ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

Social Network Embedding Method Basing Loop-free Path

Benyu Wang[a,*], Shufan Peng[b]

Department of Information and Network Security People's Public Security University of China, Beijing, China
[a]201621430015@stu.ppsuc.edu.cn, [b]8454484102@qq.com

Abstract: *Network embedding's goal is to learn low-dimensional feature representations of nodes in a network and use the learned features in various analysis tasks of the network, such as node classification, link prediction, community discovery and recommendation. Existing network embedding methods do not make sufficient use of higher-order structural information in social networks and are not ideal for application in social networks. A Loop-free Path Combined with Attributes Network Embedding (LFNE) method is proposed to solve these problems. The algorithm first calculates the node higher-order structure similarity based on the loop-free paths between nodes, eliminates the influence of loop paths and large degree nodes on node structure similarity, and enables the network embedding method to better fuse the higher-order structure information of social networks. The node structural similarity is finally applied to the low-dimensional feature representation of the learned nodes in the stacked denoising autoencoder. Experimental comparisons with representative algorithms in recent years on three social network datasets show that the LFNE algorithm can achieve relatively significant results in node classification and link prediction experiments, with better network embedding performance.*

Keywords: *Network embedding, social network, loop-free path, stacked denoising autoencoder*

1. Introduction

In recent years, with the rapid development of Internet technology, the scale of social networks has been increasing, and social network analysis tasks have become more and more important, such as node classification and link prediction. The node classification task can reasonably classify the nodes in the network, which has important application value in real life, such as recommending friends to users by their category tags in social platforms, and accurately delivering ads to users who may be interested in smart advertising. The link prediction task can effectively analyze the network topology, predict the lost edges or possible future edges of the network, and has promising applications in social networks, such as social platform user expansion and e-commerce marketing. Various novel techniques have been used in social network analysis to explore potential user attributes, and network embedding[1] is one of them.

Network embedding methods based on network structure information use the network topology to capture the low-order or even high-order similarity of nodes, and thus obtain a low-dimensional feature representation of the network nodes. Perozzi et al. proposed the DeepWalk algorithm[2], which applies the word2vec model in NLP[3] to the random wandering sequences generated by the algorithm to obtain an embedding representation based on the node contextual structure. Tang et al. proposed the LINE algorithm[4], which combines the first-order and second-order structure information of the network, minimizes the KL divergence to obtain the network embedding results. Cao et al. proposed the GraRep algorithm[5], which extends the first- and second-order structural information of the LINE algorithm to higher-order structural information. Grover et al. proposed the Node2vec algorithm[6], which combines depth-first search and breadth-first search to optimize the random wandering sequence generation method in Deepwalk. Cao et al. proposed the DNGR algorithm[7] to obtain higher-order structural information representation of nodes by random surfing and then perform network embedding by stacked denoising autoencoders. Wang et al. proposed SDNE algorithm[8] to obtain the network embedding results by deep autoencoder. RIBERIO et al. proposed Struc2vec algorithm[9] to obtain the network embedding results by defining node similarity through node space structure similarity. Wang et al. proposed the edge2vec algorithm[10] to improve the node-based network embedding method to

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

obtain network embedding results by edges in the network. Since the network structure information is sparse, the single network embedding method using network structure information does not work well experimentally in large sparse networks.

Although existing network embedding methods have demonstrated their effectiveness in different scenarios, this paper finds in its research that there are still several problems with existing network embedding methods applied to social networks as follows:

Higher-order structural information of social networks is underutilized. Higher-order structure information can fully reflect the connectivity between nodes, so most structure-based network embedding methods incorporate higher-order structure information to improve the network embedding effect. However, these methods do not remove the influence of loop paths and large-degree nodes, making each node have higher similarity with large-degree nodes and the network embedding effect is not satisfactory.

To solve the above problem, this paper proposes a Loop-free Path Combined with Attributes Network Embedding (LFNE) method for social network embedding. The main contributions of this paper are as follows:

We propose a mathematical expression to calculate the number of loop-free paths between nodes in social networks and propose Similarity Metrics Based on Loop-free Path (SLP) to eliminate the influence of loop paths and large nodes, so that the similarity metrics can better integrate the information of higher-order paths between nodes in social networks. The similarity metrics can better integrate the higher-order path information between social network nodes, and the calculation results are more accurate.

Learning the low-dimensional feature representation of nodes by stacked denoising autoencoders not only learns the data distribution in the network, but also obtains highly nonlinear features of the network. Experimental results of node classification and link prediction on public datasets indicate that the LFNE algorithm can obtain a more comprehensive node feature vector.

2. Related Works

2.1. Network Basics

Given a complex network $G=(V,E)$ to represent a social network, the set of individuals in the network is represented by the set of nodes $V=\{1,2,\cdots,N\}$, $|V(G)|=N$, and the connections of individuals in the network are represented by the set of edges $E=\{e=(i,j)\,|\,1\le i,j\le N,\ i\le N,\ i\ne j\}$ to represent the edge $e=(i,j)$ by $i-j$.

Denote the adjacency matrix $A\in R^{N\times N}$ of network, defined as

$$a_{ij}=\begin{cases}1, & i-j\in E(G)\\ 0, & i-j\notin E(G)\end{cases} \tag{1}$$

Let $S(i)=\{j\,|\,j\in V(i\ne j)\wedge(i,j)\in E\}\cup\{i\}$, call $S(i)$ a neighbor node of node .

Let $d_i=\left|j\,|\,j\in V(i\ne j)\wedge(i,j)\in E\right|$, where the number of elements in the set is denoted by $|*|$, and call d_i the degree of node i.

2.2. Loop-Free Path Algorithm

Removing the loop paths in each order path of nodes makes each node in the node connectivity path different from each other, which can effectively reduce the dependence of each node on the large degree nodes, so the connectivity between nodes can be measured by the loop-free paths between nodes.

Chen et al[11] introduced an intermediate matrix to calculate the number of loop-free paths of each order between nodes. For the case where the path order is less than or equal to 5, it gives the exact mathematical expression.

Given an adjacency matrix A of a network G, A^k denotes the k th power of the adjacency matrix

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

A. A^k gives the number of paths of each order (including loop paths) between any nodes in the network G. Define P^k to denote the number of loop-free paths among the paths of each order in network G.

For the first order path of the network, which is the adjacency matrix of the network, there is no loop path, therefore $A^1 = P^1$.

For the second-order paths of the network, which are common neighbors between nodes, there is also no loop path, hence $A^2 = P^2$.

For third-order and higher paths of the network, loop paths may appear in the inter-node paths. To facilitate the calculation of the number of loop-free paths between third-order and higher nodes, Chen et al. introduced an intermediate matrix Q^k. Q^k represents a k-order path consisting of a first-order path after a k-1 order loop-free path and is defined as shown in Eqs. (2) and (3):

$$Q^k = P^{k-1}P^1 = P^{k-1}A^1 \tag{2}$$

$$q_{ij}^k = \sum_{m=1}^{N} p_{im}^{k-1} a_{mj} \tag{3}$$

where p_{im}^{k-1} denotes the k-1st order loop-free path between node i and node m, and a_{mj} denotes the connection between node m and node j. It can be seen that Q^k can be calculated from P^{k-1} and that P^1 and P^2 are known. therefore, Q^k can be found by recursive method.

Consider a third-order path shaped as $i-a-b-j$. $q_{ij}^3 = \sum_{m=1}^{N} p_{im}^2 a_{mj}$. q_{ij}^3 denotes a second-order acyclic path between nodes b, the common neighbor of nodes i and j. Although q_{ij}^3 has removed most of the cyclic paths, some cyclic paths still exist. When $a = j$, q_{ij}^3 then there is a loop path in the path shown, so removing q_{ij}^3 in the case of $a = j$ is the $i-a-b-j$ third-order acyclic path, defined as follows:

$$p_{ij}^3 = q_{ij}^3 - a_{ij}(d_j - 1) \tag{4}$$

where d_j is the degree value of node j. Based on such an idea, consider the fourth-order path $i-a-b-c-j$. q_{ij}^4 in the case of removing $a = j$ and $c = j$ is the fourth-order loop-free path, defined as follows:

$$p_{ij}^4 = \begin{cases} q_{ij}^4 - \left(d_j - 1\right) p_{ij}^2, & if\ a_{ij} = 0 \\ q_{ij}^4 - \left(d_j - 2\right) p_{ij}^2 & if\ a_{ij} = 1 \\ -\left(q_{jj}^3 - 2p_{ij}^2\right), & \end{cases} \tag{5}$$

Similarly, the number of fifth-order acyclic paths can be found and defined as follows:

$$p_{ij}^5 = \begin{cases} q_{ij}^5 + \left\{ \sum_{a \in S(i) \cap S(j)} [p_{aj}^2 - (d_j - 1)\ p_{ij}^3] \right\} & if\ a_{ij} = 0 \\ \quad + \left\{ \sum_{a \in S(i) \cap S(j)} (-q_{jj}^3 + 2p_{aj}^2) \right\}, & \\ q_{ij}^5 + \left\{ \sum_{a \in S(i) \cap S(j)} [p_{aj}^2 - \left(d_j - 1\right) - \left(d_j - 2\right) p_{ij}^3] \right\} & \\ \quad + \left\{ \sum_{a \in S(i) \cap S(j)} (-q_{jj}^3 + 2p_{aj}^2 + 2p_{ij}^2 - 2) \right\} & if\ a_{ij} = 1 \\ \quad + \left\{ -q_{jj}^4 + p_{ij}^3 + p_{ij}^2 (p_{ij}^2 - 1) \right\}, & \end{cases} \tag{6}$$

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

3. LFNE Algorithm

To eliminate the influence of loop paths and large degree nodes on the structural similarity of nodes in social networks, and make the network embedding method better fuse the higher-order structural information of social networks, this paper proposes a social network embedding method LFNE that fuses loop-free paths. LFNE mainly consists of four steps: calculating the structural similarity be-tween social network nodes based on the similarity metric SLP of loop-free paths between nodes; constructing a stacked denoising auto-coder to extract features from the matrix to obtain a low-dimensional feature representation. Similarity generates a structural probability matrix; constructs a stacked denoising autoencoder to extract features from the matrix and obtains a low-dimensional feature representation of the network.

3.1. Node Structure Similarity Calculation

To make the similarity metric can better integrate the higher-order structure information of social networks, this paper proposes a similarity metric based on the acyclic path between nodes based similarity metric SLP, which is defined as follows:

$$S_{ij}^{slp} = \sum_{k=1}^{K} \beta(k) p_{ij}^{k} (k \le 5) \tag{7}$$

In a social network, the shorter the direct connection or connectivity path between two nodes, the higher the similarity between the two nodes and the stronger the connection. The node path information of lower order will be considered more in the SLP metric compared to the node path information of higher order. Therefore, the weight decay function is introduced, and when the inter-node path increases, the path weight will decrease accordingly. We use the exponential function as the weight decay function of the SLP index, and the weight decay function is shown in equation (8):

$$\beta(k) = \omega^{k} \tag{8}$$

where, ω is the weight decay function factor, $\omega \in (0,1)$. With the help of the weight decay function, the SLP index can be made to take more into account the low-order structural information of the nodes while incorporating the high-order structural information of the nodes.

Next, the node similarity under the SLP metric is normalized to obtain the inter-node structural probability matrix, and the node i and node j structural probabilities are defined as follows.

$$H_{ij} = \frac{S_{ij}^{slp}}{\sum_{k=1}^{|V|} S_{ik}^{slp}} \tag{9}$$

3.2. Stacked Denoising Autoencoder

Autoencoders can efficiently extract data features and have been widely used in representation learning tasks in several fields. The traditional autoencoder consists of three parts: input layer, hidden layer, and output layer. The input layer data is mapped to the hidden layer space by the encoder, and then the hidden layer space data is mapped to the recon-structed data in the output layer by the decoder, and feature extraction is achieved by reducing the error between the input data and the reconstructed data. The denoising autoencoder is based on a variation of the traditional autoencoder. The denoising autoencoder can enhance the robustness of autoencoder well by training the training data with noise contamination and by training to predict the original data without contamination. The denoising autoencoder first zeroes some cells in the original input data $x_i \in R^N$ to add noise to obtain $\hat{x}_i \in R^N$,then mapping $\hat{x}_i$ to the hidden layer space by the encoder part yields $h_i \in R^d$, afterwards, a is mapped into reconstructed data b by the decoder part. Training denoising autoencoders by optimizing the loss function between metrics x_i and z_i , the definitions are as follows:

$$h_i = f(W\hat{x}_i + b) \tag{10}$$

$$z_i = f(W'x_i + b') \tag{11}$$

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

where W and b are the weight matrices and bias vectors of the encoder, and W' and b' are the weight matrices and bias vectors of the decoder. $f(\cdot)$ is the activation function, and the Relu function is used in this paper, defined as follows:

$$f(a)=\begin{cases}0 & a\le 0\\ a & a>0\end{cases} \tag{12}$$

The loss function uses a cross-entropy function, defined as follows:

$$L(X,Z)=\frac{1}{n}\sum_{i=1}^{n}[x_i\lg z_i+(1-x_i)\lg(1-z_i)] \tag{13}$$

In order to obtain a better low-dimensional feature representation of the input data, this paper uses a stacked denoising autoencoder to construct a complete deep denoising autoencoder to learn the low-dimensional feature representation of the input data. The stacked denoising autoencoder consists of several denoising autoencoders. In the training process of the stacked denoising autoencoder, each time a denoising autoencoder is trained. The hidden layer is used as the input of the next denoising autoencoder, and each denoising autoencoder is connected in turn to form a stacked denoising autoencoder. The last denoising autoencoder hidden layer representation is the low-dimensional feature representation of the input data.

3.3. Algorithm Flow

The specific method flow of LFNE, a social network embedding method that incorporates loop-free paths, is shown in Algorithm 1.

Algorithm 1: LFNE algorithm

Input: Social network G, network adjacency matrix A and setting related parameters
Output: Low-dimensional feature matrix Z

1: The structural similarity between nodes based on acyclic paths S^{slp} is calculated according to equation (7)
2: Calculate the structural probability matrix H according to equation (9)
3: Building stacked denoising autoencoders
4: For i=1 to m do:
5: Input structural probability matrix H
6: Training Stacked Denoising Autoencoder by equation (13)
7: End for
8: Return Z

First, the input part of the LFNE algorithm consists of three components: the social network G, the network adjacency matrix A, including the path weight parameter β, and the inter-node path length k; Parameters required for the stacked denoising autoencoder feature extraction stage, including the number of stacked denoising autoencoders n, dimension d, and training times m.

Rows 1 to 2 of Algorithm 1 first obtain the similarity S^{slp} between nodes based on acyclic paths through matrix A, capturing the higher-order structural similarity of the social network, and then generate matrix H.

Lines 3 to 8 of Algorithm 1 construct a stacked denoising autoencoder with input matrix H, learn the low-dimensional feature representation of H, and execute the loop m times to finally obtain the low-dimensional feature matrix Z of H.

4. Experiment

To test the feasibility and effectiveness of the algorithm proposed in this paper, the LFNE algorithm is compared with representative algorithms in recent years for experiments. Among them, Deepwalk[2], Node2Vec[6], DNGR[7], SDNE[8] are network embedding methods based on network structure information.

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

4.1. Experimental Data Set

Specific information on the data sets used in the experiments is shown in Table 1:

Table 1: Basic information of Real network dataset.

Datasets	Nodes	Edges	Label
BlogCatalog[12]	5196	171743	6
Flickr[13]	7575	239738	9
Email[14]	1005	25751	-

4.2. Parameter Setting

In the algorithm proposed in this paper, the stacked denoising autoencoder is used to learn the low-dimensional feature representation of the nodes. In order to make the stacked denoising autoencoder can better acquire the low-dimensional feature representation of nodes, different autoencoder structures are used for different data sets. The corresponding autoencoder structures for each data set are shown in Table 2:

Table 2: Structure of stacked denoising autoencoder.

Datasets	Structure of autoencoder
BlogCatalog	5196-1024-256
Flickr	7375-2048-256
Email-Eu-core	1005-512-64

The node vector dimensionality of the comparison algorithm used in the experiments is the same as that of the algorithm in this paper. The SLP metric parameter in the algorithm of this paper is taken as 0.3 with reference to the literature[11]. the stacked denoising autoencoder model is trained using the Adam optimizer with the number of iterations set to 500 and the learning rate of 0.002. The inter-node path length is taken as 5.

4.3. Node Classification

Node classification is an important task in social network analysis to predict the kinds of untagged node labels in the network from the tagged nodes, and is often used to evaluate the performance of network embedding methods. The datasets selected for the node classification experiments are BlogCatalog and Flickr datasets, and the evaluation metrics are Micro-F1 score and Macro-F1 score. The low-dimensional feature representations of the nodes are first learned by each algorithm, and then 10% to 90% (in steps of 20%) of the nodes are randomly selected as marker nodes for training, and the remaining nodes are used as tests. The experimental results are shown in Tables 3-6:

Table 3: Micro-F1 of node classification on BlogCatalog.

Model	Ratio of training sample				
	10%	30%	50%	70%	90%
Deepwalk	0.5835	0.6531	0.6743	0.6924	0.6834
Node2Vec	0.5971	0.6423	0.6701	0.6817	0.6766
DNGR	0.6663	0.6745	0.7025	0.7077	0.7123
SDNE	0.6743	0.7142	0.7356	0.7295	0.7383
LFNE	0.7374	0.7293	0.7802	0.7537	0.7592

Table 4: Macro-F1 of node classification on BlogCatalog.

Model	Ratio of training sample				
	10%	30%	50%	70%	90%
Deepwalk	0.5765	0.6513	0.6711	0.6848	0.6754
Node2Vec	0.5873	0.6374	0.6642	0.6856	0.6801
DNGR	0.6696	0.6712	0.6934	0.7012	0.7188
SDNE	0.6645	0.7076	0.7296	0.7215	0.7345
LFNE	0.7143	0.7561	0.7622	0.7547	0.7723

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

Table 5: Micro-F1 of node classification on Flickr.

Model	Ratio of training sample				
	10%	30%	50%	70%	90%
Deepwalk	0.4331	0.5021	0.5133	0.5196	0.5243
Node2Vec	0.4185	0.4795	0.5107	0.5184	0.5255
DNGR	0.4763	0.5212	0.5714	0.5963	0.6122
SDNE	0.5014	0.5417	0.5684	0.5947	0.6033
LFNE	0.5245	0.5967	0.5959	0.6453	0.6740

Table 6: Macro-F1 of node classification on Flickr.

Model	Ratio of training sample				
	10%	30%	50%	70%	90%
Deepwalk	0.4314	0.4974	0.5085	0.5166	0.5214
Node2Vec	0.4147	0.4734	0.5066	0.5121	0.5217
DNGR	0.4711	0.5167	0.5641	0.5874	0.6045
SDNE	0.4971	0.5364	0.5614	0.5876	0.5941
LFNE	0.5133	0.5757	0.5990	0.6278	0.6429

According to the experimental results, the low-dimensional feature vectors of nodes learned by the LFNE algorithm can obtain better node classification results on each dataset compared to other algorithms. The DNGR algorithm incorporates higher-order structural information between nodes without eliminating the effect of loop paths on node information. The experimental results show that the LFNE algorithm has significantly improved the node classification Micro-F1 scores and Macro-F1 scores on the BlogCatalog and Flickr datasets compared to the DNGR algorithm. This demonstrates that eliminating loop paths can improve the network embedding effect of social networks.

4.4. Link Prediction

Link prediction is also an important task in social network analysis, predicting potential or future connection processes between nodes based on existing network connections. Link prediction is also commonly used to evaluate the performance of network embedding learning methods. The datasets selected for the link prediction experiments in this section are BlogCatalog, Flickr, and Email, and the evaluation metric is AUC. In the experiments, 10% of the connected and unconnected data were randomly selected as tests, 10% of the connected data were used as validation, and the rest of the data were used as training. The results of the experiments are shown in Figure 1:

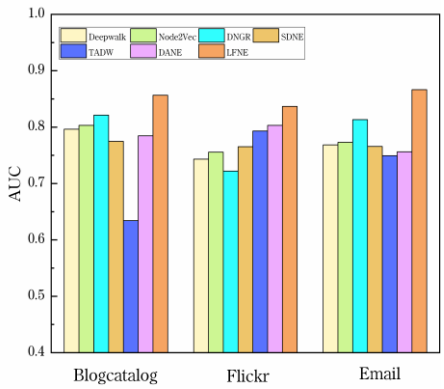

Figure 1: AUC index results of link prediction with different algorithms.

5. Conclusions

We propose a social network embedding method LFNE that fuses loop-free paths. This algorithm can eliminate the influence of loop paths and large degree nodes on node similarity and improve the social network embedding effect. The results show that the algorithm is feasible and effective. There are still several shortcomings in this paper. This paper does not find a universal algorithm to calculate the higher-order loop-free path number. This paper does not consider the interaction information between nodes for the network embed-ding aspect. In the next step, we will try to find a generalized

ISSN 2616-5775 Vol. 4, Issue 7: 73-80, DOI: 10.25236/AJCIS.2021.040711

algorithm to calculate the number of higher-order acyclic paths, improve the scalability of the acyclic path algorithm, and try to integrate the interaction information between nodes into the network embedding to obtain a better network embedding effect.

References

[1] Cunchao TU, Cheng YANG, Zhiyuan LIU, Maosong SUN. Network representation learning: an overview[J]. Scientia sinica informationis, 2017, 47(8): 980-996.

[2] Bryan Perozzi, Rami Al-Rfou, Steven Skiena. Deepwalk: Online learning of social representations [C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014: 701-710.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean. Distributed representations of words and phrases and their compositionality[J]. arXiv preprint arXiv:1310.4546, 2013.

[4] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, Qiaozhu Mei. Line: Large-scale information network embedding[C]//Proceedings of the 24th international conference on world wide web. 2015: 1067-1077.

[5] Shaosheng Cao, Wei Lu, Qiongkai Xu. Grarep: Learning graph representations with global structural infor-mation[C]//Proceedings of the 24th ACM international on conference on information and knowledge management. 2015: 891-900.

[6] Aditya Grover, Jure Leskovec. node2vec: Scalable feature learning for networks[C]//Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016: 855-864.

[7] Shaosheng Cao, Wei Lu, Qiongkai Xu. Deep neural networks for learning graph representations [C]//Proceedings of the AAAI Conference on Arti-ficial Intelligence. 2016, 30(1).

[8] Daixin Wang, Peng Cui, Wenwu Zhu. Structural deep network embedding[C]//Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016: 1225-1234.

[9] Leonardo F.R. Ribeiro, Pedro H.P. Saverese, Daniel R. Figueiredo. struc2vec: Learning node representations from structural identity[C]//Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. 2017: 385-394.

[10] Changping Wang, Chaokun Wang, Zheng Wang, Xiaojun Ye, Philip S. Yu. Edge2vec: Edge-based Social Network Embedding[J]. ACM Transactions on Knowledge Discovery from Data (TKDD), 2020, 14(4): 1-24.

[11] Jianshe Wu, Nan Cheng, Chaojie Zhou, Hefei Che, Chunlei Han, Qin Liu. Computing the Number of Loop-free k-hop Paths of Networks[J]. IEEE Transactions on Services Computing, 2020.

[12] Xiao Huang, Jundong Li, Xia Hu. Accelerated attributed network embedding[C]//Proceedings of the 2017 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2017: 633-641.

[13] Xiao Huang, Jundong Li, Xia Hu. Label informed attributed network embedding[C]//Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. 2017: 731-739.

[14] HUANG X, LI J, HU X. Label informed attributed network embedding[C]//Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. Cambridge, Feb 6-10, 2017. New York: ACM, 2017: 731-739.