Friend Recommendation based on Pagerank Scores and other features in Facebook Social Network (DSA5101 Project)

Zhong Zhaoxiang

Abstract

In our project, we construct a Facebook social network, apply PageRank algorithm to the nodes and visualize the network graph. Then we build two neural network models for general and personalized friend recommendation, respectively, and analyze the testing results.

Keywords:
Pagerank, Recommendation System, Neural Network

1. Introduction

Friend recommendation is a popular topic in social networking platforms, such as Facebook, X (Twitter) and Instagram. This specific genre of recommendation system is for suggesting friends to users based on their features and feedback from social network [1]. In social network, there are usually two kinds of links, directed and undirected. In directed graphs, the in-link represents someone "follows" or "subscribes" another user and vice versa. In undirected graphs, links are both-way which means the linked pair of users are "friends" in the platform. In our work, we analyze the undirected social network datasets of Facebook which are accessible in Stanford SNAP.

In social platforms, the primary recommendation task of server is to identify potential missing links (between users or pages). Thus, friend recommendation can also be seen as a link prediction task based on both topological and non-topological features. Numerous machine learning methods have been employed by researchers to extract these features and train predictive models [2, 3]. For our project, we will be using neural network to make predictions.

The remainder of this paper is organized as follows: In Section 2, we load the Facebook datasets and construct the Facebook-graph. We then present

descriptive analysis of the data, including information of ego-users and their social circles. Moreover, we calculate the PageRank scores of nodes in the Facebook-graph and visualize the graph. In Section 3, we introduce the general and personalized friend recommendation systems. After defining the predictive features and generating samples, we build two binary classification neural network models using different training and testing datasets.Then we analyze the testing results and test the model for application. Lastly, we draw the conclusions in Section 4.

2. About the datasets

2.1. Descriptive analysis

The datasets of our project are from Stanford SNAP[4], which contain ten nodeId.edges files, ten nodeId.circles files, ten nodeId.feat files, ten nodeId.egofeat files, ten nodeId.featnames files and one Facebook combined.txt file. Facebook combined.txt contains all the nodes and edges in our network datasets. Each pair of nodeId.edges and nodeId.circles files construct an ego network with different social circles. And each combination of nodeId.feats, nodeId.egofeat files contain the feature vector of each user in the corresponding ego network and ego-user himself. The anonymized feature names could be found in nodeId.featnames file. Next, we read all the files and load the datasets. After combining all the edges files along with the combined.txt, we construct the Facebook-graph with 4039 nodes and 88234 edges and save all the information of circles and feats in dictionaries.

We present all the descriptive information of our datasets in Table 1. For each ego-user, we denote the dimension of feature vectors in corresponding .feats file(equal to the number of featnames in the corresponding .featnames file), the total number of members in all of their social circles, the number of users in corresponding .feats file and the number of their "friends" in Facebook-graph by $D_{ego-user}$, $S_{ego-user}$, $U_{ego-user}$ and $F_{ego-user}$.

From the statistics above, we find that there are non-ego-users that don't belong to any social circle of any ego-user. And the feature vectors in the same nodeId.feats file share the same dimension but the dimension varies in different files. This could be seen from different nodeId.featnames files as well.

Table 1: Descriptive analysis

egouser	$D_{ego-user}$	$S_{ego-user}$	$U_{ego-user}$	$F_{ego-user}$
0	224	286	347	347
107	576	481	1045	1045
1684	319	769	792	792
1912	480	710	755	755
3437	262	97	547	547
348	161	220	227	229
3980	42	58	59	59
414	105	139	1159	159
686	63	170	170	170
698	48	54	66	68

2.2. Pagerank Scores

After constructing the Facebook-graph, we compute Pagerank socres for the nodes in the graph. In undirected graphs, the link can be considered as one in-link and one out-link between nodes. And the nodes with largest Pagerank socres usually tend to be the most popular users. And popularity of users can be a important feature in friend recommendation since popularity implies credibility. We will use Pagerank scores in our friend recommendation model in the next section. And Figure 1 shows the top 5 pagerank scores and corresponding nodes in our Facebook-graph.

```
Top 5 Pagerank scores and corresponding nodes :
3437: PageRank score = 0.007615
107: PageRank score = 0.006936
1684: PageRank score = 0.006367
0: PageRank score = 0.006290
1912: PageRank score = 0.003877
```

Figure 1: Pagerank Scores

2.3. Visualization

We constructed Facebook-graph store all the social circles. Next, we visualized the graph along with the social circles, using different colors to

represent nodes in different social circles and setting the node sizes positively correlated to their PageRank scores. And we present the result in Figure 2.

We can see from Figure 2 that user nodes are scattered around the social network and there are nodes that don't belong to any social circles (represented in black).

3. Friend recommendation systems

So far, we have loaded the datasets. To train predictive models, it is necessary to extract relevant features from these datasets. M. Al Hasan[3] utilizes both topological features, such as shortest path distance and clustering coefficient, as well as non-topological features, such as sum of neighbors(Pagerank Socres) and the number of shared features. We consider two friend recommendation systems: a general one, and a personalized one where we employ personalized Pagerank score with ego-users as the teleport set for recommendation.

3.1. General friend recommendation system

Firstly, we employ Pagerank score, clustering index and mutual friends as three features in our general friend recommendation system.

- **Pagerank score** The Pagerank score feature of users are simply their Pagerank scores in the Facebook-graph.
- **Clustering index** For users, the clustering index describes how dense their neighborhood is. A user with larger clustering index is more likely to grow more edges[3].

  $$\text{clustering index of u} = \frac{3 \times \text{number of triangles with } u \text{ as one node}}{\text{number of connected triples with } u \text{ as one node}} \tag{1}$$

  The clustering index can be computed by function nx.clustering.
- **Mutual friends** In social networks, users with a larger intersection of their friend sets are more likely to become friends.Thus, for mutual friends feature, we use the Jaccard similarity between user's neighborhood and recommendation candidate's.

  $$\text{mutual friends between A and B} = \frac{|Neigborhood_A \bigcap Neigborhood_B|}{|Neigborhood_A \bigcup Neigborhood_B|} \tag{2}$$

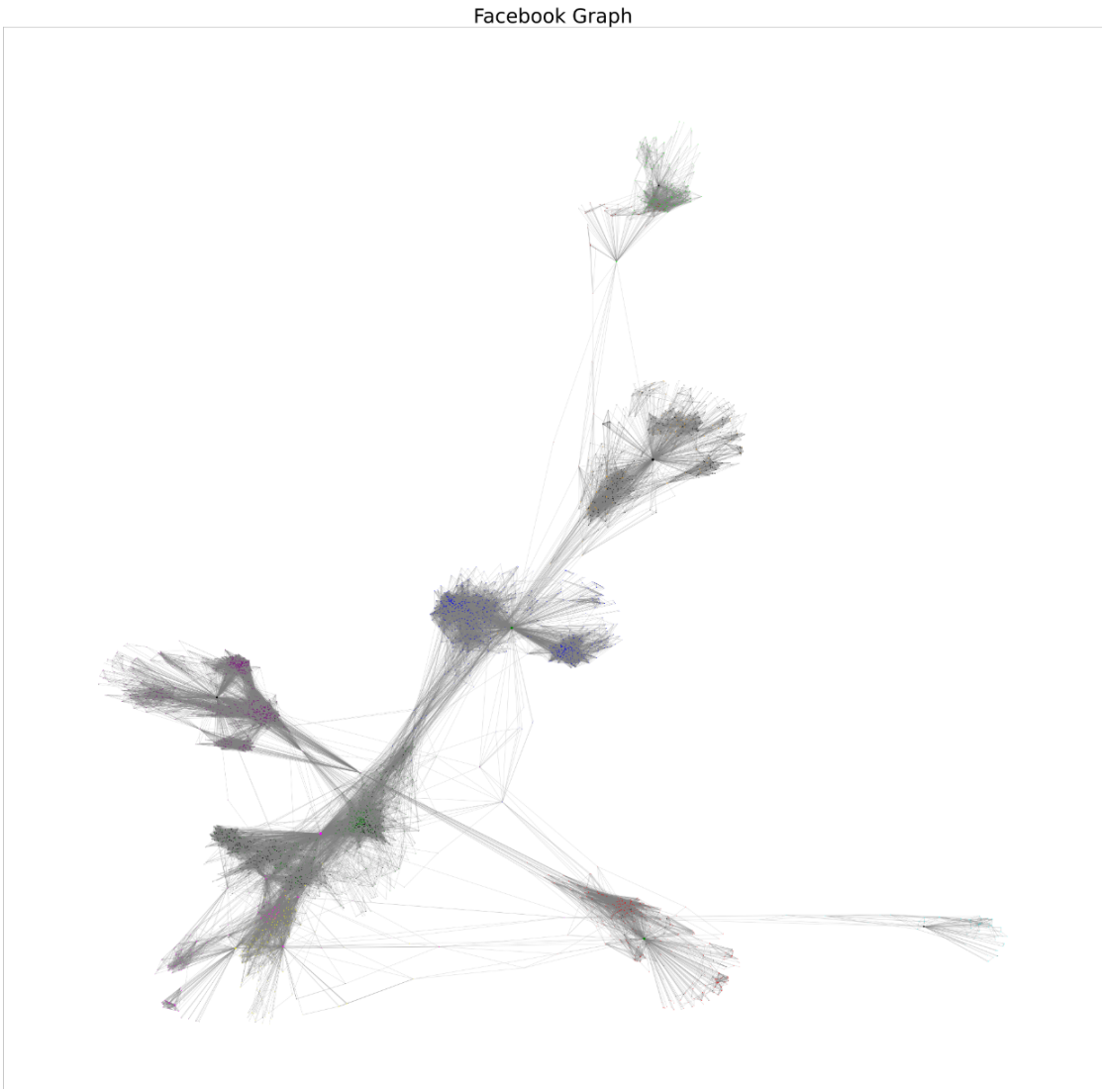
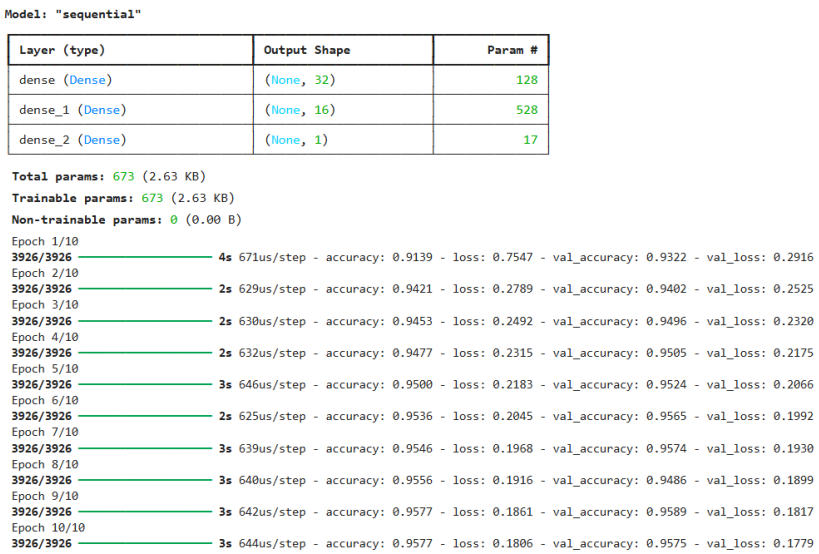


Figure 2: Visualization

We define functions to extract these features from datasets and then generate the samples. In practice, user's feedback on recommendations is necessary for generating positive and negative samples. However, such data are not available in our case. Therefore, we assume that, for given users, 50% of their friends come from recommendations they accepted, and that they also reject an equal number of recommendations as they accept.

After generating the samples, we build a binary classification neural network model, train and test it on the training set and test set respectively. We present the model structure and testing results in Figure 3 and 4.



Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 32)	128
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 1)	17

Total params: 673 (2.63 KB)
Trainable params: 673 (2.63 KB)
Non-trainable params: 0 (0.00 B)

```
Epoch 1/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 4s 671us/step - accuracy: 0.9139 - loss: 0.7547 - val_accuracy: 0.9322 - val_loss: 0.2916
Epoch 2/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 2s 629us/step - accuracy: 0.9421 - loss: 0.2789 - val_accuracy: 0.9402 - val_loss: 0.2525
Epoch 3/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 2s 630us/step - accuracy: 0.9453 - loss: 0.2492 - val_accuracy: 0.9496 - val_loss: 0.2320
Epoch 4/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 2s 632us/step - accuracy: 0.9477 - loss: 0.2315 - val_accuracy: 0.9505 - val_loss: 0.2175
Epoch 5/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 3s 646us/step - accuracy: 0.9500 - loss: 0.2183 - val_accuracy: 0.9524 - val_loss: 0.2066
Epoch 6/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 2s 625us/step - accuracy: 0.9536 - loss: 0.2045 - val_accuracy: 0.9565 - val_loss: 0.1992
Epoch 7/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 3s 639us/step - accuracy: 0.9546 - loss: 0.1968 - val_accuracy: 0.9574 - val_loss: 0.1930
Epoch 8/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 3s 640us/step - accuracy: 0.9556 - loss: 0.1916 - val_accuracy: 0.9486 - val_loss: 0.1899
Epoch 9/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 3s 642us/step - accuracy: 0.9577 - loss: 0.1861 - val_accuracy: 0.9589 - val_loss: 0.1817
Epoch 10/10
3926/3926 ━━━━━━━━━━━━━━━━━━━━ 3s 644us/step - accuracy: 0.9577 - loss: 0.1806 - val_accuracy: 0.9575 - val_loss: 0.1779
```

Figure 3: Model 1

We observe from Figure 4 that our model is performing effectively with high precision, recall, and F1 scores. The confusion matrix and high AUC also indicate that the model can distinguish between friends and non-friends. Hence, it's a promising candidate model for friend recommendation. However, there is a potential risk of over-fitting, we have mitigated this risk by employing L2 regularizer with coefficient $\lambda = 0.1$ on both hidden layers. Furthermore, it is important to note that we randomly sampled 50% of friend pairs along with an equal number of non-friend pairs, assuming these pairs represent the recommendation outcomes. As such, the model functions more as a binary classifier between friends and non-friends. To improve the model, we require data derived from user's feedback.

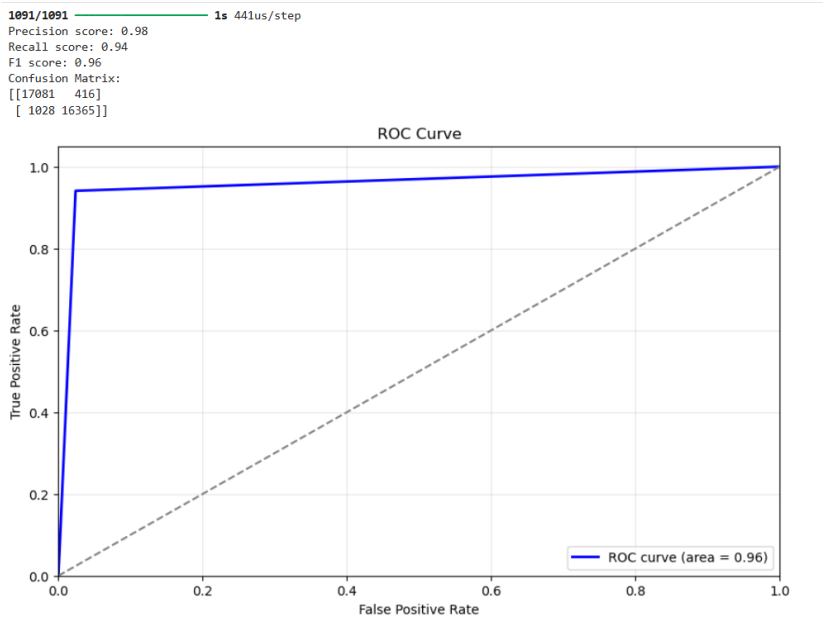


Figure 4: Testing results 1

During the sample generation process, we also produce feature values of users that are neither user 0's friends nor users that have already been rejected by user 0 as candidate samples. For application test, we apply Model 1 to these candidates and output the top 5 recommendations for user 0 which are represented in Figure 5.

```
110/110 ━━━━━━━━━━━━━━━━━━━━ 0s 471us/step
Candidate User 3437 with score: 0.97
Candidate User 1684 with score: 0.96
Candidate User 1912 with score: 0.88
Candidate User 3980 with score: 0.56
Candidate User 686 with score: 0.56
```

Figure 5: Recommendation for user 0

3.2. Personalized friend recommendation system

In Section 3.1, all users in the Facebook-graph can serve as candidates for friend recommendations to others. However, some users may be reluctant to receive an overwhelming number of recommendations or have themselves recommended widely across the network. In practice, Facebook allows users to adjust privacy settings to receive personalized recommendations, such as limiting recommendations to those who share at least one mutual friend. Therefore, we shift our focus to a personalized friend recommendation system. Additionally, we introduce the number of shared features between two users as a new feature for prediction.

For simplicity, and due to the varying dimensions of feature vectors across different ego.feats files, we only consider users who are mutual friends of the same ego-user as users or candidates for recommendation. We also observed that neighborhoods of different ego nodes have intersections, and not all neighbors of an ego node appear in the corresponding ego.feats file. Therefore, we exclude nodes that are neighbors to multiple ego nodes and nodes that are friends of a ego-user but are not listed in the corresponding ego.feats file.

After processing the data, we can count number of shared features between user and candidate now. For other 3 features, we construct a graph of the neighbors of ego nodes(after exclusion) and denote it by ego-user-graph. We calculate clustering index and mutual friends with the same method before, except in the ego-user-graph instead of Facebook-graph. For Pagerank score, we employ personalized Pagerank with the corresponding ego-user as teleport set. So far, we have ego-specific Pagerank score, clustering index, mutual friends and shared features(number of shared features) as four features for personalized recommendation.

With features defined, we extract them from datasets and then generate the samples just like we did in Section 3.1. Given the high computational cost of generating samples in all ego-user-graphs, we only choose the 5 ego-user-graphs with the fewest nodes (698, 686, 414, 348, 0) for sample generation. After generating the samples, we build a binary classification neural network model, train and test it on the training set and test set respectively. The model structure and testing results are shown in Figure 6 and 7.

We can see from Figure 7 that our model performs well, with decent precision, recall and F1 scores. The confusion matrix and AUC indicate a good ability to distinguish between positive and negative classes. However, the model is not as strong as Model 1 in terms of testing results. This is

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 16)	80
dense_4 (Dense)	(None, 8)	136
dense_5 (Dense)	(None, 1)	9

Total params: 225 (900.00 B)

Trainable params: 225 (900.00 B)

Non-trainable params: 0 (0.00 B)

```
Epoch 1/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.6937 - loss: 0.5659 - val_accuracy: 0.8493 - val_loss: 0.3638
Epoch 2/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 681us/step - accuracy: 0.8618 - loss: 0.3372 - val_accuracy: 0.8570 - val_loss: 0.3284
Epoch 3/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 709us/step - accuracy: 0.8643 - loss: 0.3209 - val_accuracy: 0.8551 - val_loss: 0.3233
Epoch 4/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 707us/step - accuracy: 0.8728 - loss: 0.3084 - val_accuracy: 0.8560 - val_loss: 0.3250
Epoch 5/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 698us/step - accuracy: 0.8726 - loss: 0.2994 - val_accuracy: 0.8589 - val_loss: 0.3177
Epoch 6/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 678us/step - accuracy: 0.8687 - loss: 0.3067 - val_accuracy: 0.8570 - val_loss: 0.3204
Epoch 7/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 690us/step - accuracy: 0.8750 - loss: 0.2946 - val_accuracy: 0.8589 - val_loss: 0.3163
Epoch 8/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 679us/step - accuracy: 0.8687 - loss: 0.2980 - val_accuracy: 0.8551 - val_loss: 0.3173
Epoch 9/10
293/293 ━━━━━━━━━━━━━━━━━━━━ 0s 697us/step - accuracy: 0.8800 - loss: 0.2890 - val_accuracy: 0.8599 - val_loss: 0.3145
Epoch 10/10
```

Figure 6: Model 2

```
82/82 ━━━━━━━━━━━━━━━━━━━━ 0s 748us/step
Precision score: 0.87
Recall score: 0.88
F1 score: 0.88
Confusion Matrix:
[[1171  161]
 [ 149 1122]]
```

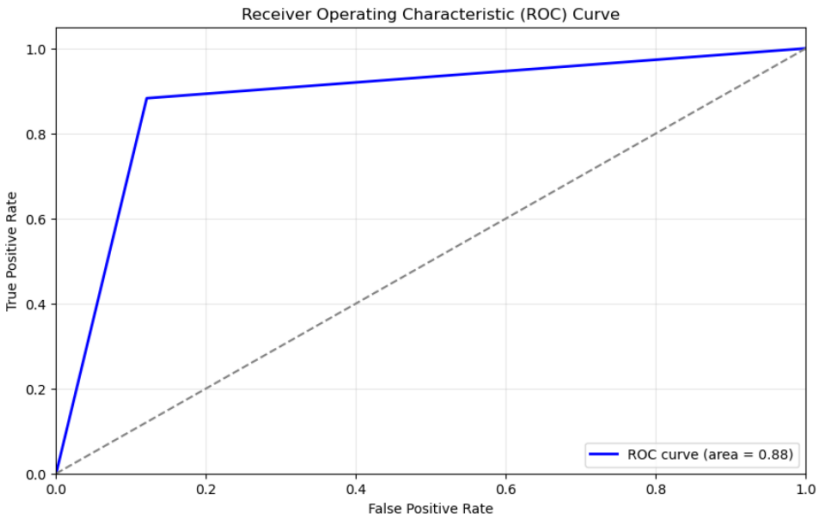


Figure 7: Testing results 2

probably because the training/testing samples for Model 1 are generated from the entire Facebook-graph, while the samples for Model 2 are generated from different ego-user graphs, leading to biases and inconsistency in the sample sets.

Similarly, for application test, we generate candidate sets from non-friends of user 3981 within ego-user 3980's ego-user-graph. Then we apply Model 2 to these candidates and output the top 5 recommendations for user 3981, as presented in Figure 8.

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
Candidate User 3981 with score: 1.00
Candidate User 4030 with score: 0.98
Candidate User 3982 with score: 0.97
Candidate User 4002 with score: 0.92
Candidate User 4014 with score: 0.87
```

Figure 8: Testing results 2

4. Conclusion

In this project, we constructed a social network graph called "Facebook-graph." We applied the PageRank algorithm to the nodes, and employed the resulting PageRank scores to the friend recommendation system. We extracted several predictive features from both the graph and the datasets for recommendation. Subsequently, we developed general and personalized friend recommendation systems, building two neural network models for each system. Finally, we analyzed the testing results of both models. The findings indicate that both models perform effectively. This suggests their potential to provide good friend recommendations for users. However, data obtained from user's feedback on the recommendations are required for better recommendation and model improvement in practice.

References

[1] Adamic, L. A., & Adar, E. (2003). Friends and neighbors on the Web. *Social Networks*, 25(3), 211-230. https://doi.org/10.1016/S0378-8733(03)00009-1

[2] Parveen, R., & Varma, N. S. (2021). Friend's recommendation on social media using different algorithms of machine learning. *Global Transitions Proceedings*, 2(2), 273-281. https://doi.org/10.1016/j.gltp.2021.08.012

[3] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *SDM06: Workshop on Link Analysis, Counter-Terrorism and Security*, vol. 30, pp. 798-805, 2006.

[4] J. Leskovec and A. Krevl, "SNAP: Stanford Network Analysis Project – Facebook Ego Network Dataset," Available: `https://snap.stanford.edu/data/ego-Facebook.html`.