Community Detection in Social Media

Vaishali[1], Manish Bhardwaj[2]

[1]*B.Tech Scholar, Dept. of Computer Engineering, Poornima Group of Institutions, Jaipur,*[2]*Assistant Professor, Dept. of Computer Engineering, Poornima Group of Institutions, Jaipur*

[1]*2014pgicsvaishali@poornima.org,*[2]*manishbhardwaj@poornima.org*

Abstract

Social Media comprises of large number of websites like Facebook, twitter, YouTube, LinkedIn and many more. In today's world almost every people spend countless hours on social media for their own interest and reasons. As for example either to communicate, to share something, for job searching and for many purposes. These all activities generate a large amount of raw data in social media which are differing from traditional data. Analysing these raw data and generating the content which are relevant to the interest of users is very important and this process is called social media mining. For analysing any social media it is very important to detect communities. As forming communities improves the overall performance of that site. As in the case of twitter previously topic model algorithm were used for generating semantics pattern but it works for the tweets of certain length only. So, Users are grouped together on the basis of their interest and network in the form of community. These communities are the subset of large network which are less noisy and hence improves the performance of the site. The communities on various sites can be formed on the basis of attribute of the members which are member based community detection and on the basis of attribute of whole community which are group based community detection. In this paper, we present member based and group based community detection algorithms for detecting communities on various social media networks.

Keywords: *Brute-force identification, Clique-Percolation , Girvan Newman Algorithm*

1. Introduction

In real world communities are formed when people with same interest, location, characteristics etc. come together under same geographical area. On various social media sites communities are formed when people with same interest come together and start interacting with each-other. These communities are called virtual communities. On different sites different types of communities are formed. As on Facebook various groups are formed where users can post message, image, comment on each other message, On LinkedIn various professional groups are formed where users can post and share information about their area and about various jobs in their area. Similar groups are formed on other social media sites as well. Detecting these groups or communities helps in analysing various sites. The problem that occurs in detecting these communities is finding the set of vertices that are most densely connected from the rest of the vertices in the network. But, in spite of many problems it helps a lot in many ways as while analysing individuals it is important to analyse the group with which they belong to. Second point is some behaviour is only seen at group level and not at individual level. Third, the characteristics of individuals are flexible and can change whereas the characteristics of group or community are more robust to change. Fourth, groups provide a global view of interaction whereas individuals provide local view of

interaction. Detecting these communities provides valuable insight to a number of business applications like marketing intelligence and competitive intelligence. Because of these features and applications , detecting groups or communities not only helps in understanding what makes individuals comes together but also helps in understanding the overall structure and properties of various social media websites and thereby helps in improving their performance.

2. Literature Review-

Yuan He [1] proposes a topic model named SILDA to mine the coherent topic of tweet collections. SILDA model is an improvement over previous topic model which works for the tweet of certain length only. SILDA model works by forming the groups of based on the interest of users and their retweet network which forms a small sub-sets of large network of twitter which are less noisy and these sub-sets are then applied to topic model to improve their performance.

James hardy [2] proposes an event evolution model named HEE model. This model considers the user interest distribution, and uses the short text data in the social network like micro blogging to model the posts and applies the recommend methods to discover the user interest. This model is an improvement over existing approaches which does not consider user interest. The improved model first filter out all the important events from the general events and then applied clustering algorithm to cluster all the short text into cluster of similar topic which is similar to forming communities on other social sites. At last the model combine the short text of each cluster into large document to simplify the interest of each user during the evolution of important events.

Girvan and Newman [6] have introduced a divisive approach in which the edges are removed depending upon their distance value. By repeatedly cutting the edges with greatest distance value it gets an optimized division of the network with the time complexity of O (m3).

Huan Liu [3] proposes the Brute force clique identification for identifying the clique on a large network and then apply some constraints such that problem can be solved in polynomial time. It then apply a clique percolation method on the cliques to identify all the overlapping communities in the network.

Other algorithms [4, 6] include the one proposed by Nan Du which does not require any prior knowledge about the number of divisions in the community and works efficiently in the large network with a running time complexity of O(C *T2) where C is the number of detected communities and T is the number of triangles in the given network for the worst case.

3. Community Detection Algorithms-

Community detection algorithms detect the communities either on the basis of attribute of its members or the basis of attribute of whole community. The former is detected by member based community detection and the latter by group based community detection.

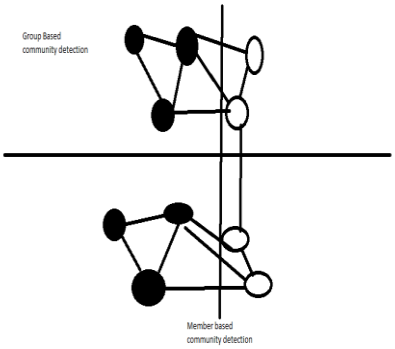

Figure 1

Consider, an example of a group of 10 individuals in which 6 are wearing black colour t-shirt and 4 are wearing white colour t-shirt, if we group the individuals on the basis of t-shirt colour only then it is detected member based community detection as it forms the community on the basis of attribute of its members. If we divide the members on the basis of density of interaction among them then it is detected by group based community detection.

3.1 Member-based community detection-

In member- based community detection members with the same characteristics are grouped in the same community. Member based community detection algorithm group members based on their attributes or measures such as similarity, degree, or reachability. In graph G (V, E), only those subgraphs that have nodes with specific characteristics are declared as community. These node characteristics are node similarity, node degree and node reachability. Node degree means those subgraph in which every node has certain degree, reachability means paths between the nodes should have specific characteristics and similarity means the most similar nodes belongs to the same community.

To find communities in a given graph G(V, E) , we find the maximal clique in the graph, maximal clique is a largest subgraph of the given graph in which every pair of nodes are connected to each-other, for this purpose we use brute-force clique identification algorithm to identify all the cliques in a given graph.

3.1.1 Brute force clique identification

Require: Adjacency Matrix A, vertex Vx

1: return Maximal Clique C containing Vx

2: CliqueStack = {{Vx}}, Processed = {};

3: while CliqueStack not empty do

4: C=pop (CliqueStack); push (Processed, C);

5: Vlast= Last node added to C;

6: N (Vast) = {Vi|Avlast,vi = 1}.

7: for all Vtemp∈ N (Vlast) do

8: if CU {Vtemp} is a clique then

9: push (CliqueStack, CU {Vtemp});

10: end if

11: end for

12: end while

13: Return the largest clique from Processed

The algorithm starts with an empty stack of clique. Initially, the stack is initialized with the value Vx. Then, from the stack a clique is popped. The last node i.e., Vlast which is added to C is selected. Then, all its neighbour are added to C sequentially and if the set of nodes from a larger clique and then it pushed to the stack. This procedure is repeated until all nodes can no longer be added. This algorithm works for the small network, but becomes impractical for large networks. Consider, a graph having 200 nodes, then the algorithm will generate 2199 different cliques in any case. To overcome, with this situation we can either relax the clique structure or use the clique as a seed or core of community.

For relaxing clique, we will use K-plex concept. Using K-plex concept, if there is a clique of size k, then all the nodes have a minimum degree that is not necessarily k-1. In graph G(V,E), for a set of vertices V, the structure is called K-plex if-

Dv>=|V|-k,,∀ v∈ V, where Dvis the degree of v.

As the size of k in k-plex increases, the structure becomes more relaxed because, we can remove more edges from the clique.

While using clique as a seed or core of community, we will use clique percolation method (CPM), in which we will assume that communities are formed from a set of clique. The algorithm for this approach is given below-

3.1.2 Clique Percolation Method (CPM)

Require: parameter k

1: return Overlapping Communities

2: Cliquesk = find all cliques of size k

3: Construct clique graph G (V, E),

Where |V|=|Cliquesk|

4: E ={eij |clique i and clique j share k−1 nodes}

5: Return all connected components of G

The algorithm starts by finding all the cliques of size k. A graph is then formed by using all the cliques as nodes and connecting them via an edge. Communities are then found by searching all the connected component of this graph. As this algorithm works for small k. Relaxations are used to enable the algorithm to perform faster. Then, clique percolation method returns all the overlapping communities.

3.2 Group-based community detection-

In group-based community detection, groups with same characteristics are put in the same community. Group based community detection algorithm group communities based on certain characteristics such as balanced, robust, modular, dense or hierarchical. In robust we find the subgraph that are robust i.e., on removing some of the nodes or an edge does not disconnect the subgraph. In modularity, we see that how far the community structure detected is created at random. In dense, we find the subgraph in which the nodes have more frequent interactions. In hierarchical, each community can have super or sub community. Girvan-Newman algorithm is designed for group based community detection.

3.2.1 Girvan-Newman Algorithm-

```
1. import networkx as nx
2. import itertools
3. defcommunities_brute(G)
4. nodes=G.nodes()
5. n=G.number_of_nodes()
6. first_community=[]
7. for iin range(1,n/2+1)
8. comb=[list(x) for x in itertools.combination
   (nodes,i)]
9. first_community.extend(comb)
10. second_community=[]
11. fori in range(len(first_community))
12. l=list(set(nodes)_-set(first_community)
13. second_community.append(l)
14. num_intra_edges1=[]
15. num_intra_edges2=[]
16. num_inter_edges=[]
17. num_intra_edges1.append(G.subgraph(first_community[i].number_of_edges))
18. for i in range(len(second_community))
   :num_intra_edges2.append(G.subgraph(second_community[i]).number_of_edges())
19. e=G.number_of_edges()
20. for i in range(len(first_community)
21. num_inter_edges.append(e-num_intra_edges1[i]-num_intra_edges2[i])
22. for i in range (len(first_community))
23. ratio.append(float(num_intra_edges1[i]+num_intra_edges2[i]/num_inter_edges[i])
24. max_value=max(ratio)
25. max_index=ratio.index(max_value)
26. print'('first_community[max_index],')','('second_community[max_index],')'
27. G=nx.barbell_graph(5,0)
28. communities_brute(G)
   To run- import networkx as nx
   G=nx.barbell_graph(5, 0)
   nx.draw(G)
   importmatplotlib.pyplot as plt
   plt.show()
   G=nx.barbell_graph(4,1)
```

```
nx.draw()
Plt.show()
```

The algorithm works as follows:

Initially, a function named communities_brute(G) is defined on the graph G. G.nodes() and G.number_of_nodes() returns all the list of nodes and the number of nodes in graph G respectively. Objective of this algorithm is to divide the nodes of this graph into two communities say first community and second community. In this brute-force approach, we will use all the possibility where first Community has 1 node and second community has n-1 nodes, first community has 2 nodes and second community has n-2 nodes and so on. In this algorithm, we will check all the possible cases. In case of first community, we have taken the range in the function as n/2 +1. This is so because, if we use 1 up to n, then it goes up to n-1 only. So, we add +1 to use exactly n/2 nodes. Next, we have used the combination () function as itertools-.combination (nodes,i), which returns all elements which are up to i. The combination () function is defined in the package itertools.

Ex= importitertools

Itertools.combination [(1, 2, 3, 4),2]

The given line of code returns all the possible combination of length 2 i.e., combination () function returns number of members in list which goes from 1 to length of the list. It does not return the list, it returns the object. So, if we have to see the tuples of object, we have to use for loop, as we have use in our function, the function works as follows-

Ex=for i in itertools.combination [(1, 2, 3, 4), 2]

Print i.

The given line of code gives the output as (1,2),(1,3),(1,4),(2,3),(2,4),(3,4).If we want to see the tuples in the form of list we will use the function list ().

Ex=for i in itertools.combination[(1,2,3,4),2]

Print list (i)

The given line of code gives the output as [1,2], [1,3], [1,4], [2,3], [2,4], [3,4].Next, we have used the extend () function in our code as-first_community.extend(comb), this line of code add all the elements of comb list to first community. If we had used append () function in place of extend () function, it gives the output as-

Ex= l1.append (l2)

The output will be like [1, 2, [3, 4]]. So, we have use extend () function to get the output in the form of [1, 2, 3, 4].After, adding list of elements to the first community, the second community is initialized, to get the elements of second community we subtract the elements of first community from all the nodes of the graph and for this purpose we have to convert list to set. So, next in our algorithm we have use l=list(set(nodes)-set(first_community)), which first convert all the list of nodes in the graph into set of nodes and then subtract set of first community nodes from all the set of nodes, and put this set into l. In the next line we have use append () function to get all the set of nodes in second community. Now, both the first and second community get the set of nodes in their communities.

Edges with the nodes of same community are called intra community edges which are dense and edges with the nodes of other community are called inter community edges which are less.Next, in our algorithm we have to find out which division is best and for to satisfy that the division is good, intra must be higher than inter. So, we have initialized both the intra edges in first and second community and inter edges. To find, the number of intra edges in both communities, next in our algorithm we have use the function subgraph (), we takes the list of nodes from and returns the induced subgraph over this graph. Now, we get the intra edges in both the communities, next we have to find the number of inter edges and for this purpose, we subtract the intra edges of both the communities from total number of edges. So, in our algorithm we have use the line as-num_inter_edges.append (e-num_intra_edges1 [i]-num_intra_edges2 [i]). Now, we have intra edges of both communities and inter edges as well. So, next we will find out the ratio by using formula as-

Ratio= (num_intra_edges1+num_intra_edges2)/num_inter_edges

So, next in our algorithm after finding the ratio, we will find the maximum value of this ratio by using max () function. Next, we will find out the index of this maximum value by using index () function as in our algorithm we have use index (max_value), this is used to find the best division of all the possible division. After, finding the best division, at last we will print the division of nodes into two communities.

4. Conclusion

In the case of member based community detection, Brute-force approach for finding the maximal clique but it has certain limitations as it can work for a graph have small number of nodes only, so some restrictions and clique percolation method are added to overcome with this limitations. But, in spite of that working on this algorithm is costly. In case of group based community detection also brute-force approach is used to find out the best division. Though, the algorithm is written in python which contains some predefined functions to work on graph but still it is very length, which increases the time complexity of the algorithm. So, we expect some refinement of existing methods and introduction of some new methods to cope up with the increasing scale of social media data.

References

[1] Y He, C Cheng, and CChangjun(2017, June 7). Mining Coherent Topic with Pre learned interest knowledge in Twitter, 5, 10515-10523, China: Shanghai.IEEE.

[2] L Shi, L Liu, Y Wu, L Jiang, and J Hardy (2017, March 1). Event Detection andUser Interest Discovering in Social Media Data Streams, 5, 20953-20962, China: Zhenjiang. IEEE.

[3] S Papadopoulos, Athena Vakali, and P Spyridonos (2011, June 14). Community detection in Social Media Performance and Application Consideration, Greece:Thessaloniki.Springer.

[4] S Rai, S Chaturvedi, and A Jain (2017 March). Community Detection on Social Media: A Review, 3, 29-33, India:Bhopal. International Journal of Scientific Research and Technology.

[5] N Du, B Wu, X Pei, B Wang, and L Xu (2007, August 12). Community Detection in Large-Scale Social Networks, 16-24, Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia Beijing University of Posts and Telecommunications, China.

[6] M. Girvan and M. Newman. Finding and evaluating community structure in networks. Physical Review E, 69(026113), 2004.