Multimedia Tools and Applications (2022) 81:34183–34194
https://doi.org/10.1007/s11042-020-09984-2

Graph convolutional networks of reconstructed graph structure with constrained Laplacian rank

Mengmeng Zhan[1] · Jiangzhang Gan[2] · Guangquan Lu[1] · Yingying Wan[1]

Received: 22 February 2020 / Revised: 26 August 2020 / Accepted: 24 September 2020 /
Published online: 12 October 2020

Abstract
Convolutional neural networks (CNNs) have achieved unprecedented competitiveness in text and two-dimensional image data processing because of its good accuracy performance and high detection speed. Graph convolutional networks (GCNs), as an extension of classical CNNs in graph data processing, have attracted wide attention. At present, GCNs often use domain knowledge (such as citation recommendation system, biological cell networks) or artificial created fixed graph to achieve various semi-supervised classication tasks. Poor quality graph will lead to suboptimal results of semi-supervised classification tasks. We propose a more general GCN of reconstructed graph structure with constrained Laplacian rank. First, we use hypergraph to establish multivariate relationships between data. On the basis of the hypergraph, In virtue of Laplacian rank constraint to the graph matrix, we learn a new graph structure which has c connected components (where c is the number of classification), and then we construct an ideal graph matrix which is more suitable for the task of semi-supervised classification on GCNs. Finally, the data and the new graph are input GCNs model to get the results of classification. Experiments on 10 different datasets demonstrate that this method is more competitive than the comparison method.

Keywords Graph convolutional networks · Adaptive graph · Hypergraph · Semi-supervised classification · Graph structure

1 Introduction

At present, deep learning has performed well in solving computer vision [24], image processing [12], speech recognition [1] and other tasks [33]. Among different types of deep learning models, convolutional neural networks (CNNs) have been widely studied and applied because of its high performance in various tasks. LeCun et al. [20] and others

✉ Guangquan Lu
lugq@mailbox.gxnu.edu.cn

[1] College of Computer Science and Information Technology, Guangxi Normal University, Guilin, Guangxi, 541004, People's Republic of China

[2] School of Natural and Computational Sciences, Massey University Auckland Campus, Auckland, 0745, New Zealand

established the basic framework of CNN model inspired by the cognitive mechanism of biological natural vision. Krizhevsky et al. [19] proposed the AlexNet network and won the championship in the 2012 ImageNet, making CNNs become the core algorithm model in image classification. After that, CNNs developed rapidly. Many CNNs models, such as ResNet [13], NasNet [33] and GoogleNet [27], have been proposed by researchers, which show strong recognition ability in target detection, digital classification and other tasks.

In particular, although CNNs have good performance and high efficiency, it also has great limitations. CNNs can only process data of the grid type. In the real world, the data we collect are often defined on irregular grids, even in non-Euclidean spaces, such as biomolecular networks, social networks, etc. Unlike image data and time series on regular grid, it is obvious that we can process these data better in the form of graph. However, there are some challenges in using CNNs to perform regular convolution operation in graph structure tasks. The convolution kernel of CNNs require that the number of adjacent nodes of each node remain unchanged, and the number of nodes is orderly. In graph structure data, each node has different number of adjacent nodes and the positions of nodes are different. In social networks, for example, each person has different relations. Therefore, it is impossible to directly use fixed rectangular filter to process graph structure data. For this reason, Researchers try to modify the convolution kernel to extract the features of irregular graph structure data and propose a variety of graph convolutional networks (GCNs) methods. Generally speaking, GCNs are divided into spatial-based and the spectral-based strategies. For spatial-based strategies, GCNs define convolution kernels directly on the connection relationship of each node, which is similar to the convolution kernel of traditional CNNs. Hamilton et al. [11] proposed a new representation of graph nodes by aggregating the features of neighbor nodes. Atwood and Towsley [3] proposed a diffusion based representation learning method, which extends CNNs to solve general graph structure data. For spectral-based strategies, the convolution operation on topological graph is realized by the theory of spectrum. Bruna et al. [4] proposed to define the graph convolution in the Fourier domain based on the characteristic decomposition of the graph Laplacian matrix. Besides, Defferrard et al. [7] proposed to get the filter through the approximate expansion of the Chebyshev polynomial of Laplacian, which reduced the computational complexity. Kipf and Welling [18] used the first-order approximation of the Chebyshev polynomial to get the kernel convolution, and proposed a simple graph convolution network.

The above GCNs have been widely used in various classification tasks, but these methods rarely pay attention to the development of graph structures. There is no doubt that one of the core of GCNs is the graph which can describe the neighborhood relationship of original data. A proper graph structure plays an important role in GCNs learning. In deep learning tasks, we often provide GCNs with a known graph structure, such as biomolecular network and social network. However, the graph structure obtained from domain knowledge may not be accurate or even available. For instance, in a social network, a user may have thousands of friends, but only a few close friends. When analyzing such social network data, it is not appropriate to create thousands of edges for the user to represent the nearest neighbor relationship. Therefore, how to construct a more suitable graph based on the known graph for GCNs classification is valuable. Futhermore, in some cases, the graph structure of the data may be unknown. We can use the traditional graph data model (such as kNN to build the graph) to obtain the graph structure of the data. But kNN only considers the binary relationship between data, and the data collected in reality often contain complex multivariate relationships. Complex relationships are simply represented as paired relationships,which can easily lead to the loss of important relevant information [32]. For the sake of

overcoming the information loss problem of traditional graph model in the knowledge of complex multiple relationship, hypergraph model came into being. Yu et al. [29] proposed to generate a set of hyperedges for each image by changing the size of the neighborhood to achieve the image classification task. Jiang et al. [16] introduced a network evolution method based on a hypergraph model with multi-dimensional connections, and further explore the new relationship between data through the features extracted by graph convolution.

Inspired by these views, We consider the advantages of hypergraphs and how to create an ideal graph structure, and propose a novel GCNs of reconstructed graph structure with constrained Laplacian rank. First of all, we build a hypergraph of data as the initial adjacency matrix. Furthermore, by imposing some constraints, we learn a new graph matrix based on the initial graph matrix, making the new graph matrix more suitable for subsequent semi-supervised classification tasks on GCNs. Extensive experiments show that the graph generated by this method is superior to other common graphs in semi-supervised classification tasks and our method has the following advantages:

- In view of the current GCNs methods, it is not necessarily the most appropriate to obtain graphs from the knowledge domain. This paper introduce a novel GCNs of reconstructed graph structure with constrained Laplacian rank. By introducing hypergraphs and Laplace rank constraints, we construct a graph structure that is more suitable for subsequent classification tasks of GCNs. Experiments on multiple data sets show that our proposed method has achieved more competitive results in semi-supervised classification tasks compared with the comparison method.
- The traditional graph structure only considers the binary relationship between data. In practical application, there are many complex relationships between nodes. The traditional method of constructing graph can not describe the complex relationship between data effectively, which affects the quality of graph. In this paper, we use hypergraph to define the initial graph and consider the multiple relations of data.

The rest of this paper is arranged as follows. In Section 2, we briefly introduce the related knowledge of hypergraph learning and GCNs. In Section 3 and Section 4, we describe our proposed method of constructing optimal graph and related experimental results, respectively. Conclusion and future work are presented at the end of the paper.

2 Related work

In this section, we describe graph learning and hypergraph related knowledge, and further introduce a main model of GCNs.

2.1 Hypergraph and graph learning

With the progress of technology and the richness of social life, the real data collected is no longer a two-dimensional grid structure or even a simple one-dimensional structure. But high-dimensional complex structure data, such as biomolecular network data and commodity recommendation data, how to effectively analyze and apply these data with high-dimensional complex structure has become a challenge for current researches [14]. As a kind of generalized data structure, The graph is used to described the complicated node relations among samples. Many machine learning methods try to use graph structure to represent complex data [31]. For example, in the field of chemical analysis, people

use graph structure to describe the relationship between chemical molecules [6], and in the commodity recommendation system, people use graph structure to describe the relationship among different people's hobbies and commodities [21]. Real graph network is often high-dimensional and difficult to process. Researchers have proposed LLE [25], SDNE [28], Node2Vec [10] and other graph embedding methods. For example, Hu et al. [14] and Zhu et al. [31] designed novel graph embedding methods for SVM classification and spectral clustering, respectively. The traditional graph structure only considers the binary relationship between data. In practical application, the structure of data is often complex multiple relationship, so the traditional point-to-point graph structure cannot effectively describe the complex relationship between data, thus affects the quality of the graph. The hypergraph is the generalization of the common binary graph [30]. The hypergraph algorithm is initially applied to the very large-scale integrated electricity [2]. Zhou et al. [30] extended spectral clustering to hypergraphs and carried out tasks such as classification and clustering. In the deep learning task, Shi et al. [26] proposed to use hypergraph to establish the high-order correlation of visual data for CNNs vision classification . In this paper, hypergraph with multiple information is used to replace the ordinary graph of binary relation. A large number of literatures show that the richer the information learned from the data, the better the performance of real experiments [5, 30].

2.2 GCNs model

We review the GCNs for semi-supervised proposed by kipf and welling [18]. GCNs are similar to CNNs and other neural network architectures, including multiple graph convolutional hidden layers. Given data set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\} \in \mathbb{R}^{n\times d}$, n and d represent the number of samples and features respectively. The graph adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ corresponding to data set $\mathbf{X}$. Then GCNs propagate layer by layer according to the following functions:

$$\mathbf{X}^{(k)} = \sigma(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\mathbf{X}^{(k-1)}\mathbf{U}^{(k-1)}), \tag{1}$$

where k is the number of hidden layers, $\tilde{\boldsymbol{A}} = \mathbf{A} + \mathbf{I}_N$ is an adjacency matrix with a ring undirected graph, and $\mathbf{D}$ is a diagonal matrix with $\tilde{\mathbf{d}}_i = \sum_{j=1}^{n}\tilde{\mathbf{A}}_{ij}$. $\sigma(\cdot)$ is an activation function. $\mathbf{U}^{(k-1)} \in \mathbb{R}^{d_{k-1}\times d_k}$ is defined the $k-1$ layer's weight matrix that changes iteratively with the number of layers. We apply softmax function to the output feature $\mathbf{X}^{(K)}$ of the final K-layer to obtain the prediction label of each sample:

$$\mathbf{P} = \text{softmax}(\mathbf{X}^{(K)}), \tag{2}$$

where $\mathbf{P} \in \mathbb{R}^{n\times c}$ is the predicted label matrix, c is the number of real classes, and $\mathbf{P}_i$ is the prediction label of the i-th sample. For semi-supervised classification tasks, The weight matrices $\{\mathbf{U}^1, \mathbf{U}^2, ..., \mathbf{U}^K\}$ are updated by the cross entropy loss function, which is as follows:

$$Loss = \sum_{i\in\mathbf{L}}\sum_{j=1}^{c}\tilde{p}_{ij}\ln p_{ij}, \tag{3}$$

where p_{ij} represents the i-th label predicted as j-th class and $\tilde{p}_{ij}$ shows the i-th belonging to the real j-th class. and $\mathbf{L}$ is the set of labeled samples.

3 Our method

In this section, for the convenience of understanding, we define some notations in this paper, and then describe the proposed method in Section 3.2, Section 3.3, respectively. Besides, we introduce the corresponding optimization steps in Section 3.4.

3.1 Notations

For the convenience of understanding, we have the following definitions. For a matrix $\mathbf{S}$, the i-th row and the i,j-th of matrix $\mathbf{S}$ are denoted as $\mathbf{s}_i$ and s_{ij}. We denote the Frobenius norm of matrix $\mathbf{S}$ as $||\mathbf{S}||_F = \sqrt{\sum_{ij} |s_{ij}|^2}$. Finally, we denote $\mathbf{S}^T$ and $tr(\mathbf{S})$ as the transpose of matrix $\mathbf{S}$ and the trace of matrix $\mathbf{S}$, respectively.

3.2 Initial graph

Graph representation learning is widely used in machine learning and artificial intelligence tasks. However, the traditional graph structure has some limitations in the expression of data correlation. Data correlation may be more complex than point-to-point relationship. The traditional graph structure modeling does not describe the complex relationship between data sufficiently [30]. In this case, the traditional graph restricts the application of graph convolution neural network. For the sake of overcoming this limitation, hypergraph model came into being [30].

Hypergraph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$ contains three parts: vertex set $\mathbf{V}$, hyperedge set $\mathbf{E}$, and hyperedge weight matrix $\mathbf{W}$. it is defined that hyperedge e has a weight $w(e)$. the link relationship of hypergraph $\mathbf{G}$ is expressed as the correlation matrix $\mathbf{H} \in \mathbb{R}^{|V|\times|E|}$, which is defined as follows:

$$h(v, e) = \begin{cases} 1 & v \in e, \\ 0 & v \notin e. \end{cases} \tag{4}$$

The above formula shows that if a vertex v is located in a hyperedge e, then $h(v, e) = 1$, otherwise $h(v, e) = 0$. According to the definition of correlation matrix, the degree matrix of hypergraph vertex and hyperedge can be further expressed as:

$$\begin{aligned} d\,(v) &= \sum_{e\in\mathbf{E}} w(e)h(v, e) \\ \delta(e) &= \sum_{v\in\mathbf{V}} h(v, e). \end{aligned} \tag{5}$$

Furthermore, we define two diagonal matrices $\mathbf{D}_v$ and $\mathbf{D}_e$ to represent the diagonal degree matrix of hypergraph vertex and hyperedge respectively. Based on spectral graph theory and literature knowledge [30] , we define the adjacency matrix $\mathbf{A}$ of hypergraph as follows:

$$\mathbf{A} = \mathbf{HWH}^T - \mathbf{D}_v, \tag{6}$$

In this paper, the diagonal elements of $\mathbf{W}$ are all defined as one. Hypergraph has been proved to be able to completely represent the relationship between objects compared with ordinary graph [26, 30]. In this paper, we choose hypergraph as the initial graph adjacency matrix to reserve the multivariate relationship of data.

3.3 Adaptive graph learning

In graph learning theory, many works have proposed how to design a high quality graph structure. Nie et al. introduced the CLR method to learn the similarity matrix of data for

clustering task and achieved excellent results [23]. In this paper, we use hypergraph as initial adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and CLR model to learn a graph matrix $\mathbf{S} \in \mathbb{R}^{n\times n}$ more suitable for our subsequent semi-supervised classification tasks on GCNs. First, we define the minimum loss function between the optimal graph $\mathbf{S}$ and the initial graph $\mathbf{A}$ by establishing the l_2-norm distance formula, that is, to minimize the following regularization terms. Besides, it is worth considering that some rows of the matrix $\mathbf{S}$ obtained in the above formula may be all zero. We construct the constraint condition $\mathbf{s}_i^T\mathbf{1} = 1$, thus we have:

$$\begin{aligned} &\min_{\mathbf{S}} \|\mathbf{S}-\mathbf{A}\|_F^2 . \\ &s.t. \quad \mathbf{s}_i^T\mathbf{1} = 1, \mathbf{s}_i \geq 0 \end{aligned} \tag{7}$$

In order to allow the adjacency matrix suitable for classification tasks, we apply Laplacian rank constraint $rank(\mathbf{L}_S) = n - c$ to the matrix $\mathbf{S}$. According to the properties of Laplace matrix, the number of zero eigenvalues of Laplace matrix $\mathbf{L}_S$ is equal to the number of connected domains of affinity matrix. So the adjacency matrix just has c connected components. We can get:

$$\begin{aligned} &\min_{\mathbf{S}} \|\mathbf{S}-\mathbf{A}\|_F^2 . \\ &s.t. \quad \mathbf{s}_i^T\mathbf{1} = 1, \mathbf{s}_i \geq 0, rank(\mathbf{L}_S) = n - c \end{aligned} \tag{8}$$

Because the rank constraint to matrix $\mathbf{S}$ is a complex constraints, in order to facilitate the subsequent optimization, according to the literature method [22, 23], it is assumed that ω_i is the i-th smallest eigenvalue of Laplacian matrix $\mathbf{L}_S$. Since $\mathbf{L}_S$ is a semi-positive definite matrix, if there is a sufficiently large value λ , Eq. (8) can be further equivalent to:

$$\begin{aligned} &\min_{\mathbf{S}} \|\mathbf{S}-\mathbf{A}\|_F^2 + 2\lambda \sum_{i}^{c} \omega_i . \\ &s.t. \quad \mathbf{s}_i^T\mathbf{1} = 1, \mathbf{s}_i \geq 0 \end{aligned} \tag{9}$$

Further, according to KyFan's Theorem [8], we know that $\sum_{i}^{c} \omega_i = \min_{\mathbf{Y}^T\mathbf{Y}=\mathbf{I}} tr(\mathbf{Y}^T\mathbf{L}_S\mathbf{Y})$ and $\mathbf{Y} \in \mathbb{R}^{n\times c}$, and finally we get our objective function as follows:

$$\begin{aligned} &\min_{\mathbf{S},\mathbf{Y}} \|\mathbf{S}-\mathbf{A}\|_F^2 + 2\lambda tr(\mathbf{Y}^T\mathbf{L}S\mathbf{Y}), \\ &s.t. \ \mathbf{s}_i^T\mathbf{1} = 1, \mathbf{s}_i \geq 0, \mathbf{Y}^T\mathbf{Y} = \mathbf{I} \end{aligned} \tag{10}$$

where the matrix $\mathbf{S}$ is a higher quality graph, $\mathbf{A}$ is the initial adjacency matrix introduced in Section 3.2 and $\mathbf{L}_S$ is the Laplace matrix about the matrix $\mathbf{S}$.

Based on the above objective function (10) , we can get an ideal matrix $\mathbf{S}$ by introducing $\mathbf{S}$ into GCNs model for classification task, Therefore, the convolution layer of GCNs changes as follows:

$$\mathbf{X}^{(k)} = \sigma(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\mathbf{X}^{(k-1)}\mathbf{U}^{(k-1)}), \tag{11}$$

where $\mathbf{S}$ represents a new graph structure that is more suitable for subsequent classification tasks. It is obtained by alternating iterative optimization of Eq. (10), and $\mathbf{D}$ is a diagonal matrix with $\tilde{\mathbf{d}}_i = \sum_{j=1}^{n} \mathbf{S}_{ij}$.

3.4 Optimization

In this part, we mainly optimize and solve the objective function Eq. (10) to get the final matrix $\mathbf{S}$.

1) Optimize $\mathbf{Y}$ by fixing $\mathbf{S}$

When $\mathbf{S}$ is fixed, the objective function Eq. (10) becomes:

$$\begin{aligned} &\min_{\mathbf{Y}} tr(\mathbf{Y}^T \mathbf{L}_S \mathbf{Y}), \\ &s.t.\ \ \mathbf{Y}^T \mathbf{Y} = \mathbf{I} \end{aligned} \tag{12}$$

The closed form solution of $\mathbf{Y}$ consists of c eigenvectors corresponding to c minimum eigenvalues of $\mathbf{L}_S$.

2) Optimize $\mathbf{S}$ by fixing $\mathbf{Y}$

When $\mathbf{Y}$ is fixed, for the second term in the formula, we can get:

$$tr(\mathbf{Y}^T \mathbf{L}_S \mathbf{Y}) = \frac{1}{2} \sum_{i,j} \left\| \mathbf{y}_i - \mathbf{y}_j \right\|_2^2 s_{ij}. \tag{13}$$

It is easy to understand that the value of matrix $\mathbf{S}$ is independent for each i. for Eq. (11), it is expanded by components as follows:

$$\begin{aligned} &\min_{\mathbf{S}} \sum_j (s_{ij} - a_{ij})^2 + \lambda \sum_j \left\| \mathbf{y}_i - \mathbf{y}_j \right\|_2^2 s_{ij}. \\ &s.t.\ \ \mathbf{s}_i^T \mathbf{1} = 1, s_{ij} \geq 0 \end{aligned} \tag{14}$$

We define vector $\mathbf{u}_i = \left\| \mathbf{y}_i - \mathbf{y}_j \right\|_2^2$, and the above formula is expanded by vector as follows:

$$\begin{aligned} &\min_{\mathbf{S}} \left\| \mathbf{s}_i - (\mathbf{a}_i - \tfrac{\lambda}{2} \mathbf{u}_i) \right\|_2^2. \\ &s.t.\ \ \mathbf{s}_i^T \mathbf{1} = 1, s_{ij} \geq 0 \end{aligned} \tag{15}$$

Equation (15) can be solved by an iterative algorithm [15]. To sum up, we optimize and obtain the ideal graph $\mathbf{S}$ for semi-supervised classification on GCNs, and the related pseudo code is described in algorithm 1.

Algorithm 1 The optimization steps to solve our objecitive function in Eq. (10).

Input: $\mathbf{X} \in \mathbb{R}^{n \times d}$, λ, and classes number c;
Output: $\mathbf{S} \in \mathbb{R}^{n \times n}$;
1.1 Initialize adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ of hypergraph by Eq. (6);
1.2 Initialize $\mathbf{Y} \in \mathbb{R}^{n \times c}$ by Eq. (12) .
2. **repeat:**
2.1. Update $\mathbf{S}$ by Eq. (15) ;
2.2. Update $\mathbf{Y}$ by Eq. (12) ;
until convergence

4 Experimental

We set up several groups of experiments to verify the performance of our proposed method. In Section 4.1 , we introduce the data sets and experimental methods used in the experiment, and in Section 4.2 , based on the experimental results, we analyze the superiority of the proposed method.

Table 1 Details of the public datasets

Datasets	Samples	Features	Classes
Musk2	6598	166	2
Ozone	2536	72	2
Chess	3196	36	2
Waveform	2746	21	3
Robotnavigation	5456	24	4
MinistData10	6996	784	10
Arrhythmia13	452	279	13
Diabetes	768	8	2
Dig1-10	1797	64	10
MinistData05	3495	784	10

4.1 Datasets and experimental methods

To verify the performance of the proposed method on classification tasks, we test it on 10 real datasets, including the handwriting recognition datasets (MinistData05 and MinistData10), sound datasets (Waveform), etc. We describe information about all data sets in Table 1 , and the public data sets were collected from UCI Machine Learning Repository [1] and the CSDN blog [2] .

About the experimental setup, we refer to some of the experimental settings of the previous work [9, 18]. Because of the different sizes of data sets, for all data sets, we randomly selected 10%, 20% and 30% samples as labeled samples, and used another 10% samples as validation samples. The remaining 80%, 70% and 60% were used as test samples. We use the Adam optimization method [17] to define that the number of training iterations using GCNs model is no more than 200, and set the learning rate to 0.01. If the cross entropy loss value of the validation set remains unchanged for 20 consecutive times, the training is stopped. Besides, we set dropout rate to 0.5. For the comparison methods, we set up KNN-GCN (KGCN) and Hypergraph-GCN (HGCN) methods respectively. For the KGCN method, kNN method is used to define the graph adjacency matrix of data, while the HGCN method uses Section 3.2 method to define the graph adjacency matrix of data. In addition, we set the neighborhood size of all datasets to 15. We mainly used ACC as the evaluation index of the experiment, and all the reported results were the average results with different training, verification and test data segmentation in 10 times.

4.2 Experiment results

The experiment results of the comparison methods KGCN, HGCN and the proposed method in this paper are shown in Table 2 and Fig. 1 respectively. Table 2 shows the average accuracy and the corresponding standard deviation of all models for 10 datasets under three different data partitions. It can be clearly seen that our proposed model has achieved the best or more advantageous results in all data sets. For example, For Musk2 datasets, when the label rate is 10%, our method is higher than HGCN and KGCN by 0.16% and 0.08%

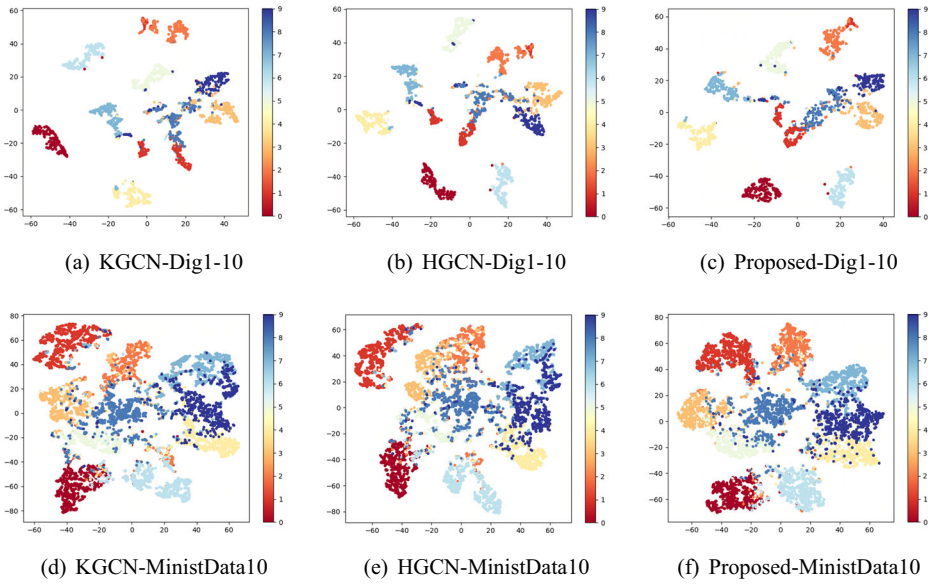
[1] http://archive.ics.uci.edu/ml/.
[2] https://blog.csdn.net/qq_32892383/article/details/104424358

Table 2 Mean classification results (ACC ± Std (%)) under 10 different datasets, and the best performance are indicated by bold

Dataset	Musk2			Ozone		
Label rate	10%	20%	30%	10%	20%	30%
KGCN	84.51 ± 0.2	84.40 ± 0.2	84.57 ± 0.4	97.08 ± 0.1	97.04 ± 0.2	97.12 ± 0.1
HGCN	84.59 ± 0.1	84.55 ± 0.2	84.65 ± 0.4	97.08 ± 0.1	97.15 ± 0.2	97.25 ± 0.2
Proposed	**84.67 ± 0.2**	**84.71 ± 0.2**	**84.82 ± 0.3**	**97.11 ± 0.1**	**97.17 ± 0.3**	**97.26 ± 0.2**
Dataset	Chess			Waveform		
Label rate	10%	20%	30%	10%	20%	30%
KGCN	76.89 ± 3.6	78.99 ± 1.5	80.21 ± 2.3	80.40 ± 1.2	80.83 ± 0.5	81.12 ± 0.9
HGCN	71.50 ± 3.5	72.60 ± 1.0	72.07 ± 1.6	79.60 ± 0.8	79.63 ± 0.6	79.90 ± 0.9
Proposed	**79.60 ± 3.6**	**80.03 ± 2.2**	**81.19 ± 1.5**	**80.55 ± 1.3**	**81.37 ± 0.9**	**81.48 ± 1.0**
Dataset	Robotnavigation			MinistData10		
Label rate	10%	20%	30%	10%	20%	30%
KGCN	62.43 ± 1.5	**63.16 ± 1.2**	63.27 ± 1.2	82.20 ± 0.6	82.84 ± 0.4	83.15 ± 0.7
HGCN	61.07 ± 1.7	61.74 ± 0.8	62.20 ± 1.7	79.45 ± 0.8	79.69 ± 0.7	80.32 ± 0.8
Proposed	**63.04 ± 2.1**	62.59 ± 1.5	**64.17 ± 1.2**	**83.82 ± 0.7**	**84.69 ± 0.5**	**85.02 ± 0.4**
Dataset	Arrhythmia13			Diabetes		
Label rate	10%	20%	30%	10%	20%	30%
KGCN	53.89 ± 1.4	53.65 ± 1.8	54.02 ± 2.2	64.88 ± 0.7	64.28 ± 1.6	64.98 ± 0.9
HGCN	53.59 ± 0.6	53.78 ± 2.0	54.02 ± 2.4	64.59 ± 0.7	64.76 ± 0.8	65.01 ± 1.0
Proposed	**54.11 ± 1.6**	**54.29 ± 2.5**	**54.79 ± 1.5**	**65.34 ± 0.6**	**65.62 ± 1.5**	**65.53 ± 1.7**
Dataset	Dig1-10			MinistData05		
Label rate	10%	20%	30%	10%	20%	30%
KGCN	**88.38 ± 1.8**	89.35 ± 0.9	89.94 ± 1.4	79.17 ± 1.0	81.12 ± 1.3	81.27 ± 0.9
HGCN	88.22 ± 1.4	88.29 ± 0.9	88.28 ± 1.5	76.67 ± 1.2	77.05 ± 1.0	77.32 ± 1.7
Proposed	88.16 ± 2.0	**90.01 ± 2.0**	**90.65 ± 1.0**	**80.64 ± 1.7**	**83.11 ± 0.9**	**83.58 ± 0.7**

respectively, and when the label rate is 20%, our method is higher than HGCN and KGCN by 0.25% and 0.17% respectively. Besides, with the label rate growing up, the classification performance of our method is also improving. In addition, the accuracy of our method is 81.19%, 81.48%, 85.02% and 97.26% on Chess, Waveform, Ministdata10 and Ozone datasets, respectively. This further shows that our method has excellent performance on semi-supervised classification tasks. This is because this paper not only considers the multiple information between the data, but adaptively assigns the optimal neighborhood to each node through the Laplacian rank constraint. Finally, we get the most suitable data graph representation matrix for GCNs.

Figure 1 shows the 2D T-SNE visualization with the convolution output feature of KGCN, HGCN and our method on Dig1-10 dataset and Ministdata10 dataset respectively. Through comparison, the data distribution of different types is more clear in our method, while several types of data distribution in KGCN show aggregation distribution,



(a) KGCN-Dig1-10 (b) HGCN-Dig1-10 (c) Proposed-Dig1-10

(d) KGCN-MinistData10 (e) HGCN-MinistData10 (f) Proposed-MinistData10

Fig. 1 2D T-SNE visualization of the features output by KGCN, HGCN and our proposed method are performed on Dig1-10 dataset and Ministdata10 dataset

which represents that our method has better performance in graph node representation and semi-supervised classification tasks.

5 Conclusion

We propose a novel GCN of reconstructed graph structure with constrained Laplacian rank in this paper. First of all, we use hypergraph with complex multivariate relations to establish the initial graph between data. Based on the initial graph, we construct a new graph representation matrix which is more suitable for semi-supervised classification tasks on GCNs. Different from the previous methods, Our method is more general, especially when the graph structure of data is missing and unavailable. The experimental results on 10 real datasets show that the method has excellent performance compared with the comparison methods. In future work, we will try to embed the graph learning method proposed in this paper into the hidden layer of GCNs, so as to dynamically adjust the graph representation matrix in each iteration. In addition, we can build a better adaptive graph model to improve the performance and practicability for semi-supervised classification on GCNs.

Acknowledgements This work is partially supported by the Key Program of the National Natural Science Foundation of China (Grant No: 61836016); the Natural Science Foundation of China (Grants No: 61876046, 81701780, 61672177 and 61972177); the Project of Guangxi Science and Technology (GuiKeAD17195062); the Guangxi Natural Science Foundation (Grant No: 2017GXNSFBA198221); the Guangxi Collaborative Innovation Center of Multi-Source Information Integration and Intelligent Processing; the Research Fund of Guangxi Key Lab of Multisource Information Mining & Security (18-A-01-01); 2019 basic scientific research capability enhancement project for middle-aged teachers in guangxi university (2019KY0062); and Innovation Project of Guangxi Graduate Education (Grants No:YCSW20201008, JXXYYJSCXXM-008).

References

1. Abdelhamid O, Mohamed A, Jiang H, Deng L, Penn G, Yu D (2014) Convolutional neural networks for speech recognition. IEEE Trans Audio Speech Lang Process 22(10):1533–1545
2. Agarwal S, Lim J, Zelnik-Manor L, Perona P, Kriegman D, Belongie S (2005) Beyond pairwise clustering. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), volume 2, pp 838–845. IEEE
3. Atwood J, Towsley D (2016) Diffusion-convolutional neural networks. In: Advances in neural information processing systems, pp 1993–2001
4. Bruna J, Zaremba W, Szlam A, LeCun Y (2013) Spectral networks and locally connected networks on graphs. arXiv:1312.6203
5. Bulò SR, Pelillo M (2009) A game-theoretic approach to hypergraph clustering. In: Advances in neural information processing systems, pp 1571–1579
6. Coley CW, Jin W, Rogers L, Jamison TF, Jaakkola TS, Green WH, Barzilay R, Jensen KF (2019) A graph-convolutional neural network model for the prediction of chemical reactivity. Chem Sci 10(2):370–377
7. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in neural information processing systems, pp 3844–3852
8. Fan K (1949) On a theorem of weyl concerning eigenvalues of linear transformations i. Proc Natl Acad Sci USA 35(11):652
9. Franceschi L, Niepert M, Pontil M, He X (2019) Learning discrete structures for graph neural networks. arXiv:1903.11960
10. Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp 855–864
11. Hamilton W, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. In: Advances in neural information processing systems, pp 1024–1034
12. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 2961–2969
13. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
14. Hu R, Zhu X, Zhu Y, Gan J (2019) Robust svm with adaptive graph learning. World Wide Web
15. Huang J, Nie F, Huang H (2015) A new simplex sparse learning model to measure data similarity for clustering. In: Twenty-Fourth International Joint Conference on Artificial Intelligence
16. Jiang J, Wei Y, Feng Y, Cao J, Gao Y (2019) Dynamic hypergraph neural networks. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence (IJCAI), pp 2635–2641
17. Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: International conference on learning representations (ICLR)
18. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv:1609.02907
19. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
20. LeCun Y, Bengio Y et al (1995) Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks 3361(10):1995
21. Mirza BJ, Keller BJ, Ramakrishnan N (2003) Studying recommendation algorithms by graph analysis. Journal of intelligent information systems 20(2):131–160
22. Nie F, Wang X, Huang H (2014) Clustering and projected clustering with adaptive neighbors. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 977–986
23. Nie F, Wang X, Jordan MI, Huang H (2016) The constrained laplacian rank algorithm for graph-based clustering. In: Thirtieth AAAI Conference on Artificial Intelligence
24. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp 91–99
25. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. science 290(5500):2323–2326
26. Shi H, Zhang Y, Zhang Z, Ma N, Zhao X, Gao Y, Sun J (2018) Hypergraph-induced convolutional networks for visual classification. IEEE transactions on neural networks and learning systems 30(10):2963–2972

27. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
28. Wang D, Cui P, Zhu W (2016) Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp 1225–1234
29. Yu J, Tao D, Wang M (2012) Adaptive hypergraph learning and its application in image classification. IEEE Trans Image Process 21(7):3262–3272
30. Zhou D, Huang J, Schölkopf B (2007) Learning with hypergraphs: Clustering, classification, and embedding. In: Advances in neural information processing systems, pp 1601–1608
31. Zhu X, Gan J, Lu G, Li J, Zhang S (2019) Spectral clustering via half-quadratic optimization. World Wide Web
32. Zhu X, Li X, Zhang S, Ju C, Wu X (2017) Robust joint graph sparse coding for unsupervised spectral feature selection. IEEE Transactions on Neural Networks and Learning Systems 28(6):1263–1275
33. Zoph B, Vasudevan V, Shlens J, Le QV (2018) Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8697–8710