COMP4332 Project 2 Report - Social Network Mining

Shiu-hong Kao*, Shi-heng Lo^
*20657378, ^20654493; {skao, sloab}@connect.ust.hk

Introduction

In computer science, Link Prediction is a graph problem aiming to predict the existence of a(n) directed/indirected edge between two entities. This problem has been extended to the application of social network to investigate the mutual connection of two individuals.

In this project, we will explore the link prediction problem on a small-scale dataset, where the training, validation, and testing set contain 8328, 5440, 5452 nodes and 100000, 19268, 40000 edges respectively. We embed each node with Node2Vec algorithm[1] as it was presented in the sample code of project description. We stored the embedding model for further use and then constructed an multi-layer perceptron (MLP) to predict the existence of each edge. Finally, our scheme achieves 0.9711 AUC on the validation dataset. We also provide the prediction score of testing dataset for reference.

Methodology

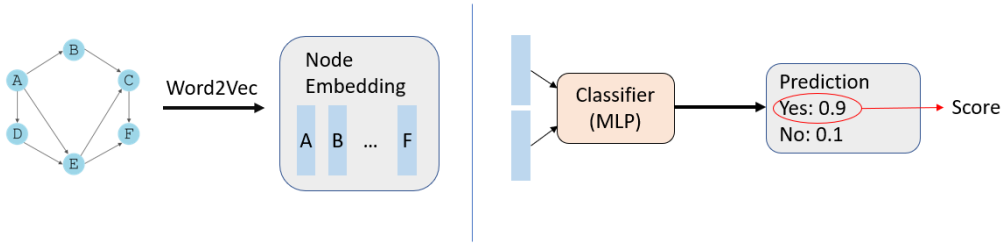

Figure 1: Designed workflow.

As shown in Figure 1, we followed the same step as in the sample code to generate the node embedding model. In specific, we used Node2Vec to embed each node. After these, we used a classifier to predict the link between two nodes using their embedded features, regarding the prediction problem as a 2-class classification issue. We randomly generated a dataset of unexisting edges and splitted it into training and validation set. These edges were labelled with 0, while the data in the original datasets were labelled 1. We then trained our classifier with cross entropy loss. We inserted a softmax function on the output logits and treated the probability of positive label as our prediction score.

Experiment

We will describe our experimental details in this section. We will first introduce the parameters and model used in our experiments and demonstrate the result in the following part.

Node2Vec. We used grid search to find the optimal configuration for node embedding for Node2Vec and set number_of_walks = 13, walk_length = 50, p = 100, and q = 0.01. As shown in Figure 2, larger window size and embedding dimension of Word2Vec generally contribute to higher performance. Considering the training complexity, we tuned the window size to 40 and node_dimension to 32. This resulted in an AUC score of 0.954 under cosine similartiy. From Figure 2, we found out that:

1. num_walk = 13: This was close to the average degree of a node (total edge = 100000, total node = 8328). We could think of it this way: it was good to walk to every possible path to collect a sufficient amount of information between nodes.
2. p = 100, q = 0.01: These parameters favored walking away from the start node. We found out that it was better to walk in a Depth-first-search (DFS) manner. If we had walked in a Breadth-First-Search (BFS) manner, it would have resulted in walking in loops and not collecting enough information from nodes that were some distance away, especially since the walk_length was tuned to quite a large number.

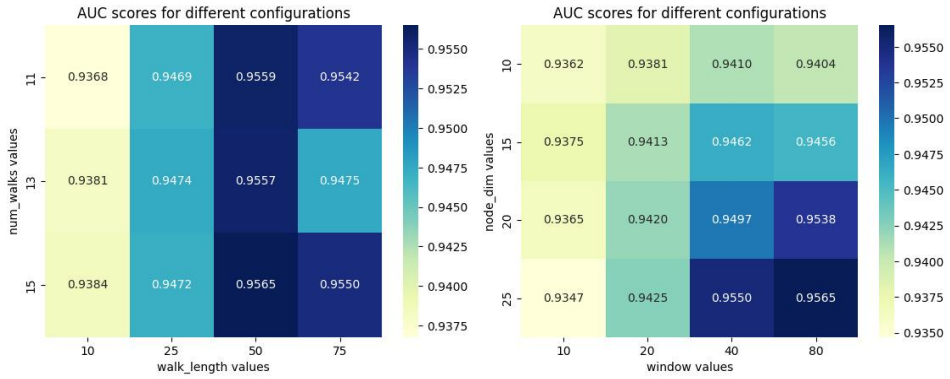

Figure 2: Heatmap of Node2Vec hyperparameters.

Random data. We randomly generated 100000 and 20000 unexisting edges in the training dataset and the validation dataset with label 0 for 2-class classification. Figure 3 shows some randomly sampled source points with their corresponding relationship to other points. We projected each embedding to the euclidean plane using Principal Component Analysis (PCA) [2] for better visualization. As shown in Figure 3, our Node2Vec methods successfully embedded useful information of node to vectors in some degree.

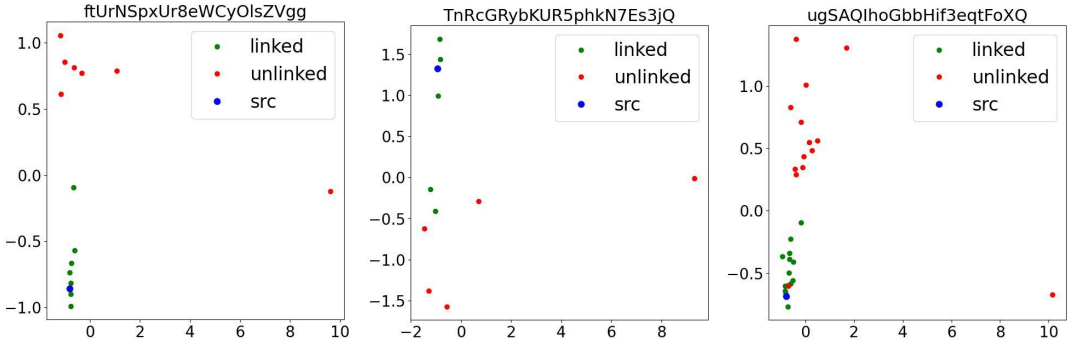

Figure 3: Illustration of node embedding with some sample datapoints.

Classifier. From high-level prospective, multi-layer perceptron (MLP) projects the features to some higher-dimension space and predicts the link existence using the weighted sum of high dimensional features. We used an MLP with 3 fully-connected layers, where both hidden dimensions were set 512. ReLU activation was inserted after each hidden layer. The last layer generates a 2-dimensional output.

Classifier training details. We optimized the classifier with the SGD optimizer in PyTorch with momentum 0.9 and weight decay 0.001. The learning rate was set 0.1 with a warmup learning rate 0.0005 and minimum learning rate 0.005 of cosine annealing strategy. We trained the model with 40 epochs, and the warmup epoch was set 10. We also included the early stopping strategy to save the best model during the training process.

Result. Our scheme achieves AUC score of 0.9711 on the validation dataset.

Code. Our code is available at https://github.com/DanielSHKao/comp4332proj2.

Conclusion

In this project, we examined the provided dataset with our proposed method composed of a Node2Vec model and an MLP classifier. The proposed method obtained AUC score of 0.9711, which is higher than the strong baseline on the validation dataset and provided a clear insight of link prediction.

References

[1] A. Grover and J. Leskovec, "Node2vec," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[2] Principal component analysis - Wikipedia