2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA) | 978-1-6654-2877-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICIBA52610.2021.9688120

Node classification without features using graph convolutional network

Qingju Jiao[1,2,3], Yuanyuan Jin[1], Yongge Liu[1,2,3]
1. College School of Computer & Information Engineering, Anyang Normal University, Anyang, China
2. Key Laboratory of Oracle Bone Inscriptions Information Processing, Ministry of Education of China, Anyang, China
3. Henan Key Laboratory of Oracle Bone Inscriptions Information Processing, Anyang, China
qjjiao@aynu.edu.cn, jinyy@aynu.edu.cn, ay_liuyongge@163.com
Corresponding Author: Qingju Jiao Email: qjjiao@aynu.edu.cn

***Abstract*—Graph convolutional network (GCN) is proposed to deal with graph-structured data, and is applied to many fields successfully, such as computer vision, natural language processing and biology. Most current graph convolutional networks and its derivative methods need node features for classification. In fact, it is a formidable task to obtain node feature or other information. Therefore, we mainly focus on improving accuracy of GCN on the graphs with node features. In order to resolve the issue, we propose a local shortest path GCN (LpGCN) in this work. The proposed method only regard *q-hop* shortest path distance as node feature, and then employ GCN to realize node classification. The experiment results show that the proposed method can significantly improve the accuracy of node classification in the networks without node features.**

Keywords—graph convolutional network; node classification; node feature; shortest path

I. INTRODUCTION

Many complex systems can be described by graphs (or networks), and then be researched easily in different fields, such as sociology, biology and computer science. A number of graph-related problems are resolved by traditional methods. For example, node classification, link prediction, community detection and network control. The breakthroughs in deep learning have led to a paradigm shift in artificial intelligence and machine learning [1]. On the one hand, numerous old problems have been revisited with deep neural networks and huge progress has been made in many fields. For example, as a vital model of deep learning, convolution neural networks (CNN) [2] have been successfully applied to computer vision and natural language processing. But, many tasks of graph are resolved relaxedly because the non-Euclidean nature of the graph-structured data [3].

Graph neural network (GNN) is neural model that captures the dependence of graphs via message passing between the nodes of graphs [4], and fills the gap between graph and deep learning. A variant of GNNs is graph convolutional network (GCN) [5] which integrates the connectivity patterns and feature attributes of graph-structured data, and outperforms many state-of-the-art methods significantly on some benchmarks [1]. In fact, Bruna et al first applied the notion of convolution to graph based on spectral graph theory as early as 2013 [6]. And then, deuterogenic methods of GCN have been proposed. As with GCN, Li et al propose an adaptive graph convolution network (AGCN) with a new spectral graph convolution layer to process the data of arbitrary graph structure and size [7]. In order to reduce labeled training data, Zhuang et al devise dual graph convolutional network (DGCN) which jointly considers local consistency and global consistency on graphs [8]. DGCN uses two convolutional neural networks to capture the two consistencies and adopts an unsupervised loss to ensemble them. Different from the previous three methods including GCN, AGCN and DGCN using spectral convolution operator, large-scale learnable graph convolutional networks (LGCNs) is proposed by spatial convolution operator [9]. LGCNs define a learnable graph convolutional layer which can select a fixed number of neighboring nodes for each feature to reduce excessive memory and computational resource. Another classic spatial graph convolutional networks is GraphSAGE [10]. GraphSAGE is a general inductive framework that generates embeddings by sampling and aggregating features from a node's local neighborhood.

Inspired from the better performance of deep CNNs in computer vision, sever methods are proposed to build deep GCNs. Unfortunately, Deep GCNs may cause over-smoothing problem. Over-smoothing means that the features of nodes cannot be distinguished with increment of network layers [11]. By adding highway network gates, Rahimi et al propose a highway GCN of which performance peaks at 4 layers [12]. Furthermore, Li et al propose more deep GCNs (named DeepGCNs) whose best results are achieved at 56 layers [13]. DeepGCNs solve the problems of vanishing gradient and over-smoothing by incorporating residual connections and dense connections. Residual is an effectively method to overcome over-smoothing problem, but initial residual alone is not sufficient to extend GCN to a deep model [14]. Chen et al combine Initial residual and Identity mapping to relieve the problem of over-smoothing [14]. DropEdge which randomly removes a certain number of edges from the input graph at each training epoch is able to alleviate both over-fitting and over-smoothing issues [11].

978-1-6654-2877-4/21/$31.00 ©2021 IEEE

In order to train large scale real-world graphs, Gao et al develop a sub-graph training method [9], which is a simple approach to allow the training of deep learning methods on large-scale graph data. The method applies regular convolutional operations to graphs, and obtains better performance on node classification task. Hamilton et al present a general inductive framework (named GraphSAGE) that leverages node feature information to generate node embeddings for previously unseen data [10]. Experimental results show that GraphSAGE consistently outperforms state-of-the-art baselines. To learn graph representation fast, some several sampling-based methods have been proposed. For example, fastGCN samples the receptive field for each layer by selecting important nodes [15]. Huang et al develop an adaptive layer-wise sampling method for accelerating the training of GCNs [16]. Cluster-GCN samples a block of nodes that associate with a dense subgraph, and restricts the neighborhood search within this subgraph [17]. The strategy of the sample leads Cluster-GCN to train large-scale network and to achieve comparable test accuracy with previous algorithms.

Most node classification methods based on GCN need node feature information mention above. But, the node feature information in real world is difficult to obtain. In the paper, we focus on the node classification without node features. In graph, the shortest path captures the position of nodes [18]. Therefore, we employ the matrix of *q-hop* shortest path distance [18] between nodes as feature matrix to node classification. Experiment results show that the proposed methods can significantly improve the accuracy of node classification in the networks without node features.

II. METHODS

Given a graph (or network) $G=(V,E,\mathbf{X})$, where $V=\{v_1,v_2,\cdots,v_n\}$ is the set of n nodes, E is the set of edges and $\mathbf{X}=[x_1^T,x_2^T,\cdots,x_n^T]\in\mathbb{R}^{n\times d}$ is the node feature matrix where d is the number of features and x_i indicates the node features of v_i. The graph can also be represented by an adjacency matrix $\mathbf{A}$. If there is an edge between node v_i and node v_j, $\mathbf{A}(i,j)$ is the weight of the edge; otherwise $\mathbf{A}(i,j)=0$. The feed forward propagation in graph convolutional network is described as [5]:

$$\mathbf{H}^{(l+1)}=\sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}) \tag{1}$$

$\mathbf{H}^{(l)}$ is the matrix of activations in the l^{th} layer; $\mathbf{H}^{(0)}=\mathbf{X}$, $\sigma(\bullet)$ denotes an activation function, such as the $\mathrm{Re}LU(\bullet)=\max(0,\bullet)$; $\tilde{\mathbf{A}}=\mathbf{A}+\mathbf{I}$ ($\mathbf{I}$ is identity matrix) and $\tilde{\mathbf{D}}=\sum_{j=1}^{n}\tilde{\mathbf{A}}_{ij}$.

The graph convolutional network is applied filed of semi-supervised classification. The model used is two layer graph convolutional network with a softmax classifier on the output features:

$$Z=soft\max(\hat{\mathbf{A}}\,\mathrm{Re}LU(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(0)})\mathbf{W}^{(1)}) \tag{2}$$

where $\hat{\mathbf{A}}=\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$, $soft\max(x_i)=\frac{1}{\sum_i\exp(x_i)}\exp(x_i)$, the weights $\mathbf{W}^{(0)}$ and $\mathbf{W}^{(1)}$ are trained using gradient descent. The loss function is defined as the cross-entropy error over all labeled examples:

$$L=-\sum_{l\in y_L}\sum_{f=1}^{F}Y_{lf}\ln Z_{if} \tag{3}$$

where y_L is the set of node indices that have labels, F is the dimension of the output features and is equal to the number of classes. $Y\in R^{|y_l|\times F}$ is a label indicator matrix.

In LpGCN, we use *q-hop* shortest path distance [18] feature ($\mathbf{M}^{SP}$) to replace the identity matrix in the input graph convolutional network. The elements in $\mathbf{M}^{SP}$ with size of $n\times n$ is defined as formula (4).

$$\mathbf{M}_{ij}^{SP}=\begin{cases}d_{sp}(i,j) & if\ d_{sp}(i,j)\le q\\ 0 & otherwise\end{cases} \tag{4}$$

In formula (4), $d_{sp}(i,j)$ represents the shortest path between node i and node j. The parameter q is set 3 because the most real-world networks have local topological structure.

III. RESULTS

In order to verify the performance of our method, two methods including GCN [5] and LpGCN are tested on five datasets (see table 1). These datasets are Cora [19], Citeseer [19], Karate [20], Dolphins [21] and Polbook (http://www-personal.umich.edu/~mejn/netdata/, Books about US politics). The former two datasets are citation networks in which each node has label and features. The after three datasets are networks with strong community structure and the nodes in these networks have no label and features. In the work, we regard the nodes in the same community as the same class.

For GCN and LpGCN, first, the feature matrixes are set identity matrixes and the *q-hop* shortest path distance matrixes respectively. Second, for hyper-parameters in GCN and LpGCN, we set the same values as in GCN [5]: a learning rate of 0.01, 200 maximum epochs, 0.5 dropout rate, 5×10^{-4} $L2$ regularization weight, 2 convolutional layers, and 16 hidden units. Third, for Core and Citeseer networks, the order of the nodes in the feature matrix is the same as the order of the data in the GCN, and we test GCN and LpGCN with 1%, 5%, 10% and 15% training size and with 10% test size respectively. For other three small networks, the order of the nodes in the feature matrixes are rearrange by labels of class, that is, we rank

the nodes alternately using the labels of class. Due to the sizes of the three networks are small, we test GCN and LpGCN with 10%, 20%, 30% and 40% training size and with 60% test size respectively. The accuracy of GCN and LpGCN are shown in Table 2. Overall, the accuracy of LpGCN on all five networks are significantly improved comparing GCN. For GCN and LpGCN, the accuracy of Citeseer are lower than the other four datasets probably because the Citeseer network is unconnected and has large value of diameter. Since the three networks (Karate, Dolphins and Polbook) have strong community structure, both GCN and LpGCN show better performance.

TABLE I. SOME CHARACTERISTICS OF FIVE NETWORKS

Networks	Nodes	Edges	Classes	Diameter	Connectivity
Core	2708	5429	7	19	No
Citeseer	3327	4732	6	28	No
Karate	34	78	2	5	Yes
Dolphins	62	159	2	8	Yes
Polbook	105	441	3	7	Yes

TABLE II. THE ACCURACY OF GCN AND LPGCN

Networks	Training size	Test size	Accuracy(GCN)	Accuracy(LpGCN)
Core	1% (27)	10% (271)	0.3578	**0.6559**
	5% (135)		0.5296	**0.7660**
	10% (271)		0.6222	**0.7922**
	15% (406)		0.6834	**0.7989**
Citeseer	1% (33)	10% (331)	0.1618	**0.2358**
	5% (166)		0.2021	**0.4397**
	10% (331)		0.2073	**0.3727**
	15% (497)		0.2373	**0.4024**
Karate	10% (3)	60% (20)	0.9368	**1**
	20% (7)		0.8789	**1**
	30% (10)		0.8842	**1**
	40% (14)		0.9474	**1**
Dolphins	10% (6)	60% (37)	0.9444	**0.9722**
	20% (12)		0.9444	**0.9722**
	30% (19)		0.9444	**0.9722**
	40% (25)		0.9444	**0.9722**
Polbook	10% (11)	60% (63)	0.7694	**0.9065**
	20% (21)		0.8839	**0.9242**
	30% (32)		0.8484	**0.9097**
	40% (42)		0.8984	**0.9306**

IV. CONCLUSIONS

Many complex systems can be represented graph, and then some problems of complex systems can be resolved easily. GNNs, especially GCNs, which fuse deep learning concept have become a widely applied graph analysis method recently [4]. GCN model that integrates the connectivity patterns and feature attributes of graph-structured data has excellent performance on node classification [1]. Unfortunately, it is difficult to feature attributes of graph-structured data in many fields. In order to improve the accuracy of node classification using GCN without node feature attributes, this work propose a new method named LpGCN which regards *q-hop* shortest path distance as node feature. Experiment results show that the proposed methods can significantly improve the accuracy of node classification in the five networks without node features. In future work, we plan to insight into the principle of LpGCN and propose a new method with deep architectures to overcome over-fitting and over-smoothing issues.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grant No. 61806007,

Henan Provincial Colleges and Universities Youth Key Teacher Training Plan under Grant No. 2021GGJS129, the National Social Science Fund Major Entrusted Project of China under Grant No. 16@ZH017A3 and the Program for Changjiang Scholars and Innovative Research Team in University under Grant No. 2017PT35.

REFERENCES

[1] Q.A. Li, Z.C. Han, and X.M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning. The Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[2] R. Girshick, Fast R-CNN. IEEE International Conference on Computer Vision, 2015.

[3] S. Zhang, H.H. Tong, J.J. Xu, and R. Maciejewski, Graph convolutional networks: a comprehensive review. Computational Social Networks, vol. 6, pp. 11, 2019.

[4] J. Zhou, G.Q. Cui, S.D. Hu, Z.Y. Zhang, C. Yang, Z.Y. Liu, L.F. Wang, C.C. Li, and M.S. Sun. Graph neural networks: A review of methods and applications. AI Open, vol. 1, pp. 57-81, 2020.

[5] T. N. Kipf, and M. Welling, Semi-supervised classifification with graph convolutional networks. International Conference on Learning Representations, 2017.

[6] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, Spectral networks and locally connected networks on graphs. International Conference on Learning Representations, 2013.

[7] R.Y. Li, S. Wang, F.Y. Zhu, and J.Z. Huang, Adaptive graph convolutional neural networks. The Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[8] C.Y. Zhuang, and Q. Ma, Dual graph convolutional networks for graph-based semi-supervised classification. The 2018 World Wide Web Conference, 2018.

[9] H.Y. Gao, Z.Y. Wang, and S.W. Ji, Large-scale learnable graph convolutional networks. The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018.

[10] W.L. Hamilton, R. Ying, and J. Leskovec, Inductive representation learning on large graphs. 31st Conference on Neural Information Processing Systems, 2017.

[11] Y. Rong, W.B. Huang, T.Y. Xu, and J.Z. Huang, Dropedge: towards deep graph convolutional networks on node classification. International Conference on Learning Representations, 2020.

[12] A. Rahimi, T. Cohn, and T. Baldwin, Semi-supervised user geolocation via graph convolutional networks. The 56th Annual Meeting of the Association for Computational Linguistics, 2018.

[13] G.H. Li, M. Muller, A. Thabet, and B. Ghanem, Deepgcns: can gcns go as deep as cnns? IEEE International Conference on Computer Vision, 2019.

[14] M. Chen, Z.W. Wei, Z.F. Huang, B.L. Ding, and Y.L. Li, Simple and deep graph convolutional networks. The 37th International Conference on Machine Learning, 2020.

[15] J. Chen, T.F. Ma, and C. Xiao, Fastgcn: fast learning with graph convolutional networks via importance sampling. International Conference on Learning Representations, 2018.

[16] W.B. Huang, T. Zhang, Y. Rong, and J.Z. Huang, Adaptive sampling towards fast graph representation learning. 32nd Conference on Neural Information Processing Systems, 2018.

[17] W.L. Chiang, X.Q. Liu, and S. Si, Cluster-GCN: an efficient algorithm for training deep and large graph convolutional networks. The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019.

[18] J.X. You, R. Ying, and J. Leskovec, Position-aware graph neural networks. The 36 th International Conference on Machine Learning, 2019.

[19] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, Collective classification in network data. AI magazine, vol. 29, issue 3, pp. 93, 2008.

[20] W.W. Zachary, An information flow model for conflict and fission in small groups. Journal of Anthropological Research, vol. 33, pp. 452–473, 1977.

[21] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, and S.M. Dawson, The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behavioral Ecology and Sociobiology. Vol. 54, pp. 396–405, 2003.