

Mining Node Relationships in Graphs based on Neural Network

Li Lin*, Linlin Luo

College of Southwest Minzu University, Chengdu 610000, China.

Abstract

Mining Node Relationships in Graphs is one of the important methods for network research. It can use network nodes and some network structure information to predict the possibility of linking between two nodes that have not yet formed a relationship in the network. It can also be used in relationship prediction, recommendation systems, etc. The field has high application value. However, the traditional graph node expression has the problems of low expression efficiency and poor link prediction accuracy. In response to these problems, this paper uses graph neural network to learn node characteristics, improves on GraphSAGE, introduces Attention mechanism to give neighbor nodes different weights; at the same time, introduces Drop mechanism to deepen the number of network layers to better characterize node information, so as to achieve higher accuracy of link prediction in a graph.

Keywords

Graph Neural Network; Attention; Relationship Prediction; Drop.

1. Introduction

Link prediction in the network refers to predicting the possibility of hidden relationships between nodes based on the existing information in the network. Predicting the relationship between nodes in the network has strong practical significance, and different application scenarios can be derived for different graphs. For example, mining graph data in social networks to realize community discovery [1]; In the recommendation system, personalized recommendations can be made according to the user's hobbies [2] and so on.

For the research on graphs, graph representation learning is a more commonly used method in recent years, most of the methods based on graph representation learning are based on Skip-Gram[3] or matrix decomposition[4], such as DeepWalk[5], Node2Vec[6], LINE[7], etc. These algorithms use the learned representations for downstream machine learning tasks. Compared with the traditional method, its accuracy is improved. However, this method based on graph representation learning still has problems such as limited expressive ability and inability to adapt to dynamic network graphs.

With the introduction of neural networks, it has made good achievements in image detection, text analysis, etc. Because the core of CNN[8] lies in its kernel, the kernel can be translated on the picture, and features are extracted through convolution, realize parameter sharing. Someone migrated it to graph data, which stimulated the expansion of methods for studying graph data, and gradually developed graph neural networks. It performs tasks on graph data by transforming the network structure into a low-dimensional representation.

In 2016, the GCN proposed by Kipf et al [9] used the concept of convolution in convolutional neural networks to directly convolve the graph according to the connected structure of the graph. The formula is very simple. The GCN input of each layer is the adjacency matrix A and the node's feature H . The two are directly made into an inner product, and then multiplied by the parameter matrix W . After the activation function, it is equivalent to a simple nerve network layer. Its expression is as 1.

$$f(H^{\wedge}((l)), A) = \sigma(AH^{(l)}W^{(l)}) \quad (1)$$

However, this simple model has limitations. The diagonal of A is all 0. When multiplying with the feature matrix H , only the weighted sum of the features of all neighbors of a node is calculated, and its own features are ignored. The identity matrix I is added to make the corner element becomes 1. A is a matrix that has not been normalized, which will change the distribution of features. Therefore, a standardization process is required, multiplied by the inversal D , and finally a symmetric and normalized matrix is obtained. The model structure of GCN is shown in Figure 1.

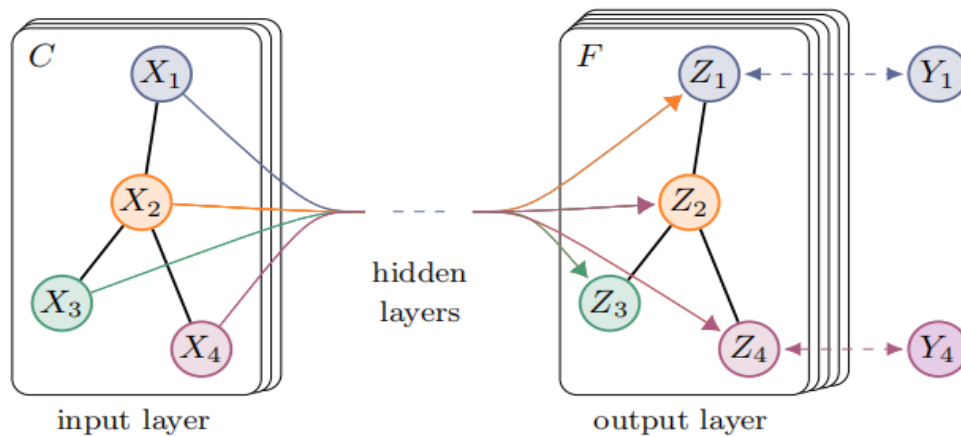


Figure 1. The model structure of GCN[9]

One year after when GCN proposed, the GAT proposed by Velickovic et al. [10] further achieved better accuracy in the task of node classification. However, this two methods require complete graph network data as the input of model, which is difficult to train and poor in scalability, and is not well applicable to large-scale networks. At the same time, the parameter of GAT is larger than that of GCN, and it is prone to overfitting when using high-order neighbor nodes.

These methods directly learn Embedding of each node on a fixed graph, but in most cases the graph is not static. When the network structure changes and new nodes appear, directly push learning such as GCN and GAT will need to be retrained. The problem with retraining is that the computational complexity will become higher and it may cause embedding to shift. Therefore, it is difficult to land on a machine learning system that needs to quickly generate unknown node Embedding. In response to this problem, in 2017, the GraphSAGE method proposed by Hamilton et al. [11] is more suitable for dynamic graphs. It solves the input problem by sampling the neighborhood of a fixed size, and adds an aggregation mechanism, and uses two-level neighborhood node information to represent the node.

2. Methodology

2.1. GraphSAGE

The GraphSAGE framework is different from GCN in that it mainly learns an aggregation function by training the function of aggregating node neighbors, expanding GCN into an inductive learning task, and generalizing unknown nodes. The main process of GraphSAGE is shown in Figure 1: First, the neighbor nodes of each node in the graph are sampled; then the feature information contained in the neighbor nodes is aggregated according to the aggregation function; finally, the vector representation of each vertex in the graph is obtained for downstream tasks to use. Figure 2 shows the main process of the model.

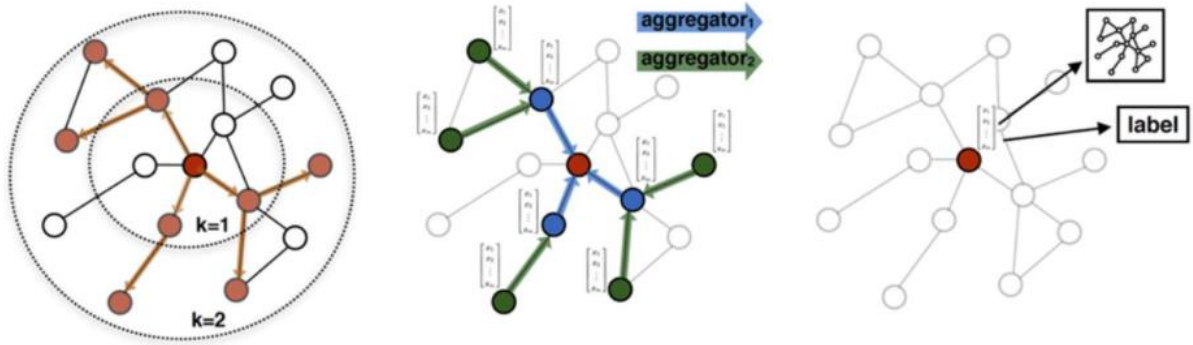


Figure 2. The main process of GraphSAGE[11]

In GraphSAGE, the sampling method is to assume that the number of samples is k . When the number of neighbors of a node is less than k , the sampling method with replacement is used until k vertices are sampled; when the number of neighbors of the node is greater than, it is used Sampling method without replacement. The above sampling method considers the calculation efficiency, but there is a problem of information loss. If the calculation efficiency is not considered, it is possible to use all neighbor nodes for information aggregation for each node, so that the node neighbors can be fully utilized. Characteristic information to ensure that the information is not damaged. Due to the special reasons of the data set, in this article, a fixed number of samples is used.

There are many options for GraphSAGE's aggregation function. The average aggregator considered in this article is shown in equations 2-3. It firstly averages each dimension in the neighbor's Embedding, and then performs a nonlinear transformation after concating with the target node.

$$h_{N(v)}^k = \text{mean}(\{h_u^{k-1}, u \in N(v)\}) \quad (2)$$

$$h_v^k = \sigma(W^{(k)} \cdot \text{CONCAT}(h_v^{k-1}, h_{N(v)}^k)) \quad (3)$$

In the formula, v is the node, k is the number of layers, and h is the feature.

2.2. Attention Mechanism:

Attention mechanism is a technology that allows the model to focus on relatively important information and fully learn. In the graph data, the relative importance of different neighbors of the node is also different, so consider adding the Attention [12] module. The module first calculates the similarity coefficient between its neighbors and itself for each vertex, that is, uses a shared parameter to increase the dimension of the vertex's features, then splices the transformed features, and finally maps the high-dimensional features to on a real number.

Among them, the shared parameters can be obtained through learning. In short, by stacking network layers, the neighborhood characteristics of each node are obtained, and then different nodes in the neighborhood are assigned different weights. The attention mechanism is applied to all edges in the graph in a shared manner, so it does not rely on pre-access to the global graph structure, nor does it rely on pre-access to the features of all nodes.

2.3. Drop Mechanism:

Different from other applications in the general deep learning field, such as images, voice, text, due to the particularity of graph data, the effect of the model cannot be enhanced by increasing the depth of the network. In order to deepen the graph convolutional network, this paper adopts the mechanism principle of DropEdge [13], which randomly cuts the edges in the original graph during model training, so as to alleviate the depth problem of the graph

convolutional network and reduce the over-smooth convergence. Speed and the loss of information caused by it.

2.4. Process of improved model:

The model process proposed in this paper is shown in Figure 3. The node features are represented by GraphSAGE, but the Attention mechanism is added to its first-order neighbor, The different attention to each node is reflected in the figure as the thickness of the connection. At the same time, it emphasized its own characteristics, that is, adding self-loop. After the node's characterization is obtained, the downstream link prediction task is performed, and the predicted result is finally obtained.

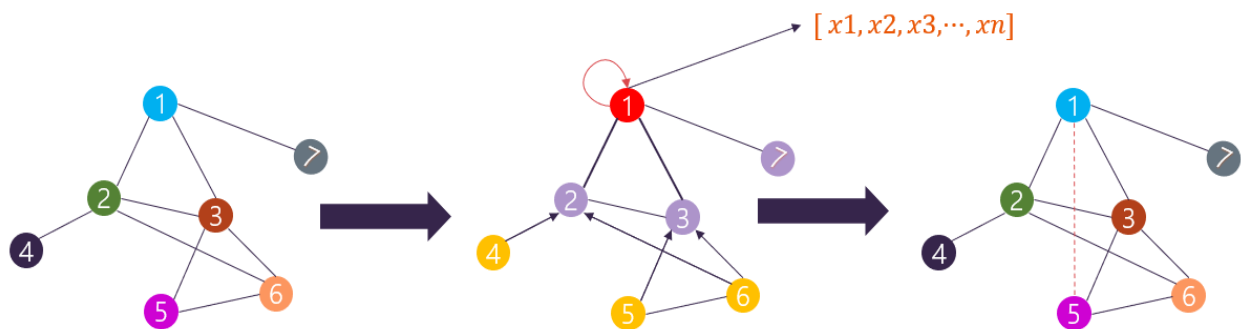


Figure 3. The link prediction process in this article

3. Experiment and discussion

3.1. Experimental data

The experiment in this article uses the common Cora data set and PubMed data set. Table 1 shows the relevant information of the two data sets.

Table 1. Data set related information

Dataset	Nodes	Edges	Features	Classes
Cora	2708	5429	1433	7
PubMed	19717	44338	500	3

The Cora data set consists of scientific papers, with a total of 2708 papers. Each paper is represented by a 1433-dimensional word vector, which means that each sample point in the data set has 1433 features. All papers are divided into 7 categories. In the Cora data set, each paper cited at least one other paper, or was cited by other papers.

The PubMed dataset consists of 19,717 scientific publications on diabetes from the PubMed database. Each sample has a feature dimension of 500 and is divided into 3 categories.

In the experiment of this article, we complete the task of link prediction by dividing the data set. Part of the edges is used as the training set, and some of the edges are used as the test set.

3.2. Results and discussion

The experimental results are shown in Table 2. According to the experiment, the use of graph neural network for link prediction has a relatively higher effect than traditional DeepWalk and other methods. And we found that the proposed improved GraphSAGE model is also effective in improving the accuracy of link prediction tasks. The number after the basic model in Table 2 indicates the number of layers of the basic network.

Table 2. Experimental results

Method	Cora	PubMed
DeepWalk	0.85	0.84
Node2Vec	0.80	0.81
LINE	0.52	0.53
GraphSAGE	0.89	0.87
GraphSAGE(Drop)	0.90	0.89
GraphSAGE2	0.87	0.89
GraphSAGE2(Drop)	0.90	0.90
GraphSAGE2(Drop)+Att	0.92	0.91

According to the experimental results, adding the Drop mechanism to the one-layer network and the two-layer network can improve the effect of the model to a certain extent, which proves the effectiveness of the Drop mechanism. When the number of network layers is deepened, the accuracy will decrease, and adding the Attention mechanism can also improve the effect of the model to a certain extent.

4. Conclusion

In this paper, based on the graph node relationship mining task, the model is improved based on GraphSAGE, and the Attention mechanism is added to the GraphSAGE model to highlight the assignment of different weights to different neighbors of the node; and the Drop mechanism is used to randomly delete links to deepen the network. Finally, experiments have proved that the accuracy of the relationship prediction between nodes in the graph is improved. The model has been improved in comparison with the traditional and some newer methods, and good results have been obtained.

References

- [1] Shuliang Xu, Shenglan Liu, Lin Feng. Manifold graph embedding with structure information propagation for community discovery[J]. Knowledge-Based Systems, 2020, 208.
- [2] Guang Xing Lye, Wai Khuen Cheng, Teik Boon Tan, Chen Wei Hung, Yen-Lin Chen. Creating Personalized Recommendations in a Smart Community by Performing User Trajectory Analysis through Social Internet of Things Deployment[J]. Sensors, 2020, 20(7).
- [3] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vectorspace. CoRR, abs/1301.3781, 2013.
- [4] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. Computer, 42(8):30–37, 2009.
- [5] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pages 701–710. ACM, 2014.
- [6] Grover, A. & Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 855–864 (ACM, 2016).
- [7] Tang, J. et al. Line: Large-scale information network embedding. In Proceedings of the 24th International Conference on World Wide Web, 1067–1077 (International World Wide Web Conferences Steering Committee, 2015).
- [8] LeCun, Yann, Bengio, Yoshua, and Hinton, Geoffrey. Deep learning. Nature, 521:436–444, 2015.
- [9] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

- [10] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. arXiv preprint arXiv:1710.10903, 2017.
- [11] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs[C]//Advances in Neural Information Processing Systems. 2017: 1024-1034.
- [12] Jonas Gehring, Michael Auli, David Grangier, and Yann N. Dauphin. A convolutional encoder model for neural machine translation. CoRR, abs/1611.02344, 2016.
- [13] Kenta Oono and Taiji Suzuki. On asymptotic behaviors of graph cnns from dynamical systems perspective. arXiv preprint arXiv:1905.10947, 2019.