A Link Prediction Method Based on Graph Neural Network Using Node Importance

Luomin Du, Yan Tang(✉), and Yuan Yuan

College of Computer and Information Science,
Southwest University, Chongqing 400715, China
{duluomin5686,y947136085}@email.swu.edu.cn, ytang@swu.edu.cn

Abstract. Link prediction is a challenging task in complex networks and data mining. Its primary purpose is to predict the possibility of links in the future. Link prediction has many application scenarios, such as product recommendations on e-commerce platforms, friend mining on social platforms, etc. Existing link prediction methods focus on utilizing neighbor and path information, ignoring the contribution of link formation of different node importance. For this reason, we propose a novel link prediction method based on node importance. The importance of node is calculated by using the topology structure of the directed network and the path information between nodes, and a graph convolutional network model suitable for directed graphs is designed. The importance of nodes is used to control the model to aggregate the neighbor information, thereby generating the vector representation of the node and obtaining the prediction score through the multi-layer perceptron (MLP). We investigate the proposed method and conduct extensive experiments on 6 real-world networks from various domains. The experiments results illustrate that the proposed method outperforms existing state-of-the-art methods.

Keywords: Link prediction · Graph convolutional networks · Data mining

1 Introduction

Link prediction aims to predict the possibility of a connection between two nodes that have not yet generated edges through the existing information of the network, including but not limited to network structure, node attributes, edge attributes, etc. Link prediction has many practical application scenarios, such as product recommendations on e-commerce platforms [13] and friend relationship mining on social platforms [1].

The existing link prediction methods can be roughly divided into two categories [7]: heuristic-based methods and learning-based methods. Heuristic methods often use a limited variety of network features to predict, for example, the common neighbor (CN) index, which argues that the more shared neighbors

G. Memmi et al. (Eds.): KSEM 2022, LNAI 13369, pp. 337–348, 2022.
https://doi.org/10.1007/978-3-031-10986-7_27

between two nodes, the more likely it is to generate link. Therefore, it is based on the shared information between nodes as a feature. Based on this idea, several other similar indicators are derived, including the Salton index [9], Jaccard [16], etc. These methods are still very popular in link prediction due to their advantages of low computational complexity and strong interpretability. However, because these methods usually only consider a limited number of specific network characteristics, they lack universality for networks of various types. For example, the common neighbor index has good performance in social networks. However, it performs poorly in biological networks [23].

With the rapid development of machine learning [12,18,20], more and more scholars consider utilizing the deep learning for link prediction. The methods of deep learning do not specify characteristics of the network but use the strong fitting ability to learn the characteristics of the network automatically. Theoretically, they can adapt to networks of various types, not limited to a specific type of network. Zhang et al. [33] combine link prediction with the graph classification task of deep learning. Graph convolutional networks (GCNs) [5,11,17,28,37] are specially used to deal with graph-related tasks [15,22]. Zhang et al. [34,35] proposed a novel graph convolutional network and used it to replace multi-layer perceptron for link prediction, which achieved better results. Cai et al. [6] went further by converting the subgraph to be predicted into a line graph. In this way, the link will be converted into a node, and the graph classification task will be converted into a node classification task, further reducing the computational overhead.

However, most methods ignored the difference in the contribution of the links formed by nodes of different importance. For example, in social networks, opinion leaders can influence the attitudes of ordinary users more, resulting in more links. Therefore, it can be considered that the important information of the node has a significant influence on link formation.

In order to use the importance information of nodes to improve the accuracy of link prediction, we propose a link prediction method based on the node importance (LPNI) in this paper. We examine the network from two different perspectives, defining the global and local importance of nodes. The global importance represents the node's importance for the entire network, while the local importance represents the node's importance for different nodes in the local curtain scope. The local importance of nodes constitutes the local importance matrix. We use it to construct a Laplacian matrix which is suitable for directed graphs and further design a graph convolutional network model suitable for directed graphs. After the graph convolutional network, the node vector representation containing the importance information is obtained, and we utilize the MLP to calculate the prediction result. We investigate the proposed method and conduct extensive experiments on 6 real-world networks from various domains. The experiments results illustrate that the proposed method outperforms existing state-of-the-art methods.

The rest of the paper is organised as follows. In Sect. 2, we state some relevant prerequisites for link prediction tasks. In Sect. 3, we detail the proposed method.

In Sect. 4, we present our experiments and analyze the experimental results. Finally Sect. 5 gives some concluding remarks with possible future directions.

2 Preliminaries

Given a network $G(V, E)$, $V = \{v_1, v_2, \cdots v_N\}$ denotes the set of nodes of the network, where N represents the number of nodes, and $E = \{e_1, e_2, \cdots e_M\}$ denotes the set of edges of the network, where M represents the number of edges. A pair of nodes can also represent the edges of the network, that is, $e = \{v_x, v_y\}$ represents the edge between node v_x and v_y. We focus on the simple directed networks, which satisfy the following four conditions:

(1) There are no self-loop edges. That is, there will be no edges such as $e = \{v_x, v_x\}$ in the network.
(2) There are at most two edges in different directions between nodes. That is, for any two edges, there will be no situation where $e_i = e_j = \{v_x, v_y\}$.
(3) There is a directionality between the edges, that is, $\{v_x, v_y\} \neq \{v_y, v_x\}$.
(4) There can be weights on the connected edges. If there is an unweighted network, the weight of all existing connected edges is 1.

The task of link prediction is to give a network G, the predict method f assigns a score for certain unconnected nodes. The highest probability of node pair is connected.

3 Method

3.1 Global and Local Node Importance

The symbol $\boldsymbol{I}_G(v_x)$ is used to represent the global importance of node v_x, that is, the importance of the node v_x to the entire network. $\boldsymbol{I}_G \in \mathbb{R}^N$ represents the vector of the global importance of all nodes.

In some cases, the global importance of a node cannot well reflect the individual importance of different nodes. For example, in a scientist cooperation network, the data mining network includes a sub-network for link prediction. Experts of the link prediction sub-network usually have a more substantial influence in the field and do not necessarily significantly influence the whole data mining field. Therefore, The local importance of a node represents the importance of a node in a specific local range of the network. Since mining subnets is cumbersome, in order to simplify the representation of this information, the symbol $\boldsymbol{I}_L(v_x, v_y)$ is used to represent the importance of node v_y in the directed graph to the node v_x, that is, the local importance of the node. Therefore, in this paper, the local importance of a node refers to the relative importance between two nodes. Our study is based on directed networks, therefore, $\boldsymbol{I}_L(v_x, v_y) \neq \boldsymbol{I}_L(v_y, v_x)$. And the $\boldsymbol{I}_L \in \mathbb{R}^{N \times N}$ represents the local importance matrix formed by all node pairs.

The research of Zareie et al. [32] shows that the influence of a node is related to the positions of the node and the neighbor nodes in the network. Therefore, we use the k-shell decomposition algorithm [8] to divide the network into different sub-networks. The nodes in each sub-network have their corresponding ks values. The larger the value, the closer the distance to the network core. By examining the ks value of the node itself, calculate the node's global importance with the ks value of the neighbor node. Use the notation $ks(v_x)$ to denote the ks value of node v_x.

It is considered that the neighbor nodes also represent the important information of the node to a certain extent. Therefore, we further examine the ks value of neighbor nodes. The Shannon entropy of the distribution of ks values of neighbor nodes is calculated as representing the diversity of neighbor nodes. The calculation formula of neighbor diversity of node v_x is as follows

$$diversity(v_x) = -\sum_{k} p(k) \cdot \log p(k) \tag{1}$$

the above formula means to traverse all possible k values, where $p(k)$ represents the probability that the ks value of the neighbor node is k.

Consider a situation where two nodes v_x and v_y, have three neighbor nodes with different ks values. For example, the neighbor ks values of node v_x are 1, 2, 3, and the neighbor ks values of node v_y are 4, 5, 6, and Eq. 1 calculates that the neighbor diversity values of nodes v_x and v_y are the same. Therefore, it is not only necessary to consider the diversity of neighbor nodes, but also to measure the ks value of neighbor nodes. Consider using the mean ks value of neighbor nodes to construct the global importance of node. The formula for calculating the mean ks value of neighbors of node v_x is as follows:

$$mean\_ks(v_x) = \frac{\sum_{v_y \in \Gamma(v_x)} ks(v_y)}{d(v_x)} \tag{2}$$

where $d(v_x)$ represents the degree of node v_x. $\Gamma(v_x)$ represents the set of neighbor node.

The global importance calculation formula of node v_x is as follows:

$$\boldsymbol{I}_G(v_x) = ks(v_x) \cdot diversity(v_x) \cdot mean\_ks(v_x) \tag{3}$$

Note that in the process of calculating the global importance of nodes, we deliberately ignore the direction of the link for simplicity.

The theory of small-world networks [3] argues that some people who do not know each other in social networks can be linked together through a very short chain of acquaintances. It can be considered that the shorter the chain of acquaintances, the greater the probability that two people know each other, so we consider computing the local importance by computing the shortest path between two nodes. Define a specific path between nodes v_x and v_y as

$$path_{v_x, v_y} = \{v_1, e_1, v_2, e_2, \cdots, e_{l-1}, v_l\} \tag{4}$$

where $e_i = \{v_i, v_{i+1}\}$.

In a directed graph, the path is also directed. That is $path_{v_x,v_y} \neq path_{v_y,v_x}$. The definition symbol $|path|$ represents the length of the path, that is, the number of nodes on the path, and the distance between two nodes is defined as the length of the shortest path. The normalized global importance of all nodes on the shortest path are multiplied together as the local importance of nodes:

$$\boldsymbol{I}_L(v_x, v_y) = \prod_{v_z \in path_{v_x,v_y}} \boldsymbol{I}_{G-norm}(v_z) \tag{5}$$

where $\boldsymbol{I}_{G-norm}(v_z) = \boldsymbol{I}_G(v_z)/\max\{\boldsymbol{I}_G(v)\}$. If there are multiple shortest paths, choose the one with the largest local importance.

Therefore, it can be seen that all node pairs constitute a local importance matrix $\boldsymbol{I}_L \in \mathbb{R}^{N\times N}$, where N represents the count of node.

3.2 GCN with Local Node Importance

Considering the excellent performance of graph convolutional networks on graph-related tasks, we combine node importance information with graph convolutional networks (GCN). Existing graph convolutional network models usually have two combination forms for this kind of data structure. The first is to use each row of the nodes' local importance matrix as the attribute vector of the node, and the second is to use the nodes' local importance matrix as a special adjacency matrix that controls the calculation process of the information aggregation stage. Adopting the second form brings the following advantages:

1. Decouple the attribute information of the node from the scale of the network. That is, the dimension of the attribute vector of the node will not change with the scale of the network.
2. The graph convolutional network model can learn the information of neighbor nodes farther away, which alleviates the over-smoothing problem [19] to a certain extent and reduces the training parameters.

Inspired by the literature [21], we regard matrix $\boldsymbol{I}_L$ as the adjacency matrix of the network to construct the Laplacian matrix of the graph convolutional network. First, insert the auxiliary node v_ξ into the adjacency matrix, and let all nodes generate reciprocal links with it, so that the network is transformed into a strongly connected network, the weight from other nodes to the auxiliary node v_ξ is α, and the weight from the auxiliary node to other nodes is $1/N$. The weight of other links is reduced by $(1-\alpha)$ times. Use the symbol $\tilde{\boldsymbol{I}}_L \in \mathbb{R}^{(N+1)\times(N+1)}$ to denote the local importance matrix after inserting auxiliary node. The calculation formula is as follows:

$$\tilde{\boldsymbol{I}}_L = \begin{pmatrix} (1-\alpha)\boldsymbol{I}_L & \alpha\mathbf{1} \\ \frac{1}{N}\mathbf{1} & 0 \end{pmatrix} \tag{6}$$

This operation makes the local node importance matrix of the network aperiodic and irreducible, achieving Perron's theorem's precondition [14]. Using Perron's theorem, the left eigenvector $\tilde{\boldsymbol{\pi}} \in \mathbb{R}^{N+1}$ that owns all positive entries can be

obtained. We can decompose the $\tilde{\boldsymbol{\pi}}$ vector into two parts, namely $\tilde{\boldsymbol{\pi}} = \{\boldsymbol{\pi}_L, \pi_\xi\}$, $\boldsymbol{\pi}_L \in \mathbb{R}^N$ represents the approximation of the Perron vector composed of N nodes in the original network, and $\pi_\xi \in \mathbb{R}^1$ represents the Perron vector part of the auxiliary node. Diagonalize the $\boldsymbol{\pi}_L$ vector to get $\boldsymbol{\Pi}_L \in \mathbb{R}^{N\times N}$. And use $\boldsymbol{\Pi}_L$ to construct a Laplacian matrix suitable for directed graphs:

$$\mathcal{L} = \boldsymbol{E} - \frac{1}{2}(\boldsymbol{\Pi}_L^{\frac{1}{2}} \cdot \boldsymbol{I}_L \cdot \boldsymbol{\Pi}_L^{-\frac{1}{2}} + \boldsymbol{\Pi}_L^{-\frac{1}{2}} \cdot \boldsymbol{I}_L^T \cdot \boldsymbol{\Pi}_L^{\frac{1}{2}}) \tag{7}$$

The definition symbol $\hat{\boldsymbol{A}}$ is expressed as:

$$\hat{\boldsymbol{A}} = \frac{1}{2}(\boldsymbol{\Pi}_L^{\frac{1}{2}} \cdot \boldsymbol{I}_L \cdot \boldsymbol{\Pi}_L^{-\frac{1}{2}} + \boldsymbol{\Pi}_L^{-\frac{1}{2}} \cdot \boldsymbol{I}_L^T \cdot \boldsymbol{\Pi}_L^{\frac{1}{2}}) \tag{8}$$

We define our graph convolutional network model as two convolutional layers:

$$\boldsymbol{Z} = f(\boldsymbol{X}, \boldsymbol{I}) = Softmax(\hat{\boldsymbol{A}} ReLU(\hat{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}^0)\mathbf{W}^1) \tag{9}$$

where $\boldsymbol{Z} \in \mathbb{R}^{N\times d}$ is the vector representation matrix of nodes. $\boldsymbol{X} \in \mathbb{R}^{N\times 3}$ represents the matrix formed by the attribute vector of the node. We use the three-dimensional vector formed by the out-degree $d_{out}(v_x)$ and in-degree $d_{in}(v_x)$ and the global importance $\boldsymbol{I}_G(v_x)$ as the attribute vector of the node v_x. $\boldsymbol{W}^0 \in \mathbb{R}^{3\times h}$ and $\boldsymbol{W}^1 \in \mathbb{R}^{h\times d}$ represent trainable parameters, respectively. The details of process of the graph convolutional network are shown in Fig. 1.

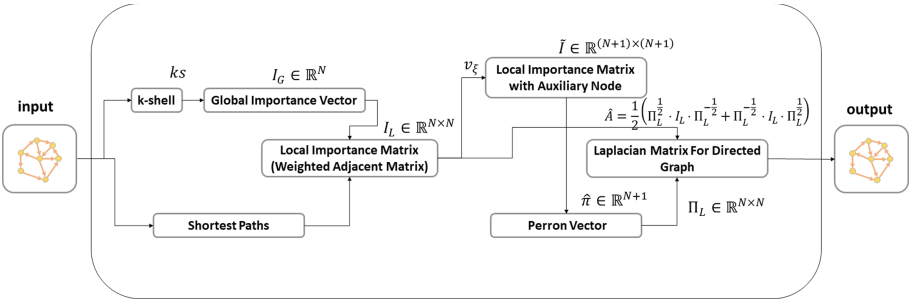

Fig. 1. Details of process of the graph convolutional network.

3.3 Make Prediction

The traditional method of the dot product of the node vectors in the undirected graph is not applicable because of the link direction. Consider using a multi-layer perceptron to splice the vectors of the two nodes so that the splicing result $[\boldsymbol{z}_{v_x}||\boldsymbol{z}_{v_y}] \neq [\boldsymbol{z}_{v_y}||\boldsymbol{z}_{v_x}]$, so the direction information is preserved. Our method computes the prediction score using a multi-layer perceptron of the form:

$$\hat{\boldsymbol{y}} = Softmax(\left[\boldsymbol{z}_{v_x} || \boldsymbol{z}_{v_y}\right] \boldsymbol{W}) \tag{10}$$

where $\hat{\boldsymbol{y}}$ is the prediction result vector, the first element represents the existing possibility of a link, and the other element represents the non-existing possibility of a link. $\boldsymbol{W} \in \mathbb{R}^{d \times 2}$ is a trainable parameter. And we utilize the cross entropy as our loss function.

$$loss = \sum_{n} \sum_{i} -y_i \log \hat{y}_i + \frac{\mu}{2n}(\boldsymbol{W} + \boldsymbol{W}^0 + \boldsymbol{W}^1) \tag{11}$$

where n is the sample size, y_i represents the ith element of the label vector $\boldsymbol{y} \in \mathbb{R}^2$, μ is index of weight decay to alleviate overfitting.

4 Experiments

4.1 Evaluation Indicators

The evaluation indicators used in the experiments are listed below.
(1) AUC. Each time an edge is randomly selected from the test set and then randomly selected another edge from the non-existent test set. After n times independent comparisons in this way, if there are n' times, the score value in the test set is greater than the non-existing edge score. There are n'' times the two scores are equal, and then the AUC calculation formula is:

$$AUC = \frac{n' + 0.5n''}{n} \tag{12}$$

(2) ACC (Accuracy). The accuracy is the proportion of correct predictions (both true positives and true negatives) among the total number of cases examined. The formula is

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

where TP =True positive; FP =False positive; TN =True negative; FN = False negative.

4.2 Baseline Methods

To verify the model's effectiveness, we use the following baseline methods to compare with our proposed method in experiments. To thoroughly verify the performance of the proposed method, these baselines are also divided into two categories, namely graph convolutional methods, and heuristic methods.

(1) DCN [36]: Directed version of CN index.
(2) DAA [36]: Directed version of AA index.
(3) DRA [36]: Directed version of RA index.

(4) Cheb [11]: A spectral-based graph convolutional network proposed by Defferrard et al. This model uses Chebyshev polynomials to achieve fast localization and low complexity, hence the name Chebyshev network.
(5) GCN [17]: A spectral-based graph convolutional network proposed by Kipf and Maxwell. This model is further simplified on the basis of Chebyshev network to form a classic graph convolutional network.
(6) GAT [28]: A spatial-based graph convolutional network proposed by Veličković et al. The model combines the attention mechanism [27] with the graph convolutional network to extract more critical information.
(7) GIN [31]: A spatial-based graph convolutional network proposed Xu et al. From the perspective of Weisfeiler-Lehman test, Xu et al. considered the expressive ability of graph neural network, and proposed this model with the same powerful ability as Weisferler-Lehman test in theory.
(8) DiGCN [26]: A spectral-based graph convolutional network for directed graphs proposed by Tong et al. This model utilizes the idea of Inception [25] and is the latest graph convolutional network for link prediction.

4.3 Datasets

Comparative experiments are carried out on 6 real datasets of different scales in various fields, namely:

(1) High-school (HIG) [10]: A social network from a high school
(2) C.elegans (C-ele) [29]: A neural network of the nematode C.elegans, node representation in the network Neurons, edges represent information transmission between neurons.
(3) SmallW (SMW) [24]: A social network within a company, the nodes in the network are represented by users, edges represent the message passing between nodes.
(4) SmaGri (SMG) [4]: A citation network in which nodes represent papers and edges represent citations.
(5) Political blogs (PB) [2]: An American political blog network, where nodes in the network represent Blog page, the edge represents the hyperlink jump relationship existing between blogs.
(6) Air traffic control (ATC) [24]: An aviation network from the US Flight Control Center. The nodes in the network represent airports or service centers, and the edges represent recommended routes (Table 1).

4.4 Experiment Settings

We divide the edge set of the network into the training set and test set, and the division ratio is 10%. That is, the number of test edges accounts for 10% of the total number of edges. All existing edges have positive labels. Then randomly sample the same number of negative edges from the non-exiting edge of network, and their labels are negative.

Table 1. Statistical properties of the datasets. $|V|$ represents the number of nodes, $|E|$ represents the number of edges, ρ represents the density of the network, $\langle k \rangle_{out}$ represents the average out degree, and cc represents the clustering coefficient of the network.

Name	$\|V\|$	$\|E\|$	$\langle k \rangle_{out}$	ρ	cc	Type
HIG	70	366	5.228	0.0758	0.3624	Social network
C-ele	131	764	5.832	0.0449	0.1495	Neural network
SMW	181	756	4.176	0.0232	0.3426	Social network
SMG	1024	4919	4.803	0.0047	0.154	Citation network
PB	1224	19025	15.543	0.0127	0.2184	Hyperlink network
ATC	1226	2615	2.133	0.0017	0.0404	Aviation network

The number of layers of all the above graph convolutional networks is set to 2, the vector representation dimension of nodes is 16, the dropout rate during training is set to 0.5.

The dimension of the representation vector is 16, and the feature of the node is a 2-dimension vector composed of the in-degree and out-degree of the node. We set $\alpha = 0.05$ and the hidden layer dimension $h = 32$ and $d = 16$ in experiments.

4.5 Results and Analysis

All results are the average of 10 independent experiments. The experiment results are shown in Table 2 and Table 3, the proposed method represented by "LPNI." The highest value is shown in bold characters.

Table 2. The AUC results of experiments.

Method	HIG	C-ele	SMW	SMG	PB	ATC
DCN	0.5123	0.7212	0.8714	0.6865	0.8836	0.5455
DAA	0.5185	0.7188	0.8734	0.6845	0.8815	0.5452
DRA	0.5185	0.7173	0.8716	0.6835	0.8771	0.5472
Cheb	0.5790	0.7291	0.8778	0.7799	0.8640	0.6318
GCN	0.6420	0.7444	0.8323	0.7541	0.8949	0.6393
GAT	0.6788	0.7086	0.8296	0.6722	0.8139	0.6361
GIN	0.6508	0.6905	0.6987	0.7068	0.8507	0.5677
DiGCN	0.7129	0.7812	0.9118	0.8847	0.8942	0.6278
LPNI	**0.7411**	**0.8147**	**0.9143**	**0.8909**	**0.8986**	**0.6518**

Heuristic methods include DCN, DAA, and DRA, which only have ranking results and cannot define a threshold for calculating the ACC value. Therefore, their ACC results are not available in Table 3.

Table 3. The ACC results of experiments.

Method	HIG	C-ele	SMW	SMG	PB	ATC
Cheb	0.5778	0.6932	0.8043	0.6962	0.7661	0.5471
GCN	0.6587	0.7292	0.7776	0.6792	0.8191	0.6363
GAT	0.6841	0.6786	0.7651	0.6265	0.7561	0.6259
GIN	0.6207	0.6734	0.6603	0.6601	0.7612	0.5761
DiGCN	0.7127	0.7448	0.8428	0.8079	0.8103	0.6388
LPNI	**0.7241**	**0.7622**	**0.8603**	**0.8179**	**0.8274**	**0.6495**

As can be seen from Table 2 and Table 3, our proposed method outperforms other baseline methods in two evaluation metrics and in all datasets, proving that considering the information of node importance in link prediction can improve the prediction accuracy.

As can be seen from Table 2, comparing the graph convolutional network-based methods (such as LPNI, DiGCN [26], and Cheb [11]) with traditional heuristic methods (such as DCN [36], DAA [36], and DRA [36]), it can be found that the graph convolutional network-based prediction methods show higher accuracy in various types of networks, indicating that methods based on graph convolutional networks can capture the characteristics of different types of networks and are suitable for different types of networks.

As shown from Table 2 and Table 3, DiGCN [26] has the highest prediction accuracy compared with all baseline methods. DiGCN also uses the PageRank algorithm to calculate the personalized PR value of nodes, which can be regarded as a node importance value, which indicates that considering the importance information of nodes can improve the link prediction accuracy. However, its accuracy is lower than that of the LPNI method, indicating our node importance information is more effective.

5 Conclusion

This paper proposed a novel link prediction method that utilizes the node importance information. We conducted many experiments on 6 real networks from various fields and compared them with other baseline methods. The results suggested that our method can work directly on the directed graph in the link prediction tasks. Our method outperforms all baseline methods, including the traditional heuristic methods and graph convolutional network based methods.

In our experiments, the datasets we used are static. That is, the time when the link is generated is ignored. The research work of Xia et al. [30] shows that considering the time when the link appears has a positive impact on the prediction accuracy, so we consider utilize the time information in future work to improve the link prediction accuracy further. In addition, Zhang et al.'s [33] research uses subgraph sampling for link prediction, which significantly reduces

the computational complexity. Therefore, our further work will also consider using subgraph sampling technology to reduce the model under the premise of complete node importance information to reduce the computational cost of training.

References

1. Adamic, Lada A., Adar, Eytan: Friends and neighbors on the Web. Soc. Netw. **25**(3), 211–230 (2003)
2. Adamic, L.A., Glance, N.: The political blogosphere and the 2004 U.S. election: Divided they blog. In: Proceedings of the 3rd International Workshop on Link Discovery, LinkKDD 2005, pp. 36–43. Association for Computing Machinery, New York, NY, USA (2005)
3. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
4. Batagelj, V., Mrvar, A.: Pajek-program for large network analysis. Connections **21**(2), 47–57 (1998)
5. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203 (2013)
6. Cai, L., Ji, S.: A multi-scale approach for graph link prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3308–3315 (2020)
7. Cai, L., Li, J., Wang, J., Ji, S.: Line graph neural networks for link prediction. IEEE Trans. Pattern Anal. Mach. Intell. (2021)
8. Carmi, S., Havlin, S., Kirkpatrick, S., Shavitt, Y., Shir, E.: A model of internet topology using k-shell decomposition. Proc. Natl. Acad. Sci. **104**(27), 11150–11154 (2007)
9. Chowdhury, G.G.: Introduction to Modern Information Retrieval. Facet Publishing (2010)
10. Coleman, J.S., et al.: Introduction to Mathematical Sociology (1964)
11. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 29. Curran Associates, Inc. (2016)
12. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
13. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: LightGCN: simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 639–648 (2020)
14. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press (2012)
15. Hu, F., Lakdawala, S., Hao, Q., Qiu, M.: Low-power, intelligent sensor hardware interface for medical data preprocessing. IEEE Trans. Inf. Technol. Biomed. **13**(4), 656–663 (2009)
16. Jaccard, P.: Étude comparative de la distribution florale dans une portion des alpes et des jura. Bull. Soc. Vaudoise Sci. Nat. **37**, 547–579 (1901)
17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
18. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)

19. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: 32nd AAAI Conference on Artificial Intelligence (2018)
20. Li, Y., Song, Y., Jia, L., Gao, S., Li, Q., Qiu, M.: Intelligent fault diagnosis by fusing domain adversarial training and maximum mean discrepancy via ensemble learning. IEEE Trans. Industr. Inf. **17**(4), 2833–2841 (2021)
21. Ma, Y., Hao, J., Yang, Y., Li, H., Jin, J., Chen, G.: Spectral-based graph convolutional network for directed graphs. arXiv preprint arXiv:1907.08990 (2019)
22. Qiu, H., Zheng, Q., Msahli, M., Memmi, G., Qiu, M., Lu, J.: Topological graph convolutional network-based urban traffic flow and density prediction. IEEE Trans. Intell. Transp. Syst. **22**(7), 4560–4569 (2021)
23. Ravasz, E., Somera, A.L., Mongru, D.A., Oltvai, Z.N., Barabási, A.L.: Hierarchical organization of modularity in metabolic networks. Science **297**(5586), 1551–1555 (2002)
24. Rossi, R., Ahmed, N.: The network data repository with interactive graph analytics and visualization. In: 29th AAAI Conference on Artificial Intelligence (2015)
25. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
26. Tong, Z., Liang, Y., Sun, C., Li, X., Rosenblum, D., Lim, A.: Digraph inception convolutional networks. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 17907–17918. Curran Associates, Inc. (2020)
27. Vaswani, A., et al.: Attention is all you need. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc. (2017)
28. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
29. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. nature **393**(6684), 440–442 (1998)
30. Xia, L., et al.: Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 4486–4493 (2021)
31. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
32. Zareie, A., Sheikhahmadi, A., Jalili, M.: Influential node ranking in social networks based on neighborhood diversity. Fut. Gener. Comput. Syst. **94**, 120–129 (2019)
33. Zhang, M., Chen, Y.: Weisfeiler-Lehman neural machine for link prediction. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 575–583 (2017)
34. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. Adv. Neural. Inf. Process. Syst. **31**, 5165–5175 (2018)
35. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: 32nd AAAI Conference on Artificial Intelligence (2018)
36. Zhang, X., Zhao, C., Wang, X., Yi, D.: Identifying missing and spurious interactions in directed networks. Int. J. Distrib. Sens. Netw. **11**(9), 507386 (2015)
37. Zhang, Z., Cui, P., Zhu, W.: Deep learning on graphs: a survey. IEEE Trans. Knowl. Data Eng. **34**(1), 249–270 (2022)