Scale Robust Community Prediction with Node Embedding based Negative Sampling

HyunJun Choi
Graduate School of NQE
KAIST
Daejeon, South Korea
juneir@kaist.ac.kr

DongHyeok Shin
Graduate School of ISysE
KAIST
Daejeon, South Korea
tlsehdgur0@kaist.ac.kr

HyunJi Lee
School of EE
KAIST
Daejeon, South Korea
alee6868@kaist.ac.kr

ABSTRACT

Recent works on representation learning for graph-structured data focuses on one entire graph or relation between the nodes in a graph. It has a limitation that the model can watch only the global or the local information on the graph. However, there are cases where we need to extract the features of the subgraph from one big graph. In this paper, we present a new method GWL. GWL can detect potential community or subgraph by making new datasets using features extracted for the current entire graph. This method is novel in that compared to previous researches or models, it can preserve the information of both the global and local information and by a rule-based negative sampling, it can convert datasets which are more prone to the unsupervised task to supervised task. Our experiments on several changes prove the righteousness of our model and by comparing it with other link prediction models, it reveals high accuracy over others.

1. INTRODUCTION

At present, there are a lot of datasets in different fields to predict future steps or phenomena. In this connection, community prediction is one of the most useful and powerful methods to predict new subgraphs from the network. Community prediction has been applied to many fields including biological networks, cost management, dynamic systems, and also present in our daily lives, suggesting people we may know in social networks or products we might be interested in.

Community prediction methods also can be used in the co-author relationship prediction problem. When we predict future co-author relationships, one of the important things is extracting reasonable features from the current graph and use the features to predict future co-author relationships. Previous community prediction methods show improvement in prediction accuracy using various methods such as divide to link prediction, formulate new equations, or propose new methods. However, they applied their methods to rich information datasets, which contain publication venues, ground-truth communities, etc with dense matrix or a relatively small number of nodes. This kind of information for each node and large information about the entire graph can help in making predictions of the community simpler.

In this report, we propose the new community prediction method, which we called GWL, to detect potential community or subgraph by making new datasets using features from the graph of the current state. We bring this idea from the current Vision task which generates a new dataset by GAN[2] if we cannot collect wanted datasets. We used the node embedding model, Node2Vec[3], to predict potential disconnected communities. Since we did not start from the random space our method of sampling is similar to conditional GAN proposed in cycleGAN[12]. By generating new datasets, we could convert dataset that is prone to use in unsupervised tasks to supervised tasks. We then used the generated datasets to extract graph embedding vectors with various scales for binary classification task of whether the subgraph is likely to exist in the future. More details about the method will be discussed in Section3.

The problem we want to solve is the following:

- GIVEN: a sparse graph with a large number of nodes and well-formed community information. We wanted to extract the information of the community itself from the whole graph to determine whether the community has a common feature or not.
- FIND: determine whether given communities also called subgraphs will be formed in the future or not. We assumed that communities that contain nodes with the common feature will have the feature itself, and those features will lead the community to be formed in the future.
- to MINIMIZE: the classification error of classifying if the communities are likely to be formed.

The contributions of our method are the following:

- 1. GWL is unique by creating a new dataset by doing rule-based negative sampling based on node embedding to convert the given dataset which is prone to unsupervised learning tasks to dataset more suitable to the supervised learning task.
- 2. GWL is robust to diverse scales of communities and is extensible to other classification models like GNN, MLP, etc rather than a simple classification model we used.

.

- 3. GWL uses new graph labeling which preserves features of community structure in enclosing subgraph from node embedding.

Unsupervised Learning Unsupervised Learning is a type of machine learning which looks for previously undetected patterns in an unlabeled dataset and without human supervision. Two of the main methods used in unsupervised learning are principal component and cluster analysis. Cluster analysis is a method of grouping the data that has not been labeled, classified, or categorized. It identifies the commonalities in the data and reacts based on the presence or absence of such commonalities in each new piece of data. There were studies of community prediction using unsupervised learning specifically clustering with datasets without labels.[8], [7]

Supervised Learning Supervised Learning is a type of machine learning which learns a function that maps an input to an output based on labeled train datasets. This new function can be later used for predicting class for new datasets commonly called as test datasets. It is used in diverse algorithms such as SVM, linear regression, logistic regression, etc. Given labeled datasets, using these algorithms are possible.

Node2Vec Node2Vec[3] learn continuous feature representation of the nodes when given any graphs. This feature representation can be used for various downstream machine learning tasks such as link prediction, community detection, and so on. The objective is flexible, and it accommodates for various definitions of network neighborhoods by simulating biased random walks. It assumes that random walks through a graph can be treated as sentences and each node can be treated as individual words. By using a skip-gram or continuous bag of words algorithm to these random walks, it can extract the feature of each node. It also uses hyperparameters, p, and q, to weight the connections between the two nodes, providing a way of balancing the exploration-exploitation tradeoff. However, it has a limitation that it could only model local similarity within a confined neighborhood and fails to learn global structural similarities that help to classify similar graphs together which concludes that extending this model directly for tasks involving whole graphs yields don't bring sufficient performance.

Graph2Vec Graph2Vec[8] is an unsupervised representation learning technique to learn distributed representations of arbitrarily sized graphs. If extracts structure-preserving, data-driven embedding which appropriately models both local and global similarities among graphs. Similar to doc2vec, when given a dataset of graphs, graph2vec considers the set of all rooted subgraphs around every node as its vocabulary. It learns the representation of each graph in the dataset by using a skip-gram training process. By this, it assigns a unique label for all the rooted subgraphs in the vocabulary and deploys the WL relabeling strategy. This algorithm is not suitable when sampling subgraphs from a big graph since it can not preserve the information of the whole graph.

WLMN Weisfeiler-Lehman Neural Machine(WLNM)[10] is a link prediction method that used subgraph labeling based on the Weisfeiler-Lehman algorithm. WLNM assumes that the enclosing subgraph which describes the "surrounding environment" of the target link contains topological information of link prediction. WLNM uses proposed palette-WL which labels using the shortest path between the target nodes and subgraph nodes. WLNM learns structural features in the form of graph patterns which help to the prediction of links. However, only link prediction is available for the WLNM model. So we developed the idea so that community prediction will also be available. We call this model as GWL which is short for Generalized-WL.

SEAL SEAL[11] present the theoretical justification of WLNM's assumption. Moreover, this paper shows that it is sufficient for learning link prediction heuristics by using only a local enclosing subgraph, not the entire network. It means we can gain sufficient information for high-order heuristics by using small hop-size, which determine the enclosing subgraph's size. We apply this idea to community prediction.

2. PROPOSED METHOD

The main motivation behind our method is to handle provided datasets carefully so that we can directly extract community features. Since a sparse graph with a large number of nodes and only positive connections are provided in the given dataset, we proposed our new methods GWL that will help in solving these problems.

Our proposed method is as follows:

- 1. We changed the overlapping clustering task to binary classification tasks. When given one big graph as a dataset for the clustering problem, it is difficult to choose the number of clusters that optimize the result. To reduce the burden of choosing the number of clusters, we tried to convert the problem to a binary classification of deciding whether the given subgraphs are likely to be connected or not in the future instead. The flow is described in Figure 2.
-
- 2. We extracted features of the subgraph that also preserves the global information using node embedding vectors of nodes in the subgraph. As proposed theoretically in SEAL[11], we assumed that community enclosing subgraphs contain enough information to learn good graph structure features. We also proved that this held in our model by experiments. (Section 3.4) Further expanding ideas from Graph2Vec and WLMN, we used the geometric mean of the embedding vector's distance to compare other nodes. Specific flows are described in Figure 3.

We made 4 assumptions before constructing a model.

- 1. given paper author graph is already formed.
- 2. communities will be formed based on similar nodes. In other words, if the two nodes are similar, they have a high possibility of being connected.
- 3. based on the second assumption, the community is a complete graph. Nodes inside a community will have a common feature and based on it, the community will contain the same feature.
- 4. due to assumptions 2 and 3, to have less variation on how we struct the base graph, we assumed both base and test dataset graphs as unweighted graphs.

To interpret this task as a clustering task, we had difficulty deciding the number of clusters and deciding the range we will consider it as a community. So we rather interpret the task as a classification task and make a new dataset by ruled-based negative sampling.

Our approach contains three stages. Figure 1 summarizes our framework. First, in constructing a network graph we

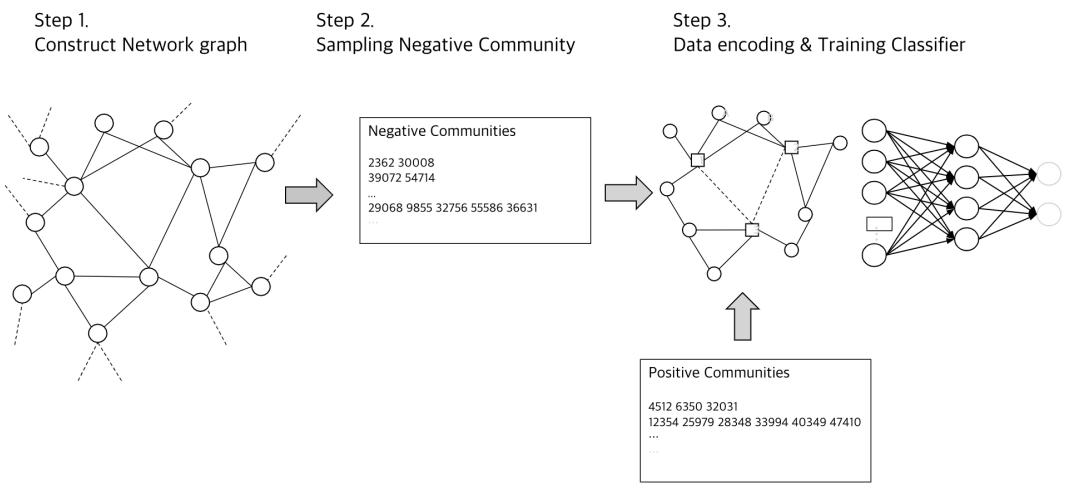


Figure 1: Overall method

construct a graph based on a given dataset. (Sec 3.1) The graph is used in node2vec for negative sampling. (Sec 3.2) Finally, both the initial graph and negative sampling are used to extract feature vectors of each subgraph. (Sec 3.3)

2.1 Construct Network Graph

To extract the embedding vectors of nodes or subgraphs, we first need a stabilized graph. Since we assumed that nodes are connected if they have similar features and communities will be created with those connected features, we presume that the nodes in the community will all be connected and form a complete graph. However, due to these assumptions, we thought that it is more reasonable to draw the graph as an unweighted graph. The output graph is then used to extract node embedding and subgraph embedding.

We draw the graph as an unweighted graph because we thought the information that the two nodes are connected more than once doesn't imply that they have more than one common feature but rather it simply means they are in the same community twice. The two communities could be connected due to the same feature. Also, if we draw a weighted graph, it will depend highly on how we sampled or how we constructed the graph which we do not prefer.

2.2 Sampling Negative Community

To get negative sampled communities based on node embedding, we trained Node2Vec using the graph structure from the previous stage.

Node2Vec We set the hyperparameters mostly based on paper and to capture sufficient information from the sparse large graph and to extract information of communities, we used the method based on DFS. From the result, we could get embedding vectors for each node and similarity between the nodes. These two pieces of information are used in further stages.

Negative Sampling Terminology of 'Negative Sampling' is used in diverse ways. In node embedding, it is often used in optimization. In optimization procedure, since it is difficult to look at all the nodes when the graph is large, it rather samples some negative nodes randomly and uses this information to update instead of looking at the entire nodes. In our method, we use the term differently. Negative sampling is used to extract the communities that are unlikely to be connected in the future so that it could be labeled together.

Simple information on our rule-based Negative Sampling is as below. Referring to the result of Node2Vec, we generate a new directed graph which will be called 'graph A'. We can pick the lowest 10 similar nodes for each node and connect directed edges between them. We considered the bidirectional pairs in graph A as the nodes that are most unlikely to be connected in the future. However, the number of bidirectional pairs was not enough so we had to divide the cases of communities with size 2 and the rest. For the communities with size 2, we randomly picked the directed edges from the graph A. In other cases, starting from the bidirectional pairs, we randomly picked an edge in graph A that could be connected to the pairs until the number of nodes connected has the same number as the target subgraph.

We decided the number 10 for connecting directed edges for each node because if the number is small, there was a problem that similar subgraphs were continuously sampled from graph A. For number larger than 10, there were too many edges to sample from so we decided the number 10 would be the optimal number.

The sampled number of communities for each size is the same as the positive community to prevent any kind of bias when training stage3(sec 3.3) using both positive communities and negative communities.

2.3 Data encoding and Training Classifier

Using the base graph from stage1(sec 3.1) and negative communities sample using our negative sampling method(sec 3.2) we extract the feature embedding vector for each subgraph in this stage.

Extract Subgraphs We assumed that community enclosing subgraphs contain enough information to learn features of graph structure based on the theory proposed in SEAL[11]. Based on this proposal, we extract the enclos-

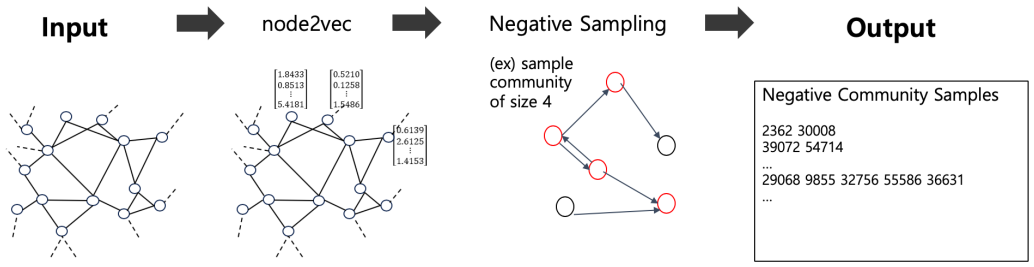


Figure 2: Based on the network graph constructed in stage1, we train Node2Vec. For communities with size of 2, we randomly pick the one from directed graph and for communities larger than 2, we make a directed graph which will be called as graph A with each node connected to 10 lowest similarity. Based on the bidirectional pairs in graph A, we expand the community until the target community size.

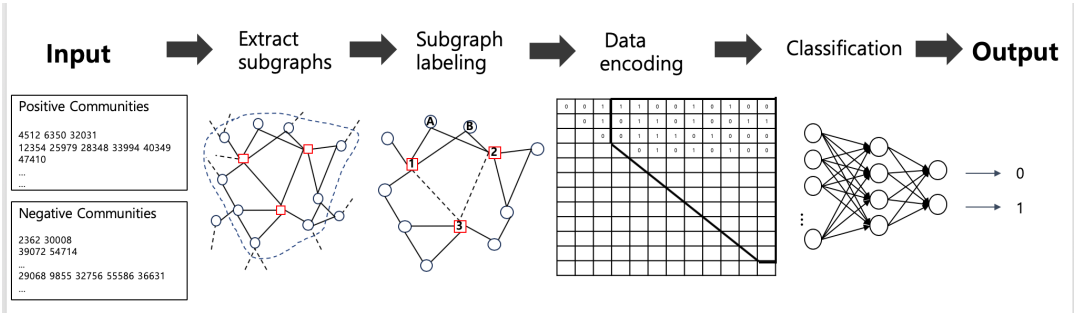


Figure 3: Data encoding and Training Classifier. Using the graph from step1 and generated dataset from step2, we extract the enclosing subgraph of the target community such that its size is proportional to the target community size. After, we conduct the subgraph labeling. By using geometric mean of embedding vector's distance, we compare the node and construct adjacency matrix with subgraph labels. We concatenate the row vectors to make it flat and train a simple linear classifier.

ing subgraph of the target community such that its size is proportional to the target community size.

Subgraph Labeling To preserve the subgraph structure, we conduct the subgraph labels. After imposing the initial label for the target community, we use the geometric mean of the embedding vector's distance difference to compare other nodes. We used Euclidean distance to compare the embedding vector's distance difference. In the Ablation Study (Section 3.4), we will show why we decided to use euclidean distance with geometric mean.

Data Encoding From the subgraph labels, we construct the adjacency matrix and concatenate the row vectors to make it flat. As shown in Figure3, we do not use the entire elements of the upper triangular matrix but exclude some information because the first square matrix provides true existence which could be a hint for prediction.

We then train a classifier with cross-entropy loss to define whether the input subgraph is likely to be connected in the future or not. To be more concise, different classifiers like GNN or MLP could be used instead. However, since we wanted to make our model be small and simple we used 4 fully connected layer classifier.

3. EXPERIMENTS

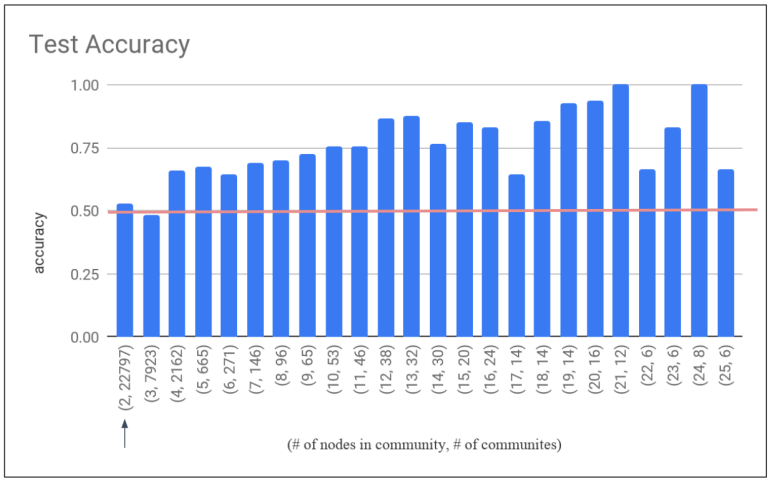
Since we tried to make a method and a model that is suitable for the given dataset, we couldn't find a model that is suitable for comparison. So instead of comparing the model with different models, we tried to prove that our idea is right with diverse experiments, and for cases where we could do the comparison, we tried to show that our model is promising. The Accuracy and F1 score below is when we trained only using the paper author dataset. Since we retrained the model using both paper author and public query datasets as our final model, we assume that both accuracy and f1 score will have increased.

3.1 Data and Experiment Setup

Datasets We used the dataset "paper-author" for train dataset. This dataset contains the author's information on past publications. There is 58646 number of authors and 137958 publications. Since by rounding there is about 3 previous collaboration information for each author and by this, we can say that there aren't large information and the graph is sparse.

We generated a new dataset that contains information on fake collaborations by Negative Sampling. As in paper-author, there is 58646 number of authors and 137958 publications. This dataset was also used for training.

For the test dataset, we used "query-public" dataset. This



(a)

TP	10939
FP	9605
TN	7370
FN	6086
Precision	0.53247
Recall	0.59747
F1 score	0.56310
Accuracy	0.5385

(b)

Figure 4: Test Accuracy on Each Community Size (a) and Evaluation on Total Dataset (b)

dataset contains both the information of fake collaboration as well as real collaborations. There is 58646 number of authors and 34479 publication information. The rate of true collaboration and fake collaboration was nearly 1 to 1.

Implementation Details For stage2 (Sec 3.2), we tried to follow hyperparameters as given in the paper[3]. We set the hyperparameters as dimension=128, walk length=80, num walks =10, p=1, q=0.5, window=10, batch words=4. Since our model is not an end-to-end model, we didn't work much on optimizing the hyperparameter but instead tried to make it work faster by training in parallel. To get enough information about the community inside the sparse graph, we decided on the values of p and q.

For stage3 (Sec 3.3), we set the hyperparameters as batch size=1000, total epochs = 50, hidden layer dimension of classifier = [128,32,8,2], hop size = 1. To find the reasonable hyperparameters of stage3, we conduct several ablation study, see more detail in Sec 3.4.

Evaluation Metrics We evaluated the model using query public dataset and mainly compared the performance of the models with F1 and Accuracy.

3.2 Accuracy on Each Community Size

From the evaluation in Figure 4, we can see high accuracy for communities with a large number of nodes. This proves that our method sufficiently captures the community's feature in the entire graph even with large size ones. However, since most of the communities in the test dataset were communities with size 2 or 3, the overall accuracy was 0.54 and the F1 score was 0.56. Since the accuracy when the number of nodes in the community is 2 or 3 is low compared to when there are 4 or more nodes in a community, we tried to check the validity of our model by comparing the link prediction results with other models using different datasets.

3.3 Link Prediction Experiments with Different Models

As shown in Figure 4, accuracy on communities with a small number of nodes was relatively low compared to the large communities. We assumed that since we use the same model for all sizes of communities, it has a weakness of capturing the features of 2 or 3 node communities which has less information. To check our prediction, we tried to compare with other link prediction models. However, we could find that our model gives sufficient results for the link prediction task as well.

The test accuracy for each model and dataset is provided in Figure 5. Since we wanted to train Link Prediction, we split the paper author and negative sampling dataset to a combination of all links and used these to train WLNM and GWL. For the test dataset, we used communities with size 2 in query public. We used the graph from stage1 for Jaccard Coefficient and Resource Allocation Index. [5], [6], [9]

Though our model has a lower accuracy than WLNM, since our model can do scalable community prediction, we thought that the difference is negligible. For the case of the Jaccard Coefficient and Resource Allocation Index, since the output value ranges from 0 to 1, we set the threshold to 0.5, and those that have value over the threshold we assumed they are connected and those below are disconnected.

We further tried the same test with U.S. Air dataset which has fewer nodes. It consists of 332 nodes and 2126 edges. Since there is a fewer number of nodes, we can see that the accuracy increases highly and for this dataset, our model GWL returns sufficient high accuracy compared to the other models.

3.4 Ablation Study

Extract Subgraphs We assumed that community enclosing subgraphs contain enough information to learn features of graph structure as proposed theoretically in SEAL[11]. We tried to prove this by experiments. For a small classifier, we put 4 linear layers as a classifier with the dimensionality of [128, 32, 8, 2] for each layer, and for a large classifier, we put 4 linear classifiers with [512, 64, 8, 2] dimensionality for each layer. We used Relu except for the last layer which we used softmax. For both cases, we used adam optimizer and sparse categorical cross-entropy as the loss function.

Test Acc. / Algorithm	Paper Author & Query Public	US Air
Jaccard coefficient	0.50	0.67
Resource Allocation Index	0.52	0.72
WLNM	0.59	0.88
GWL	**0.54**	**0.85**

Figure 5: Link Prediction Accuracy for each model

As shown in Figure 6, we can see that there are no big differences between the 4 cases which prove that hop size 1 sufficiently held information to learn the features of the graph structure. Since we used node embedding for each node, we can, therefore, get the features of the subgraph structure from one big graph.

Distance Measure and Means In WLNM[10], since they did not use the embedding vector of each node, they simply calculate the geometric mean of the shortest path length between the two nodes to construct the subgraph.

Expanding the idea from WLNM, we wanted to preserve the information of the whole graph and based on the information, we wanted to extract the feature of the subgraph inside. Therefore we decided to change the method of using the shortest path length to other methods using node embedding.

The node embeddings of Node2Vec will be similar if the two nodes are closely related and will differ otherwise. By this fact, we thought that there are two ways of calculating the distance between the nodes using node embedding. First is calculating the cosine similarity between the two node embeddings and pick the ones that maximize the value. Second is to calculate the euclidean distance between the embedding vectors and pick the ones that minimize the value.

For the second case, we used a geometric mean as given in the paper. However, for the case of cosine similarity, there was a problem of getting a negative value. Due to this problem, we instead used the arithmetic mean to normalize the value.

As shown in Figure 6, the geometric mean of euclidean distance between the embedding vector had the highest accuracy. This is because arithmetic means has the problem of converging into similar value in all cases and shortest path length have the problem of not using the embedding vector. Further, we could see that our idea of preserving the whole graph information by node embedding worked well compared to the base case of WLNM.

4. CONCLUSIONS

Our method GWL has the following advantages:

- 1. We sampled negative communities based on node embedding that was extracted from Node2Vec based on DFS to detect community information. Compared to simple random sampling, it contains information of the similarity between the nodes, which gives meaningful information on stage3. Also, since we used a random method for extending the model, it is fast and efficient.
- 2. It is scalable to any size of communities and it extracts significant features of the community that could be used for the classification task. Also, it is extensible in the sense that despite we used simple linear classifier to make the model simple, different classifiers such as GNN or MLP could be used instead.
- 3. We propose a novel method of graph labeling which preserve feature of community structure and information of the general graph.
- 4. Our model proposes sufficient accuracy for both community prediction task and link prediction task.

Though there is some weakness in our model currently such as:

- 1. for complete training, we need to go step by step
- 2. it is hard to capture the feature of the subgraph when there aren't enough nodes in the subgraph
- 3. if training Node2Vec fails, forward training may all fail

but we tried to solve the weakness if possible or prove the effectiveness of our model considering the tradeoffs by:

- 1. optimize all the steps so that the overall process will not take long
- 2. by Figure 5, we have proved that although there was some tradeoff that when we try to make one model to embed both large communities and small communities well, small communities tend to have difficulty, it works sufficiently well as compared to other link prediction algorithms especially when using the dataset with small graphs.
- 3. Figure 4 shows that our model is able to extract meaningful feature vector for each community which we can predict that node embedding vectors from Node2Vec also outputs meaningful embeddings even though optimal hyperparameter tuning was not done.

Since we further trained our model by adding Query Public dataset for our final model, we expect the accuracy to increase. Also, if delicate hyperparameter tuning is done for each stage, the accuracy would increase.

With better resources, we would be able to develop our model in diverse ways because we had difficulty with out of memory error difficulty due to large paper author graphs. Due to this problem, we had to get some tradeoff of losing accuracy to maintain less usage of the memory. We would

Hop Size \ Classifier Structure	Small Classifier	Large Classifier
1	**0.5385**	0.5318
2	0.5198	0.5214

(a) hop size and classifier structure

Distance Measure	Cos Similarity between embedding vectors	Euclidean distance between embedding vectors	Shortest path length between two nodes
Means	Arithmetic Mean	Geometric Mean	Geometric Mean
Test Accuracy	0.50	**0.54**	0.48

(b) distance measure and means

Figure 6: (a) By showing that the 4 values are similar, we can prove the proof from SEAL established in our case. (b) Our model of using node embedding preserved the information of whole graph compared to the base algorithm of WLNM

be able to change Node2Vec to an algorithm such as Watch Your Step[1] in stage2 or train with larger batch, longer epochs, and complicated classification models will be available, which we assume will lead to a more concrete model.

Also, sampling negative communities with more information such as result from Matrix Factorization[4] is likely to give better results of sampling.

For code details, visit https://github.com/amy-hyunji/GWL

5. REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alex Alemi. Watch your step: Learning graph embeddings through attention. *arXiv preprint arXiv:1710.09599*, 1(2):3, 2017.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[3] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[4] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[5] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[6] Shuxin Liu, Xinsheng Ji, Caixia Liu, and Yi Bai. Extended resource allocation index for link prediction of complex network. *Physica A: Statistical Mechanics and its Applications*, 479:174–183, 2017.

[7] Annamalai Narayanan, Mahinthan Chandramohan, Lihui Chen, Yang Liu, and Santhoshkumar Saminathan. subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs. *arXiv preprint arXiv:1606.08928*, 2016.

[8] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

[9] Sucheta Soundarajan and John Hopcroft. Using community information to improve the precision of link prediction methods. In *Proceedings of the 21st International Conference on World Wide Web*, pages 607–608, 2012.

[10] Muhan Zhang and Yixin Chen. Weisfeiler-lehman neural machine for link prediction. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 575–583, 2017.

[11] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, pages 5165–5175, 2018.

[12] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

APPENDIX

A. APPENDIX

A.1 Labor Division

The team performed the following tasks

- Search for papers and discuss ideas [all]
- Modify Node2Vec code for stage2 [HyunJi]
- Implement codes for Negative Sampling method [HyunJi]
- Implement codes for GWL [DongHyeok]
- Modify WLNM code for baseline [DongHyeok]
- Suggest ideas to develop GWL and conduct experiments [DongHyeok, HyunJi]

A.2 Full disclosure wrt dissertations/projects

A.2.0.1 HyunJun:.

He researched papers and participated in a discussion about ideas.

A.2.0.2 DongHyeok:.

He researched papers and participated in a discussion about ideas. He managed the overall process of stage1 and stage3. He worked on diverse experiments and came up with ideas to improve our model based on the results.

A.2.0.3 HyunJi:.

She researched papers and participated in a discussion about ideas. She managed the overall process of stage1 and stage2. She worked on diverse experiments and came up with ideas to improve our model based on the results.