Link Prediction in Facebook Networks

Homak Patel
School of Engineering and Applied Science
Ahmedabad University
AU1940042

Gaurav Bajaj
School of Engineering and Applied Science
Ahmedabad University
AU1940169

Shubham Jain
School of Engineering and Applied Science
Ahmedabad University
AU1940315

***Abstract*—In this project, we are aiming at solving two very important problems related to various popular social networks like Facebook, Twitter, etc. For our study, we have taken into account Facebook's social network. The first problem to solve is related to finding the top influential nodes in the network based on several centrality measures and aggregating their ranks together to get their single net ranks out of all the ranks measured through centrality algorithms. The nodes with top net ranks are considered the most influential nodes in the network. At a later stage, we also identified communities in the network and searched for the top influential nodes in each community. The extraction of structural data required for link classification is the fundamental bottleneck in link prediction techniques. We present a collection of simple, easy-to-compute structural features that can be evaluated to discover missing links in this project. We demonstrate that machine learning classifiers like SVM, Logistic Regression, etc trained on the suggested simple structural features (related to Social Network Analysis concepts) may correctly identify missing links even when applied to the difficult problem of classifying links between individuals who share at least one mutual friend.**

***Keywords*—** *Eigenvector Centrality, Betweenness Centrality, Degree Centrality, Closeness Centrality, Rank Aggregation, Markov Chain Rank Aggregation, Communities, Influential Nodes, Link Prediction, Jaccard's coefficient, feature extraction, SVM, Logistic Regression, ANN*

I. Introduction

1.1 Social Network Analysis

Social networks are inherently dynamic. They change quite swiftly over a short period of time. New relationships are constantly formed between nodes, and many old ones are broken. These relational modifications (when people become friends through shared friends), node characteristics, actor characteristics or link weights, and random unexplained occurrences in sequences all contribute to the graph's properties. SNA's main goals include multiple measures to rank nodes or edges, the link prediction problem, inferring social networks from social events, and so on.

Degree centrality, Proximity centrality, Clustering coefficient, Betweenness centrality, and Eigenvector centrality are some of the criteria used to rank nodes. Link prediction is the prediction of links that do not exist or do exist but are unknown and are likely to occur in the near future. Viral marketing is concerned with using users' social connectedness patterns to spread product awareness. Community detection entails graph partitioning based on social network activity and detecting the dense subgraphs in a social network.

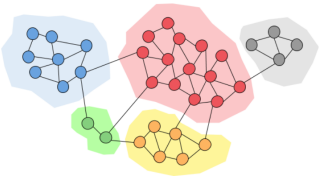

Figure 1.1.1: Community Detection in a network

1.2 Rank Aggregation

Many real-life social networks have cases of influential nodes. Influential node identification in social networks is helpful for many practical purposes. Some of these include rumor containment, viral marketing, virus spreading, information propagation, and many more. Moreover, the study and knowledge about the influential nodes in a social network are helpful for spreading information to major parts of the network which can be particularly useful for accomplishing successful advertisements. For finding the influential nodes in the network, we are employing several centrality measures like Betweenness, Closeness, Degree, and Eigenvector. Each of these will rank the nodes accordingly. So, we need a way to summarise all the rankings into a single net ranking list. This is where rank aggregation comes into play. Rank aggregation acts as a black box that outputs a combined form of all the various centrality rankings and gives a fair idea about the topmost influential nodes in the network.

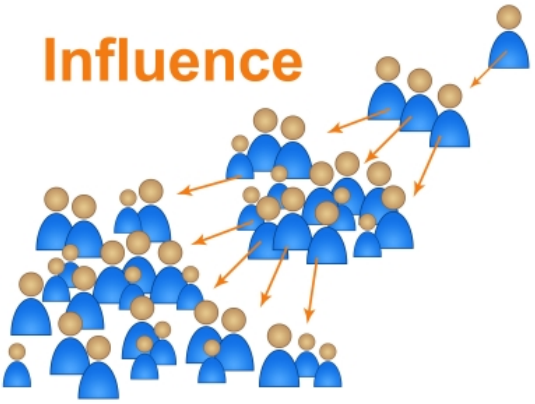

Figure 1.2.1: Influence of nodes in a social network

1.3 Link Prediction

A social network contains several types of links or edges between nodes. For instance, social interactions, phone conversations, or hyper-references. When analyzing social networks, there can be a lot of information about the connections between nodes that haven't been discovered or

aren't recognized at the time. Link Prediction is the challenge of predicting linkages that do not yet exist at the given time t or that exist but are unknown at this time. Given a snapshot of a social network (nodes and connections) at time t, we must reliably estimate the linkages that will be added to the network between time t and a future time t+n.

Figure 1.3.1: Link Prediction in Social Networks

In a variety of fields, the link prediction problem deals with the difficulty of obtaining missing links from a known network. Based on observable data, it predicts new links that are not clearly apparent now but are likely to exist in a network. It considers a static image of the network rather than network evolution and dynamics. It also takes into account individual attributes of network nodes rather than computing the power of prediction algorithms that focus on the graph topology.

1.4 Applications of Link Prediction

Aside from its function as a fundamental question in social network creation, the link prediction problem may be related to a variety of intriguing social network applications. Link Prediction has an important role in security research, owing to the difficulty of regulating terrorist networks and anticipating their future involvement. Efficient link prediction techniques in bioinformatics can be used to predict protein interactions. It aids in the development of recommendation systems in e-commerce, which aids in viral marketing and successful product awareness. In social media networks (that as Facebook), link prediction is specifically useful in recommending friends to users. These are just a few examples of applications of link prediction. However, there could be many such possible, useful, real-life applications that make link prediction important.

II. Dataset and definitions

2.1 Dataset

For our project purpose, we have selected a dataset from Snap Stanford. Our dataset is a network of the popular social media website, Facebook. The dataset is named “Social Circles: Facebook”. The dataset was obtained via recording the friend lists and circles of about 10 anonymous users in the dataset. The dataset contains separate files for egonetworks of these 10 users which contains information about edge links between alters of the ego network. There are also files that contain information about different social circles for each of these egonodes. The dataset also contains a file that shows all the possible edges in these ego networks of all 10 users. Below are some statistics related to the dataset: -

Nodes	4039
Edges	88234
Nodes in largest WCC	4039
Edges in largest WCC	88234
Nodes in strongest WCC	4039
Edges in strongest WCC	88234
Average Clustering Coefficient	0.6055

Table 2.1.1: Dataset Statistics

We have not considered separate ego-networks for our study, but the entire available network which contains all the ego-networks and the ego-nodes, as well links between different ego-networks. The nodes in the social network represent users and there exist undirected edges between 2 nodes which represents the relationship of 2 nodes being friends with each other.

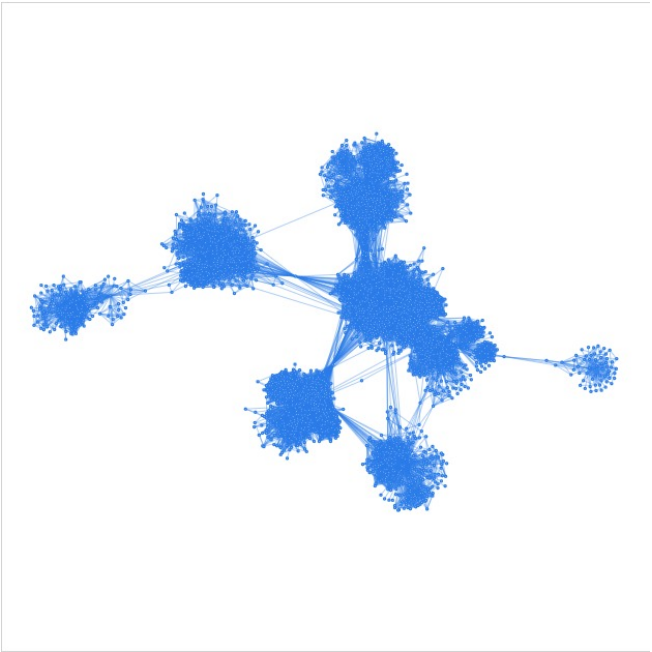

Figure 2.1.1: An overview of our Facebook dataset network

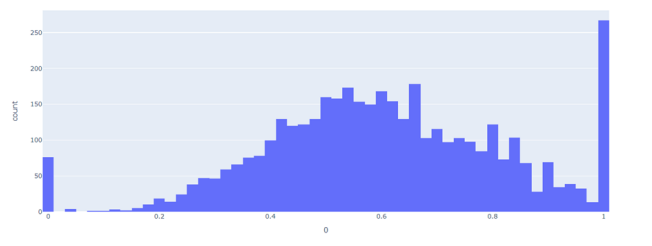

Figure 2.1.2: Clustering Coefficient Histogram

Fig. 2.1.2 depicts a histogram of the clustering coefficients evaluated over the network for each node. It shows how a majority of nodes have got a really good average clustering coefficient close to 1. This can be attributed to the fact that our network is a very tiny sub-network containing only data of about a network of 10 users (or ego-nodes) of the actual social network for Facebook. This could also be inferred as most of the alters of the egos might know each other in our small network. This could obviously vary if one considers the entire Facebook social network.

2.2 Definitions

(a) Clustering Coefficient

It's a measure to evaluate the overlap between neighbors of 2 nodes. In simple language, through the Clustering Coefficient, we aim to evaluate how many neighbors of a node are neighbors with each other.

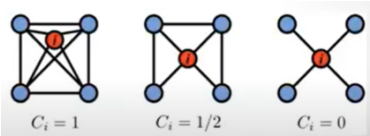

Figure 2.2.1: Clustering Coefficient Histogram

It is calculated as: -

$$C_i = \frac{2e_i}{k_i(k_i - 1)}$$

(b) Eigenvector Centrality

Eigenvector Centrality is one of the main measures of centrality for measuring the influence of a node on the network. The relationship of a node with a high-scoring node contributes more toward its eigenvector centrality score. For finding the eigenvector centrality score for vertices in the graph, we need the graph's adjacency matrix (A) and calculate the eigenvector corresponding to the largest eigenvalue.

$$\mathbf{Ax} = \lambda\mathbf{x}$$

Normally, most of the algorithms employ Power iteration for finding the eigenvector corresponding to the largest eigenvalue. The eigenvector scores for each of the vertex are normalized to obtain an absolute score for all the vertices.

(c) Betweenness Centrality

This is another important measure of the centrality of vertices in a network. It can be evaluated as the number of times a vertex lies in the shortest path between two vertices. Nodes that occur more frequently in different shortest path routes are given a higher score compared to other nodes. This measure of centrality also tells us about the nodes that act as bridges to connect different parts of the network with one another.

(d) Closeness Centrality

It is a measure of centrality which indicates the closeness of a node with respect to other nodes. It can be calculated by averaging out the shortest distance of the current node from all the other nodes in the network and reciprocating the obtained value. Closeness centrality works well in the case of connected graphs, while on the other hand, this measure of centrality suffers in terms of comparing nodes in disconnected graphs. Since our network was a fully connected network, we have taken this useful measure of centrality into account.

(e) Degree Centrality

As the name suggests, this measure of centrality is related to the degree of the node in the network, i.e., the number of edges incident on the node. Higher the degree would mean that the node is highly connected to other nodes in the network and thus, it would be assigned a high degree centrality score. This is a useful measure of centrality to consider as it is often found that nodes with high degree centrality scores also have high scores of other centralities.

(f) Common Neighbor

To predict future connections, the Common Neighbors technique calculates the intersection of the sets of neighbors of the nodes to provide a measure of similarity. It is computed as

$$CN(A, B) = |A \cap B|$$

This metric is based on the assumption that two nodes A and B have a higher probability of connecting if they share a neighbor C. This chance increases as the number of shared neighbors increases.

(g) Jaccard's Coefficient

Jaccard's coefficient is the number of features (neighbors) shared by two nodes that are proportional to the total number of features that any node has. It is a normalized version of the Common Neighbors. The Jaccard coefficient is calculated as follows:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

(h) Adamic Adar Index

It is a metric that compares the number of properties shared by two nodes. They place a higher value on products that are unique to a small group of users than on items that are shared by a large group of users. By using shared neighbors as a property, this measurement may be readily changed in the context of node neighborhood. As a result, the inverse of the logarithms of their shared neighbors' neighborhoods is

recommended as the sum of their shared neighbors. It is computed as

$$A(x,y) = \sum_{u \in N(x) \cap N(y)} \frac{1}{\log |N(u)|},$$

where N(u) is the set of nodes neighboring u.

(i) Preferential Attachment

Preferential Attachment is predicated on the assumption that a node X will acquire new neighbors faster than a node Y since Y has fewer neighbors than X. As a result, the likelihood that a node will build a new link varies with the number of its current neighbors. The number of neighbors multiplied by the likelihood of two nodes being connected by an edge based on the preferred attachment is used to calculate the likelihood of two nodes being connected by an edge based on preferential attachment. It is computed as

$$|\Gamma(x)| \cdot |\Gamma(y)|$$

(j) Resource Allocation Index

The idea behind the Research Allocation Index is that it examines a proportion of a resource, such as information or whatever else that a node can convey to another node via shared neighbors. So, let's first see how it's defined theoretically, and then we'll see what the logic behind the measure is. As a result, the Resource Allocation index of the nodes X, and Y will be the sum of one over the degree of all shared neighbors of X and Y. It is computed as

$$\text{res\_alloc}(X,Y) = \sum_{u \in N(X) \cap N(Y)} \frac{1}{|N(u)|}$$

III. Implementation for analysis

3.1 Finding Most Influential Nodes

In order to visualize and get different aspects of our problem from the network, it was important for us to first build one. The implementation of our project is done solely in Python with the help of several useful libraries like Networkx, matplotlib, plotly, etc. We first built our entire network from the dataset using the facebook_combined file. We read the edges from the file, added it one by one in the networkx graph, and generated and visualized our network.

Next, in order to look out for the most influential nodes in the network, we employed various measures of centrality - Eigenvector centrality, Betweenness Centrality, Closeness Centrality, and Degree Centrality - for each of the nodes in the network. For this, we used functions from networkx library that would fetch us normalized scores of all these centrality scores for each of the nodes in the inputted network that we built. Once we received all the different centrality scores for all the nodes, it was important for us to rank all the nodes based on their different scores. In other words, we would have to assign them integers or ranks from 0 to N (for N nodes in the network), with 0 being the highest ranks.

Now at this stage, this was a problem because all the nodes would have different ranks as per different centrality measures. It happened that for some nodes, it got a higher rank through eigenvector centrality while at the same time, betweenness centrality ranked it low compared to other nodes. In other words, we wanted to find a single, unique ranking for all the nodes in the network, and for this, it was important to perform algorithms of rank aggregation that would combine the different ranking lists of the nodes of the network into one single, unique ranking list. Through this ranking list, we can find which are the nodes that would act as the most influential nodes in the network and serve as crucial nodes for all the above-discussed applications of the problem.

There are several algorithms available that could ease our task of rank aggregation like Borda, Kemeny-Young, and Markov Chains. For rank aggregation in our project, we employed the Markov Chain type-4 algorithm. We used this approach because Markov Chains offer various benefits over other algorithms like optimized computational efficiency in terms of space and time and are also capable of handling uneven comparisons in the rankings of nodes in the network. It works as follows - We first pick a node, A, from our network, and then, we select another node B in a uniform order from the union of all rankings of the nodes obtained from different centrality measures. If for most of the ranks τ, $\tau(B) < \tau(A)$, then we move towards analyzing B. Otherwise, we remain at A and repeat the process. In this way, we move towards a higher-ranked node in the network and aggregate the ranks of all nodes in the network.

Using NetworkX greedy_modularity_communities, we also tried to find out the different communities in our network. The function works on the Clauset-Newman-Moore modularity approach that maximizes the greedy modularity of the network in the community. It begins with a node as a community and starts joining further new communities that leads way to larger and larger modularity until the total modularity approaches saturation. For each of the communities, we also evaluated which are the most influential nodes in each community.

3.2 Link Prediction using ML

There are many different methods available for link prediction which are based on Topological/Network Structure, Content/Semantic, or Individual nodes attribute approach, and Joint/Conditional Probabilistic Approach. But here we have followed the Topological/Network Structure Approach and so we have selected 5 features which are basically the node neighbor algorithms. We have selected only these 5 features because these features were related to Social Network Analysis concepts and our objective was to learn how these algorithms are used in link prediction.

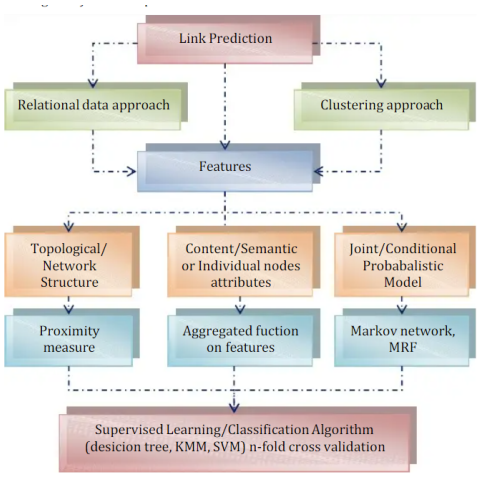

Figure 3.2.1 - Different Approaches for Link Prediction

A. Strategy to Solve Link Prediction Problems

Let's look at a dummy graph to better comprehend this concept. A 7-node graph is shown below at some time t, with the disconnected node-pairs AF, BD, BE, BG, and EG:

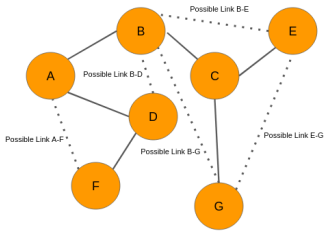

Figure 3.2.2- Graph at time t

Let's imagine we analyzed the data and came up with the graph below. A few new links (in red) have been established after some time t+n:

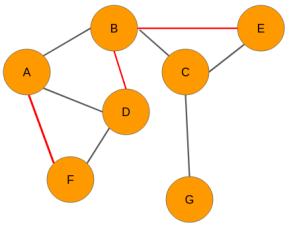

Figure 3.2.3 - Graph at time t+n

Our goal is to anticipate whether a link exists between any two disconnected nodes. We can remove the following node pairs with no linkages between them from the network at time t:

a. A-F
b. B-D
c. B-E
d. B-G
e. E-G

The next stage for us is to design features for each pair of nodes. The good news is that there are various approaches for extracting features from network nodes. Assume we utilize one of these strategies to create features for each of these couples. Examine the graph at time t+n. We can observe that the network has three new linkages for the pairs A-F, B-D, and BE, respectively. As a result, we'll give each of them a value of 1. Because there are no linkages between the nodes, the node pairings B-G and E-G will be assigned a value of 0.As a result, the data will look like this:

Features	Link (Target Variable)
Features of A-F pair	1
Features of B-D pair	1
Features of B-E pair	1
Features of B-G pair	0
Features of E-G pair	0

Now that we've identified the target variable, we can use it to train a machine learning model to predict links.So, in order to extract the target variable, namely the presence of a connection between a node pair, we must employ social graphs at two separate points in time. However, keep in mind that in real-world settings, we will only have data from the current moment.

B. Approach For Link Prediction

Step-1: Building the Graph

The graph containing nodes and edges of all the ego networks of the dataset was taken into account for building the main graph. The graph was built using the NetworkX library.

Step-2: Extracting the features

a. Jaccard's Coefficient
b. Common Neighbour
c. Adamic/Adar
d. Preferential Attachment
e. Resource Allocation Index

The above-mentioned link prediction algorithms were taken as features for ML models. The details about these link prediction algorithms are given in section-2.2. A feature in

machine learning is individual measurable property. Choosing different, independent, and informative features is considered an important step for a machine learning model. These features were selected as they are quite different in way of prediction of links and they are specifically designed for the prediction of links in graphs. Certain other features can also be taken into account for the ML models that we have not used.As the main aim was to get an insight into these link prediction algorithms, so we included only them as features.

Step-3: Model Training and Testing

Three classical machine learning models were used for the implementation i.e. SVM (Support Vector Machine), Logistic Regression, and ANN (Artificial Neural Network). The dataset was split into 4:1 ratio for preparing testing and training data sets i.e. training data was taken as 80% of the whole dataset whereas testing data was taken as 20% of the whole dataset.

3.3 Link Prediction using general link prediction algorithms

a. Jaccard's Coefficient
b. Common Neighbour Centrality
c. Adamic Adar Index
d. Preferential Attachment
e. Resource Allocation Index

Following general link prediction approaches for social networks were used also implemented for link prediction using the NetworkX library. A detailed explanation of these algorithms is given in section-2.2.

The NetworkX python library functions used return u,v, and p, where u and v are the pair of nodes and p is the probability of link formation between two nodes. All the pairs of nodes (u,v) apart from the ones that already have an edge between them. As the no of edge prediction for a graph with 4k+ nodes can be huge we have taken only one or two ego networks for link prediction using these approaches. A threshold value is also set so that nodes with a probability greater than the given threshold value are only taken into consideration for future link formation.

The results obtained by setting various threshold values on these link prediction algorithms are given in section-4.3.

IV. RESULTS AND INFERENCES

4.1 Process of Finding Top Influential Nodes in the Network

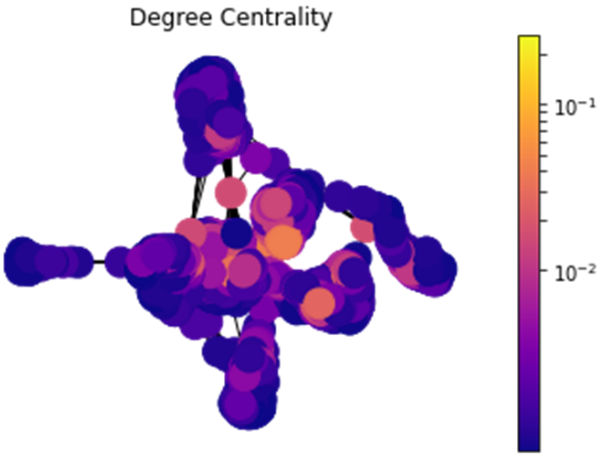

Figure 4.1.1 - Degree Centrality Visualization

Plots were made with the help of NetworkX and matplotlib library. We adopted spring layout for all the visualizations. We plotted the above graph to observe different nodes in terms of their degree centrality. Nodes with high degree centrality are colored bright yellow, while nodes with low degree centrality are colored in dark blue. We have kept the same scale for coloring of nodes based on their centrality scores for plots of all the 4 centrality scores. We can observe from the above plot that most of the nodes are with low degree centrality which indicates that most nodes have got few neighbours. Now this can be true for alter nodes as we do not have complete information about friend lists for nodes except egonodes. Egonodes are the ones that should have high degree centrality and they are the ones that are colored in yellow (which are also few in number).

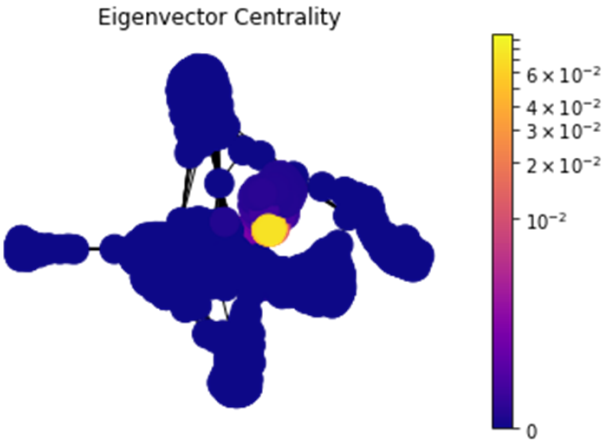

Figure 4.1.2 - Eigenvector Centrality Visualization

The above plot depicts various nodes along with their respective eigenvector centrality scores. It is very surprising to observe that only few of the 4k+ nodes have high score of eigenvector centrality, while majority of the nodes lie in the blue region suggesting that they have low scores of eigenvector centrality. While ranking nodes based on eigenvector centrality, we have to assign same ranks to many nodes in the network because they had very close eigenvector centrality score compared to each other.

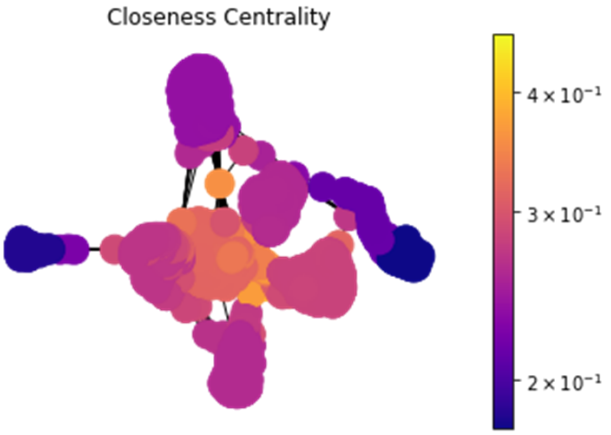

Figure 4.1.3 - Closeness Centrality Visualization

The above graph depicts closeness centrality of different nodes in the network. It can be observed that most of the nodes in the network have moderate closeness centrality, with a few having large scores and few having below average closeness centrality. Majority of the nodes with large closeness centrality can be fairly assumed to be ego-nodes, as they are the ones that are connected with most of the other nodes (alter nodes) in the network.

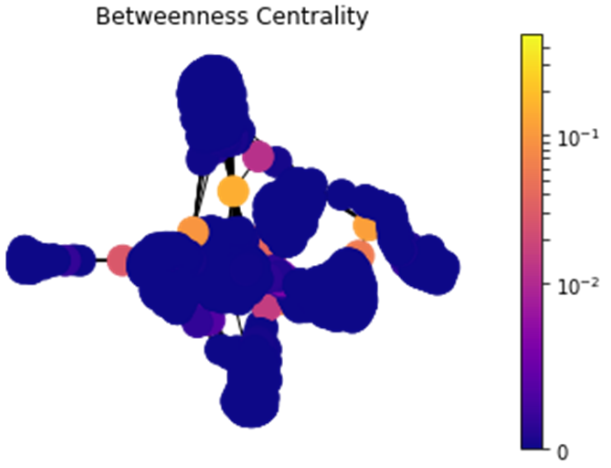

Figure 4.1.4 - Betweenness Centrality Visualization

Above plot depicts the levels of betweenness centrality across the nodes of the network. Most of the nodes have pretty low betweenness centrality, inferring that they are a part of very few shortest paths between two different nodes in the network. At the same time, we can also observe that there are a few nodes that have relatively high betweenness centrality (colored in yellow) in the network.

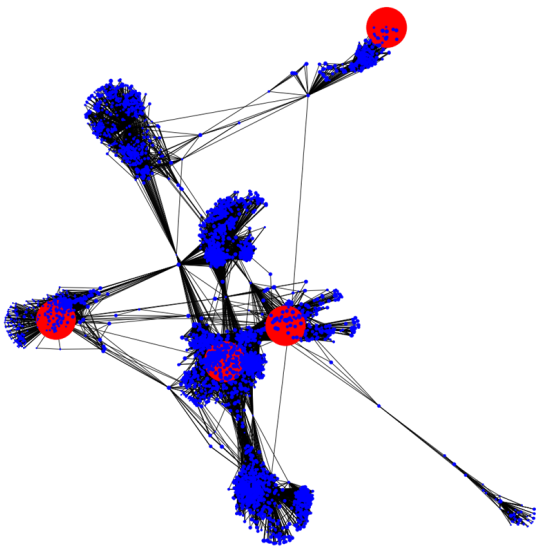

Figure 4.1.5 - Top 4 Influential Nodes in the Network

In the above plot, we have depticted the top 4 most influential nodes that we got after aggregating all the different centrality score-based ranks for each node using Markov Chain Type-4. These influential nodes are colored in red and have enlarged sizes compared to other nodes. These are the top 4 nodes that are central to the network are are really useful for different applications like viral marketing, rumour containment, information propagation, advertising, etc. But at the same time, if we look carefully, there are parts of network that go missing or unreached or untouched by these top 4 influential nodes. Thus, it becomes important to take into account more influential nodes for those applications.

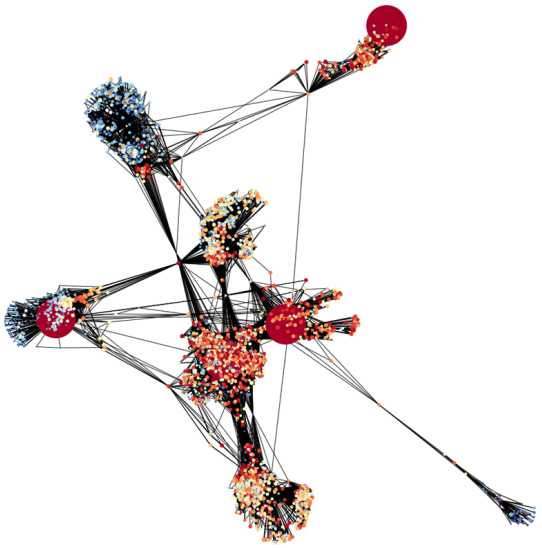

Figure 4.1.6 - Node Rankings with top 4 influential nodes Visualization

The above plot depicts ranks of different nodes after rank aggregation by different shades of red, blue and yellow, with

red being the highest rank, followed by blue and yellow.

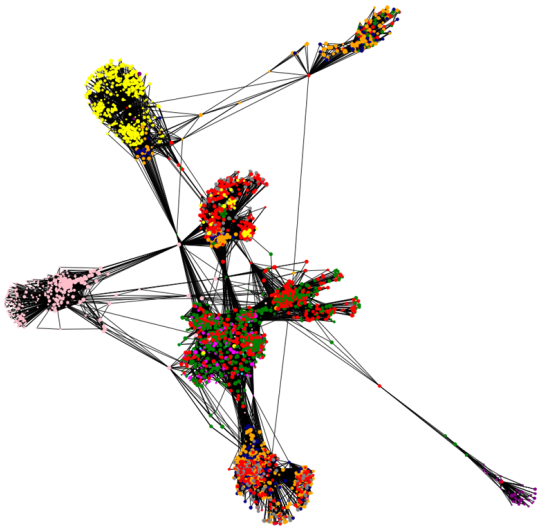

Figure 4.1.7 - Detected communities in the network

The algorithm detected about 13 different communities in the network. This is because for each ego-network in the dataset, there were many social circles available. Social circles are like a group of people who are co-related with each other from the perspective of the ego node. For instance - a person could have a social circle of school friends, another for college friends and another for professional friends in hist ego-network. All these friends are a part of his/her social network, but are segregated into different social circles. All the detected social circles are colored differently for easy visualization. It would be interesting to know an influencer for each of the community.

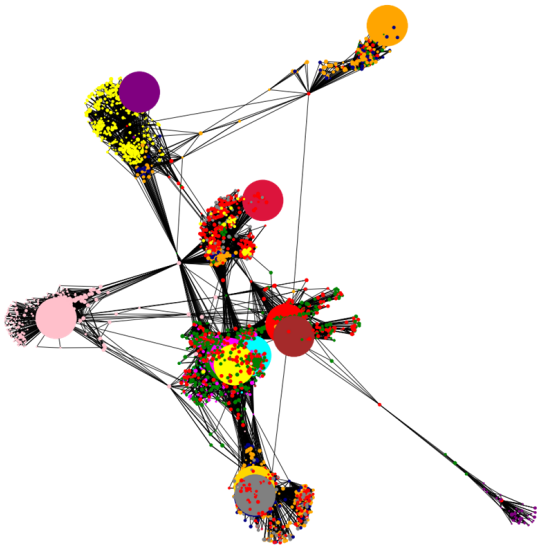

Figure 4.1.8 - Top influential node per community

The above plot depicts influential nodes per community detected. The top influential node corresponding to each community is given an enlarged size and the same color as that of the community, It was interesting to know that the maximum of the ranks amongst all these influential nodes was about 3384 while the total nodes present in the network are close to 4000. This also shows that there are several communities in the network that have multiple top influential nodes present. Hence, in order for information to propagate through every community effectively and directly, we have to look at some of the low-ranked nodes as well.

4.2 Link Prediction using ML models

```
Accuracy of SVM Model: 0.640661938534279
```

```
Accuracy of Logistics Regression Model: 0.6595744680851063
```

```
ANN Model Accuracy: 0.6572104018912529
/usr/local/lib/python3.7/dist-packages/sklearn/neural_network/
  ConvergenceWarning,
```

Figure 4.2.1- Accuracy Results obtained from ML models (SVM, LR, ANN)

The above screenshots depict the results obtained from the link prediction using machine learning. SVM model gave an accuracy of 64.06%, the Logistic Regression Model gave an accuracy of 65.95% and ANN Model gave an accuracy of 65.72%.

4.3 Link Prediction using general link prediction algorithms

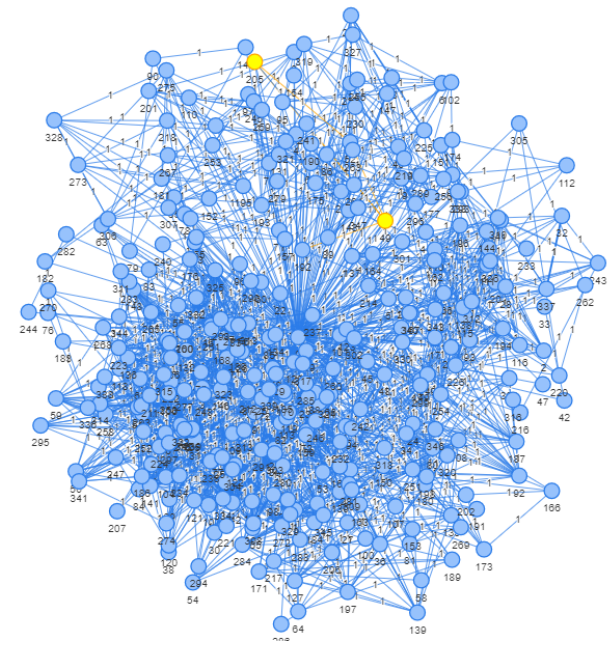

Figure 4.3.1- Link Prediction using JaccardCoefficient for EgoNetwork of node-0 at threshold value=0.8

The above graph describes the link prediction using Jaccard Co-efficient on the ego network graph of node-0. The threshold value was set as 0.8 in this case i.e. nodes having a

Jaccard co-efficient probability greater than or equal to 0.8 were taken into consideration. The total link predicted in this scenario was only one. The nodes and edges that were predicted by the algorithm are depicted in the color yellow in fig-4.3.1.

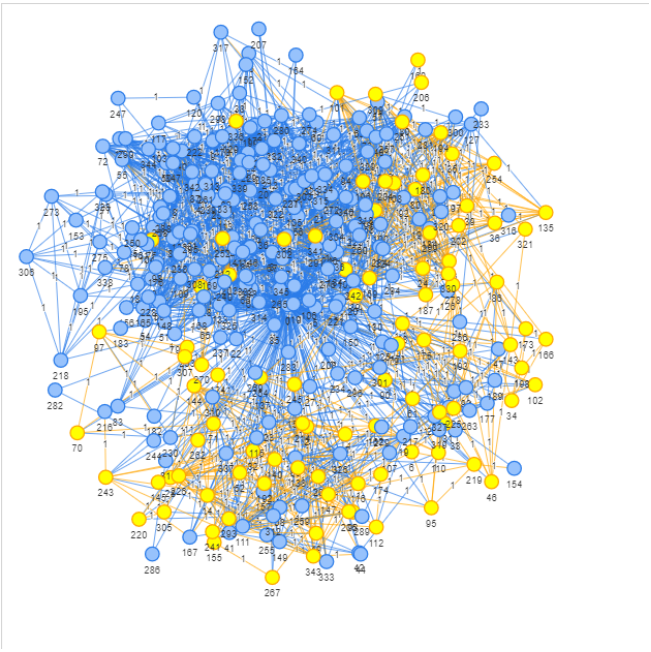

Figure 4.3.2- Link Prediction using JaccardCoefficient for EgoNetwork of node-0 at threshold value=0.5

The above graph describes the link prediction using Jaccard Co-efficient on the ego network graph of node-0. The threshold value was set as 0.5 in this case i.e. nodes having a Jaccard co-efficient probability greater than or equal to 0.5 were taken into consideration. The total link predicted in this scenario was 94. The nodes and edges that were predicted by the algorithm are depicted in the color yellow in fig-4.3.2.

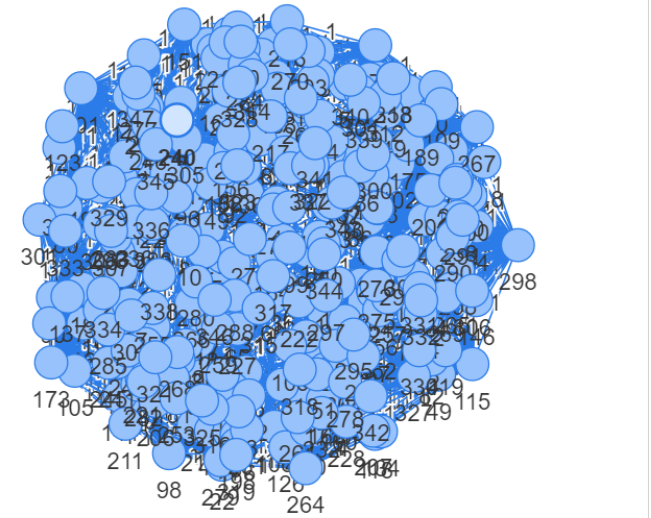

Figure 4.3.3- Link Prediction using other link prediction algorithms (Common Neighbor Centrlity, Adamic Adar Index, Preferential Attachment and Resource Allocation Index) at threshold value=0.5 and 0.8

The other link prediction algorithms i.e. Common neighbor Centrality, Adamic Adar Index, Preferential Attachment, and Resource allocation index were not able to predict link formation at a threshold value of 0.5 and 0.8 i.e. algorithms were not able to suggest or predict links having the probability equal to or greater than 0.5 and 0.8. The results of these link prediction algorithms can be descibed by the figure 4.3.3 where no yellow edges i.e. no link prediction is seen.

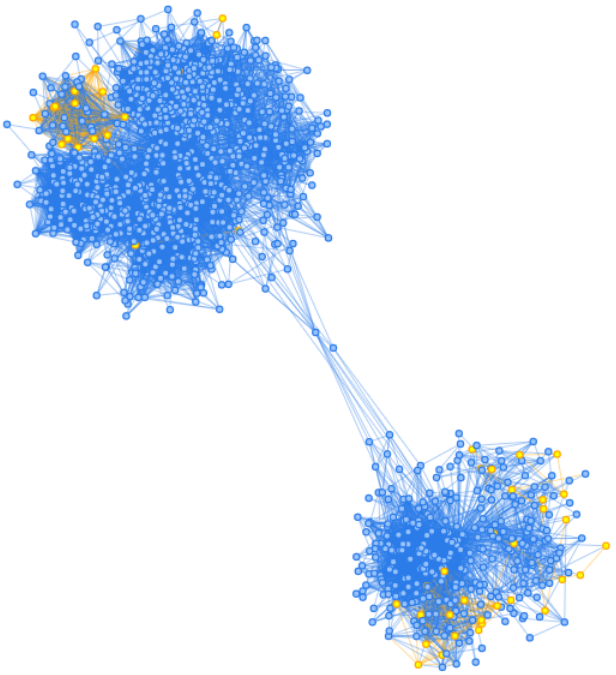

Figure 4.3.4- Link Prediction using JaccardCoefficient for graph made from the ego networks of node-0 ad node-107 at threshold value=0.7

As the Jaccard Coefficient performed well in link prediction for ego-network of node-0. We used Jaccard coefficeint to predict link formation for graph made form ego network sof node-0 and node-107.. The threshold value was set as 0.7 in this case i.e. nodes having a Jaccard co-efficient probability greater than or equal to 0.7 were taken into consideration. The total links predicted in this scenario were 29. The nodes and edges that were predicted by the algorithm are depicted in the color yellow in fig-4.3.4.

VI. Conclusion

Through the implementation and analysis of rank aggregation for finding influential nodes in the network, we got to know that most of the highly ranked influential nodes belong to same community, which calls for low ranked nodes to be taken into consideration if we want to influence every single community in the network. At the same time, we also think that we can improve our system and approach by taking into account other centrality measures like Katz centrality, etc. More the ranking lists of various centrality for each node, more is the optimality and credibility of aggregated ranks and we think that we can achieve better

results through this. Moreover, we also think that these few of these centrality scores ignore the topology of neighboring nodes which can, in turn, affect their centrality measures and hence, the overall ranking.

For link prediction we used two appoaches i.e. ML based approach and general link prediction algorithms. For ML based approach, we used SVM, ANN and LR models. However, the accuracy obtained was appox. 65% in all the models. This accuracy can be increased by taking into account more features such as other link prediction algorithms or certain other social aspects. In general link-prediction algorithms approach, it was seen that Jaccard Co-efficent link prediction algorithm performed well than the other link-prediction algorithms i.e. Common Neighbor Centrality, Adamic Adar Index, Preferential Attachment and Resource Allocation Index for a single ego-network graph.

VII. Acknowledgment

This project was a part of the course CSE533 Social Network Analysis. We would like to acknowledge all those without whom this project would not have been successful. Firstly, we would wish to thank our course instructors Professor Amit Nanavati & Professor Sougata Mukherjea who guided us throughout the project and gave their immense support. They made us understand how to successfully complete this project and without them, the project would not have been complete. This project has been a source to learn and bring our theoretical knowledge to the practical, real-life world. Thus, we would like to acknowledge their help and guidance for this project.

References

[1] Wang, T., & Liao, G. (2014, October). A review of link prediction in social networks. In 2014 International Conference on Management of e-Commerce and e-Government (pp. 147-150). IEEE.

[2] Shahmohammadi, A., Khadangi, E., & Bagheri, A. (2016). Presenting new collaborative link prediction methods for activity recommendation in Facebook. Neurocomputing, 210, 217-226.

[3] Fire, M., Tenenboim, L., Lesser, O., Puzis, R., Rokach, L., & Elovici, Y. (2011, October). Link prediction in social networks using computationally efficient topological features. In 2011 IEEE third international conference on privacy, security, risk and trust and 2011 IEEE third international conference on social computing (pp. 73-80). IEEE.

[4] Gao, F., Musial, K., Cooper, C., & Tsoka, S. (2015, June 16). *Link prediction methods and their accuracy for different social networks and network metrics*. Scientific Programming. Retrieved May 2, 2022, from https://www.hindawi.com/journals/sp/2015/172879/

[5] Link prediction: Link prediction in Social Networks. Analytics Vidhya. (2020, April 21). Retrieved May 2, 2022, from https://www.analyticsvidhya.com/blog/2020/01/link-prediction-how-to-predict-your-future-connections-on-facebook/

[6] Jon Kleinberg. Jon Kleinberg's Homepage. (n.d.). Retrieved May 2, 2022, from https://www.cs.cornell.edu/home/kleinber/

[7] Volkova, S. (2014, May 23). CIS 830 final project link prediction in Social Networks Part 1. Academia.edu. Retrieved May 2, 2022, from https://www.academia.edu/2992506/cis_830_final_project_link_prediction_in_social_networks_part_1

[8] Social Networks Analysis : Link Prediction - RCCIIT. (n.d.). Retrieved May 2, 2022, from https://rcciit.org.in/students_projects/projects/cse/2018/GR16.pdf