**Graph Formation Twitter Rank:**

In the referenced Paper we form a network(graph) of the users with the nodes being the users and the edge signifies a friend relationship. They use the below formula to decide the edge weight between users:

Pt(I,j) = |Rj|\* simt(i,j)/(∑a: si follows sa |Ra|) ----------------------------------(1)

In the above equation |Rj| is the number of tweets published by sj, and ∑a: si follows sa |Ra| sums up the number of tweets published by all of si ‘s friends. Simt(I,j) in Eq. (1) is the similarity between si and sj in topic t, which is defined as:

Simt(i, j) = 1 - |DT’it - DT’jt |

The DT’ is a row normalized matrix of DT where DT is the document term matrix. It signifies the influence of each user on each topic. As we can see |DTit’ – DTjt’| is the difference of distance between user I and user j and 1-|DTit’ – DTjt’| signifies the similarity. So if the topic influence of two users on two topics is almost the same then we can see that they have a high similarity.

We have done a similar computation for our project with the difference being that we have used the fan relationship instead of the friend relationship. In our yelp dataset we found that each user had a large number of fans instead of friends. Using the fan relationship did solve the problem of graph sparsity. We used the same formula as mentioned below but instead of tweets we counted the number of reviews. The formation of the document topic matrix has already been covered in the previous section.

**PageRank Introduction:**

Before going ahead to next section, we briefly discuss the PageRank algorithm for finding out the most influential node in a graph. First lets talk about PageRank in undirected unweighted graphs. Pagerank was first invented by google to bring ranking to the web. It is a random walk model. So consider a node and then we follow any of the edges randomly, then what is the probability of following each edge.

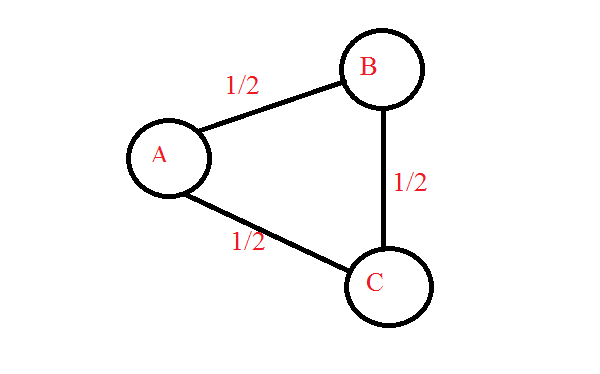


Fig 1.

In Fig 1 we see that the probability of moving from A to B and A to C is ½ as there are two edges of A. So if we do a random walk from A then we have equal probability of moving to B or C. Also pagerank has a damping factor. The damping factor signifies whether the user does continue the random walk of he again starts the teleportation. In the paper as well as in our project we have taken the damping factor as 0.85. So the probability of continuing the random walk is 0.85 and the probability of restarting the random walk is 0.15. So when we do this random walk a multiple times we get the pagrank scores of all the nodes. The pagerank score is nothing but the importance or the number of times a random surfer visits that node. So if a node has highest value then it is the most number of times a random surfer visits it and thus it is one of the most significant node.

Consider another graph shown below:

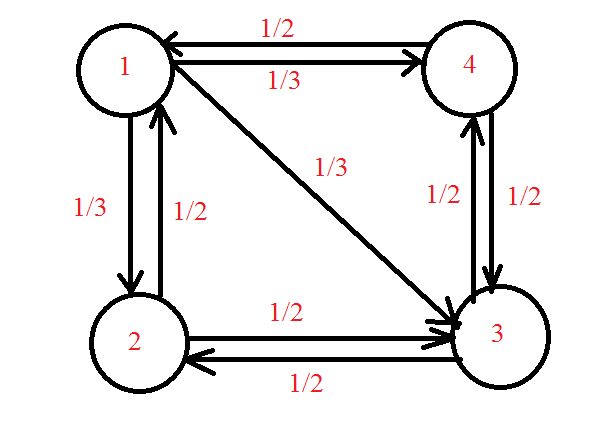


Fig 2.

Now the transition matrix for the above graph is as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | ½ | 0 | 1/2 |
| 1/3 | 0 | ½ | 0 |
| 1/3 | ½ | 0 | 1/2 |
| 1/3 | 0 | ½ | 0 |

So in this matrix A[2][3] signifies the 4th column, 3rd row which is nothing but the probability of moving from node 4 to node 3 in the random walk. So the formula for calculating the pagerank scores for the above transition matrix is as follows:

Algorithm:

1. Let the transition matrix be A.

|  |
| --- |
| 0.25 |
| 0.25 |
| 0.25 |
| 0.25 |

2. Let the Teleportation vector be B. Initially all the nodes have equal teleportation probability. Thus B for fig 2 is:

B=

3. Initialize oldB to a matrix of size B all values initialized to 0.

4. while(B !=oldB)

OldB = B;

B = A\*B

5. Output B

If we closely see here B is nothing but an eigen value and this page rank score is the eigen vector corresponding to the eigen value 1.

One of the key problems of the above matrix is that for an unconnected graph some of the nodes will never be reached. To achieve this we introduce the damping factor. So the equation for PageRank can be written as

B= (1-p)A\*B + p\*T ---------------------------------------------(1)

The p in the above equation is the damping factor

T is nothing but the Teleportation vector , in this example all the nodes have equal probability so it is a n X 1 column matrix with each value 1/n. n is the nodes in the graph.

**Calculating Twitter Rank :**

Twitter rank is similar to PageRank but there is a change in the teleportation vector. The equation for twitter rank is as follows:

TRt  = (alpha)\*Pt +(1-alpha)Et

Here Et is the teleportation vector.

Here Et  = DT’’ where DT’’ is a column normalized matrix of DT.

So if a user has a large contribution in a specific topic then there is a higher chance of teleporting to him in the random walk.

**Aggregation of Topic Specific PageRank:**

So now to get the most influential user for a specific user what we do is that we take the bag of words of the user and count the number of words in each topic. So we get the contribution of a user on each topic is given by:

Ct = (no of words in a topic t)/(total number of words by the user)

So the ration Ct is multiplied to the pagerank score of that topic and then we get the most influential users.

TR = ∑ Ct TR -----------------------------------------------------------(3).

Now the users with the highest score in TR are the mot influential user specific to the reviewer.

In the next section we see our approach to solve the similar problem.

**Our Approach:**

One of the key points of observation in the above approach was that the influential users largely depended on the number of connections in the network rather than the topic i.e the top influential users remained the same for a large number of test users even though their topic interests is the same. To try and reduce the effect of the network and give more weightage to the topic interests we propose a clustering based method. The flowchart in fig 1 has already shown the difference between our approach and the previous approach.

In our approach we apply k means clustering after forming the Document Term Matrix DT. One of the advantages of applying the clustering method is that the users with the same interest are likely to fall in the same cluster. For e.g if a user is interested in Mexican food and a quite atmosphere then such users will be in the same cluster. So it takes users who may have multiple similar interest and puts them in the same cluster. One of the key issues of k means clustering is that we have to find an optimum value of k i.e the number of clusters. To find the optimum number of clusters we use the elbow method. The elbow method plots k vs the objective function of k means clustering.

Objective Function of K Means = ∑i=1n|uj – xi |2  where n is the number of points and uj is the cluster center corresponding to the point xi. Now we plot a graph of k vs the objective function.\

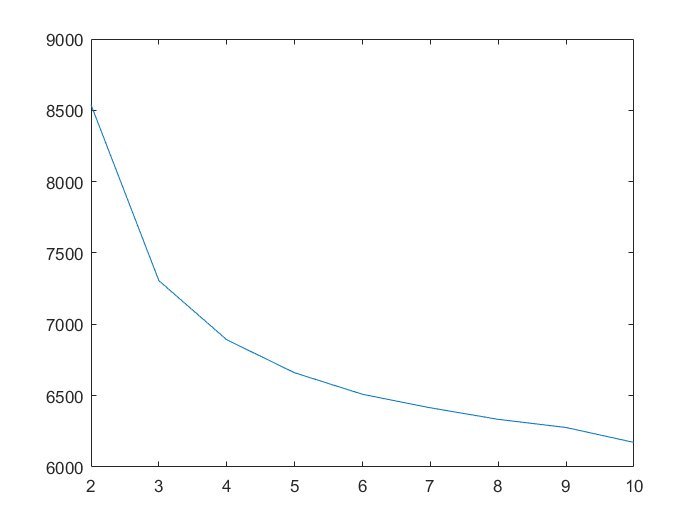
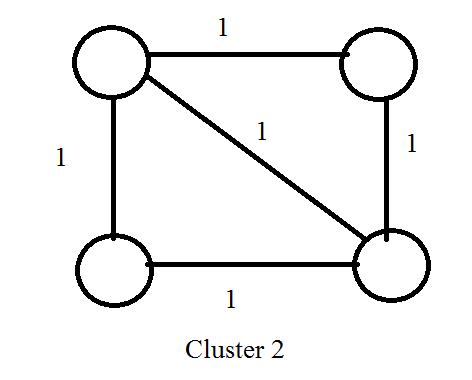
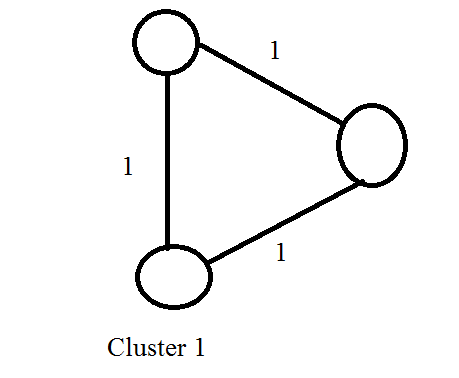


Fig 4: K vs Objective Function

Now we see that at k=5 the objective function quickly starts tapering down. So using the elbow method we can infer that k=5 is the optimum number of clusters.

**Graph Formation for K Means:**

In this approach we have already clustered the users with similar interest. Now we form a graph and the edges of the graph are only within each cluster i.e there is no edge between nodes of cluster1 and any other cluster. The edges of the graph take only two values i.e 0 or 1. If 2 users have a friend relationship and both are in the same cluster we give the value of 1 between them otherwise we give a value of 0. This can be better explained with the diagram below:



So in the above diagram we show two sample clusters and as described below an edge has a default value of 1.

**Finding Most Influential User:**

For finding the most influential user we take a test user and find the cluster in which he falls. This is found by comparing his word vector to each of the cluster centers and the one with the smallest distance is the one where he’s placed. Now in that cluster we form the graph as described above and then apply a simple page rank algorithm. The simple PageRank algorithm is similar to the one described above but there the weights here are 0 or 1 and also the teleportation vector starts with equal probability for all nodes in that cluster. The simple pagerank can be better explained with the equation below as shown in equation (1) before:

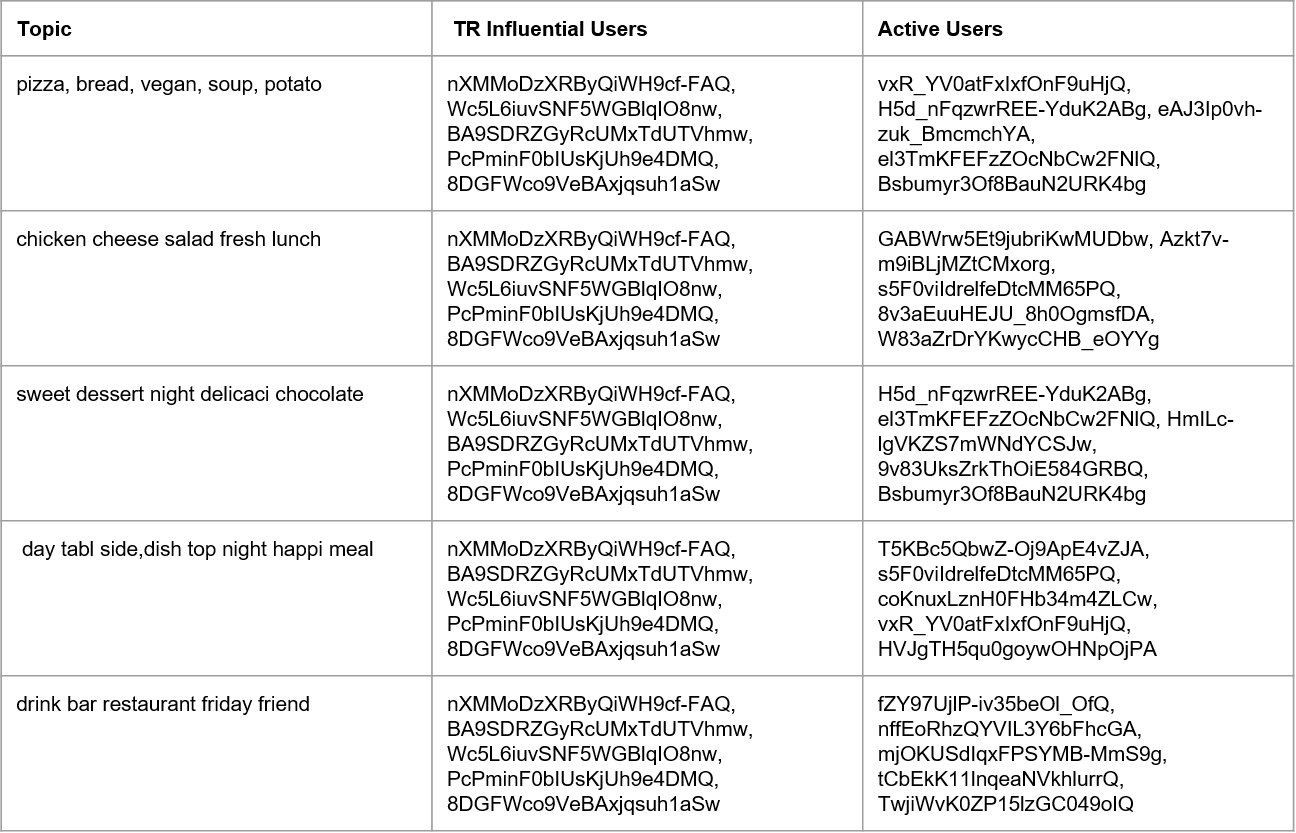
B= (1-p)A\*B + p\*T ---------------------------------------------(3)

The p in the above equation is the damping factor.

T is nothing but the Teleportation vector , in this example all the nodes have equal probability so it is a n X 1 column matrix with each value 1/n. n is the nodes in the graph.

Now the users with the highest PageRank scores are the most influential users relevant to the network.

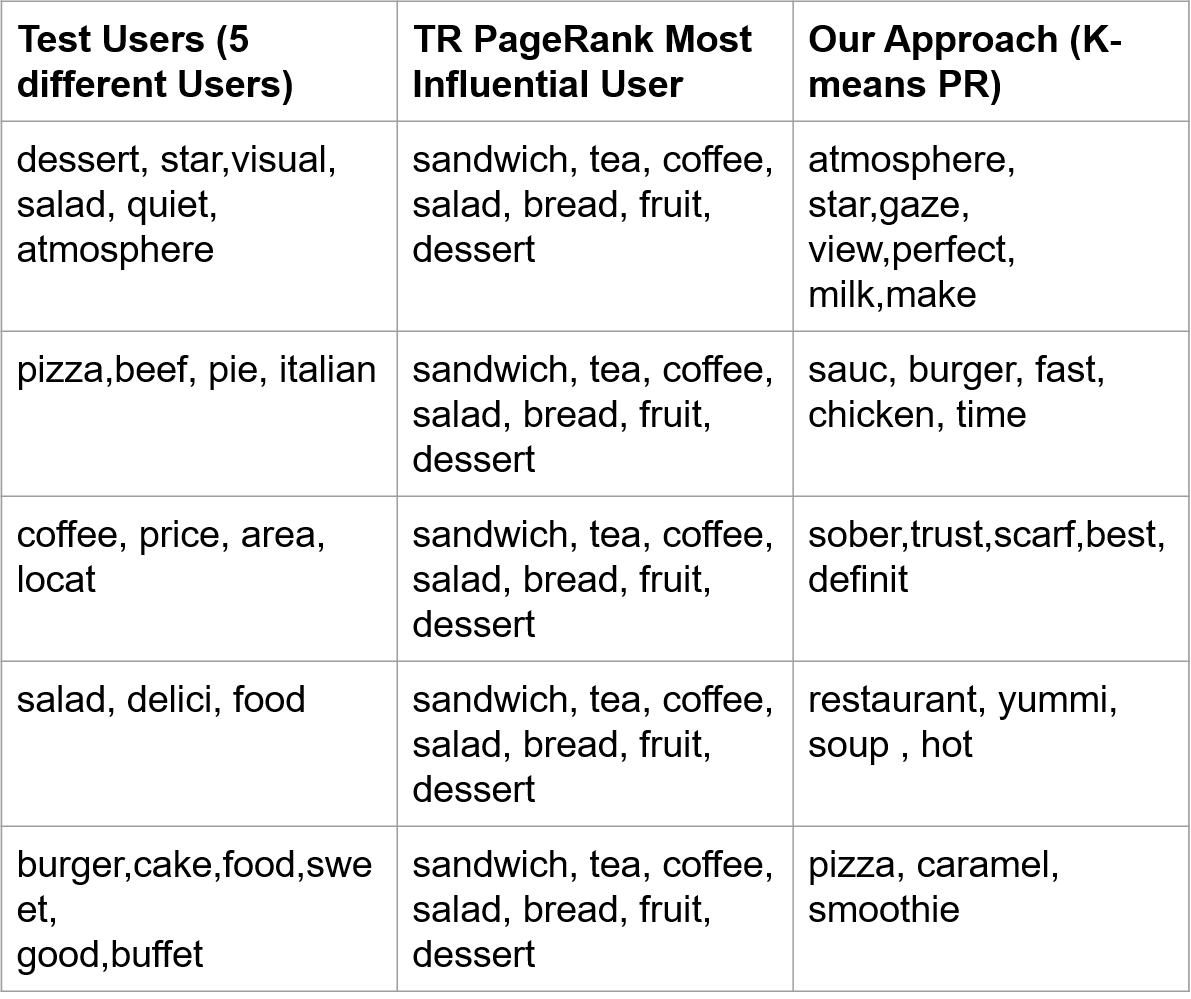
**Results:**



Active Users vs Influential Users

Here we show the active users vs the influential users, the active users are the users who have the most number of words in that topic whereas the influential users are the most influential users obtained via our first approach. We see that it is not necessary that the most influential user and the active users remain the same. Also we see that the most influential user remains the same in Influential users and this is because of the connections in the network and the teleportation vector.

Now we take 5 random test user and compare the bag of words obtained by approach 1 and approach 2.



Comparison of Approach 1 and Approach 2

In the above figure we see that the most influential user via the twitter rank approach remains the same but the influential user according to our approach changes as each of the user falls in a different cluster.

Now lets compare the bag of words obtained from approach 1 and approach 2 for a random test user

**Test data Topic words**:

flawless,dessert,fire,love,star,visual,sinc,info,good,tast,amaz,quick,gorgeou,place,dinner,person,friend,quiet,area,glass,salad,challeng,almost,atmospher,simpli,via,salon,half,keep,fill,immedi,vibey,call,filet,hill,fountain,easili,group,better,top,view,

**Topic Sensitive PR**:

food,order,friend,great,salad,healthi,littl,back,menu,sandwich,pretti,price,tea,say,option,use,mom,reason,enjoy,friendli,coffee,fri,flavor,fruit,bread,thai,rate,server,vegan,bar,asian,feel,recommend,glad,fish,last,wish,home,much,work,total,select,excel,meal,chicken,wasn,stevia,tast,dessert,pictur

**K-means PR**:

cappuccino,better,blue,heart,atmospher,tad,shop,consist,liter,master,fresh,time,spend,drive,short,tradit,want,stinkin,someon,though,star,gaze, view,perfect,sure,nice,color,milk,make,fell,love,thing,steam,back,seat,oz,afternoon,s,hello,shini,path,espresso,beaten,nutti,bake,close,expect,fri,awesom

If we see the bag of words here we see that the test data bag of words and the influential user 1 bag of words match on food tastes like dessert, salad(marked in red) whereas using our approach the bag of words match on finer details like the atmosphere of the restaurants the user likes.(the words in blue).