COMP90051 Statistical Machine Learning Project Report

Team: guesswhat

Jianeng Li (1175332) **Yixuan Liang (1094908)** **Hongzhi Fu (1058170)**

1. Introduction

This report discusses 8 prediction methods our team used for the authorship link prediction task in project 1. The task is to predict future collaborations between different authors based on previous relationships. Unlike other machine learning tasks, to maximise the use of graph information in the link prediction task, all the methods we used in this project is based on the graph structure from the training data given.

2. Related work

Many works have been conducted on finding a representation for the graph in an unsupervised manner. Embedding methods such as Node2Vec (Grover & Leskovec, 2016) provides a good feature representation of the graph structure by maximizing the likelihood of preserving network neighborhoods of nodes. On other hand, some traditional feature engineering methods are also used to represent the node information in a graph. Several measures for solving link-prediction problems (Liben-Nowell, D., & Kleinberg, J., 2007) were proposed, including common neighbors, Jaccard's coefficient, preferential attachment, adamic-adar, etc.

3. Data and Data preprocessing

3.1 Data

A text file containing 9311 lines is given as the training data for the task. Each line in the file represents a collaboration between the authors in that line. For instance, a line with '1 2 3' represents three edges in the graph, '1-2', '1-3', and '2-3'. We end up extracting 16034 edges, representing collaborations, and 3767 nodes, representing authors, in the graph.

3.2 Data preprocessing

3.2.1 Label definition

Since only very limited training samples are given, we would use all the edges given to be our positive data in the supervised task. We pick non-existed edges between two randomly picked nodes as our negative samples. In order to get a balanced dataset, we repeat this process 16034 times and have the same amount of negative samples. The disadvantage is that this approach would make our model biased since the randomly picked edges might be positive in reality but we forced it to be negative when training our model.

3.2.2 Data leakage

In this supervised task, we need to separate the data into training samples and validation samples. Since we are utilising graph information in our models, two graphs should be constructed in the process. We have one graph containing only nodes and edges in the training dataset, i.e., we remove the edges and nodes in the validation set, and another graph containing all the information given in the text file. This will prevent data leakage in the training process and avoid overfitting.

4. Feature extraction methods

4.1 Embedding

Node2Vec (Grover & Leskovec, 2016) is an algorithm for feature learning which preserves neighborhood objectives in networks. Based on the Deepwalk (Perozzi et al., 2013) approach and the idea of Word2Vec (Mikolov et al., 2014) embedding, Node2Vec combines the Breadth-first Sampling (BFS) and Depth-first Sampling (DFS) techniques into a flexible biased random walk procedure to generate sequences for graph representation and embeds them into lower dimensional vector spaces. LINE (Tang et al., 2015) is also a feature embedding technique for large networks. The approach learns d-dimensional feature representations in two separate phases, capturing both first-order proximity and second-order proximity, i.e., the local pairwise proximity between two vertices and the similarity between their neighborhood network structures.

4.2 Link prediction similarity metrics

In our experiment, we attempted to extract 6 different features to represent information between two nodes in a graph, namely, common neighbors, shortest path, Jaccard coefficient, resource allocation index , adamic adar index and preferential attachment. All of the 6 features are able to conclude the information of two nodes in a graph, so the classifier can make a prediction based on the information.

5. Feature Engineering

5.1 Graph Construction

Since we have extracted all pairs of authorship information represented by edges, we need to construct a graph based on the edge information. Networkx provides a method Graph.read_edgelist() that generates a graph given a list of all edges. Thus, feature extraction methods can better utilize the information to extract useful features, such as shortest path, common neighbors, etc.

5.2 Logistic Regression

Logistic regression is a single binary classification model that takes the weighted sum of input features and outputs a probability value evaluated by sigmoid function. In our logistic regression model, we used all 6 features to fit a model under the best combination of hyperparameters.

5.3 Multi-Layer Perceptron

Multi-layer perceptron (MLP) is another non-linear classifier that outperforms logistic regression. The result shown in Table 1 demonstrates MLP achieves a higher AUC score on both the predefined validation set and the test set.

5.4 Decision Tree

We also employed decision tree models as the baseline to classify whether there is a link between two nodes. Two types of decision trees are used. One is a normal gradient-based decision tree (GBDT), and another is XGBoost, where the latter maximizes the utility of computation resources. The test AUC scores in Table 1 confirm that GBDT and XGBoost are two powerful decision tree models.

5.5 Stacking

Stacking is an ensemble method that builds a classifier of several base classifiers, i.e the output of the base classifier is the input of the stacking model to predict the final probability, and we selected MLP, GBDT and XGBoost as three base classifiers. The final result shows the stacking model achieves a slightly higher AUC score, compared with base classifiers.

6. Experiments and Results

Data Type	**Variants**	**AUC on Public leaderboard**	**AUC on Private leaderboard**
Embedding with MLP	unweighted Node2Vec	0.8224	-
	weighted Node2Vec	**0.8462**	**0.8401**
	LINE	0.7713	-
Manually Extracted Features	Logistic Regression	0.8334	-
	MLP	0.8515	-
	XGBoost	0.8203	-
	GBDT	0.8253	-
	Stacking	**0.8595**	**0.8228**

Table 1. Performance of Two Feature Extractors on Public and Private Leaderboard

7. Discussion

In our embedding as feature experiments, we found that the best result comes from feature embedding using Node2Vec on weighted graphs. Node2Vec embeddings utilise the weights in the graph for generating random walks, which adds additional information to the vector space compared to the ones using unweighted graphs, hence the improvement of results. The result from using LINE as embedding features has the least satisfying outcome among the three, since the method cannot be tuned for optimal performance. We were aware of bias introduced by our sampling strategy. Thus, we did additional experiments on different sampling of the negative data and stacked results from them. The AUC score is not satisfactory on the public leaderboard, but achieved a higher score on the private leaderboard.

Feature engineering is a promising way to build a high-accuracy model, because the extracted features can be a good representation of node information and its lower feature space accelerates the training procedure. Another finding during the experiment is that the shortest path feature makes a greatest contribution to the final AUC score, but our final submitted model is the combination of 6 features, in case of overfitting.

The final approach is using Node2Vec to generate embedding and MLP as the classifier. Although link prediction similarity metrics are intuitive and easy to extract, the training data only contains limited features. Embedding contains rich information about the correlation between authors. Compared with the tree model and logistic regression, MLP is more suitable for learning high-dimensional feature space.

Reference

Grover, A., & Leskovec, J. (2016, August). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015, May). Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web* (pp. 1067-1077).

Liben-Nowell, D., & Kleinberg, J. (2007). The link-prediction problem for social networks. *Journal of the American society for information science and technology*, *58*(7), 1019-1031.