COMP90051 Project Report

Introduction

Link prediction is a common technique used in various circumstances, for instance, friend recommendation in social application such as Facebook. However, Facebook link prediction is based on undirected graph, but in this project, we were given a graph with directed edges to recommend potential account the user may be interested in which means some of the techniques used in undirected graph may not work properly or only work with minor amendment. Moreover, the total number of nodes and edges of the given graph are approximately 5 million and 23 million respectively which means, potentially, there are approximately 24 trillion disconnected edges in total. It will consume a very long time to process and learn from the whole graph directly.

Methods

1. Simple Heuristic

There are many simple heuristics which can be used for link prediction. I selected Katz score and Adamic/Adar score initially. However, I apply minor changes to Katz to precisely describe the local information between two nodes, as address in the followings.

a. Katz

Instead of running Katz on the whole graph for the given nodes, I firstly extract a local subgraph sG of the original graph G (Zhang, & Chen, 2018) to restrict redundant information and reduce computation time. Formally, Katz score is defined as

$$Katz(x,y) = \sum_{l=1}^{\infty} \beta^l \left| Path_{x,y}^l \right|$$

β is a penalty factor for long distance and $Path_{x,y}^l$ is the collection of paths between x and y with length l. To approximate this, the upper bound of the path length is set to 3 since the set of paths $\left\{ Path_{x,y}^l \mid l > 3 \right\}$ rarely provide information about link existence from x to y in the real world. To convert it to a probability, I eventually adopted the following formula

$$prob(x,y) = \tanh\left(2 \sum_{l=1}^{3} \frac{1}{8}^{l-1} \left| Path_{x,y}^l \right| \right)$$

In normal words, if the link from x to y indeed exists, it will guarantee to generate a probability close to 1 (at least $\tanh 2$) and simultaneously heavily penalising longer paths. This is not a bad approach and it yields a ROC-AUC score of about 0.7 (and this is the best I can get with this method by logically augmenting β and the upper bound of l).

b. Adamic/Adar

This is a simpler heuristic which only needs to consider the common neighbours. The formula is given as

$$\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log|\Gamma(z)|}$$

$\Gamma(x)$ is the set of neighbours of vertex x. Since we are making link prediction on directed graph, there is two types of neighbours, i.e. $pred(x)$ denotes the predecessors of vertex x and $succ(x)$ denotes the successors of vertex x. Apply the transformation, Adamic/Adar score is calculated by

$$AA(x,y) = \sum_{z \in succ(x) \cap pred(y)} \frac{1}{\log|succ(z)|}$$

To transform it into a probability, I apply $tanh$ again to the score given by

$$prob(x,y) = \tanh \sum_{z \in succ(x) \cap pred(y)} \frac{1}{\log(|succ(z)|+1)}$$

The reason I bias the number of successors of their common neighbours by 1 is to avoid circumstance when $|succ(z)| = 1$. In this case, $prob(x,y) = \tanh \infty = 1$ which implies

$$link(x,y) \wedge link(y,z) \Rightarrow link(x,z)$$

$link(a,b)$ means there is a directed edge from a to b. This is not logically correct in the real world. This gives a ROC-AUC score of about 0.8, and this is the final approach.

2. Random Walk Similarity

In an undirected graph, two nodes are potentially connected if they are similar. To test if this work well on a directed graph, I implement a random walker with Alias Sampling to randomly choose the candidate node from the successors of node where the random walker currently located at (the node2vec embedding of a vertex). Unsurprisingly, the ROC-AUC scores of this pure similarity-based method are all about 0.66.

3. Classification/Regression

In this type of undirected graph, all weights of the existing edges are simply 1. Therefore, binary classification or complicated regression can be applied to this problem, I only draw approximately 200 thousand existing edges accompany by the same number of non-existing edges to build the classifier and regressor. However, these generally performed worse than all other methods based on the features I extracted from the graph (around 0.6 ROC-AUC score).

Analysis

node2vec embedding for graph is generally the same as word2vec embedding for document. The walk length and number of walks affect the paths. Another factor is the number of adjacent nodes to be taken account into in a path, which is similar to the n-gram choice in word embedding. Hence it works well in an undirected graph since mutual link (such as friendship in Facebook) between two vertices is implied in such type of graph but its performance decreases on directed edge in Twitter link prediction. The method I applied Katz centrality has problem with penalisation of number of paths of a specific length. It simply counts the number of paths, but it ignored the factor of individual nodes on each path. This is also the reason Adamic/Adar method outperforms the Katz method since it rewards the rare feature. I tried to extract various features (including the heuristic scores described previously) for the classifiers and regressors, but it is still worse than the simplest method described before.

I also attempted to build a Deep Graph Convolutional Neural Network, but it fails to run due to the memory limit even though I applied the subgraph extraction.

Reference List

Adams, R. (2013, March 3). The Alias Method: Efficient Sampling with Many Discrete Outcomes. Retrieved September 2, 2018, from https://hips.seas.harvard.edu/blog/2013/03/03/the-alias-method-efficient-sampling-with-many-discrete-outcomes/

Dong, L., Li, Y., Yin, H., Le, H., & Rui, M. (2013, September 17). The Algorithm of Link Prediction on Social Network. September 4, 2018, http://dx.doi.org/10.1155/2013/125123

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable Feature Learning for Networks. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 16. doi:10.1145/2939672.2939754

Sadraei, A. (2014). Link Prediction Algorithms. Retrieved August 31, 2018, from http://be.amazd.com/link-prediction/

Zhang, M., & Chen, Y. (2018, February 27). Link Prediction Based on Graph Neural Networks. Retrieved September 2, 2018, from https://arxiv.org/abs/1802.09691

Zhang, M., Cui, Z., Neumann, M., & Chen, Y. (2018). An End-to-End Deep Learning Architecture for Graph Classification. AAAI.