tweet texts, select user IDs and texts and save them to a new data frame called table_of_tweets_jo

table_of_tweets_joe = joe Biden %>%
  select(user_id, text)

table_of_tweets_trump = donal trump %>%
  select(user_id, text)
```

4.3.2. The Network of @JoeBiden We draw @JoeBiden network diagram based on degree centrality using the Fruchterman-Reingold layout.

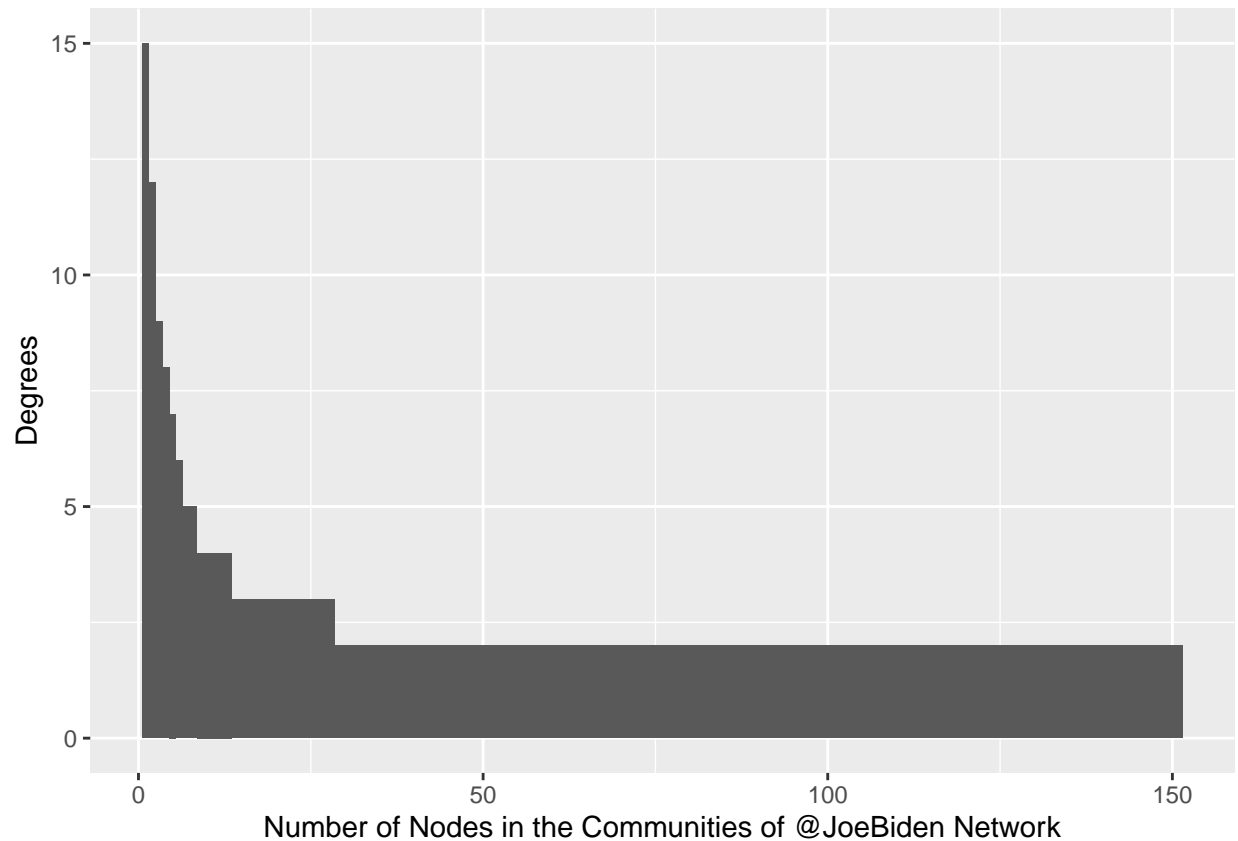
```
g1 %>%
  mutate(degree = centrality_degree(mode = 'all')) %>%
  #Create a column of degree centrality using mutate().
  left_join(all_users_joe, by = c('name' = 'user_id')) %>%
  #Use left_join() function to join the IDs with the lookup table to get a username for each ID.
  gggraph(layout = 'fr') +
  #Specify layout='fr' with gggraph() function to draw the network with the Fruchterman-Reingold layout
  geom_edge_link(size=1, alpha=0.3) +
  #Specify the size of the edge and transparency with size=... and alpha=... respectively.
  geom_node_point(aes(size = degree), alpha=0.3) +
  #Assign degree centrality value to the size of the nodes with size=degree.
  geom_node_text(aes(label = ifelse(degree >1, user_screen_name, NA)), size = 2.5, repel = TRUE) +
  #Use ifelse() function to display the username of the nodes with more than 1 degree. repel=TRUE helps
  theme(legend.position = 'none')
```



According to the network diagram of @JoeBiden, we can see a few complex communities appearing among many other smaller ones. There are 11 nodes with more than one degree (the ones with the usernames.) The rest are either the quote_ids in complex communities or the nodes with only 1 degree.

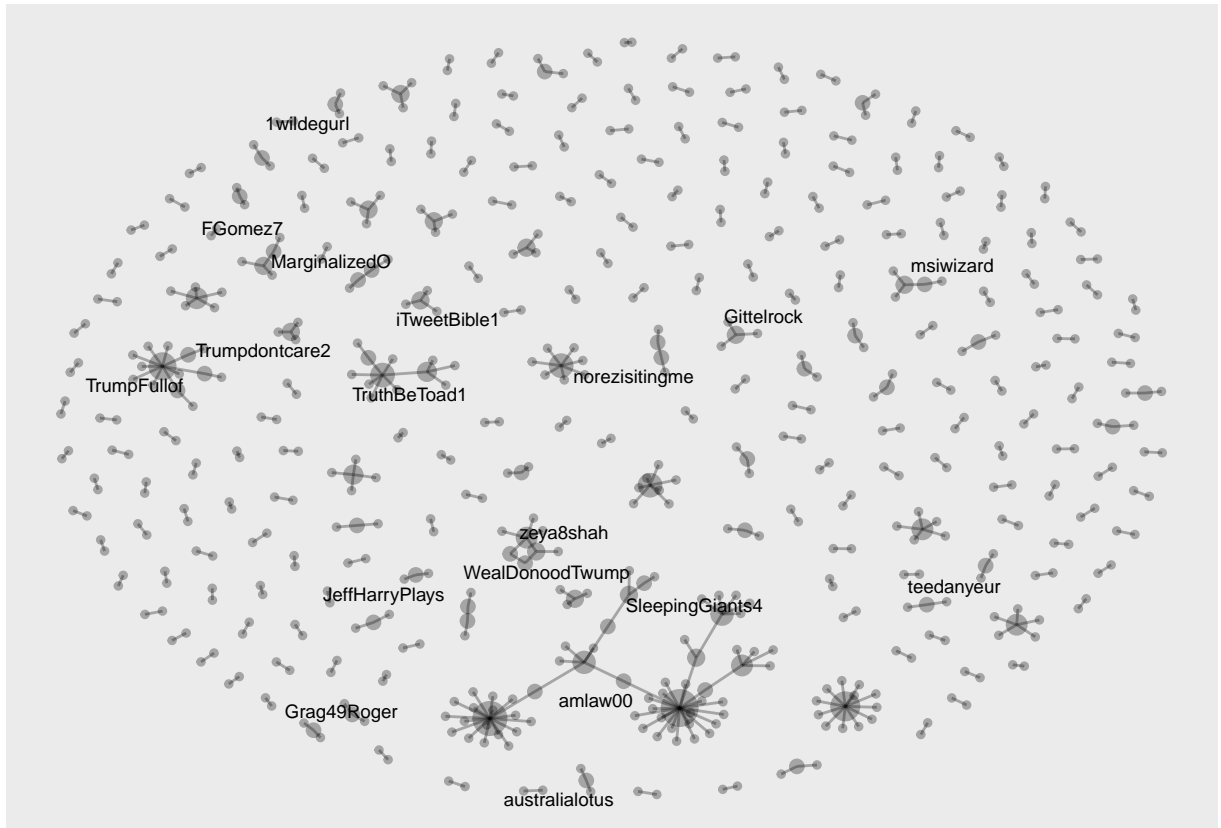
A number of nodes in all communities can be demonstrated in a bar chart format.

```
g1 %>% to_undirected() %>%
  mutate(group = group_louvain()) %>%
  #Create a column that contains communities or groups using group_louvain community detection algorithm
  as_tibble() %>%
  #Turn the object into a table.
  group_by(group) %>%
  #Count a number nodes in each group using group_by().
  tally() %>%
  #Sum each count up with tally().
  ggplot() +
  #Visualize the table with ggplot().
  geom_col(aes(x = group, y = n)) +
  #Draw a bar chart with geom_col() and name x and y axis with lab(x='...', y='...') below.
  labs(x = 'Number of Nodes in the Communities of @JoeBiden Network', y = 'Degrees')
```

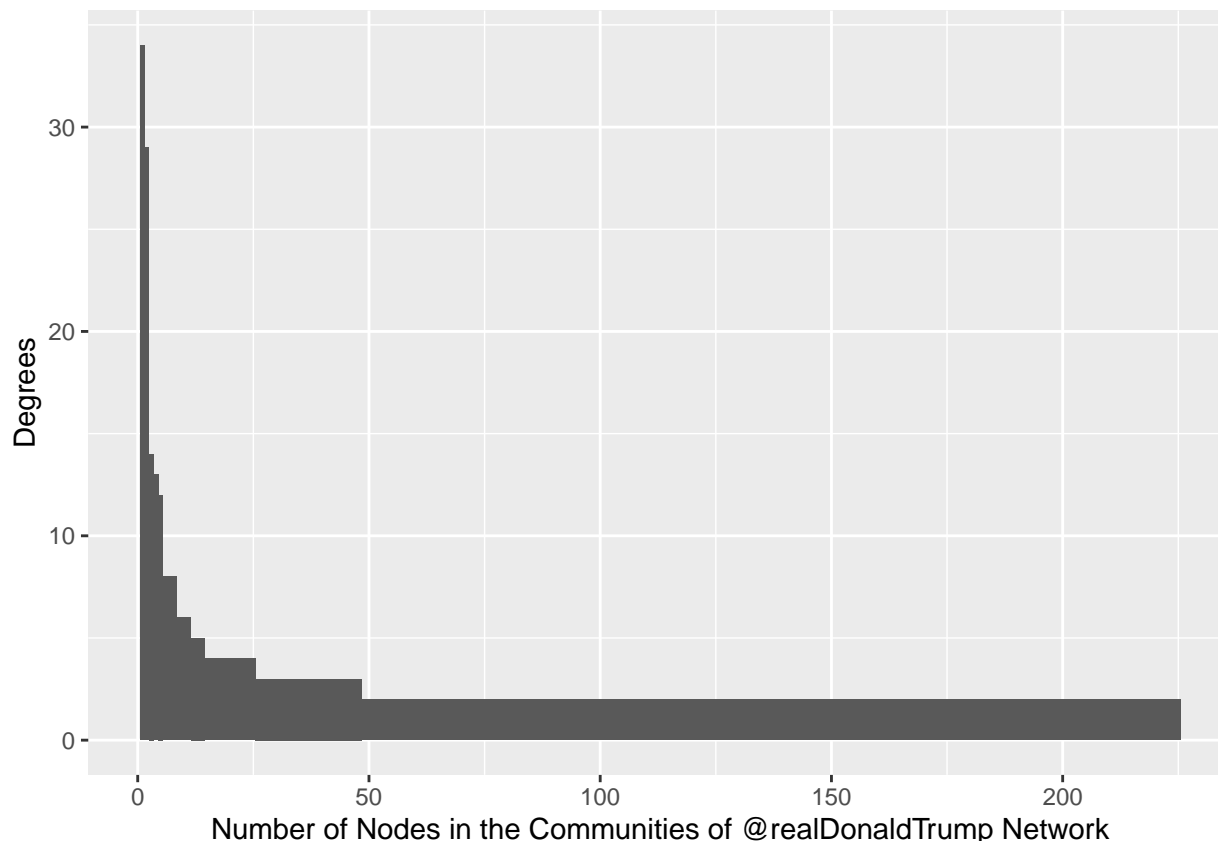



4.3.3. The Network of @realDonaldTrump We repeat the same network diagram visualizing and node-in-community bar chart drawing for @realDonaldTrump.

```
g2 %>%
  mutate(degree = centrality_degree(mode = 'all')) %>%
  left_join(all_users_joe, by = c('name' = 'user_id')) %>%
  ggraph(layout = 'fr') +
  geom_edge_link(size=1, alpha=0.3) +
  geom_node_point(aes(size = degree), alpha=0.3) +
  geom_node_text(aes(label = ifelse(degree >1, user_screen_name, NA)), size = 2.5, repel = TRUE) +
  theme(legend.position = 'none')
```



```
g2 %>% to_undirected() %>%
  mutate(group = group_louvain()) %>%
  as_tibble() %>%
  group_by(group) %>%
  tally() %>%
  ggplot() +
  geom_col(aes(x = group, y = n)) +
  labs(x = 'Number of Nodes in the Communities of @realDonaldTrump Network', y = 'Degrees')
```



Comparing both network diagrams, we can see clearer that @realDonaldTrump network has more active tweeters who tweet more often and also form a more extensive connection through quoting the text from one another. The biggest community in @realDonaldTrump network has the node with over 30 degrees, while it is only 15 for @JoeBiden.

The top biggest communities of @realDonaldTrump consist of Tweeters like amlaw00, Grag49Roger, and SleepingGiants4. Some smaller communities in @realDonaldTrump network are, for example, the ones of zeya8shah, TruthBeToad1, and msiwizard.

The main communities for @JoeBiden have usernames like DsoChoa, RayneNGrace, and KathySamira, and smaller ones are, for example, Gittelrock, MarginalizedO, and shane doug.

4.4. Community Analysis

We apply a community detection algorithm to reveal the communities within each network. The algorithm also ranks the communities based on their size. We choose to display top 10 communities for @JoeBiden and top 15 for @realDonaldTrump because @realDonaldTrump network is bigger. Before coming to a conclusion with these two numbers, we tried to adjust the numbers up and down and found that the communities beyond this top rank has only 2 or 1 degree.

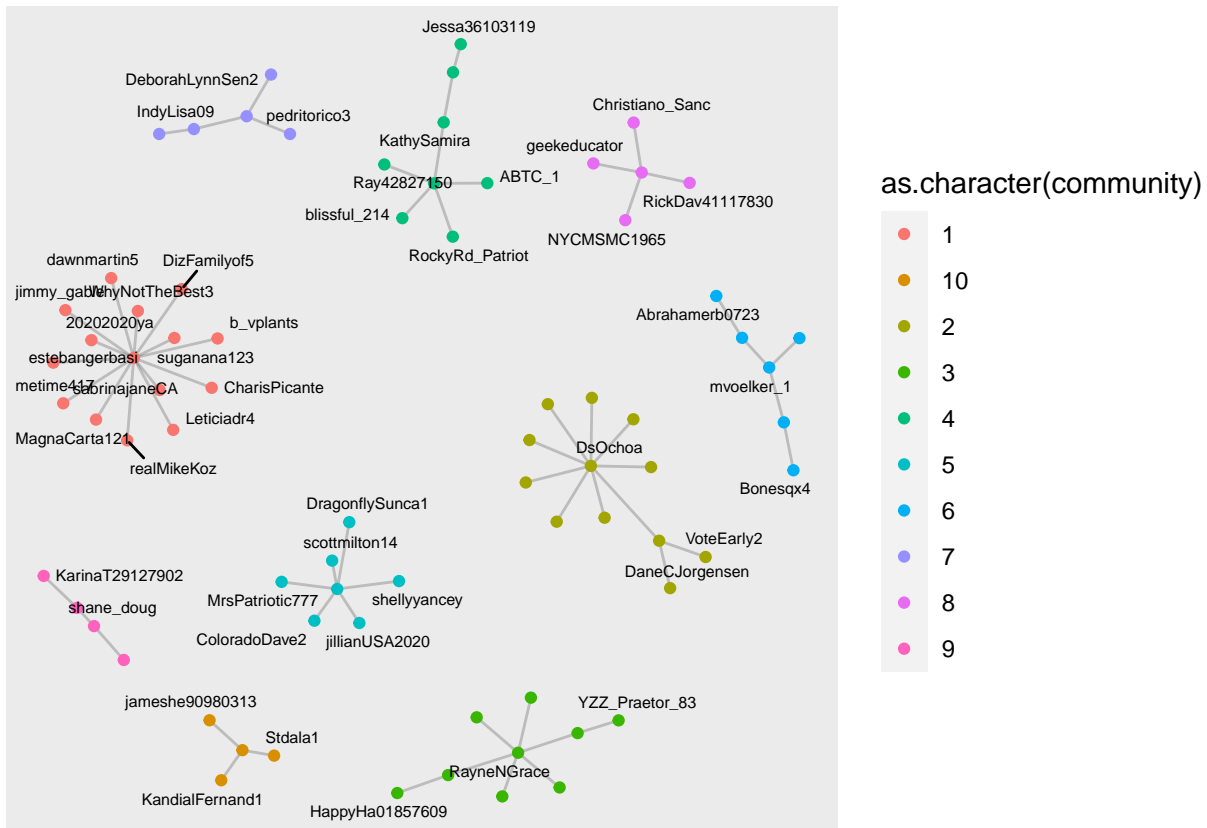
4.4.1. @JoeBiden The top 10 communities in @JoeBiden network are demonstrated below.

```
g1 %>%
  to_undirected() %>%
  #Change the graph to undirected in order to make the community detection work.
  mutate(community = group_louvain(weights =NULL)) %>%
```

```

#Calculate the group memberships.
filter(community %in% 1:10) %>%
#Filter to only include the first 10 biggest groups.
left_join(all_users_joe, by = c('name' = 'user_id')) %>%
ggraph('fr') +
geom_edge_link(alpha = 0.2) +
#Reduce the transparency of the edges with alpha=0.2.
geom_node_point(aes(color = as.character(community))) +
#Map the color aesthetic to the group label and use as.character to change it to a discrete value.
geom_node_text(size = 2, aes(label = user_screen_name), repel = TRUE)

```

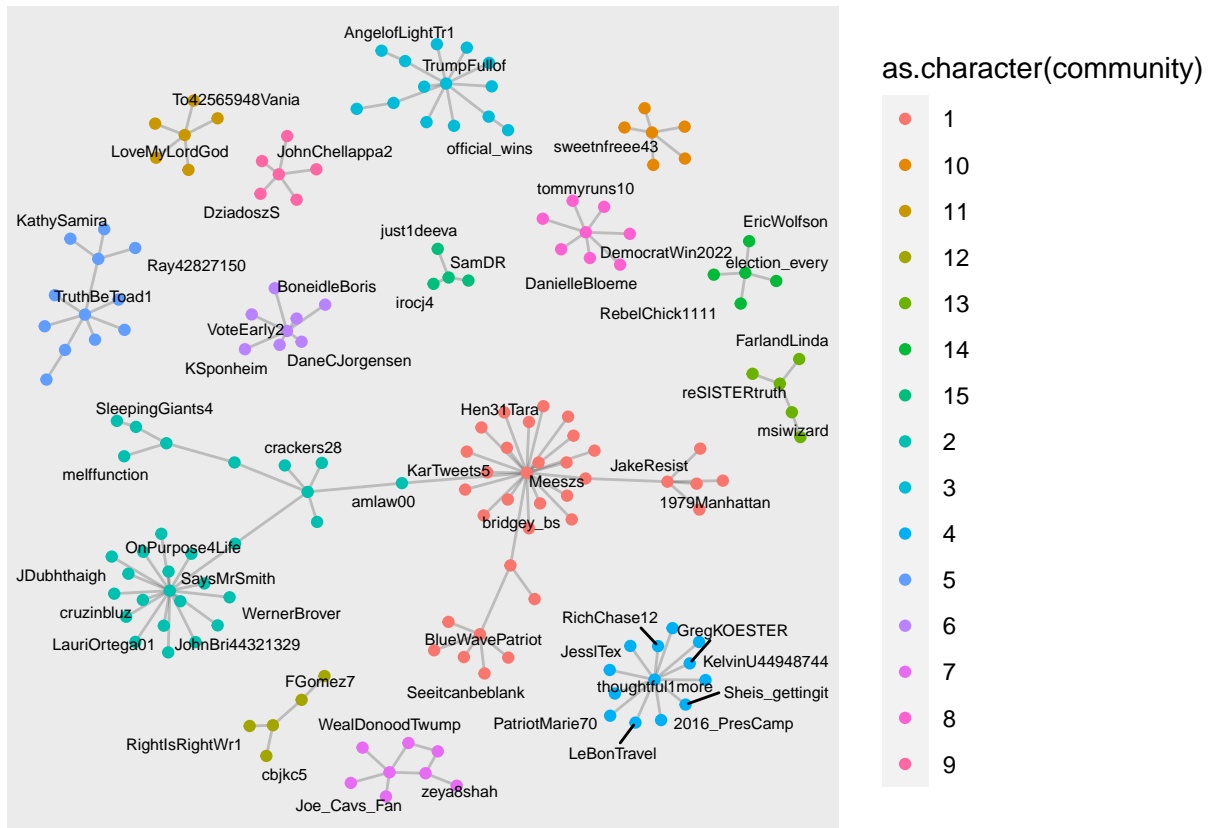


4.4.2. @realDonaldTrump Since @realDonaldTrump network has a larger number of complex communities. We increase the filter amount to 15 communities.

```

g2 %>%
to_undirected() %>%
mutate(community = group_louvain(weights = NULL)) %>%
filter(community %in% 1:15) %>%
left_join(all_users_joe, by = c('name' = 'user_id')) %>%
ggraph('fr') +
geom_edge_link(alpha = 0.2) +
geom_node_point(aes(color = as.character(community))) + geom_node_text(size = 2, aes(label = user_sc:

```



The main communities displayed in the diagrams above are showing usernames of tweeters who mention @JoeBiden and @realDonaldTrump in their tweets with their tweet texts (the nodes without a name), as well as those users who quote the same tweet.

5. Analysis

5.1. Tweet Texts and Usernames

```
# First create a lookup table with group (or community) number to be able to join it with tweet and use
```

```

as_tibble() %>%
  #Turn the result into a table, arrange by group, and select only first five groups.
  arrange(group) %>%
  filter(group %in% 1:5)

nodes_table_trump_1 = g2 %>%
  tidygraph::to_undirected() %>%
  mutate(group = group_louvain()) %>%
  as_tibble() %>%
  arrange(group) %>%
  filter(group %in% 1:5)

```

#Join the lookup table with the tweet and username table and save the result to a dataframe called text_joe

```

text_joe = nodes_table_joe_1 %>%
  left_join(table_of_tweets_joe, by = c('name' = 'user_id')) %>%
  left_join(all_users_joe, by = c('name' = 'user_id')) %>%
  as_tibble() %>%
  arrange(group)

text_joe

```

```

## # A tibble: 195 x 5
##   name                group text                user_screen_name user_description
##   <chr>              <int> <chr>              <chr>             <chr>
## 1 103616288          1 "@JoeBiden ? h~ suganana123      <NA>
## 2 103616288          1 "@whatsnewpdq @A~ suganana123      <NA>
## 3 103616288          1 "@JoeBiden ht~ suganana123      <NA>
## 4 103616288          1 "@JoeBiden What~ suganana123      <NA>
## 5 117544813          1 "@JoeBiden https~ sabrinajaneCA    ONLY original t~
## 6 117544813          1 "@JoeBiden https~ sabrinajaneCA    ONLY original t~
## 7 117544813          1 "@MisikoMichael ~ sabrinajaneCA    ONLY original t~
## 8 117544813          1 "@JoeBiden https~ sabrinajaneCA    ONLY original t~
## 9 117544813          1 "Bobulinski is @~ sabrinajaneCA    ONLY original t~
## 10 1281218003057680384 1 "@JoeBiden #Bobu~ WhyNotTheBest3  "Optimism and s~
## # ... with 185 more rows

```

```

text_trump = nodes_table_trump_1 %>%
  left_join(table_of_tweets_trump, by = c('name' = 'user_id')) %>%
  left_join(all_users_trump, by = c('name' = 'user_id')) %>%
  as_tibble() %>%
  arrange(group)

text_trump

```

```

## # A tibble: 496 x 5
##   name                group text                user_screen_name user_description
##   <chr>              <int> <chr>              <chr>             <chr>
## 1 150212862          1 "@EricTrump @realDonaldTrump~ Meeszs      <NA>
## 2 150212862          1 "@megynkelly @realDonaldTrump~ Meeszs      <NA>
## 3 150212862          1 "@realDonaldTrump Trump ha~ Meeszs      <NA>
## 4 150212862          1 "@realDonaldTrump Trump ha~ Meeszs      <NA>

```

```
## 5 150212862      1 "@realDonaldTrump Trump ha~ Meeszs      <NA>
## 6 150212862      1 "@realDonaldTrump Trump ha~ Meeszs      <NA>
## 7 150212862      1 "@seanmdav @realDonaldTrump~ Meeszs      <NA>
## 8 150212862      1 "@EricTrump @realDonaldTrump~ Meeszs      <NA>
## 9 150212862      1 "@realDonaldTrump Trump ha~ Meeszs      <NA>
## 10 150212862     1 "@endlessninth @HeyTammyBr~ Meeszs      <NA>
## # ... with 486 more rows
```

The first five biggest communities in @JoeBiden network contain 195 rows of tweet texts and usernames and 496 rows for @realDonaldTrump network.

Most tweets for both networks contain only mentions (@...) and links while only some tweets show opinions or messages. Therefore, we divide tweet text and username analysis into three main parts: mentions, links, and messages.

5.1.1. Tweet Texts and Usernames of Top 5 Communities in @JoeBiden Network

- Mentions

Politicians: @AnnCoulter, @KamalaHarris, @RaheemKassam, @fred_guttenberg, @kevinomccarthy
Institutions: @NBCNews, @CBSNews, @KellyannePolls
Famous people: @ArthurSchwartz (American composer and film producer)

- Links: Most tweets contain links to the video clips related to the debate.
- Messages
 - Criticisms
 - “And again @JoeBiden stating NO FRACKING. Wow what a liar.”
 - “Hey Texas and Pennsylvania, @JoeBiden just admitted he would transition from the oil industry, effectively killing an estimated 11 million jobs.”
 - “I don’t know why @JoeBiden thinks he can continue to lie about this. He wants to ban fracking and end all fossil fuels like coal too.”
 - “@JoeBiden yes you said it right here you were the one who locked my people up”
 - Compliments
 - “Well said, @joe Biden.” People deserve to have affordable healthcare.”
 - “Love our @JoeBiden for our POTUS 2020”
 - Election Result Updates
 - “#Debates2020 was won by @JoeBiden”
 - “Congratulations @JoeBiden on your #Debates2020 win”

“So @JoeBiden needed four days to prepare for a debate and today he drove a few blocks from his basement to read a teleprompter to cover up last night’s debate mistakes by pandering on Covid. America wants to connect with their candidate. #doyouknowmrBubanski?”

- Usernames
 - Places related to the US: sabrinajaneCA, ColoradoDave2, jillianUSA2020
 - Self description: DizFamilyof5, MrsPatriotic777
 - Politics related: MagnaCarta121, VoteEarly2

5.1.2. Tweet Texts and Usernames of Top 5 Communities in @realDonaldTrump Network Like @JoeBiden network, most tweets in @realDonaldTrump network contain mentions and links related to the debate, but @realDonaldTrump network have more tweets with textual content and opinions. Many textual tweets strongly criticize his policies, and some other tweets are sarcastic. There are also many tweets that give an update on the election result. Most of them are about the loss of Donald Trump. The mentions in @realDonaldTrump network show more variety in terms of categories and in a higher amount. For example, many celebrities and media-related people are mentioned.

There are three main categories of usernames of both networks: places related to the US like CA, Colorado, Cali, and USA, a description of oneself like DizFamilyof5, MrsPatriotic777, and xxnavygirl, and politics related like MagnaCarta121, VoteEarly2, and amlaw00. Most of the usernames in both networks are either individual's names, a series of letters, or codes, but those in @realDonaldTrump network are less meaningful and interpretable.

- Mentions

Politicians: @Mike_Pence, @EliseStefanik, @RudyGiuliani, @BernieSanders Media related: @AndrewCMcCarthy (TV director), @megynkelly (journalist), @seanhannity (politician commentator) Institutions: @CBSNews, @KellyannePolls Famous people: @MovezzzCGR (Christian Rodriguez, footballer), @seanmdav (Sean Davis, footballer)

- Links: Most tweets contain links to the video clips related to the debate.

- Messages

- Criticisms

“Im a immigrant and I came back. I may have a low IQ, but we work hard, have empathy, we build families and communities with our contributions! Wind mills cause cancer? You said!...”

“It doesn’t matter who built the fucking cages. YOU separated those children from their families. YOU!”

“Lots and lots of sane human beings disagree with you. Lots.”

“@realDonaldTrump is a racist”

“@realDonaldTrump stating that the kids in cages are being well cared for after being separated from their parents is #bullshit. He’s never toured the facility or taken even the slightest interest in how they”

- Sarcasms

“Well, there was one little section in the debate where Trump was popular”

“MUTE BUTTON”

“Wow. All you got is an unscientific online twitter poll? Sad...”

“Seems it wasn’t you.”

“Crazy old man doesn’t get any credit for trying to act normal once in his life”

“TOP TEN TRUMP LIES IN THE DEBATE. HE LIED SO MUCH THERE IS A TOP TEN”

- Election Result Updates

“Trump has lost te debate!!, Every reputable poll shows that Trump lost the debate. Someone’s crazy uncle needs to slink quietly away.”

“You’re so bad at this. You lost the debate”

- Usernames

- Places related to the US: CaliMtBear3

- Self description: 1979Manhattan, xxnavygirl

- Politics related: amlaw00, official_wins, reasonedvoices

5.2. Unique Words in Each Community

To extract the unique words in a community and not found so much in others, we employ the tidytext library to calculate the tf-idf scores and display it in a tabular form.

TF stands for Term Frequency. It calculates the unique words in each community. A high TF score means a word is unique in a community. IDF stands for Inverse Document Frequency. It calculates the common words found in all community as a whole. A low IDF indicates that a word exists more generally and is, therefore, less unique to a certain community. Both statistical numbers are inverse variations. Both figures will multiply to get tf_idf scores. High tf_idf scores indicate that a word is unique.

```
data(stop_words)

tweets_tf_idf_joe = nodes_table_joe_1 %>%
  left_join(table_of_tweets_joe, by = c('name' = 'user_id')) %>%
  left_join(all_users_joe, by = c('name' = 'user_id')) %>%
  unnest_tokens(word, text) %>%
  count(group, word) %>%
  bind_tf_idf(word, group, n)

words_joe = tweets_tf_idf_joe %>% anti_join(stop_words) %>%
  group_by(group) %>%
  slice_max(order_by = tf_idf, n = 20) %>%
  as_tibble()

words_joe
```

5.2.1. Unique Words in the Top Communities of @JoeBiden Network

```
## # A tibble: 146 x 6
##   group word                n      tf    idf tf_idf
##   <int> <chr>                <int>  <dbl> <dbl> <dbl>
## 1     1 1 9yv0h7mood             4 0.00622 1.61 0.0100
## 2     1 1 bobulinskididntkillhimself 4 0.00622 1.61 0.0100
## 3     1 1 burisma                4 0.00622 1.61 0.0100
## 4     1 1 con                  4 0.00622 1.61 0.0100
## 5     1 1 corruptjoe             4 0.00622 1.61 0.0100
## 6     1 1 creepyjoe              4 0.00622 1.61 0.0100
## 7     1 1 crookedjoe             4 0.00622 1.61 0.0100
## 8     1 1 frackingtape           4 0.00622 1.61 0.0100
## 9     1 1 hunter                4 0.00622 1.61 0.0100
## 10    1 1 hunterbiden            4 0.00622 1.61 0.0100
## # ... with 136 more rows
```

Unique words found in each top 5 community in @JoeBiden network are as follow:

- Community 1: corruptjoe, creepyjoe, sleepyjoe
- Community 2: won, bidenwon, empathetic, intelligent, patriotic
- Community 3: lied, lie, chumps, drbiden
- Community 4: news, carry, conference
- Community 5: joebidenlies, fracking, lies

```

tweets_tf_idf_trump = nodes_table_trump_1 %>%
  left_join(table_of_tweets_trump, by = c('name' = 'user_id')) %>%
  left_join(all_users_trump, by = c('name' = 'user_id')) %>%
  unnest_tokens(word, text) %>%
  count(group, word) %>%
  bind_tf_idf(word, group, n)

words_trump = tweets_tf_idf_trump %>% anti_join(stop_words) %>%
  group_by(group) %>%
  slice_max(order_by = tf_idf, n = 20) %>%
  as_tibble()

words_trump

```

5.2.2. Unique Words in the Top Communities of @realDonaldTrump Network

```

## # A tibble: 118 x 6
##   group word      n      tf      idf tf_idf
##   <int> <chr>   <int>   <dbl> <dbl>   <dbl>
## 1     1 lost      30 0.0147 0.916 0.0135
## 2     1 dataprogress 17 0.00836 1.61 0.0135
## 3     1 tonight's 17 0.00836 1.61 0.0135
## 4     1 41       16 0.00787 1.61 0.0127
## 5     1 52       16 0.00787 1.61 0.0127
## 6     1 cnn      15 0.00737 1.61 0.0119
## 7     1 yougovamerica 15 0.00737 1.61 0.0119
## 8     1 0        14 0.00688 1.61 0.0111
## 9     1 14       14 0.00688 1.61 0.0111
## 10    1 35       14 0.00688 1.61 0.0111
## # ... with 108 more rows

```

Unique words found in each top 5 community in @realDonaldTrump network are as follow:

- Community 1: lost, verified, include, viewers, beat
- Community 2: nah
- Community 3: lies, covid
- Community 4: poor, policies, ball, clock, country
- Community 5: bill, law, safe, seat

The first two communities of @realDonaldTrump network barely have any meaningful words. When looking as a whole, top communities @realDonaldTrump network tweet many negative words like poor and lies compared to a higher number of positive words in @JoeBiden network like bidenwon, intelligent, and patriotic.

Since this is the tf-idf score comparing between top communities in each network, it would be interesting to look at the data in another angle by comparing the common words in @JoeBiden with @realDonaldTrump network.

6. Conclusions and Discussions

To conclude, @realDonaldTrump network is less dense and more extensive. It also has more active tweeters who tweet often and quote each other. The content of tweets in @realDonaldTrump network is more diverse

and heated with a more variety of mentioned people, a lot of criticisms and sarcasms. Tweets in @JoeBiden network tend to be more positive. In terms of the usernames, while most usernames are not meaningful and interpretable, three categories are found: places related to the US, self description, and politics related names.

It is interesting that during the second debate, most tweets only contain mentions and links, and only a small number of tweeters express their opinion in their tweets. This passive, rather than active, participation indicate that tweeters wanted to show a participation in this political event. Most obviously chose the side but prefer not to go into details about the policies. Many express their feelings through sarcasms.

7. References

Documenting the Now. 2020. Hydrator [Computer Software]. Retrieved from <https://github.com/docnow/hydrator>.

Moreno, Jacob L. 1934. Who Shall Survive? A New Approach to the Problem of Human Interrelations, Washington D.C.: Nervous and Mental Disease Publishing Co.

R Core Team. 2020. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.

Chakrit. (2019, 28 May). tf-idf- . <https://www.softnix.co.th/2019/05/28/tf-idf-%E0%B8%97%E0%B8%B3%E0%B8%87%E0%B8%B2%E0%B8%99%E0%B8%A2%E0%B8%B1%E0%B8%87%E0%B9%84%E0%B8%87/>