# SENTIMENTAL ANALYSIS, NETWORK AND COMMUNITY DETECTION USING R PROGRAMMING

Mayukh Ghosh (18BCE0417)
School of *Computer Science and Engineering*
Vellore Institute of Engineering
Vellore, India
mayukh.ghosh2021@gmail.com

***Abstract*—This paper presents an analysis on the live tweets. It has two approaches to it. First it analyses the terms of the recent tweets and tries to find a relationship between the terms. In the second part, it analyzes the tweets as a whole. In both the cases it uses three different community detection algorithm and at the end tries to compare these three algorithms result on the basis of the sentiments of the terms and tweets respectively.**

***Keywords—Sentiment Analysis, Tweeter, Coronavirus, Network, Edge Betweenness Algorithm, Label Propagation, Greedy optimization of Modularity, Clustering, Sentiment visualization***

## I. INTRODUCTION

In this paper, we will try to analyse network of recent tweets using R on the basis of sentiments. We will collect the live tweets using the tweeter API. Then we will pre-process the tweets to make a corpus. After that we will make two networks, first network of terms, second network of tweets. On these two networks we will apply the three-community detection algorithm edge-betweenness, label propagation, greedy optimisation of modularity to detect clusters. Then we will individually analyse the clusters formed in each subpart and plot the average sentiment of the cluster. We will also find the error in the clustering using the Root Mean Square Error. At the end of the paper we will analyse the clustering and see if we can find any similarities in the cluster or difference within the clusters formed by the clustering algorithm and also try to analyse how efficiently the clustering divides the terms or the tweets based on the sentiment scores.

## II. SENTIMENT ANALYSIS

With the introduction of the internet and its access to all. The amount of text generated in a day, is impossible for humans to read and analyse. So, with the help of modern-day algorithms and advance hardware to do the massive computations, we have been able to process and analyse the contents automatically. Sentiment Analysis is a text mining technique to analyse the sentiment, emotions and views in a portion of text automatically. Sentiment analysers are of many types, some are used to extract sentiment, some use to classify them binarily, and some analysers may even summarise the opinions over a corpus of text. Some analysers are also used in every day to detect spam in our mails.

In this paper, to analyse the sentiments, we are using the dictionary-based approach. In this approach, the text is first taken as an input. The text is then processed, the stop words are removed, the root words of the words are extracted, then depending on the dictionary the n-grams are separated. Then from the bag of words that the lexicon already has, the words in the sentence are matched. And accordingly, the sentiment scores are added. Some algorithms use a cosine similarity to give a percentage value of the sentients, other algorithms may use a logistic function to give a bool type output. Whereas some others just add the sentiments per word to give a sentiment score for the whole piece of text that is supplied as input.

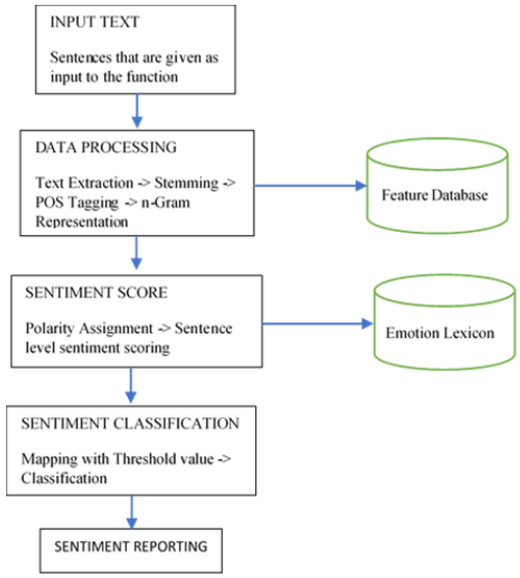


Fig.1 Sentiment Analysis Flowchart

### *A. Data Pre-processing*

- The tweets contain a lot of Latin and other special characters which are irrelevant for analysis, so to eliminate the Latin characters and other special characters, we will convert the tweets to ASCII format from the UTF-16 format and delete the Latin characters
- We will convert all the alphabet to lowercase to make sure “Mayukh” and “mayukh” are considered same in the corpus
    - We remove the punctuation from the corpus, to avoid them being considered as a word

- We remove numbers, to avoid different numbers being considered differently when they mean the same thing or same numbers connect two really different tweets
- We also remove the English stop words as they are very common in all the tweets and become a hindrance in tweets analysis as they have comparatively very large frequency compared to key words thus overshadowing the important key words
- We also remove the URLs from the tweets
- We also remove the words like 'corona', 'covid', 'positive', ‘coronavirus’ from our tweets as these were the key words with which we made the query to collect the tweets, so they will be present in all the tweets, resulting in edges between all the tweets
- We then remove the whitespaces too, to make sure ‘mayukh’ and ‘ mayukh ’ are treated the same during analysis.

### B. *NRC Emotion Dictionary*

In this paper we will be using the lexicon-based sentiment analysis and the lexicon that we will be using is the NRC Emotion Lexicon. This has 14,182 unigram words, and it was made manually by crowdsourcing on mechanical trunk. This divides the words into two sentiments Positive and Negative and eight emotions anger, anticipation, disgust, fear, joy, sadness, surprise, trust. It was made in 2010, as a emotion dictionary for general domain and is used inside the R package of ‘syuzhet’, that we have used in this paper to analyse the sentiments. The sentiment analyser that we have used accumulates the unigram scores that are present in the corpus and then gives us the output as the summation of all the scores.

### C. *Analysis on the tweets*

Now we will use the R function get_nrc_sentiment, to get the sentiment

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| anger | 0 | 0 | 0 | 0 | 0 | 1 |
| anticipation | 0 | 1 | 0 | 0 | 1 | 0 |
| disgust | 0 | 1 | 0 | 0 | 0 | 1 |
| fear | 0 | 0 | 0 | 0 | 1 | 1 |
| joy | 0 | 1 | 0 | 0 | 0 | 0 |
| sadness | 0 | 1 | 0 | 0 | 1 | 0 |
| surprise | 0 | 0 | 0 | 0 | 0 | 1 |
| trust | 0 | 1 | 0 | 1 | 1 | 0 |
| negative | 0 | 1 | 0 | 0 | 1 | 1 |
| positive | 0 | 1 | 0 | 1 | 1 | 0 |

Fig.2 Sentiment table of tweets

We can see that the function has found out the presence of the 10 emotions from each tweet, those are anger, anticipation, disgust, fear, joy, sadness, surprise, trust, negative, positive.
We plot the bar chart of the frequency of sentiments that appear over all the 1000 tweets that we are analysing

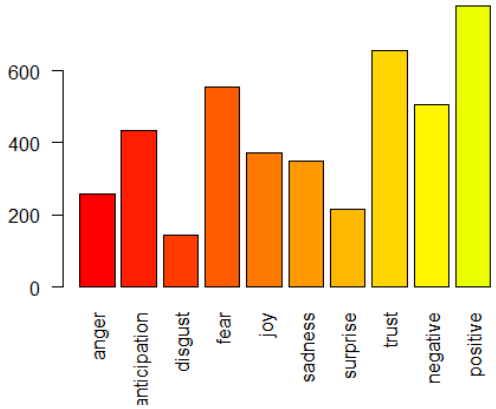


Fig.3 Accumulated sentiment score graph

We will be using the sentiment table of tweets later on in our report to analyze the clusters of tweets

## III. Cluster Detection

The definition of what is a community is not very clear. But it means a way to partition a network in such a way that the connections between the communities is minimised and the connection within the community is maximum. The connection is depicted as an edge in a network, it can depict anything in a real-life scenario such as phone calls, business deals, money transaction, research paper citation etc.

### A. *Edge-betweeness algorithm*

Edge betweenness determines the number of shortest paths that uses that edge in the network. If there are more than one shortest between a pair of nodes, then each path is assigned equal weight, such that the summation of weight of all the paths is 1.

Now the algorithm that uses this measure to find out the communities is known as Girvan-Newman Algorithm. The Girvan-Newman Algorithm is a very expensive algorithm computationally, so, it is used in small networks only.

Algorithm:

Calculate edge-betweenness for all the edges in the graph

Remove the edge with the highest brightness (in case of two edges with highest betweenness, we will remove one from them)

Recalculate the betweenness for all the edges affected by the edge removal.

Repeat until all the edges are removed.

### B. *Label Propagation*

The label propagation algorithm works in a near linear time complexity and is thus a right choice for large networks. The main idea of this algorithm is that a node would take the label which majority of its neighbor have, with the ties broken uniformly randomly.
Algorithm:

Start by initializing random labels to all the nodes
Set t = 1
Make a unique random arrangement say X
For every node, x $\in$ X,

label of $x_t$ = Max (neighbors($x_{(t-1)}$)) , select any random label if there are more than one highest random label

When there is no difference in the labels between t and (t-1), then stop the algorithm

### C. *Greedy Optimisation of Modularity*

Modularity value of a network is a significant way to analyse how strong the community is.

$$Q = \frac{1}{2m}\sum_{i,j}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta\left(c^{(i)}, c^{(j)}\right)$$

A is the adjacent matrix

m is the number of edges in the network

$k_i$ is the degree of vertex i

C(i) is the cluster to which the vertex belongs to

$\delta(x,y)$ is the Kronecker delta function

$$\delta(x,y) = 1 \; if \; x = y$$

$$0, \text{otherwise}$$

Modularity satisfies $-1 \leq Q \leq 1$

When Q > 0.3, it means that the community structure is significant

Now to optimise the algorithm and to do it for a large network in a resourceful way, a greedy optimisation has been made over this algorithm known as Louvain method.

Algorithm:

Start with every node in its own community

Phase 1: Modularity Optimisation

Order the nodes and do the following for each i

Move i to the community of neighbour j that leads to maximum ΔQ

If all ΔQ < 0 then i remains in its current community.

Repeatedly cycle through all nodes until ΔQ = 0

Phase 2: Community Aggregation

Create a weighted network of community from Phase 1.

Nodes = Communities of phase 1

Edge weights = sum of weights of edges between communities

Edge between a community become 2 self-loops

Repeat:

Apply Phase 1/Phase 2 to resulting network, and so on until ΔQ = 0

## IV. Network of Terms

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

### A. *Data Pre-processing and Visualising the corpus*

Now to make a network of terms, we need to know which terms are having common documents, for that we need to make a term matrix.

We will use the already pre-processed corpus from section II -A and make a term matrix from it

| | Docs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Terms | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| dms | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| make | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| slide | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| sure | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| vaccinated | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| youre | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| bcoz | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| can | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| cancelboardexams | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| cancelled | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Fig.4 Full term document matrix

Now the resultant matrix is quite big and some words will be irrelevant for our analysis due to their very less frequency, so we will choose only those words, who have an overall frequency of more than 30 in the whole corpus
Now the size of the matrix comes down from 37.3 MB, to a 581.1 kb, which will be absolutely okay for analysis with the limited computational power that we have.

| | Docs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Terms | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| can | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| biden | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| china | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| media | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| must | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 |
| origin | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| pandemic | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| president | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| state | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| wuhan | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

Fig.5 Filtered term document matrix with term frequency more than 30

We will use the word cloud to get a clearer picture


Fig.6 Word-cloud of the terms of the corpus

To get a clearer view of the corpus, we would plot the word frequency histogram, as human perception of length is more compared to size, so we will get an even more cleared picture of how much these key words are being discussed

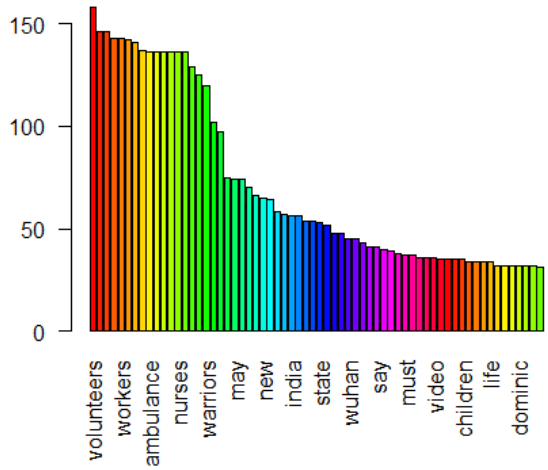


Fig.7 Term frequency histogram of the terms of the corpus

Now we have got a view of the text corpus we have as well as a sneak peek of the data in them, we will create the term edge adjacency matrix, for that we will have to multiply term matrix with transpose of it i.e.,

Let, 'tdm' be term document matrix, and

'termM' be term edge adjacency matrix, then

termM = tdm X t(tdm), where t(x) means transpose of x

| Terms | can | biden | china | media | must | origin | pandemic | president | state | wuhan |
|---|---|---|---|---|---|---|---|---|---|---|
| can | 42 | 0 | 0 | 0 | 3 | 0 | 5 | 0 | 0 | 0 |
| biden | 0 | 46 | 17 | 17 | 17 | 36 | 17 | 20 | 36 | 35 |
| china | 0 | 17 | 33 | 17 | 18 | 17 | 18 | 17 | 17 | 19 |
| media | 0 | 17 | 17 | 36 | 18 | 18 | 17 | 17 | 17 | 17 |
| must | 3 | 17 | 18 | 18 | 34 | 18 | 21 | 17 | 17 | 17 |
| origin | 0 | 36 | 17 | 18 | 18 | 39 | 18 | 17 | 35 | 36 |
| pandemic | 5 | 17 | 18 | 17 | 21 | 18 | 93 | 17 | 18 | 19 |
| president | 0 | 20 | 17 | 17 | 17 | 17 | 17 | 30 | 17 | 17 |
| state | 0 | 36 | 17 | 17 | 17 | 35 | 18 | 17 | 53 | 35 |
| wuhan | 0 | 35 | 19 | 17 | 17 | 36 | 19 | 17 | 35 | 44 |

Fig.8 term edge adjacency matrix

The above Fig.8 is of the term edge adjacency matrix, where the counts signify the number of tweets, they were present together.
All the cells with row_number = columne_number i.e., same terms signify their appearance in the whole corpus that we are dealing with.

### *B. Building and analysing the Term Network*

Now we will form the graph from the termM matrix, that we have made in section IV-A.

```
[1] can  --can        can  --must       can  --pandemic    can  --first
[5] can  --now        can  --may        can  --vaccination can  --vaccines
[9] can  --live       can  --new        can  --help        can  --lockdown
[13] can  --cummings  can  --dominic    can  --even        can  --doctors
[17] can  --amp       can  --health     can  --will        can  --today
[21] can  --inquiry   can  --say        can  --says        can  --people
[25] can  --get       can  --government can  --need        can  --life
[29] can  --many      can  --vaccine    can  --children    can  --video
```

Fig.9 un-simplified term network edges

As we can see there are self-loops, that will make out graph look messy and clustering will be formed such that each node will be a cluster in itself, as self-loops signify their number of occurrences in the whole corpus (whenever a term appears, it appears with itself). So, we will remove these self-loops.

```
[1] can  --must       can  --pandemic    can  --first       can  --now
[5] can  --may        can  --vaccination can  --vaccines    can  --live
[9] can  --new        can  --help        can  --lockdown    can  --cummings
[13] can  --dominic   can  --even        can  --doctors     can  --amp
[17] can  --health    can  --will        can  --today       can  --inquiry
[21] can  --say       can  --says        can  --people      can  --get
[25] can  --government can  --need       can  --life        can  --many
[29] can  --vaccine   can  --children    can  --video       biden--china
```

Fig.10 simplified term network edges

Now the self-loops have been removed from the graph as we can see above

We will examine the degree of the nodes of the term network

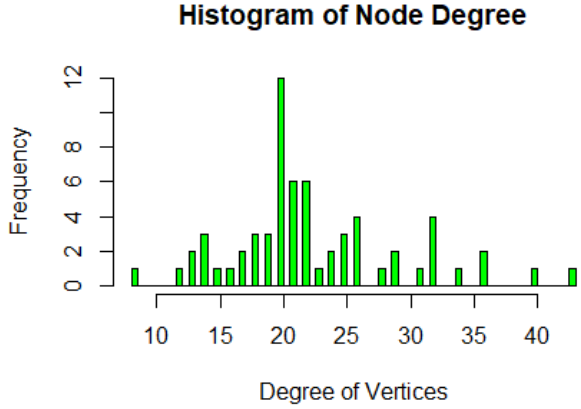


Fig.11 Node degree histogram of network of terms

We will plot the network of terms


Fig.12 Network of terms with label

Which looks quite unreadable with the labels, so we will plot another graph without the labels to see the structure

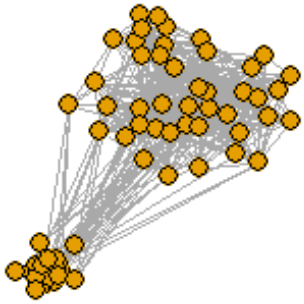

Fig.13 Network of terms without label

Now to analyse along with the communities, we will need the labels, so we will also use the tkpolt in R, to make the plot interactive, so that we can drag the nodes out of the mess when we wish to examine the respective labels

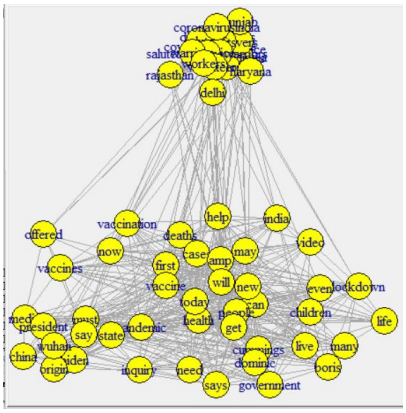


Fig.14 Interactive Netowrk of terms

Now we will use the edge betweenness algorithm to find out the clusters that are present in these nodes.


Fig.15 Edge-betweenness algorithm on Network of terms (with label)

As we can see that the number of giant communities formed is 2, but it is not clear if some small communities also exist, for that we will remove the label and plot the graph again.

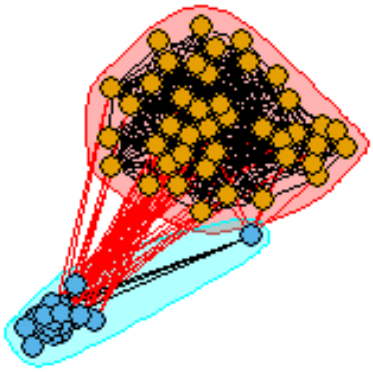

Fig.16 Edge-betweenness algorithm on Network of terms (without label)

Now as we can see that there are 2 groups only, now we analyse the groups

Average sentiment in the first cluster is

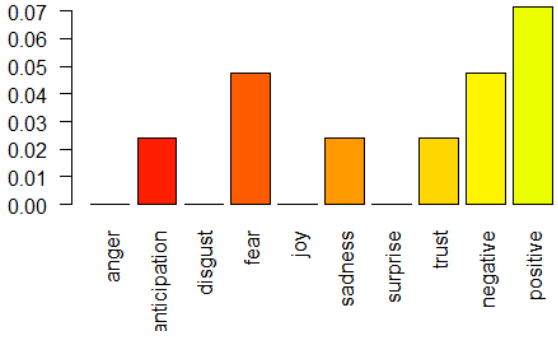


Fig.17 Average sentiment bar plot of group 1

Average sentiment in second cluster is

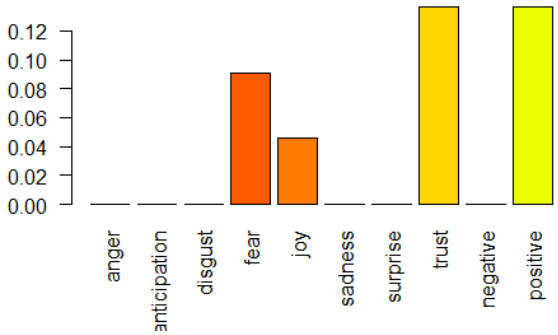


Fig.18 Average sentiment bar plot of group 2

And if we check for the error in clustering these groups using RMSE, we get

```
[1] "Group  1  : "
[1] " Number of memebers :  42"
[1] "RMSE :  0.0239789887510402"
[1] "Group  2  : "
[1] " Number of memebers :  22"
[1] "RMSE :  0.0436630428083952"
```

Fig.19 RMSE of the clusters with Edge-betweenness

Now we will go for clustering using label propagation

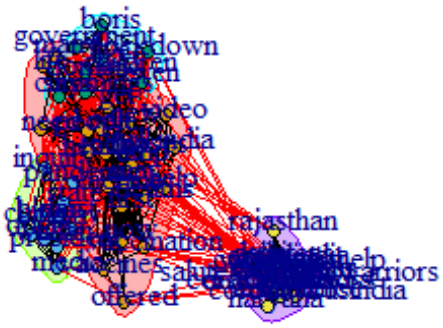


Fig.20 Label Propagation algorithm on Network of terms (with label)

This one also detects four groups; we will remove the labels to get a clear view to the structure of the network

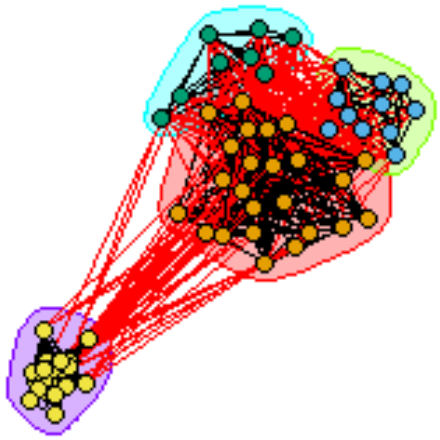

Fig.21 Label Propagation algorithm on Network of terms (without label)

Now we will analyse the groups, in group 1, the average sentiment is as shown in Fig. 22

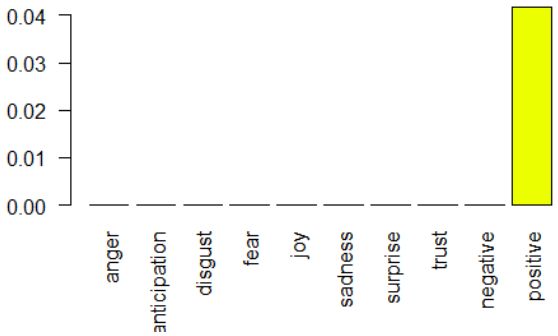


Fig.22 Average sentiment bar plot of group 1

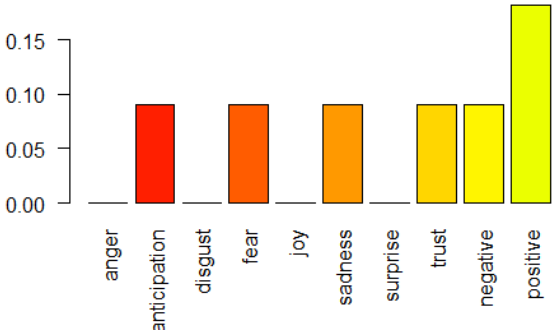


Fig.23 Average sentiment bar plot of group 2

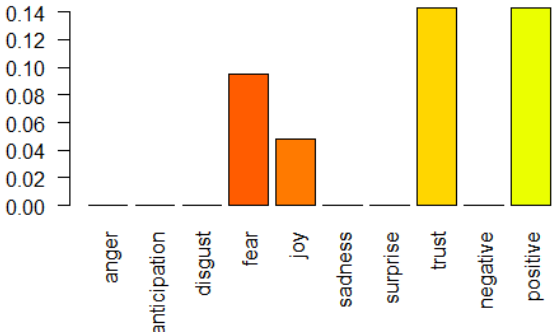


Fig.24 Average sentiment bar plot of group 3

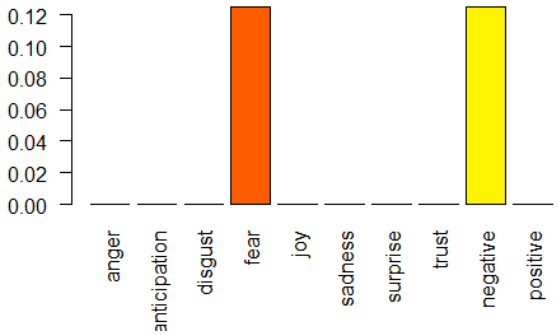


Fig.25 Average sentiment bar plot of group 4

And if we check for the error in clustering these groups using RMSE, we get

```
[1] "Group  1  : "
[1] " Number of memebers :  24"
[1] "RMSE :  0.013447858840998"
[1] "Group  2  : "
[1] " Number of memebers :  11"
[1] "RMSE :  0.0755644922357849"
[1] "Group  3  : "
[1] " Number of memebers :  8"
[1] "RMSE :  0.0592927061281571"
[1] "Group  4  : "
[1] " Number of memebers :  21"
[1] "RMSE :  0.0448154261006819"
```

Fig.25 RMSE of the clusters with Label propagation

Now we will go for detection of community via greedy optimisation of modularity

Fig.26 Greedy optimisation of modularity algorithm on Network of terms (with label)

Here we get two clusters that are forming the clusters, we will remove the nodes to get a clearer view of the structure of the network

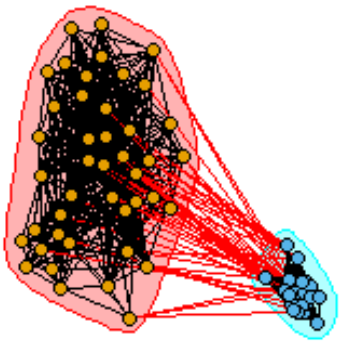

Fig.27 Greedy optimisation of modularity algorithm on Network of terms (without label)

Now we will analyse this group as well and see if the result is comparable with the other two algorithms that we have used before in the later part of this paper

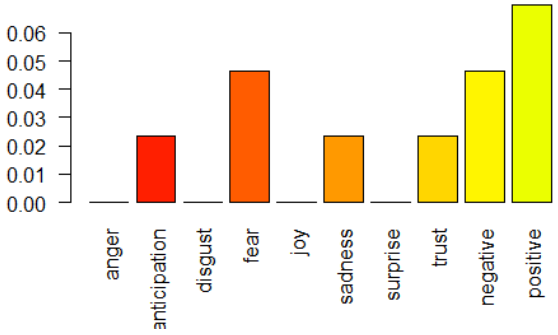


Fig.28 Average sentiment bar plot of group 1

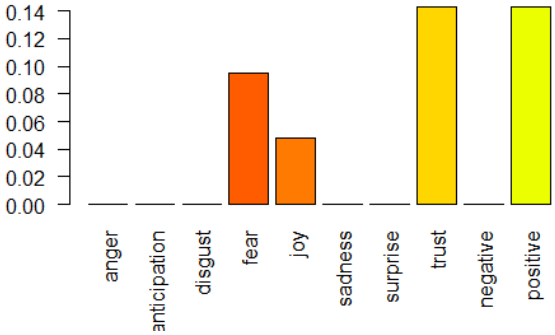


Fig.29 Average sentiment bar plot of group 2

And to check for the correctness of the clustering we will go with RMSE calculation and that gives us the result

```
[1] "Group  1  : "
[1] " Number of memebers :  43"
[1] "RMSE :  0.0233637301430807"
[1] "Group  2  : "
[1] " Number of memebers :  21"
[1] "RMSE :  0.0448154261006819"
```

Fig.30 RMSE of the clusters with Greedy optimisation of Modularity

At the end of our analysis of network of terms, we will go for plotting the plot with degrees as the parameter of the size of the nodes in the network. This will give us a global idea of the frequency of appearance of the words in the tweets. Thus, put light on the current hot topics of discussion.

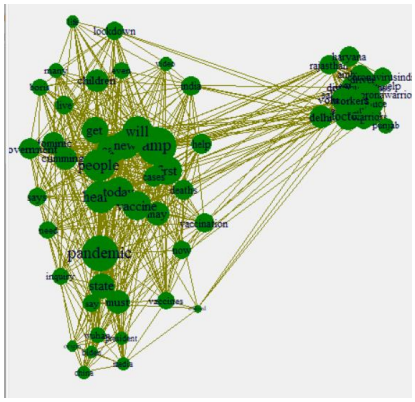


Fig.31 Hilighting degrees of nodes

## V. Network of Tweets

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

### *A. Data pre-processing*

To form the network of tweets, we will have to find out which documents have some common words, those documents are expected to be similar, as many similar words, as similar the document/ tweets are expected to be like.

We will look at the term document matrix again (Fig. 4)

Now we will transpose the term document matrix, it would look something like this

| Docs \ Terms | dms | make | slide | sure | vaccinated | youre | bcoz |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Fig.32 Transpose of term document matrix

The term document matrix formed is of size 37.3 MB, which is huge, so keeping in mind the limited resource we have we will select those tweets that have words at least more than 20 and after those choose words which are present in at least more than 15 tweets.

This reduces the matrix size to 584.1 kb, with 194 terms and 353 tweets.

| Docs \ Terms | make | vaccinated | can | spread | well | times | america |
|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 2 | 1 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 28 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 29 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Fig.33 Reduced Transpose of term document marix

Now as we can see, some words appear more than once in a tweet, we will reduce them to 1, and use the term document matrix like a bool type matrix, that does not count the frequency of occurrence of terms in the tweets, but just indicates the presence or absence of it.

Now to make the tweetM matrix, that will signify the edges between the tweets with respective weight we will use the logic that we used before for make the network of terms in section IV-A

Let, 'tdm' be term document matrix, and

'termM' be term edge adjacency matrix, then

termM = tdm X t(tdm), where t(x) means transpose of x

It looks something like shown below in Fig. 34

| Docs \ Docs | 2 | 5 | 8 | 9 | 14 | 17 | 18 | 26 | 28 | 29 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 23 | 1 | 2 | 0 | 0 | 1 | 0 | 0 | 0 |
| 8 | 0 | 1 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 2 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 5 | 0 | 1 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 11 | 1 | 0 | 0 | 0 |
| 18 | 0 | 1 | 0 | 0 | 1 | 1 | 4 | 0 | 0 | 0 |
| 26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 20 | 16 | 18 |
| 28 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 17 | 15 |
| 29 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 18 | 15 | 18 |

Fig.34 tweet edge adjacency matrix

### *B. Building and analysing the Term Network*

Now we will make a graph out of the tweetM matrix that we made in section V-A.

```
 [1] 2--2   2--86  2--95  2--106 2--142 2--176 2--202 2--209 2--254 2--354
[11] 2--405 2--417 2--425 2--462 2--479 2--520 2--537 2--543 2--596 2--641
[21] 2--661 2--665 2--686 2--732 2--745 2--763 2--772 2--798 2--809 2--849
[31] 2--881 2--888 2--894 2--897 2--946 2--950 2--971 2--996 2--998 5--5
[41] 5--8   5--9   5--18  5--48  5--53  5--87  5--91  5--95  5--105 5--116
[51] 5--117 5--118 5--149 5--154 5--159 5--165 5--172 5--173 5--176 5--178
[61] 5--183 5--193 5--202 5--207 5--216 5--218 5--222 5--255 5--256 5--269
[71] 5--272 5--281 5--282 5--324 5--332 5--338 5--342 5--346 5--349 5--353
```

Fig.35 un-simplified tweet network edges

The graph edges shown above has a lot of self-loops, which will make the graph looks messy and useless. So, we will remove them.

```
 [1] 2--86  2--95  2--106 2--142 2--176 2--202 2--209 2--254 2--354 2--40
[11] 2--417 2--425 2--462 2--479 2--520 2--537 2--543 2--596 2--641 2--66
[21] 2--665 2--686 2--732 2--745 2--763 2--772 2--798 2--809 2--849 2--88
[31] 2--888 2--894 2--897 2--946 2--950 2--971 2--996 2--998 5--8   5--9
[41] 5--18  5--48  5--53  5--87  5--91  5--95  5--105 5--116 5--117 5--11
[51] 5--149 5--154 5--159 5--165 5--172 5--173 5--176 5--178 5--183 5--19
[61] 5--202 5--207 5--216 5--218 5--222 5--255 5--256 5--269 5--272 5--28
[71] 5--282 5--324 5--332 5--338 5--342 5--346 5--349 5--353 5--366 5--37
```

Fig.36 simplified tweet network edges

Now we will plot a histogram of the degree of the nodes to see the business of the network

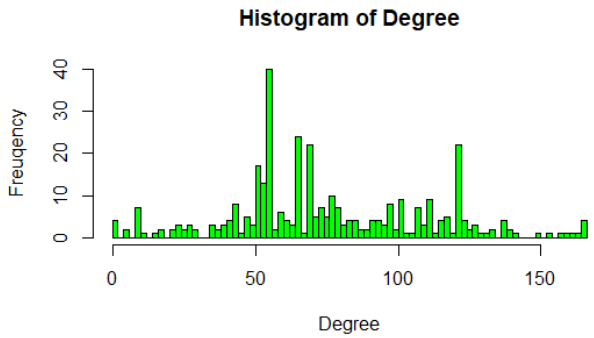


Fig.37 Node degree histogram of network of tweets

Now we will plot the graph to look at its structure

Fig.38 Network of tweets with label

To make the graph more visually pleasing and to understand the underlying structure of the graph, we will remove the labels in the graph

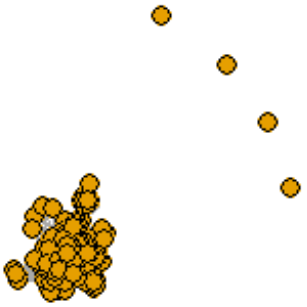

Fig.39 Network of tweets without label

Now we will use the community detection algorithms to find out the communities present in these tweets. The very first algorithm that we will be using is the edge betweenness algorithm.

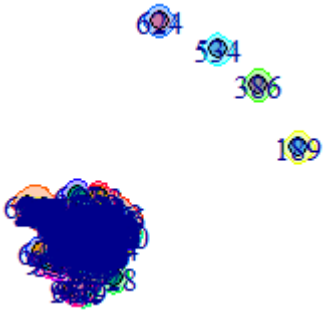

Fig.40 Edge-betweenness algorithm on Network of tweets (with label)

Now we will remove the labels from the graph to get a better view of the clustering

Fig.41 Edge-betweenness algorithm on Network of tweets (without label)

We can see that 50 clusters have been formed, now we will analyze the clusters

Average sentiment in the first cluster is

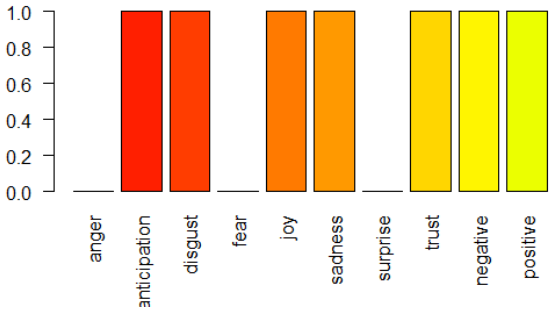


Fig.42 Average sentiment bar plot of group 1

Average sentiment in the second cluster is

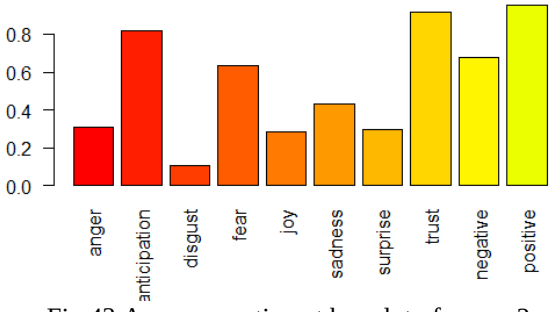


Fig.43 Average sentiment bar plot of group 2

Average sentiment in the third cluster is

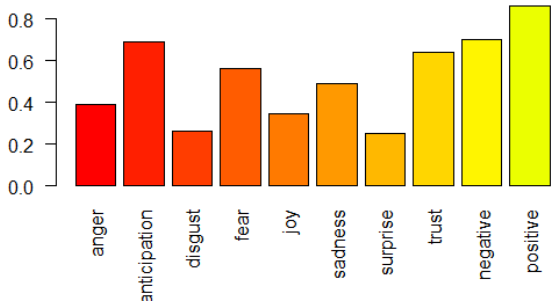


Fig.44 Average sentiment bar plot of group 3

Average sentiment in the fourth cluster is

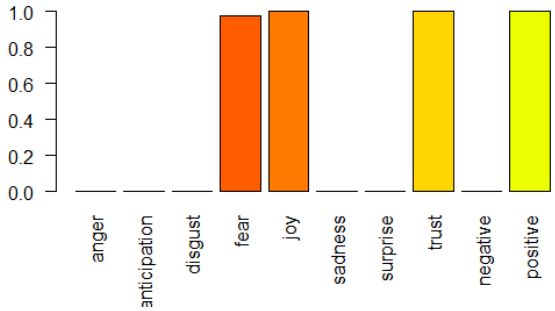


Fig.45 Average sentiment bar plot of group 4

All the other clusters up to 50 have only one member in the groups, and are thus not relevant for visual analysis

Now to see the error in the clustering we would use the RMSE calculations

```
[1] "Group  1  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
[1] "Group  2  : "
[1] " Number of memebers :  84"
[1] "RMSE :  0.0640327390009765"
[1] "Group  3  : "
[1] " Number of memebers :  180"
[1] "RMSE :  0.0405337756500666"
[1] "Group  4  : "
[1] " Number of memebers :  42"
[1] "RMSE :  0.105888347232776"
[1] "Group  5  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
[1] "Group  6  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
```

Fig.46 RMSE of the clusters with Edge-betweenness

Now we will go for community detection using label propagation

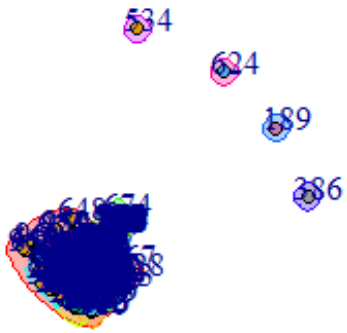


Fig.47 Label Propagation algorithm on Network of tweets (with label)

This is the resultant graph that we get in Fig. 48, we will remove the labels from the graph to get a better view of the structure and the clusters

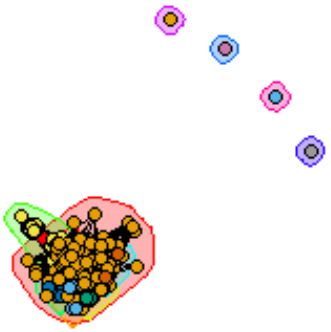

Fig.48 Label Propagation algorithm on Network of tweets (without label)

As we can see using label propagation algorithm, we get 10 clusters. Now we will analyze these clusters.

We will see the mean sentiment score of first clusters to seventh cluster.

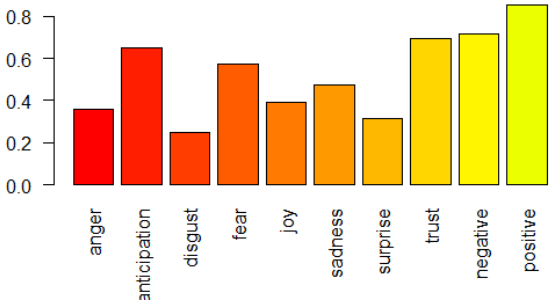


Fig.49 Average sentiment bar plot of group 1

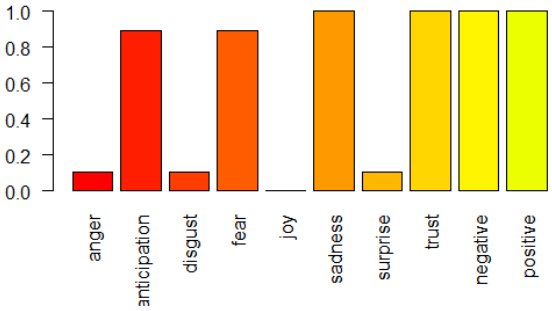


Fig.50 Average sentiment bar plot of group 2

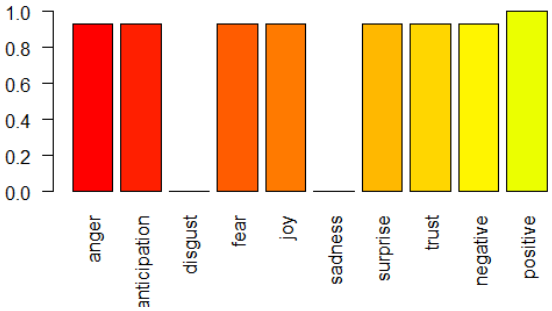


Fig.51 Average sentiment bar plot of group 3

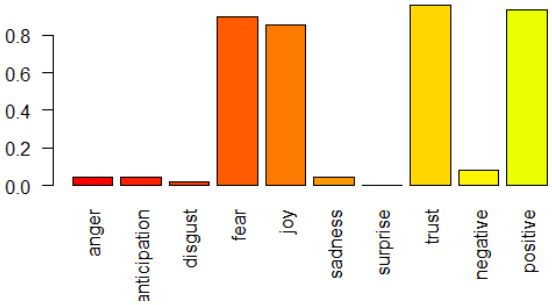


Fig.52 Average sentiment bar plot of group 4

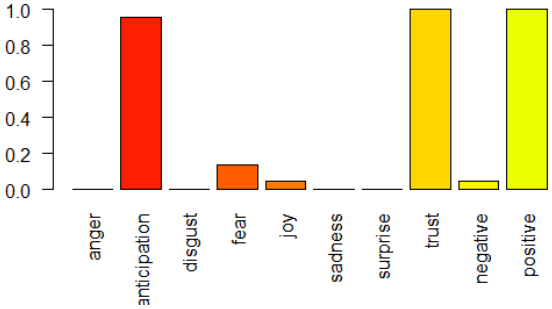


Fig.53 Average sentiment bar plot of group 5

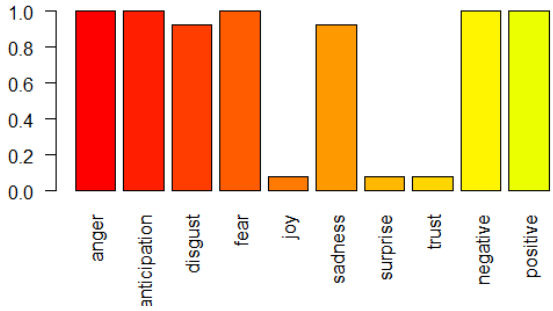


Fig.54 Average sentiment bar plot of group 6

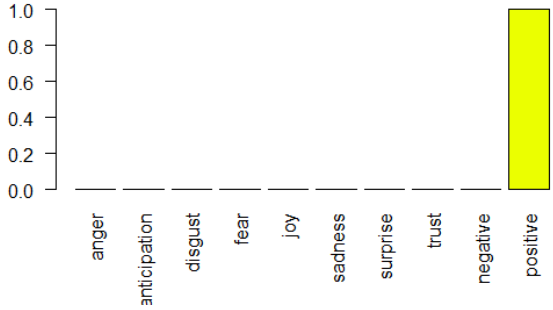


Fig.55 Average sentiment bar plot of group 7

And from group 7 to 10, the clusters formed are single member clusters.

So, it is not relevant to show them all in this paper.

Now we will look at the error that the clustering algorithm made using the label propagation algorithm using the RMSE calculations, below in Fig 55

```
[1] "Group  1  : "
[1] " Number of memebers :  231"
[1] "RMSE :  0.0348673626029116"
[1] "Group  2  : "
[1] " Number of memebers :  19"
[1] "RMSE :  0.147684662663208"
[1] "Group  3  : "
[1] " Number of memebers :  15"
[1] "RMSE :  0.143191143942996"
[1] "Group  4  : "
[1] " Number of memebers :  49"
[1] "RMSE :  0.0932215458505485"
[1] "Group  5  : "
[1] " Number of memebers :  22"
[1] "RMSE :  0.137665957494009"
[1] "Group  6  : "
[1] " Number of memebers :  13"
[1] "RMSE :  0.171740401738904"
[1] "Group  7  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
[1] "Group  8  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
[1] "Group  9  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
[1] "Group  10  : "
[1] " Number of memebers :  1"
[1] "RMSE :  0"
```

Fig.55 RMSE of the clusters with Label propagation

Now we will go for our third clustering algorithm, the greedy method

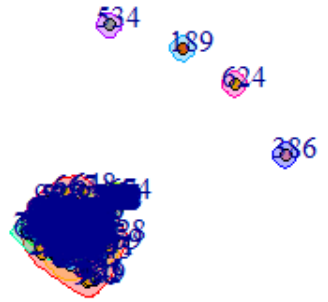


Fig.56 Greedy optimization of modularity algorithm on Network of tweets (with label)

We will remove the labels int the graph to analyze the structure in the cluster in a better way

Fig.57 Greedy optimization of modularity algorithm on Network of tweets (without label)

Now we can see that in the clustering there are 9 clusters to analyze, we will analyze them separately as done before.

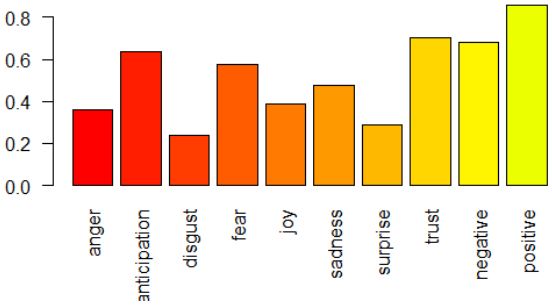


Fig.58 Average sentiment bar plot of group 1

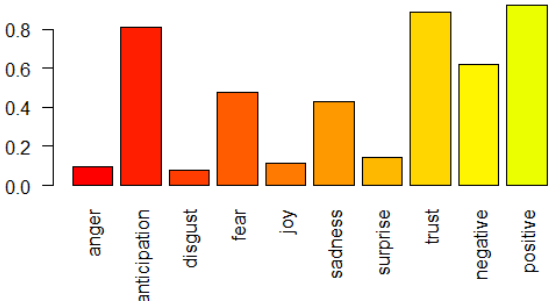


Fig.59 Average sentiment bar plot of group 5

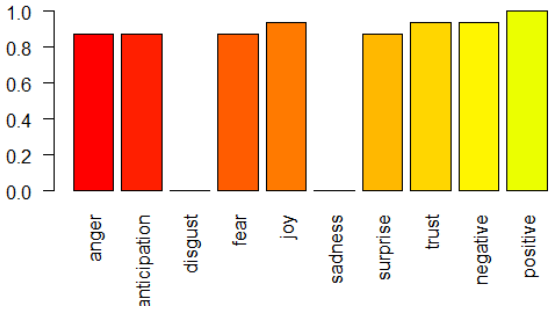


Fig.60 Average sentiment bar plot of group 4

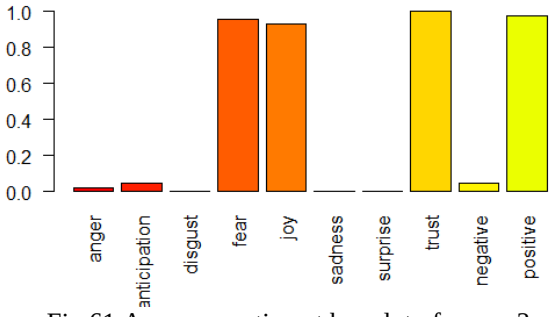


Fig.61 Average sentiment bar plot of group 3

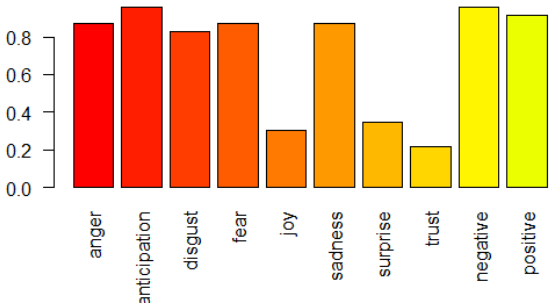


Fig.61 Average sentiment bar plot of group 2

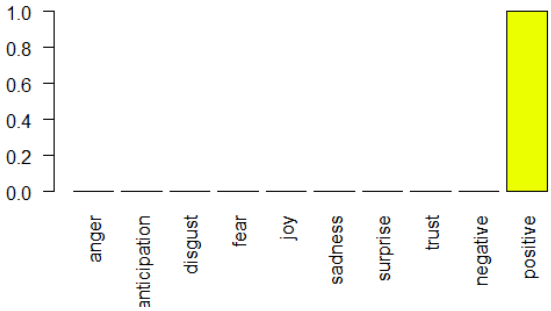


Fig.62 Average sentiment bar plot of group 6

The clusters 6,7,8,9 have only one member so it won't be relevant to plot the graph.

Now we will analyze the error in the clustering using RMSE calculation

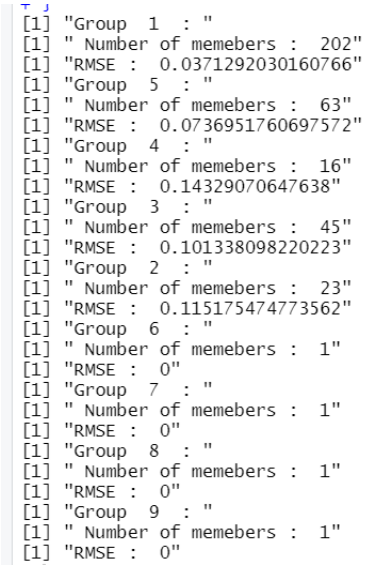

Fig.63 RMSE of the clusters with Greedy optimization of Modularity

## VI. Evaluation of Sentiments of Cluster

We try to evaluate the sentiment of the cluster as whole using the average sentiment plots that we have plotted in the last section and then we use the Root Mean Square Error detection, to evaluate the clusters.

RMSE of a group can be defined as

$$\sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}$$

Where in our case the x is the sentment vector of the term/ tweet as per the context.

### *A. Network of Terms*

Looking at Fig. 11, we can say that most of the terms in the corpus that we formed have degree of 12, and all the terms' degree lies in the 0 to 40. This means that the words which appear with some other word in a tweet are usually 12. But have maximum of 40 and minimum of 0, which is trivial.

Looking into the network plot in Fig. 13, it looks like there are two clusters.

And on analyzing the network with edge-betweenness algorithm we see it results in 2 cluster as well. The first cluster has mostly positive sentiment with little mix of Negative as well. There are elements of surprise, fear, sadness, anticipation as well. On the other hand, the second cluster has no negative sentiment, It has some element of fear but the terms used are of joy and trust (more compared to the other one). The RSM error (Fig.19) in group 1 is 0.023 with 42 members and 2nd group has error of 0.04 with 22 members.

In label propagation the number of clusters formed is 4, with 24, 11, 8 and 21 members respectively (Fig. 25). Group 1 consists of positive sentiment words only, whereas in group 2 we can see mostly positive sentiment words with a little mix of anticipation, fear, sadness, trust, negative. Group 3 and 4 also has unique score of sentiment (different from other clusters) as can be seen in Fig. 24 and Fig.25. The error rate is 0.013, 0.075, 0.059 and 0.044 in group 1 to 4 respectively.

In greedy optimization of modularity also, two clusters are detected. The clusters have unique sentiment plot comparing to each other. The first group has words which are negative as well as positive, it has anticipations, fear, sadness, trust. In the second group, there is no negative terms, there is an emotion of joy in the words. The RMSE can be read from Fig. 30, both the clustering have a very low error of 0.023 and 0.044 in a group of 43 and 21 terms respectively.

From Fig. 31, where the degrees are highlighted, we can see most of the tweets use the term pandemic, vaccine, other relevant terms are workers, volunteers, ambulance etc. all these hot words point to the present situation topics of debate, such as high price of ambulance, concern about the corona workers etc.

### *B. Network of Tweets*

Fig. 37 plots the degree of nodes in the network of tweets, that gives us the idea that the range of the degree is in between 0 to 170, and maximum degree of tweets can be found around 50.

Fig. 38 and 39 plots the graph that shows that the tweets are divided to 5 clusters with one huge cluster and 4 cluster of single nodes.

Using the edge-betweenness algorithm we find there are 50 clusters, of which cluster 2,3,4 have 84, 180 and 42 members are rest all have 1 member. The 1 member clusters have RMSE of 0, which is trivial and cluster 2,3,4 have RSME of 0.064, 0.040 and 0.10 respectively.

The sentiment score of all the clusters were found to be unique.

Now using the label propagation algorithm in the network of tweets gives us 10 clusters. Of which cluster 1,2,3,4 5 and 6 have 231, 19, 15, 49, 22 and 13 tweets and rest of the clusters have 1 tweet. The clusters with one tweet have RMSE of 0 trivially and all other 6 groups have RMSE of way below 1 (Fig.55).

Using the greedy optimization on clustering using modularity we can get result somewhat similar to the label propagation. It gives us 9 clusters and each of the clusters have unique sentiment plot (Fig. 58 onwards). 5 clusters have formed clusters of size more than 1 and all of them have a RMSE of less than 0.2.

## VII. Result and Discussion

From Fig. 6 and 7, we can say that the terms which occur most often in the corpus are volunteers, workers, ambulance, nurses, children etc. all these indicate to the current hot topics that are on debate in the social media these days (26.05.2021).

Now coming to the analysis that we did on the clusters, we can conclude that our social media users have varied opinion and a vast number of sentiments moving on around the current day topics, as we can see the average sentiment plots plotted for each cluster formed using the three clustering algorithms. The sentiments are mostly positive even in these negative times, which supports the fact that people love to share their positive feeling or happening on social media more than the negative. There are almost 40% of the clusters formed which have anticipations as an emotion. which is of importance that not all things viewed

on these social platforms are not fact, some are anticipations as well. And it can be seen very clearly from the sentiment plots that all the sentiment score plots for different clusters are different, that strengthens our aim of showing the difference among clusters on the basis of sentiment scores. On the other side of the coin, if we look into the RSME errors in the clusters formed, we can see that they are way below 1, which shows the similarity in the clusters formed, thus supporting our second goal of this experiment

Talking about the comparing the three algorithms, we can say that all of them give reliable results irrespective of the act that in some cases their results differ. But talking of the usability, edge-betweenness algorithms are feasible for small networks, because of their complexity of order 3. On the other hand, the other two algorithms can solve the problem in nearly linear time. In the edge-betweenness algorithm, the result will come the same each time you run the algorithm in whatever order. But in the other two algorithms the choice of the order of nodes can create a difference in the structure of clusters formed.

## VIII. Future Scope

The sentiment analyzer that we used is not a very strong one. It uses unigrams only for analysis, thus terms like ‘not happy’ gets considered as positive and negative both. We have limited h/w resources so we had to take a small dataset of 1000 tweets and perform our experiment. We can use a better h/w support and work with a larger dataset to get a broader view of the present scenario using the analysis of tweets. We had to analyze the tweets manually by looking into the program outputs or the plots. In future we may use NLP techniques to tabulate the analysis based on the comparisons made between the algorithms implicitly by the program only.

## References


[1] Vishal A. Kharde, S.S. Sonawane, “Sentiment Analysis of Twitter Data: A Survey of Techniques” *International Journal of Computer Applications Volume 139 – No.11, April 2016*

[2] Saif M. Mohammad, Peter D. Turney, “Emotions Evoked by Common Words and Phrases:Using Mechanical Turk to Create an Emotion Lexicon” *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text.*

[3] Tanvi Hardeniya 1, Dilipkumar A. Borikar, “Dictionary Based Approach to Sentiment Analysis - A Review” *International Journal of Advanced Engineering, Management and Science Vol-2, Issue-5, May- 2016*

[4] P. De Meo, E. Ferrara, G. Fiumara and A. Provetti, "Generalized Louvain method for community detection in large networks," *11th International Conference on Intelligent Systems Design and Applications, 2011*

[5] U. Raghavan, R. Albert, S. Kumara, “Near linear time algorithm to detect community structures in large-scale networks” Physical Review E 2007

[6] Seunghyeon Moon, Jae-Gil Lee and Minseo Kang, "Scalable community detection from networks by computing edge betweenness on MapReduce," *International Conference on Big Data and Smart Computing 2014*