Link Prediction with Enclosing Subgraph

Zihuan Diao
Stanford University
353 Serra Mall, Stanford, CA 94305
diaozh@stanford.edu

Abstract

Link predication is a classical task in the field of network analysis. A good link prediction model would be useful in building a recommendation system that could be applicable in social networks and online shopping sites. There are two main ways to tackle the link prediction problem, one is based on ranking links based on heuristics of their end nodes, and the other is to transform the problem into a supervised learning problem and train machine learning models to predict link formations. In this paper, we will survey previous publications on the topic of link prediction and propose a new supervised learning model to solve this problem.

1 Introduction

The goal of link prediction is that given a graph to predict the new edges that would form in the future or the missing edges in the present observation. Studying the link prediction problem could help us understand the underlying dynamics of the network, and therefore there are many existing work around this topic. There are two main types of approaches researchers take to tackle the link prediction problem, one is using unsupervised methods and the other is to formalize the problem as a supervised machine learning problem and build a model to predict link formation. For the supervised learning methods, the main idea is to transform the link predicting problem into a binary classification problem. The problem is defined as given two nodes in the base graph that don't have an existing edge between them, predict whether an edge would form between them given some input features.

A link's formation is determined both by its end nodes' local and global network structures. In order to capture the local network structure, the enclosing subgraph of a candidate link is used, and it is defined to be a fixed-size subgraph containing multi-hop neighbors of both end nodes. Enclosing subgraph with its nodes' features extracted from the original network would help include both local and global network characteristics. In the project, we will explore existing solutions as the baseline and develop a enclosing subgraph based supervised learning model. The model would extract features based on the enclosing subgraph of the two end nodes and train an binary classification model using those features. We also evaluate the performance of the model and analyze the result for future improvements.

2 Related Work

There are many existing publications related to the topic of link prediction, and we present you some of those papers that contribute to the ideas in this paper.

2.1 Heuristic Methods

David Liben-Nowell and Jon Kleinberg formlize the problem of link prediction in a network and propose a class of method based ranking with similarity scores of the end nodes of a potential link.[1]

The link prediction problem is defined as given a graph $G = \langle V, E \rangle$ at time stamp t, we would like to predict new edges that would form from t to a future time stamp $t^{'}$. Also, in this problem setting, the link predication problem only focuses on the new edges whose end nodes existing in the graph at t.

Using the intuition that two entities in the social network are more likely to interact with each other if they share many common neighbors. The authors propose using heuristics to measure the similarity between the two end nodes X, Y and rank all candidate edges using this similarity score $score(X, Y)$ of their end nodes. The authors experiment with three different kinds of similarity scores such as common neighbors, Jaccard's coefficient, Katz, PageRank, and SimRank. The paper shows that the similarity score based method could greatly improve the link predication performance comparing to a random model.

2.2 Supervised Learning Methods

After David and Jon presented the link predication problem in their paper[1], a new suite of machine learning method is introduced to solve the problem. Mohammad Al Hasan and the coauthors formalize the link prediction problem as a binary classification problem and use supervised learning models to solve it.[2] The problem is defined as taking node pairs in the training graph where no edge appear between them as input data points, and the label for the classification is defined to be whether the node pair has an edge in the testing graph.

For each of the node pairs, the model extracts three kinds of features: proximity features based on the extra information in the dataset, aggregated features, and topology features. Then, the paper uses different machine learning models to build a classification model using these features. Some of the models in this paper are decision tree, SVM, KNN, and Bagging. The paper evaluates the result of each models following the common practice for the general classification models, using Accuracy, Precision, Recall, F-value and Squared Error. The paper shows some promising F-value for most of the models.

2.3 Enclosing Subgraph

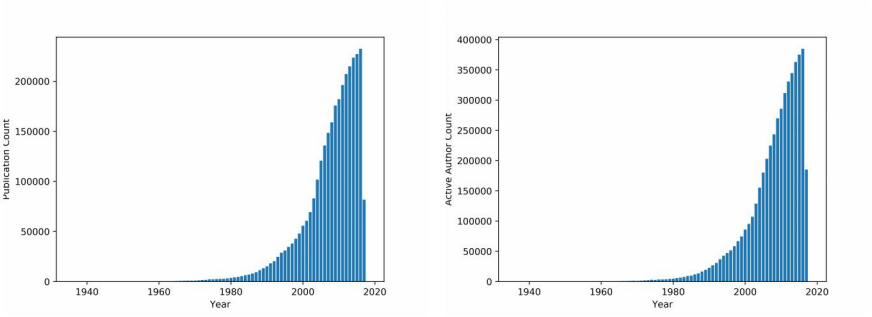
Jumping to the recent years, Muhan Zhang and Yixin Chen introduce a new idea for supervised learning in link prediction in their paper using enclosing subgraph.[3] Instead of focusing on the two end nodes for the candidate link, this paper propose we look at the enclosing subgraph of fixed size k for the link. The enclosing subgraph is defined as the graph containing only nodes that are n-hop neighbors of the end nodes for the candidate edge where we will start from $n = 1$ until we have enough nodes to reach size k.

The Weisfeiler-Lehman Neural Machine models works in the following way. First, for each candidate link, the model extract the enclosing subgraph of a size k, and orders the nodes in this subgraph using Pallette-WL algorithm[3]. Then, it generates an adjacency matrix for the subgraph using the ordering and feed the flattened adjacency matrix as the input feature to a neural network. The paper evaluates the model's performance on 8 networks, and the model achieves higher AUC comparing to most of the baseline models.

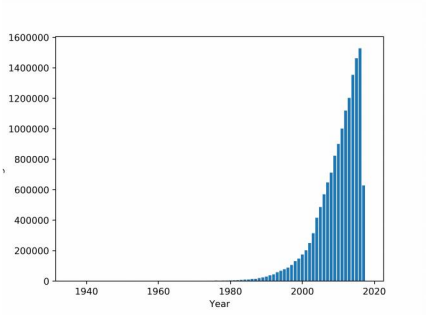
3 Dataset

In this project, we will be working with the undirected collaboration network derived from DBLP-Citation-network V10 [4]. This network contains papers extracted from the DBLP database where publication's date is from 1940 to 2018. In this dataset, there are 3,079,007 publications, 1,766,547 unique authors, and around 14,724,453 collaboration edges between authors. Note that the count for edges is an estimate as we are including duplicate edges. The distribution of the publications, active authors, and edges are shown in figure 1. We could find that the numbers of all these three items increase as time progresses. Based the distribution over the years, we could then select a subgraph whose size is feasible for this project.

In this paper, we will be studying the collaboration relationship between authors. Therefore, the nodes in our graph would be unique authors, and the edges denotes whether two authors has collaborated or not.



(a) (b)



(c)

Figure 1: (a) paper distribution, (b) active author distribution, (c) collaboration edge distribution

Table 1: Graph construction parameters

PARAMETER	VALUE
authors in t_i	1995
base graph t_0 to t_1	from 1995 to 1999
prediction graph $t_1 + 1$ to t_2	from 2000 to 2004

3.1 Preprocessing

Due to the huge size of the original network, we would only be using a subset of the network in this project. To extract the subgraph, we find a period of time t_i and extract all the authors that published paper during t_i. This set of authors V would be the set of nodes for our base graph and prediction graph. We will then pick a suitable year range from t_0 to t_1 to construct the base graph G with nodes in V, and we will then find a year t_2 such that $t_2 > t_1$, and record the graph from t_0 to t_2 minus the new nodes and existing edges in the base graph which will be the prediction graph $G^{'}$. The parameters used to construct the base graph and the prediction graph are shown in table 1.

In addition to choosing a reasonable graph size, we also remove nodes with degree less than 3 when constructing the base graph and the prediction graph. This is based on the idea that nodes with degree less than 3 have little information for us to make a informed prediction.

We randomly select 60% of the edges in the prediction graph as training set, 20% as validation set, and 20% as test set. In order to make the dataset applicable for both the similarity score base methods and supervised learning method, we will add the same number of randomly sampled negative sample, which means node pair that does not have edges in the base and prediction graph, to each of the dataset.

Table 2: Base Graph Characteristics

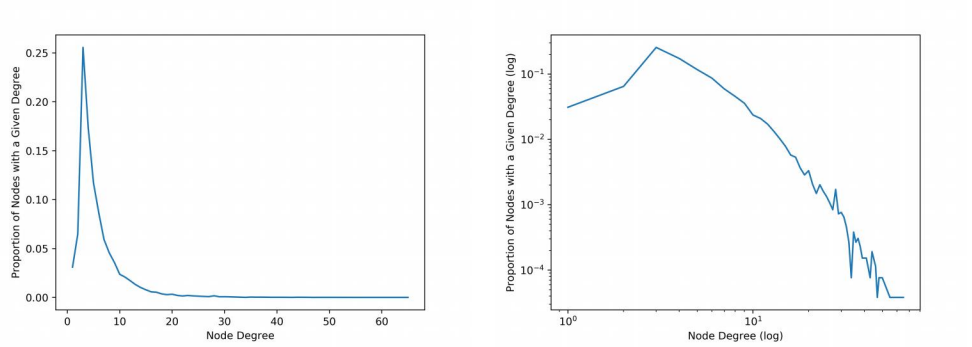
PARAMETER	VALUE
Nodes	26431
Edges	75691
Clustering coefficient	0.632587
Number of weakly connected components	799
Edges in the largest weakly connected component	68151
Nodes in the largest weakly connected component	22335

Table 3: Prediction Graph Characteristics

PARAMETER	VALUE
New edges	18623
Fraction of new edges	$\frac{18623}{94314} = 0.1975$

3.2 Dataset Characteristics

After extracting the two graphs that we will be working on, we study the characteristics about the graphs. For the base graph, we could find the degree distribution shown in figure 2. Also, other graph characteristics could be found in table 2.



(a) (b)

Figure 2: (a) Degree distribution for base graph, (b) degree distribution for base graph (log-log scale)

We could find that for the base graph, the majority of the nodes have a degree of less than 10. Looking at the clustering coefficient, we notice that it is around 0.6 meaning that the graph tend to form closely tied subgraphs. Also, based on the size of the of the largest weakly connected component, we find that the majority of the graph is connected.

4 Method

4.1 Similarity Score Heuristics

In this section we will define the similarity scores of two nodes that will be used in the project. The similarity scores are used both in the ranking based unsupervised methods and the supervised methods as input features. The similarity score between node A and node B should provide a similarity measurement that would be a value in $\mathbf{R}$.

Graph distance. The graph distance similarity for A and B would be the negate of the length of the shortest path between node A and node B.

Common neighbors. First, let's define $\Gamma(A)$ to be the set of neighbors of node A. The common neighbors similarity for A and B would be

$$|\Gamma(A) \cap \Gamma(B)|$$

Jaccard's coefficient. The Jaccard's coefficient[6] for A and B would be

$$\frac{|\Gamma(A) \cap \Gamma(B)|}{|\Gamma(A) \cup \Gamma(B)|}$$

Preferential attachment. The Preferential attachment[8] for A and B would be

$$|\Gamma(A)| \cdot |\Gamma(B)|$$

Adamic and Adar. Adamic and Adar[7] introduce a similarity measure based on the generic features of two entities. In our link prediction setting, the Adamic and Adar similarity for A and B would

$$\sum_{Z \in \Gamma(A) \cap \Gamma(B)} \frac{1}{\log(|\Gamma(Z)|)}$$

Katz. Katz[9] defines a similarity based on the weighted sum of number of paths with certain length between A and B. In our project, we would be using the unweighted version of this similarity as the graph we are studying is unweighted undirected graph. The Katz similarity between A and B would be where β is damping parameter

$$\sum_{l=0}^{\infty} \beta^l \cdot |path^l_{A,B}|$$

This similarity could also be expressed in the matrix form with the adjacency matrix M,

$$(I - \beta M)^{-1} - I$$

4.2 Edge Embedding

Edge embedding refers to ways that characterize a potential edge to a fixed-size feature vector. Node2vec[10] is a node embedding algorithm that utilizes parametric random walk specified with two parameters p and q. In this project, we would use node embedding from node2vec and generate edge embedding to perform link prediction.

For a candidate edge e with end nodes n_1 and n_2. We could use a node2vec algorithm with $p = 1$ and $q = 0.5$ to get a node level embedding of size 64. The idea is that with $p = 1$ and $q = 0.5$, the node2vec random walk would be more BFS-like thus letting the embedding capture structural similarity. After getting embedding for the two node n_1 and n_2, we then use the Hadamard product of the two nodes' embedding as the embedding for e, as Hadamard product has yielded better performance in experiments performed with node2vec[10]. The edge embedding used in the project is a vector of size 64.

4.3 Enclosing Subgraph

One of the key idea in this project is to incorporate rich information from the candidate links' surrounding subgraph to help the model predict link formations. For node A and B, we define the enclosing subgraph for the two nodes as the subgraph from the original graph containing only node in set EN where EN is constructed using the following algorithm[3]. Note that instead of getting subgraph with a fixed size k, we could also extract subgraph that only contains 1-hop to k-hop neighbors of the end nodes.

The other challenge with using information from the enclosing subgraph is how to encode the information. One idea would be to aggregate the node level features into a fixed size feature for the subgraph. For instance, we would extract node features and sum them into a features vector for the subgraph. The other way would be to order the nodes and concatenate each node's features into a

Algorithm 1 Enclosing Subgraph Extraction

Input link(A, B), size k, graph G
Output EN for (A, B)
F = A, B
EN = $\emptyset$
while $|EN| < k$ **do**
  F = $[\cup_{v \in F} \Gamma(v)] \setminus EN$
  $EN \cap = F$
end while
return EN

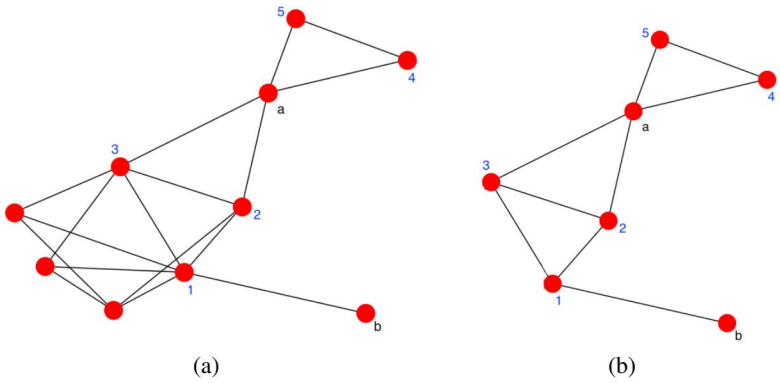


Figure 3: (a) Sample Graph, (b) subgraph of size 5 for node a and b

large subgraph feature vector. In this project, we will focus on a method that is in the middle. We will aggregate the node features by taking the average within groups of nodes that have the same minimum distance to the end nodes. Then, we will order the aggregated features according the groups' distance to the end nodes. In other words, nodes that are m-hop neighbors to the end nodes would be treated as a group, and their per node features will be aggregated to generate the features for this group.

4.4 Baseline

We implement the similarity score based methods as baselines. The similarity score heuristics we use are introduced in the previous sections. For each of the similarity scores, we will use it to rank the the testing set node pairs, and the top half of the pairs would be our prediction since the test set consists of half negative and half positive samples.

4.5 Neural Network with Edge Embedding

The edge embedding defined in the previous section is shown to perform well in link prediction tasks. For a candidate link, we use its edge embedding as input feature and train a fully connected neural network to perform binary classification on whether the link will form in the future.

The neural network used in this part has 5 layers with size 64, 32, 8, 4, and 1 where the first 4 layers has Relu activation functions and the final output layers has a sigmoid activation function.

4.6 Neural Network with Enclosing Subgraph Features

The main goal of the project is explore enclosing subgraph based supervised learning models. We first extract the enclosing subgraph that contains the 1-hop and 2-hop neighbors of the end nodes and then get the node2vec features for each node in the subgraph. The configuration for the node2vec is the same as we used to edge embedding. With the node level features, we encode the features for

Table 4: Method performance on test set

METHOD	PRECISION	AUC
Graph distance	0.6510	0.5865
Common neighbor	0.5519	0.5926
Jaccard's coefficient	0.5648	0.5888
Preferential attachment	0.6545	0.7074
Adamic and Adar	0.5498	0.5983
Katz with $\beta = 0.005$	0.6623	0.6133
Edge embedding	0.7832	0.8704
Subgraph features	0.7573	0.8232
Subgraph features w/ Edge embedding	**0.8416**	**0.9190**

the subgraph using the techinque introduced in the previous section. We will have subgraph features of size $2 \cdot 64 = 128$. Finally, we also experiment with concatenating the edge embedding features to the subgraph features creating a feature vector of size 192.

For the neural network, we decide to use the same fully connected neural network structure as used in the edge embedding section. The idea is to keep every part of the model the same and only vary the input features. In this way, we could study if the subgraph features could help with the task of link prediction.

5 Results

In this section, we present the result of evaluating different methods on the test set. The two metrics used in the part are the precision and AUC which is the area under the Receiver operating characteristic curve.

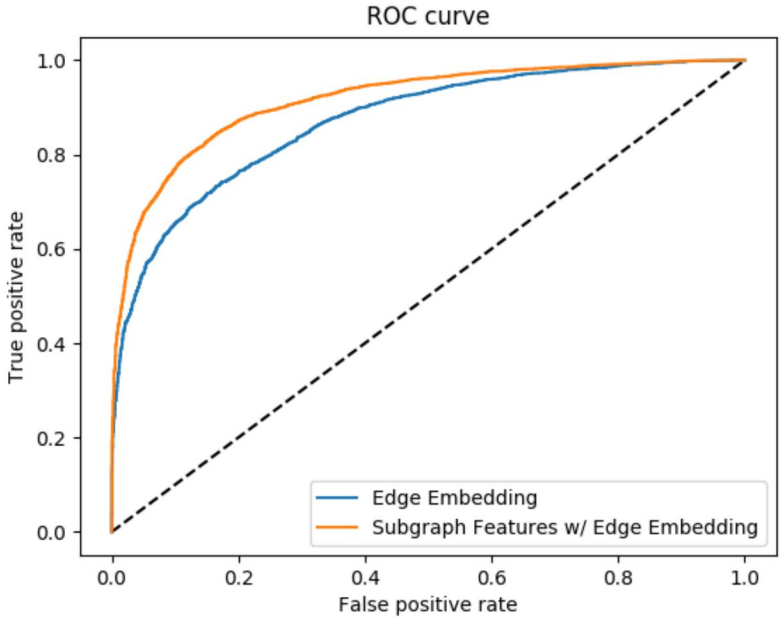


Figure 4: ROC curve for the edge embedding and subgraph features w/ edge embedding models on test set

5.1 Performance Analysis

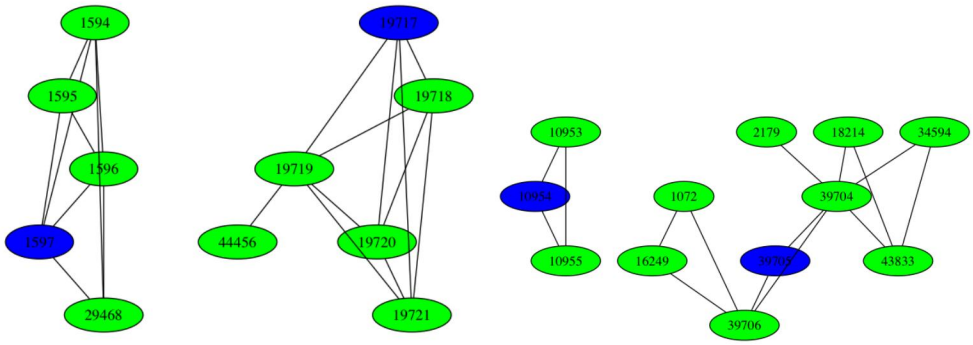
From the precisons and AUC results, we could find that the three supervised learning method using neural network outperforms the ranking based methods using different similarity heuristics. The model with only the subgraph features perform the worst among the three while the model with both subgraph features and edge embedding perform the best.

Moreover, since we are using the same neural network structure and node2vec embeddings for the edge embedding and subgraph features, we could conclude that the additional features extracted from the end nodes' enclosing subgraphs has provided additional information that has imporved link prediction performances.

5.2 Sample Analysis

In this section, we will examine the samples where an edge embedding model made a wrong prediction but the model with additional enclosing subgraph features made a correct prediction. Analyzing the samples could help us understand how adding subgraph features help with link prediction.

For the test set, our model using subgraph features and edge embedding is able to make the right prediction on 883 samples where the edge embedding only model gets wrong. Among them, there are 506 samples with a label of 1 and 377 samples with a label of 0. This indicates that the subgraph features has the similar effect on helping the model to predict better on both positive and negative samples.



1597_19717_1

(a)

10954_39705_0

(b)

Figure 5: (a) and (b) are visualization of the enclosing subgraphs of candidate pairs where the subgraph features improved comparing to edge embedding. (a) is a positive pair, and (b) is negative. The blue nodes are the end nodes for the candidate link

After studying the subgraph structure from the improved samples, we find that most of the samples have a enclosing subgraph that is disconnected. In others words, the samples that show an improvement are the ones whose end nodes are far from each other or disconnected in the original graph. Since the graph has a large weakly connected components, it is not a common case for samples to have disconnected end nodes or end nodes that are far from each other.

For instance, we could take a closer look at the visualization in figure 5. This is an interesting observation where we could learn some hints about how subgraph features are contributing to link prediction. In this case, since we have edge embedding that is based on node2vec, if two end nodes are close to each other and having a connected enclosing subgraph, then due to the nature of the random walk in node2vec, the node2vec embedding should be able to capture the correlation between the two end nodes. Therefore, when the end nodes have connected enclosing subgraphs, edge embedding could work well on its own. On the other hand, when two end nodes are far from each other and having a disconnected enclosing subgraph, the node2vec algorithm would be less likely to capture the correlation between the nodes. However, when incorporate the embedding

from the subgraph, we are essentially looking further from the end nodes and thus getting more improvements on cases where the two end nodes are having a disconnected enclosing subgraph.

6 Conclusion

In this project, we explored different methods to perform link prediction tasks. We focus on studying if and how enclosing subgraph features help with link prediction. We find that node2vec is a good feature to perform link prediction, and we could improve the model using only node2vec features from the links' end nodes by incorporating aggregated node2vec features from the links' enclosing subgraph. We also study the cases where subgraph features help improve the model's performance and come up with a explanation that subgraph features help with link predictions by incorporating information far away from the end nodes which is particularly helpful in cases where end nodes are disconnected or have a large distance between them.

7 Future Work

In this project, we focus on using a neural network model to test if enclosing subgraph features improve the performance on link prediction. The problem with neural networks is that they have less interpretability comparing to other machine learning models. Therefore, we will have to rely on analysis on individual samples to reason about enclosing subgraph features' effects on the task. Testing the enclosing subgraph features by designing different feature extraction and encoding methods could help shine more light on this topic. Also, building different models with more interpretability like SVM or decision trees could help improve our understanding on the contribution of enclosing subgraph on link prediction.

Contribution

This project is done individually by Zihuan Diao. He is responsible for composing the reports and posters, and also preparing the dataset, building models, designing experiments, and analyzing results.

Code Repository

`https://github.com/diaozh/cs224w_project`

References

[1] Liben-Nowell, D. & Kleinberg, J. (2003) The Link Prediction Problem for Social Networks. *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, pp. 556-559. New York, NY: ACM.

[2] Al Hasan, M. & Chaoji, V. & Salem, S. & Zaki, M. (2006) Link Prediction using Supervised Learning. *Workshop on Link Analysis, Counter-terrorism and Security (at SIAM Data Mining Conference)*

[3] Zhang, M. & Chen, Y. (2017) Weisfeiler-Lehman Neural Machine for Link Prediction. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 575-583. New York, NY: ACM.

[4] Tang, J. & Zhang, J. & Yao, L. & Li, J & Zhang, L. & Su, Z. (2008) ArnetMiner: Extraction and Mining of Academic Social Networks. *In Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'2008)*, pp.990-998

[5] Lecun, Y. & Bottou, L. & Bengio, Y. & Haffner, P. (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE, Volume: 86, Issue: 11*, pp. 2278-2324

[6] Salton, G. & McGill, M.. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[7] Adamic, L. & Adar, E. Friends and neighbors on the web. *Social Networks*, 25(3):211230, July 2003.

[8] Newman, M. Clustering and preferential attachment in growing networks. *Physical Review Letters E*, 64(025102), 2001.

[9] Katz, L. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):3943, March 1953.

[10] Grover, A. & Leskovec, J. node2vec: Scalable Feature Learning for Networks. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.