Multigraph Fusion for Dynamic Graph Convolutional Network

Jiangzhang Gan, Rongyao Hu, Yujie Mo, Zhao Kang, Liang Peng, Yonghua Zhu, and Xiaofeng Zhu, *Senior Member, IEEE*

***Abstract*— Graph convolutional network (GCN) outputs powerful representation by considering the structure information of the data to conduct representation learning, but its robustness is sensitive to the quality of both the feature matrix and the initial graph. In this article, we propose a novel multigraph fusion method to produce a high-quality graph and a low-dimensional space of original high-dimensional data for the GCN model. Specifically, the proposed method first extracts the common information and the complementary information among multiple local graphs to obtain a unified local graph, which is then fused with the global graph of the data to obtain the initial graph for the GCN model. As a result, the proposed method conducts the graph fusion process twice to simultaneously learn the low-dimensional space and the intrinsic graph structure of the data in a unified framework. Experimental results on real datasets demonstrated that our method outperformed the comparison methods in terms of classification tasks.**

***Index Terms*— Data fusion, dimensionality reduction, graph convolutional networks (GCNs), graph learning.**

I. Introduction

GRAPH convolutional network (GCN) has been widely applied in the graph data such as social networks and citation networks as it conducts representation learning while taking the structure information of the data (i.e., the graph information) into account [1]–[4]. The GCN and its variants deal with the graph data by two key steps, i.e., graph learning constructing a graph from the original data and representation learning improving the discriminative ability of the original representations [5]. The graph can be provided by the original data or constructed by the k nearest neighbors (kNN) method [6]. Current GCN methods are focusing on representation learning, i.e., designing different convolution operators to improve the effectiveness of representation learning while ignoring the graph learning. For example, Chen *et al.* [7] proposed to conduct representation learning by multiple convolution operations and Jiang *et al.* [8] proposed graph convolution methods for feature propagation.

In real applications, due to the noise and outliers, the initial graph often has wrong connections, which can degrade the robustness of the subsequent representation learning [9]–[11]. Therefore, improving the quality of the initial graph can further improve the performance of the GCN model. The goal of graph learning is to output the high-quality graph for guaranteeing the quality of representation learning. The quality of graph learning can be affected by many factors, such as noise and redundancy in the original data and the structure preservation.

First, the original data often contain noise and redundancy, which seriously affects the robustness of graph learning. In traditional machine learning methods, a number of solutions have been proposed to remove or reduce the influence of noise and redundancy, including feature selection and subspace learning. For example, Sun *et al.* [12] employed the feature selection method to remove irrelevant and redundant features in classification tasks. Salem and Hussein [13] designed subspace learning techniques to remove the noise of the data. However, current deep learning methods pay little attention to the issue of noise and redundancy on the original high-dimensional data, thus limiting the robustness of deep learning models.

Second, previous GCN methods focused on preserving the local structure of the data to construct the graph. However, the structural information of real-world data is complex, and a single characterization (i.e., either the global structure or the local structure) may not be sufficient to capture the intrinsic structure of the data [14]–[16]. The studies of graph learning in traditional machine learning [15], [17] demonstrated that both the local structure preservation and the global structure preservation are important for graph learning as they provide the complementary information to each other [11], [18]. The local structure describes the internal organization of the original data (i.e., the similarity between data points), while the global structure reflects the overall information of

Manuscript received 22 May 2021; revised 6 April 2022; accepted 28 April 2022. Date of publication 16 May 2022; date of current version 5 January 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0102200, in part by the National Natural Science Foundation of China under Grant 61876046, and in part by the Guangxi "Bagui" Teams for Innovation and Research. *(Corresponding author: Xiaofeng Zhu.)*

Jiangzhang Gan is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the School of Mathematical and Computational Science, Massey University Auckland, Auckland 0632, New Zealand.

Rongyao Hu is with the School of Mathematical and Computational Science, Massey University Auckland, Auckland 0632, New Zealand.

Yujie Mo, Zhao Kang, and Liang Peng are with the Center for Future Media and the School of Computer Science and Technology, University of Electronic Science and Technology of China, Chengdu 611731, China.

Yonghua Zhu is with the School of Computer Science, The University of Auckland, Auckland 1010, New Zealand.

Xiaofeng Zhu is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518000, China (e-mail: seanzhuxf@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3172588.

Digital Object Identifier 10.1109/TNNLS.2022.3172588

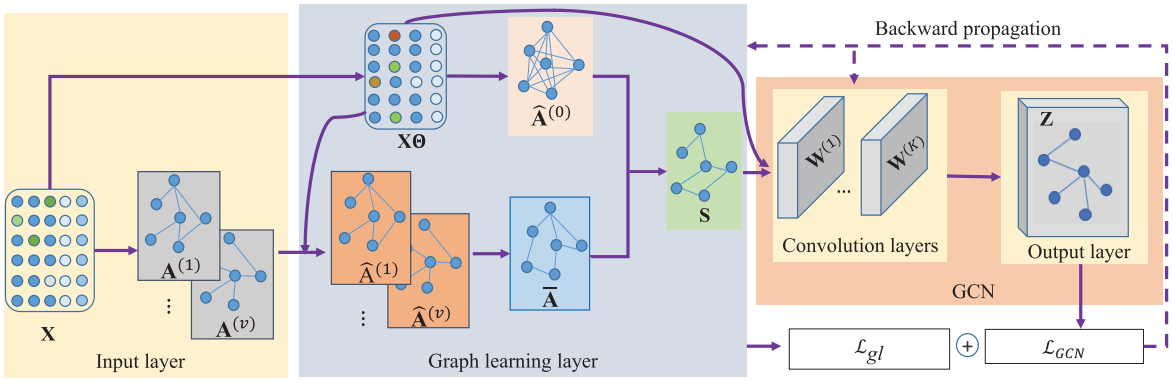

Fig. 1. Flowchart of our proposed method. Specifically, our method uses the feature matrix $\mathbf{X}$ of the original dataset to generate multiple sparse graphs $\mathbf{A}^{(v)}(v = 1, \ldots, V)$ as well as the low-dimensional data matrix $\mathbf{X}\Theta$ and then combines each $\mathbf{A}^{(v)}(v = 1, \ldots, V)$ with $\mathbf{X}\Theta$ to generate its local graph $\hat{\mathbf{A}}^{(v)}(v = 1, \ldots, V)$, followed by unifying all local graphs to generate the unified local graph $\bar{\mathbf{A}}$. The global graph $\hat{\mathbf{A}}^{(0)}$ learned from the low-dimensional data matrix $\mathbf{X}\Theta$ is further integrated with $\bar{\mathbf{A}}$ to output the initial graph $\mathbf{S}$ for the GCN model. The learned graph $\mathbf{S}$ and the low-dimensional data $\mathbf{X}\Theta$ are then fed to a two-layer GCN model for representation learning. It is noteworthy that $\mathbf{S}$ keeps invariant in GCN (i.e., the orange block) and varies in each epoch due to the backpropagation process.

the original data (i.e., the relationship between categories). In graph representation learning, considering both the global and local structure of the data allows the graph representation model to be more robust to noisy and sparse data [19]. To our knowledge, few literature of previous GCN models focused on simultaneously considering the local structure preservation and the global structure preservation.

In this article, we propose a novel GCN method as shown in Fig. 1 to jointly conduct graph learning and representation learning in a unified framework by: 1) fusing multiple local graphs on the low-dimensional space of the original high-dimensional data to obtain the unified local graph, where noise and redundancy are removed as much as possible, and 2) fusing the unified local graph and the global graph to output the initial graph for the GCN model. The first fusion step integrates the common information among multiple local graphs to guarantee the correctness of the edges in the unified local graph, i.e., the correctness of the local structure preservation. The second fusion step tries to find the missed edges in the local graphs from the global graph. Moreover, the proposed method simultaneously learns the projection matrix converting the original high-dimensional data to its low-dimensional space, two kinds of graph structures, and the representations of all samples. As a result, the update of each of them pushes the adaptive adjustment of representation learning so that it guarantees our proposed method to output discriminative representations.

Compared to previous methods, we list the main contributions of our method as follows.

First, we propose a new GCN method, which jointly conducts graph learning and representation learning from the low-dimensional space of the original data. To produce a high-quality graph, the proposed method explores the global structure and local structure of the data, with the aim of obtaining a correct connection for every node. Second, we propose a novel graph fusion mechanism to integrate the multiple graphs information. Compared to the traditional graph fusion method, the proposed graph fusion method dynamically learns the weighted for each graph during the fusion process so that the adversary effect of noise is effectively reduced and conducts the process of graph fusion twice where the first fusion focuses on guaranteeing the correction of the edges and the second fusion focuses on finding the missed edges in the graphs from the original data.

II. Related Work

A. Graph Learning

Previous graph learning methods can be categorized into traditional graph methods and deep graph methods. Specifically, traditional graph methods usually use traditional machine learning methods to exploit simple structure information of the data, while deep graph methods employ deep learning models to capture complex structure among the samples. Moreover, each category includes two kinds of techniques, i.e., static graph learning and dynamic graph learning, depending on the fact whether the graph structure is fixed or dynamically updated during the model training [11], [16].

Traditional graph methods include kNN graph, ϵ-graph, and so on. For example, Takai *et al.* [20] proposed using the hypergraph structure to preserve the high-order relations among samples so that the samples in the same cluster are highly similar. The graph in previous methods such as the kNN graph and the ϵ-graph is obtained from the original data as well as fixed in the process of the model training so that it is easily influenced by noise and redundancy. To address the above issue, dynamic mechanism is widely embedded in graph learning methods to mine inherent correlation of the data in the low-dimensional feature space. For example, Xiong *et al.* [21] proposed using the adaptive neighbor learning to capture the neighborhood information in the low-dimensional subspace, Li *et al.* [22] explored the local structure information to adaptively assign each data point with optimal neighbors, and Luo *et al.* [23] incorporated the exploration of the local structure into the procedure of feature selection to learn the optimal graph. Recently, Wang *et al.* [24] extended the

conventional label diffusion to the label pair diffusion for image segmentation.

Deep graph methods take use of a multilayer nonlinear architecture to extract complex pattern from underlying data, aiming at learning complicated node representation or graph representation [3]. GCN is a well-known static deep graph method, where the graph is used to preserve the local structure of the data during the process of representation learning [25]. The graph quality is still the key issue in deep graph methods. Recently, Jiang *et al.* [26] proposed integrating both the graph learning and the graph convolution in a unified network architecture so that the graph can be updated by the representation learning during the training process, Differently, Zhang *et al.* [27] proposed using the self-attention mechanism to learn a parameterized adjacency matrix tailored to a specific task, thus effectively improving the performance of the cosaliency detection task. Peng *et al.* [3] proposed to conduct a deep reverse graph learning based on the GCN for conducting graph learning on the low-dimensional space of original data.

B. Graph Fusion

Graph fusion techniques are frequently used to generate a common graph structure by multiple feature representations or multiple graph structures. A typical example using fusion techniques is shown in [28], which enforces the features across all the views to construct a common similarity graph. Traditional graph fusion methods employed a graph learning method to perform graph fusion via minimizing the difference between the desired common graph and predefined multiple graphs [29], [30]. For example, Tang *et al.* [31] proposed to capture both the common information and distinguishing knowledge across different views, Tong *et al.* [32] conducted multimodality classification using a nonlinear graph fusion method, and Tang *et al.* [33] took the traditional predefined graph matrices of different views as input and learned an improved graph for each single view. Recently, Lindenbaum *et al.* [34] proposed using a random walk process to conduct multiview dimensionality reduction. In deep graph fusion methods, Zhuang and Ma [35] designed a dual GCN architecture to learn a robust node representation by simultaneously taking into account the local consistency and the global consistency, However, this method employed the fixed graph technique and thus may limit the robustness of classification tasks. To perform consistent learning across multiple graphs, Jiang *et al.* [36] proposed a fusion method by conducting dynamic GCN and consistent learning on multiple graphs in a unified framework.

III. Method

A. Motivation

Details of the symbols used in this article are given in Table I. Given an initial graph $\mathbf{A} \in \mathbb{R}^{n\times n}$ storing the graph structure of all samples of the feature matrix $\mathbf{X} \in \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n\times d}$ where n and d, respectively, indicate the number of the samples and the features, the GCN method taking both $\mathbf{A}$ and $\mathbf{X}$ as the inputs passes several hidden layers and one fully connected layer to output the new representation $\mathbf{Z} \in \mathbb{R}^{n\times c}$ of $\mathbf{X}$, where c is the number of the classes.

TABLE I
Details of the Symbols Used in This Article

$\mathbf{X}$	Feature matrix
$\mathbf{A}^v (v = 0, ..., V)$	Initial kNN graph
$\hat{\mathbf{A}}^v (v = 0, ..., V)$	Update local graph
$\hat{\mathbf{A}}^0$	Update global graph
$\mathbf{A}$	Unified local graph
$\mathbf{S}$	Fused graph
$\mathbf{p}, \boldsymbol{\Theta}, \mathbf{W}^{(1)}, ..., \mathbf{W}^{(M)}$	Trainable parameters

Specifically, the representation learned by the mth layer of the GCN method can be obtained by

$$\mathbf{H}^{(m)} = \sigma\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(m-1)}\mathbf{W}^{(m)}\right) \tag{1}$$

where $\mathbf{H}^{(m)} \in \mathbb{R}^{n\times d_m}$ denotes the outputted representation in the mth hidden layer, $\mathbf{D} \in \mathbb{R}^{n\times n}$ is the diagonal matrix of $\mathbf{S}$, d_{ii} is the summation of all elements in the ith column of $\mathbf{A}$, $\mathbf{W}^{(m-1)} \in \mathbb{R}^{d_{m-1}\times d_m}$ is the weight matrix, which needs to be trained in the (m)th layer, d_m is the number of hidden units in the mth layer, and $\sigma(.)$ is the activation function. The last layer of the GCN model is the classification layer with a softmax function

$$\mathbf{Z} = \text{softmax}\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(M-1)}\mathbf{W}^{(M)}\right) \tag{2}$$

where $\mathbf{W}^{(M)} \in \mathbb{R}^{d_{M-1}\times c}$ denotes the weight matrix in the Mth hidden layer and M is the number of hidden layers. The weight parameters $(\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(M)})$ of the GCN are trained by minimizing the following cross-entropy loss function:

$$\mathcal{L}_{\text{GCN}} : -\sum_{i\in\mathcal{Y}}\sum_{j=1}^{c} y_{ij}\ln z_{ij} \tag{3}$$

where $\mathcal{Y}$ is the label set and y_{ij} and z_{ij}, respectively, denote the ground truth and the prediction label for the jth class and the ith sample.

Recently, dynamic GCN methods have been proposed to jointly conducting graph learning and representation learning in a unified framework. As a result, the representation can be updated based on the optimized graph, thus guaranteeing to produce discriminative representation for the original data. For example, Jiang *et al.* [37] employed the graph learning technique in traditional machine learning to add one more regularization term (i.e., $\mathcal{L}_{\text{GL}}$) [26] into the original GCN method, i.e., $\mathcal{L}_{\text{GCN}}$ in (3). More specifically, given the initial graph $\mathbf{A} \in \mathbb{R}^{n\times n}$, the regularization term $\mathcal{L}_{\text{GL}}$ is defined as

$$\begin{aligned}\mathcal{L}_{\text{GL}} : \min_{\mathbf{S}} &\sum_{i,j=1}^{n} \left\|\mathbf{x}_i - \mathbf{x}_j\right\|_F^2 s_{ij} + \lambda_1\|\mathbf{A} - \mathbf{S}\|_F^2 \\ &\text{s.t.} \quad \forall i,\ \mathbf{s}_i\mathbf{1} = 1,\ s_{ij} \geq 0\end{aligned} \tag{4}$$

where $\mathbf{S} = \{s_{ij}\}_{i,j=1}^{n} \in \mathbb{R}^{n\times n}$ is the updated graph based on the initial graph $\mathbf{A}$ and $\|\|_F$ indicates the Frobenius norm. Finally, the objective function of the DGCN is

$$\mathcal{L}_{\text{DGCN}} = \mathcal{L}_{\text{GCN}} + \lambda\mathcal{L}_{\text{GL}} \tag{5}$$

where λ is the nonnegative tuning parameter. Both graph $\mathbf{S}$ and the new representation $\mathbf{Z}$ are updated iteratively by minimizing (5). As a consequence, even though the original

graph $\mathbf{A}$ is low quality, DGCN can finally output discriminative representation of the original data $\mathbf{X}$.

The DGCN methods have been paying much attention on revising the regularization term $\mathcal{L}_{\mathrm{GL}}$ to meet different requirements [38]–[40]. However, these methods have at least two drawbacks. First, both the initial graph $\mathbf{A}$ and the updated graph $\mathbf{S}$, i.e., the second term in (5), are learned from the original data. Second, many previous GCN methods only focused on exploring the local structure of the samples such as kNN graph and the ϵ-neighborhood graph, by ignoring the global structure of the samples. However, Yan *et al.* [41] pointed out that both the local structure and the global structure are important for data analysis as they provide the complementary information to each other to improve the effectiveness of data analysis.

B. Multigraph Learning

In this section, we visualize the proposed framework in Fig. 1 and introduce the proposed multigraph fusion method in detail. First, we propose a projection matrix $\Theta \in \mathbb{R}^{d\times d'}$ to convert the original data $\mathbf{X}$ into the low-dimensional data $\tilde{\mathbf{X}} = \mathbf{X}\Theta$ (where $d' < d$) where the noise and the redundancy are removed as much as possible. Second, we explore the local and global structure of the data through our proposed multigraph fusion method. Specifically, we use multiple kNN graphs $\hat{\mathbf{A}}^{(v)}(v = 1, \ldots, V)$ (where V is the number of graphs) to learn the local structure of the data and employ the self-representation method to learn the global graph $\mathbf{A}^{(0)}$ by preserving the global structure of all samples. Compared with previous graph learning methods, our multigraph fusion method can capture the diversity of the graph structure in $\mathbf{X}$ because the multigraph structure can provide rich edge information. Finally, we integrate the graph learning process and the representation learning process into a framework to realize the dynamic update of the graph structure and the data representation.

1) Initial Graph Generation: We employ the kNN graph method to generate multiple initial graphs. Specifically, the similarity between two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is first defined as

$$a_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\sigma}} \tag{6}$$

where σ is a nonnegative tuning parameter. After calculating the similarity of all samples, we then keep the similarities of the top-k neighbors for each sample and set other similarities as zeros, to finally obtain a sparse kNN graph $\mathbf{A}$, i.e., an initial graph for $\mathbf{X}$.

Obviously, we obtain V initial graphs $\mathbf{A}^{(v)}$ $(v = 1, \ldots, V)$ by setting different k values. The graphs $\mathbf{A}^{(v)}$ $(v = 1, \ldots, V)$ preserve the local structure as each node in the graph connects only k nearest neighbors. We let $\mathbf{A}^{(v)} = (\mathbf{A}^{(v)} + \mathbf{A}^{(v)^T})/2$ to guarantee that the learned graphs are asymmetric.

2) Local Structure Learning: The local structure of the data, i.e., the local neighborhood relationship of a dataset, is important to maintain the manifold structure of high-dimensional data [42] and is often characterized through the k nearest neighbors of each data point in this work. Traditional local structure learning methods include locality preserving projection (LPP) and locally linear embedding (LLE). However, these methods have the disadvantages as follows. First, it is difficult to determine the value of k. Second, the initial kNN graphs are learned from the original data. Hence, we propose to update the initial kNN graph from the low-dimensional space of the original data by the following objective function:

$$\begin{aligned} \min_{\hat{\mathbf{A}}^{(v)},\Theta} & \sum_{i,j=1}^{n} \left\|\mathbf{x}_i\Theta - \mathbf{x}_j\Theta\right\|_F^2 \hat{a}_{ij}^{(v)} + \lambda_1 \left\|\hat{\mathbf{A}}^{(v)} - \mathbf{A}^{(v)}\right\|_F^2 \\ \text{s.t.} \quad & \forall i,\ \hat{\mathbf{a}}_i^{(v)}\mathbf{1} = 1,\ \hat{a}_{ij}^{(v)} \geq 0 \end{aligned} \tag{7}$$

where λ_1 is a nonnegative tuning parameter and $\mathbf{1}$ indicate the all-one-element vector. $\hat{\mathbf{A}}^{(v)} \in \mathbb{R}^{n\times n}$ is the updated graph matrix of the initial graph $\mathbf{A}^{(v)}$ and $\Theta \in \mathbb{R}^{d\times d'}$ is the projection matrix, which converts the high-dimensional data $\mathbf{X}$ into the low-dimensional space. In (7), the graph $\hat{\mathbf{A}}^{(v)}$ and the projection matrix Θ are iteratively updated. Regarding that all the initial graphs are learned from the original data which may contain noise and redundancy, we use the low-dimensional data $\tilde{\mathbf{X}} = \mathbf{X}\Theta$ to update them.

3) Global Structure Learning: In (7), $\hat{\mathbf{A}}^{(v)}$ only takes the local structure of the data into account. Recent literature shows that the preservation of the global structure of the data is of great importance for some machine learning tasks, i.e., feature selection and classification. The reason is that the global structure effectively contains the discriminative information, which can provide the complementary information different from the local structure preservation to capture the intrinsic structure of the data [43]. Self-representation property has been applied to many real applications and demonstrates its capability in capturing the global structure of data [44]. Specifically, the self-representation property assumes that each data point can be linearly reconstructed from weighted combinations of all

Algorithm 1 Pseudo of Our Proposed Method

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, the label $\mathcal{Y}$, λ_1, λ_2, λ_3, η, and β;

Output: The model weights and classifying unlabelled samples;

```
Generate multiple graphs A^(v) ∈ R^{n×n} (v=1, ..., V) by the kNN graph method from X;
Initialize the model weights (Θ, W^(1), ..., W^(M));
while epoch < 5000 do
    epoch = epoch + 1;
    Â^(v)(v = 1, ..., V) ← {X, Θ, A^(v)} by Eq. (7);
    Â^(0) ← {X, Θ} by Eq. (8);
    S ← {Â^(v')(v = 0, 1, ..., V)} by Eq. (11);
    H^(m)(m = 1, ..., M) ← {S, H^(m−1), W^(m−1)} by Eq. (1);
    Z ← {S, W^M, H^M} by Eq. (2);
    L ← {S, Â^v'(v' = 0, 1, ..., V), Z, Y} by Eq. (13);
    Back-propagate L to update the model weights (p, Θ, W^(1), ..., W^(M));
end while
```

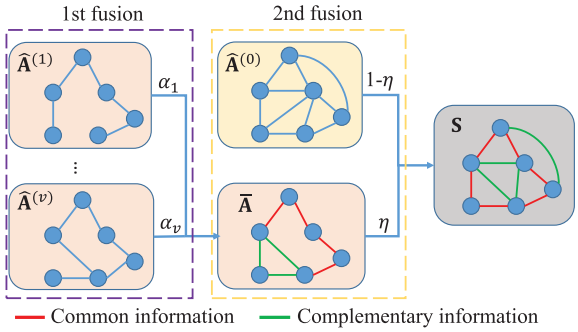

Fig. 2. Visualization of the proposed multigraph fusion method. Specifically, the first fusion (i.e., the purple dot rectangle) outputs the common and complementary information among the local graphs but may have missed edges. The second fusion (i.e., the yellow dot rectangle) outputs the common and complementary information among the local graphs and the global graph as well as adds the missed edges in the first fusion.

other data points, i.e., $\mathbf{x}_i = \hat{\mathbf{a}}_i^{(0)}\mathbf{X} + \mathbf{e}$, where $\hat{\mathbf{a}}_i^{(0)}$ indicates a weight coefficient vector between $\mathbf{x}_i$ and $\mathbf{X}$ and the vector $\mathbf{e}$ is the noise bias. To this end, we propose the following objective function to conduct the global structure learning:

$$\begin{aligned} &\min_{\hat{\mathbf{A}}^{(0)},\Theta} \left\|\mathbf{X}\Theta - \hat{\mathbf{A}}^{(0)}\mathbf{X}\Theta\right\|_F^2 + \lambda_2\left\|\hat{\mathbf{A}}^{(0)}\right\|_1 \\ &\text{s.t.} \quad \forall i,\ \hat{\mathbf{a}}_i^{(0)}\mathbf{1} = 1,\ \hat{a}_{ij}^{(0)} \geq 0 \end{aligned} \tag{8}$$

where λ_2 is a nonnegative tuning parameter. In (8), the first term is used to generate a dense graph $\hat{\mathbf{A}}^{(0)}$, but the ℓ_1-norm on the global graph $\hat{\mathbf{A}}^{(0)}$ pushes to generate sparse representation while preserving the global structure.

4) Local and Global Preservation Learning: Equations (7) and (8), respectively, conduct the local structure learning and the global structure learning. Moreover, two kinds of learning are based on the low-dimensional data $\mathbf{X}\Theta$. Hence, in this article, we integrate the local structure learning and the global structure learning with the projection matrix learning to have

$$\begin{aligned} \mathcal{L}_{\text{gl}}: \min_{\hat{\mathbf{A}}^{(0)},\hat{\mathbf{A}}^{(1)},\ldots,\hat{\mathbf{A}}^{(V)},\Theta} & \sum_{v=1}^{V} tr\left(\Theta^T\mathbf{X}^T\mathbf{L}^{(v)}\mathbf{X}\Theta\right) \\ &+ \lambda_1\left\|\mathbf{X}\Theta - \hat{\mathbf{A}}^{(0)}\mathbf{X}\Theta\right\|_F^2 \\ &+ \lambda_2\sum_{v=1}^{V}\left\|\hat{\mathbf{A}}^{(v)} - \mathbf{A}^{(v)}\right\|_F^2 + \lambda_3\left\|\hat{\mathbf{A}}^{(0)}\right\|_1 \\ \text{s.t.} \quad & \forall i,\ \hat{\mathbf{a}}_i^{(v')}\mathbf{1} = 1,\ \hat{a}_{ij}^{(v')} \geq 0,\ \left(v' = 0, 1, \ldots, V\right) \end{aligned} \tag{9}$$

where $\mathbf{L}^{(v)} = \mathbf{D}^{(v)} - \hat{\mathbf{A}}^{(v)}$ is the Laplacian matrix of $\hat{\mathbf{A}}^{(v)}$, $\mathbf{1}$ is a vector whose elements are one, and tr(·) is the trace operator. There are $(V + 2)$ variables in (9), i.e., $\hat{\mathbf{A}}^{(v')}(v' = 0, \ldots, V)$ and Θ, where Θ is the trainable parameter update by the backpropagation process, $\hat{\mathbf{A}}^{(v')}(v' = 0, \ldots, V)$ are updated by (10). The details of the optimization of (9) are listed in Algorithm 1.

First, if we fix Θ, the optimization with respect to $\hat{\mathbf{A}}^{(v')}(v = 1, \ldots, V)$ is (7). To reduce the complexity, we follow [26] to approximately optimize $\mathbf{p} \in \mathbb{R}^{r\times 1}$ by

$$\hat{a}_{ij}^{(v)} = \frac{\exp\left(\text{ReLU}\left(\mathbf{p}^T\left|\mathbf{x}_i\Theta - \mathbf{x}_j\Theta\right|\right)\right)}{\sum_{j=1}^{n}\exp\left(\text{ReLU}\left(\mathbf{p}^T\left|\mathbf{x}_i\Theta - \mathbf{x}_j\Theta\right|\right)\right)} \tag{10}$$

where ReLU(.) is the activation function. If ${a_{ij}}^{(v)}$ is unavailable, we can set ${a_{ij}}^{(v)} = 1$ in the above update rule. After optimizing (17), we obtain the updated graphs $\hat{\mathbf{A}}^{(v')}$ $(v' = 0, 1, \ldots, V)$ on the low-dimensional features $\mathbf{X}\Theta$, rather than on the high-dimensional original feature matrix as in previous DGCN methods. Moreover, the projection matrix and the graphs are iteratively optimized, and thus, (17) considers the local structure and the global structure to conduct dynamic graph learning.

C. Objective Function

With the assumption that both the local graph structure and the global graph structure can provide complementary information different from others, we combine the learned local graphs and the global graph to obtain the input graph $\mathbf{S}$ for the GCN method as follows:

$$\mathbf{S} = \eta\bar{\mathbf{A}} + (1 - \eta)\hat{\mathbf{A}}^{(0)} \tag{11}$$

where $\bar{\mathbf{A}} = \sum_{v=1}^{V}\alpha_v\hat{\mathbf{A}}^{(v)}$, α_v denotes the weight (importance) of $\hat{\mathbf{A}}^{(v)}$, and η is a tuning parameter, which can be used to control the weight between two different kinds of graphs.

In (11), we combine multiple local graphs with a global graph by two weights, i.e., α and η. The parameter η can be tuned in the implementation and the value of the parameter α_v can be obtained by

$$\alpha_v = \frac{\exp\left(\sum_{ij}\hat{a}_{ij}^{(v)}\hat{a}_{ij}^{(0)}\right)}{\sum_{v=1}^{V}\exp\left(\sum_{ij}\hat{a}_{ij}^{(v)}\hat{a}_{ij}^{(0)}\right)}. \tag{12}$$

In (12), each local graph has individual distributions. Moreover, the larger the value of α_v, the more important $\hat{\mathbf{A}}^{(v)}$.

Finally, we list our proposed objective function as follows:

$$\mathcal{L} = \mathcal{L}_{\text{GCN}} + \beta\mathcal{L}_{\text{gl}} \tag{13}$$

where β is a nonnegative tuning parameter.

The proposed method in (13) conducts graph fusion twice, as shown in Fig. 2. The first fusion combines all sparse graphs to preserve the local structure and the parameter $\boldsymbol{\alpha}$ automatically learns the weight of every graph. Specifically, if some edges are only found in a small part of all $\hat{\mathbf{A}}^{(v)}(v = 1, \ldots, V)$, the first fusion may automatically assign them with a small or even zero weight. Hence, they can be regarded as noisy edges and are removed out of $\bar{\mathbf{A}}$. If some edges are found in some sparse graphs with large weights, the first fusion may keep them in $\bar{\mathbf{A}}$ as the complementary information across the local graphs. If some edges are found in most graphs, the first fusion regards them as the common information among all local graphs kept in $\bar{\mathbf{A}}$. As a result, $\bar{\mathbf{A}}$ is a high-quality graph preserving the local structure but could still miss some edges. The second fusion integrates the local graph $\bar{\mathbf{A}}$ and the global graph $\hat{\mathbf{A}}^{(0)}$ to address this issue as follows. Specifically, the common information (i.e., the common edges between $\bar{\mathbf{A}}$ and $\hat{\mathbf{A}}^{(0)}$) and the complementary information (i.e., the

edges appearing in either $\bar{\mathbf{A}}$ or $\hat{\mathbf{A}}^{(0)}$) are outputted. Hence, our proposed method conducts the graph fusion twice to guarantee the quality of the learned graph $\mathbf{S}$.

D. Time Complexity

The proposed model has $V+1$ trainable graphs i.e., $\hat{\mathbf{A}}^{(v')}(v'=0,\ldots,V)$, the time complexity of learning an graph matrix is $\mathcal{O}(n^2d')$, where n is the number of samples and d' is the number of features of the original data after dimensionality reduction. However, $V+1$ graph matrices can be trained at the same time in our proposed method, and thus, the total time complexity of graph learning in our proposed method is $\mathcal{O}(n^2d')$. The time complexity of GCN is $\mathcal{O}(Mn^2d'+Mnd'^2)$, where M is the number of hidden layers; hence, the overall time complexity of our proposed model is $\mathcal{O}(n^2d'+Mn^2d'+Mnd'^2)$.

IV. Experimental Analysis

We conducted extensive experiments on 12 real datasets to compare our method with seven comparison methods in terms of semisupervised node classification.

A. Experiment Setting

1) Datasets: In our experiments, the used public datasets included four network datasets (i.e., Citeseer, Cora, Pubmed, and Wiki-CS), seven image datasets (i.e., CIFAR10, SVHN, MNIST, Scene15, AWA, Handwritten, and Caltech), and a text dataset (i.e., BBC). The details of all used datasets are listed as follows.

1) Citeseer includes 3327 nodes distributed in six classes, where each node is represented by a 3703-D feature descriptor.
2) Cora has 2708 nodes, which is represented by a 1433-D features. Moreover, all samples were classified into six classes.
3) Pubmed includes 19 717 nodes and each node has 500 features. Moreover, all nodes were classified into three classes.
4) Wiki-CS has 11 701 nodes distributed in ten classes, where each node is represented by 300-D features.
5) CIFAR10 includes 50 000 natural images distributed in ten different classes.
6) SVHN has 5000 images, where each class has 500 images.
7) MNIST consists of 5000 images of hand-written digits from "0" to "9" in our experiments. Moreover, each class has 500 images.
8) Scene15 includes 4485 image distributed in 15 different classes. We followed [45] to obtain the feature of the samples.
9) BBC consists of 2225 news documents within five categories, i.e., business, entertainment, politics, sport, and technique.
10) Handwritten extracts normalized bitmaps of handwritten digits from a preprinted form. In our experiments, it has 2000 images that are in ten classes.
11) Caltech is an image dataset, including 1474 images that are distributed in seven classes.
12) AWA describes 50 animals by 4000 images. Each image is represented by 2688-D color histogram features.

2) Comparison Methods: The comparison methods included one baseline method, i.e., GCN [25], and six state-of-the-art method, i.e., graph attention network (GAT) [46], joint learning graph convolutional network (JLGCN) [47], graph learning convolution networks (GLCNs) [26], deep iterative and adaptive learning graph neural network (DIAL-GNN) [40]), multiple graph learning, convolutional networks (M-GLCNs) [36], and graph structure learning for robust graph neural networks (GLR-GNN) [48].

1) GCN encodes the feature information and the structure information of the data by the defined graph convolution operators. Moreover, it includes two convolutional layers (each of which had 20 units), followed by the softmax normalization for the label prediction.
2) GAT uses mask self-attention layers to assign different weights to different nodes within a neighborhood without relying on prior knowledge of the graph structure.
3) JLGCN jointly learns the graph structure and feature representation in the GCN framework. Specifically, it optimizes an underlying graph kernel via distance metric learning with the Mahalanobis distance. Moreover, the metric matrix is decomposed into a low-dimensional matrix and a graph Laplacian regularizer.
4) GLCN integrates graph learning operations and traditional graph convolution structures in a unified network. To do this, it designs a graph learning regularization term to learn the optimal graph structure.
5) DIAL-GNN casts the graph structure learning problem as a similarity metric learning problem as well as leverages an adapted graph regularization for controlling smoothness, connectivity, and sparsity of the generated graph. DIAL-GNN designs a novel iterative method for searching for a hidden graph structure that augments the initial graph structure.
6) M-GLCN jointly learns the multigraph structure and the feature representation in the GCN framework. To do this, it designs a multigraph learning mechanism to learn the internal relationship among the multigraph structure.
7) GLR-GNN targets on exploring the graph properties of sparsity, low rank, and feature smoothness, aiming at designing robust graph neural networks. To do this, it uses a regularization term to learn the clean graph structure from the noisy graph data and jointly learn parameters for the robust graph neural network and the clean structure.

GCN conducts representation learning by preserving the local structure in the initial graph, which is invariant in the whole training process. The methods (e.g., JLGCN, GLCN, GLR-GNN, M-GLCN, and DIAL-GNN) jointly conduct graph learning and representation learning by designing different methods to improve the quality of the initial graph. GAT uses attention coefficients to combine its features with the features from its neighborhood to obtain a new representation.

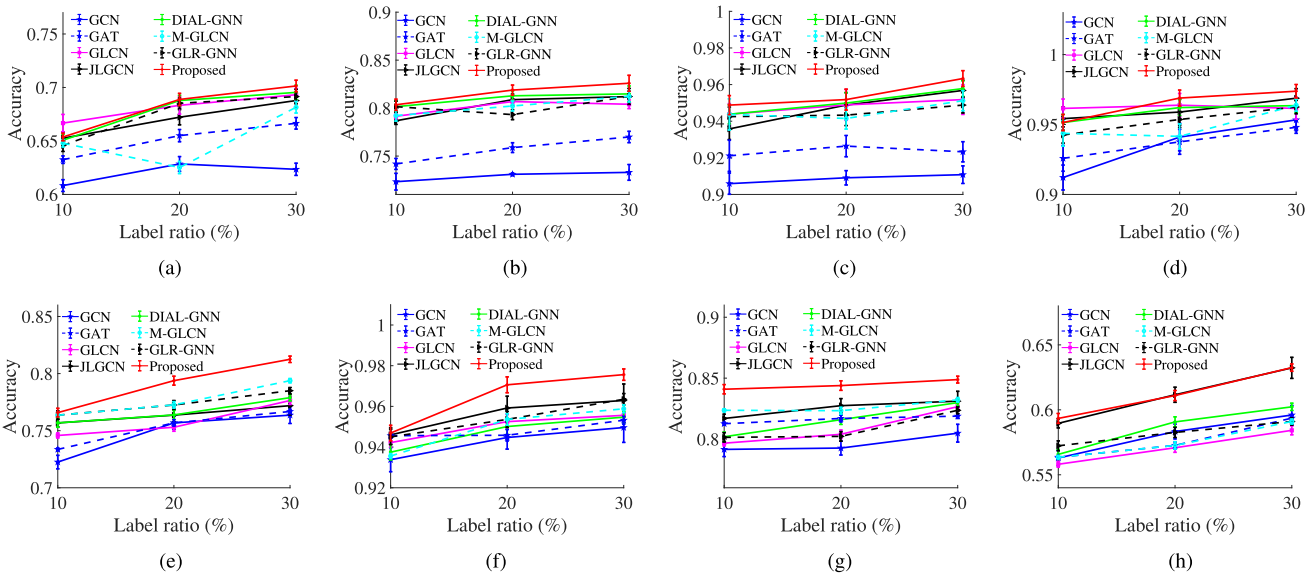

Fig. 3. Classification accuracy of all methods on eight datasets. (a) CIFAR10. (b) SVHN. (c) MNIST. (d) Scene15. (e) BBC. (f) Handwritten. (g) Caltech. (h) AWA.

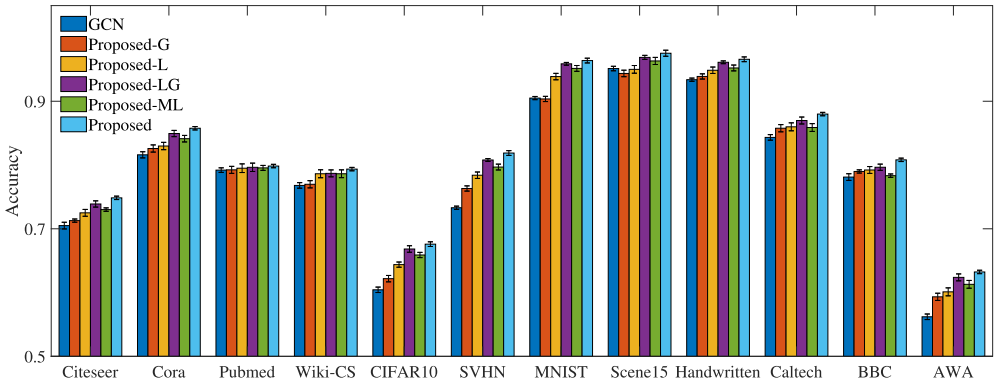

Fig. 4. Classification accuracy of our methods and GCN on all 12 datasets.

TABLE II
CLASSIFICATION RESULTS (%) OF ALL METHODS ON FOUR CITATION NETWORK DATASETS (I.E., CITESEER, CORA, PUBMED, AND WIKI-CS)

Methods	Citeseer	Cora	Pubmed	Wiki-CS
GCN	70.50 ± 0.26	81.60 ± 0.48	79.20 ± 0.32	76.80 ± 0.12
GAT	72.60 ± 0.21	83.18 ± 0.15	79.14 ± 0.87	77.11 ± 0.11
GLCN	72.50 ± 0.68	85.26 ± 0.58	78.80 ± 0.33	77.39 ± 0.23
JLGCN	73.58 ± 0.54	83.77 ± 0.21	79.32 ± 0.84	77.88 ± 0.13
DIAL-GNN	73.89 ± 0.50	84.50 ± 0.39	79.31 ± 0.30	77.64 ± 0.15
M-GLCN	74.15 ± 0.18	84.25 ± 0.34	79.11 ± 0.26	76.13 ± 0.32
GLR-GNN	74.22 ± 0.23	84.75 ± 0.39	79.31 ± 0.14	78.75 ± 0.28
Proposed	**74.85** ± 0.42	**85.77** ± 0.52	**79.84** ± 0.29	**79.36** ± 0.18

3) Setting Up: There are 12 datasets in our experiment. For four citation network datasets (i.e., Citeseer, Cora, Pubmed, and Wiki-CS), we follow the previous literature [25] to divide the datasets into a training set, validation set, and test set. For eight datasets (i.e., CIFAR10, SVHN, MNIST, Scene15, BBC, Handwritten, Caltech, and AWA), we randomly select 10%, 20%, and 30% of the samples as the training set. For remaining samples, we select 30% of the samples for the validation set and 30% of the samples for the test set. All methods were verified by the tenfold cross-validation scheme. Moreover, we repeated this scheme ten times and reported the averaged results and the corresponding standard deviations of 100 experiments as the final result. For the model selection, we referred to the corresponding literature to make them output the best performance. In our experiment, our method was set two hidden layers, each of which had 30 units. We set the maximum of epochs as 5000 for the training process with the Adam optimizer and stopped the training process if the loss did not decrease for 100 consecutive epochs. In addition, we set both the initial learning rate and the weight decay as 0.005. For seven image datasets and a text dataset, we constructed the initial graph by a kNN graph (i.e., $k = 10$) and set the k value as 5, 10, and 15, to obtain multiple graphs. Moreover, the network weights of all methods were initialized by the Glorot initialization.

B. Result Analysis

We report the results of four citation network datasets in Table II and the results of eight datasets in Fig. 3.

First, our method achieved the best performance, followed by GLR-GNN, DIAL-GNN, M-GLCN, GLCN, JLGCN, GAT, and GCN. For example, in Table II, our method improved

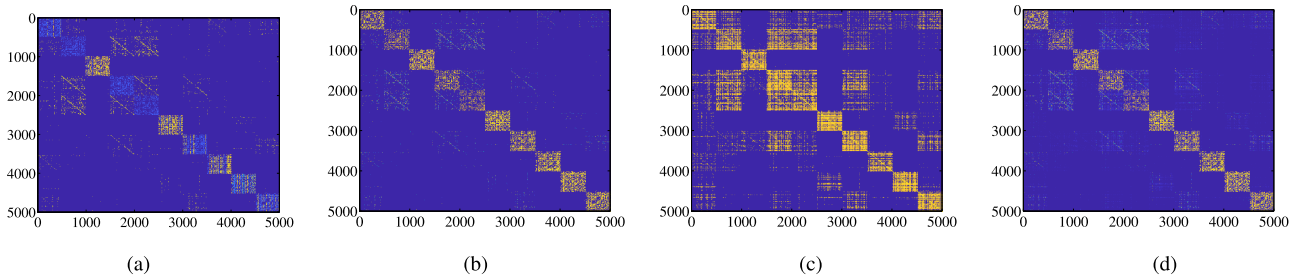

Fig. 5. Visualization of four types of graphs. (a) Original graph. (b) Local graph. (c) Global graph. (d) Fused graph.

by, on average, 0.63%, 1.02%, 0.53%, and 0.61% compared to the best comparison method GLR-GNN, in terms of classification accuracy. In Fig. 3, our method improved by, on average, 0.89%, 1.17%, and 1.56%, compared to the comparison method GLR-GNN, in terms of different label ratios, i.e., 10%, 20%, and 30%, on eight datasets. The possible reason is that our method considers both multigraph learning and representation learning. As a result, each graph provides the complementary information different from other graphs to find the edges excluded in some graphs and all graphs provide the common information to remove the noisy edges only existed in a small part of the graphs.

Second, both our method and M-GLCN are designed to learn the graph structure of the data from multiple graphs, while the other five methods directly adjust an initial graph. As a result, the multigraph methods (i.e., our method and M-GLCN) outperformed others (i.e., GCN, GAT, GLCN, JLGCN, DIAL-GNN, and GLR-GNN). For example, the multigraph methods improved by, on average, 2.35%, compared to others, on eight datasets with different label ratios. This implies the feasibility to consider the multigraph learning for graph learning in deep learning models. In addition, our proposed method is superior to M-GLCN. The possible reason is that our method considers both the global structure and the local structure of the data by a regularization term.

Third, the DGCN methods (i.e., our method, DIAL-GNN, GLCN, GLR-GNN, M-GLCN, and JLGCN) beat GCN. For example, these methods improved by, on average, 4.89%, 4.36%, 3.69%, 4.68%, 4.51, and 3.38%, compared to GCN, on all datasets. This clearly demonstrates that dynamic graph learning is helpful to improve the performance of GCN methods.

C. Ablation Study

1) Sensitive Analysis of Different Graphs: Our method learns multiple graphs from low-dimensional space of the original data to preserve their local structures and global structure. In this section, we verified the effectiveness of different graphs by extending our method in (17) to the following methods and list the detail of five proposed methods in Table III.

1) Proposed-L considers the local structure in one graph by removing the second term in (17) and setting $k = 5$.
2) Proposed-G considers the global structure by removing the first term in (17).

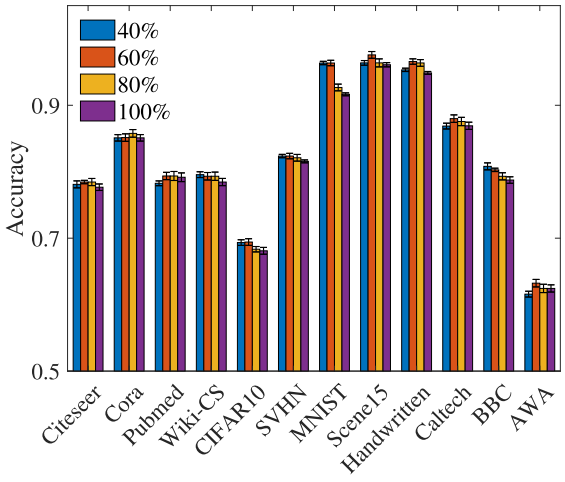

Fig. 6. Classification accuracy of our methods and proposed-RO on six datasets.

3) Proposed-LG considers the local structure in one graph by setting $k = 5$ and the global structure.
4) Proposed-ML considers the local structure in multiple graphs by removing the second term in (17) and setting $k = [5, 10, 15]$.

We reported the classification results of six methods (i.e., Proposed-L, Proposed-G, Proposed-ML, Proposed-LG, GCN, and Proposed) in Fig. 4. It is noteworthy that the results of both proposed and GCN are the same as them in Fig. 3 and Table II. First, Proposed-L outperformed Proposed-G, improving by, on average, 1.35% on 12 datasets. This shows that the local structure is more useful than the global structure, similar to the conclusion in [19]. However, Proposed-G beats GCN on all datasets, implying the effectiveness of dynamic graph learning and global structure preservation. Second, Proposed-ML improved by, on average, 1.14%, compared to Proposed-L, on 12 datasets. It contributes to the fact that a single graph often cannot capture all the information of the data [18]. This clearly demonstrates the feasibility of multigraph learning in our method. Third, the proposed one outperformed Proposed-LG as the latter only uses a single original graph and the proposed one uses multiple graphs. This verifies that the learning with multiple graphs may be better than the learning with the single graph [18].

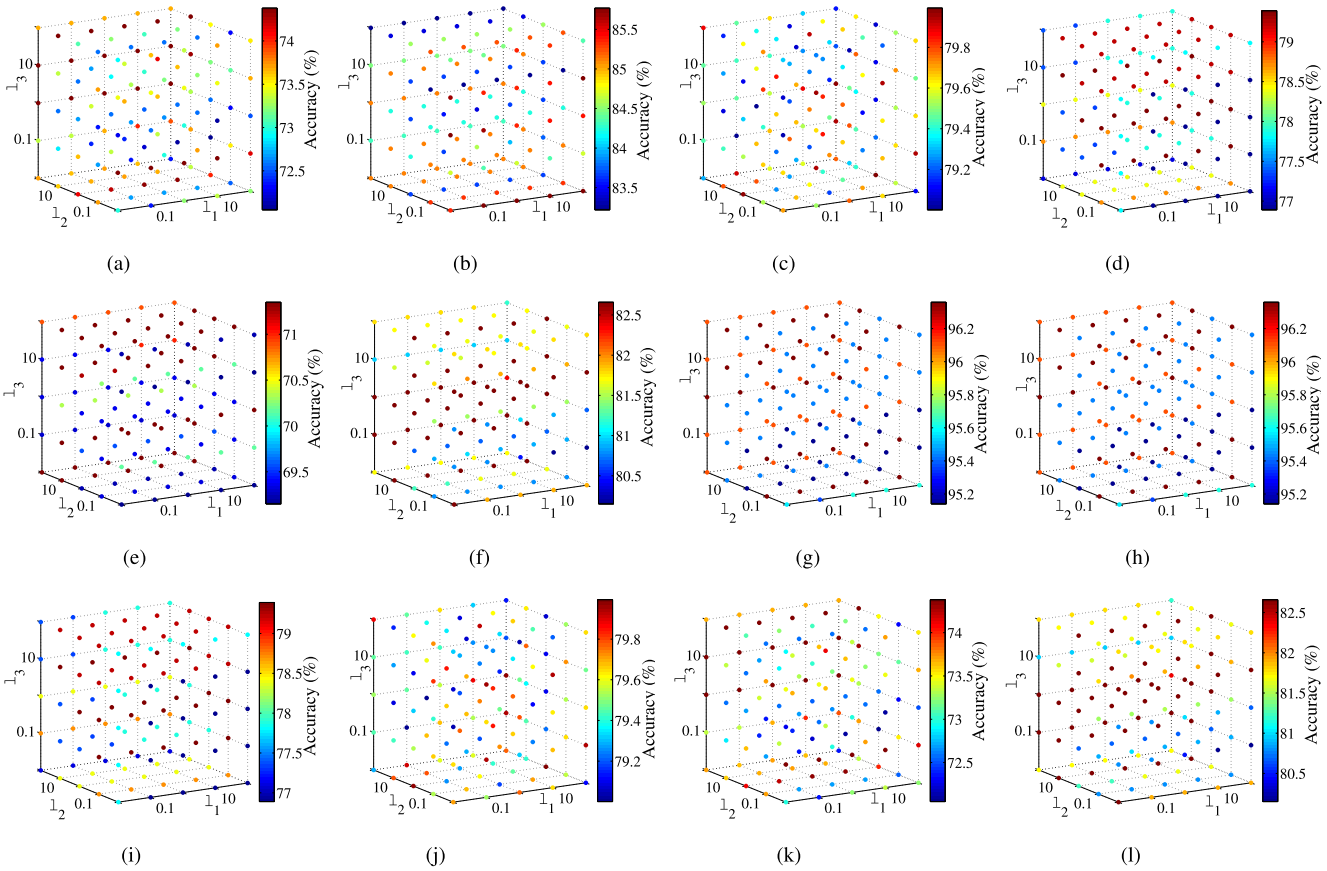

Fig. 7. Classification accuracy of our method at different parameter settings (i.e., λ_1, λ_2, and λ_3) on 12 datasets. (a) Citeseer. (b) Cora. (c) Pubmed. (d) Wiki-CS. (e) CIFAR10. (f) SVHN. (g) MNIST. (h) Scene15. (i) BBC. (j) Handwritten. (k) Caltech. (l) AWA.

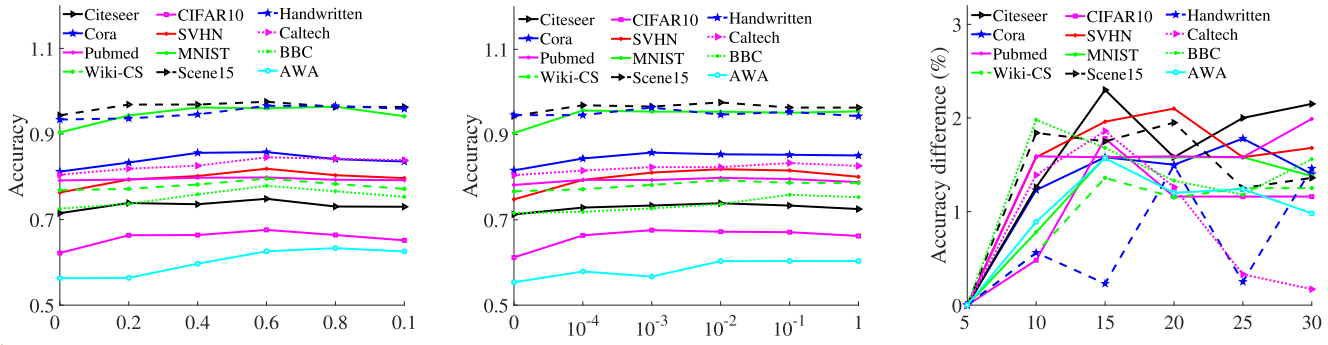

Fig. 8. Results of our method at different variations of parameter η (left), parameter β (middle), and parameter k (right).

2) Visualization of Different Graphs: In this section, we compare the difference among four graphs (i.e., the local graph, the global graph, the fused graph, and the original kNN graph) on the dataset MNIST, where we set $k = 200$ and visualized the results in Fig. 5.

From Fig. 5, we have the observations as follows. First, either the global graph or the local graph has more clear structures, compared to the original kNN graph. This indicates the effectiveness of either local structure learning or global structure learning, as shown in the literature of traditional graph learning [49]. Second, the local graph is more clear compared to the structure of the global graph. This shows that local information is more important than the global structure in classification tasks. Third, the fused graph is the most clear one among all the graphs. This shows that it is reasonable for our method to consider both the local structure and the global structure.

3) Effectiveness of the Dimensionality Reduction: In this section, we conduct experiment to verify the effectiveness of dimensionality reduction of our method by varying the value of d' in $\Theta \in \mathbb{R}^{d \times d'}$. We report the classification accuracy of the proposed method with different values of $d' \in \{40\% * d, 60\% * d, 80\% * d, 100\% * d\}$ in Fig. 6.

TABLE III
DETAIL OF THE FIVE PROPOSED METHODS (× INDICATES THAT THE METHOD DOES NOT CONTAIN THIS PART AND √ MEANS THAT THE METHOD CONTAINS THIS PART)

Methods	Global learning	Local learning	Multi-graph learning
Proposed-L	√	×	×
Proposed-G	×	√	×
Proposed-LG	√	√	×
Proposed-ML	√	×	√
Proposed	√	√	√

From the results, the proposed method achieves the worst classification results on 12 datasets with the setting of $d' = 100\% * d$. This clearly demonstrates that dimensionality reduction is effective in our method and it is feasible to learn the local structure of the data from a low-dimensional space, as redundant features in the original high-dimensional space may influence the model robustness.

D. Parameter Sensitivity Analysis

We evaluated the effectiveness of our method at different settings of the parameters, i.e., λ_1, λ_2, and λ_3 in (17), η in (11), and β in (13).

We summarized the classification results of our method with different settings of the parameters (e.g., λ_1, λ_2, and $\lambda_3 \in \{10^{-2}, 10^{-1}, \ldots, 10^{2}\}$) on 12 datasets in Fig. 7, in which color represents the classification accuracy and the color bar stands for the range of variation of classification accuracy with different parameter settings. Obviously, our method is sensitive to the settings of three parameters as different parameter settings made different accuracies. Specifically, the variation ranges of accuracy were about 2.45%, 2.38%, 0.84%, 2.10% 1.88%, 2.17%, 1.19%, 1.08%, 2.04%, 1.89%, 2.45%, and 1.88% on datasets Citeseer, Cora, Pubmed, Wiki-CS, CIFAR10, SVHN, MNIST, Scene15, BBC, Handwritten, Caltech, and AWA respectively. This shows: 1) our method can learn useful information from the original graph; 2) the sparsity of $\hat{\mathbf{A}}^{(0)}$ may affect the performance of the model; and 3) our method easily achieves good performance by setting the ranges of λ_1, λ_2, and λ_3 as [0.1, 10].

The parameter η in (11) conducts the tradeoff between the local structure graph $\bar{\mathbf{A}}$ and the global structure graph $\hat{\mathbf{A}}^{(0)}$. We reported the classification results of our method with different settings of the parameter η on the left subfigure of Fig. 8. The larger the value of η, the less importance the global structure. Specifically, the accuracy of our method increased with the varied values of η from 0 to 0.6 and decreased with the varied values of η from 0.6 to 1. This indicates that: 1) the local structure is more important than the global structure in our method and 2) our method needs both of them because either the complementary information in the individual graph or the common information among them may improve the quality of the graph in our method.

We reported the classification accuracy of our method with the variations of the parameter β in the middle subfigure of Fig. 8. Clearly, if β was set to 0, the performance of our method is similar to GCN. However, the accuracy increased, while the values of β were varied from 10^{-4} to 1. This verifies that the regularization term is important in our method. Hence, the graph learning regularization term is essential in our method, similar to the conclusion in the literature of the DGCN methods [3], [47].

The right subfigure of Fig. 8 reports the accuracy difference between the accuracy of our method under different k settings (e.g., $k \in \{5, 10, 15, 20, 25, 30\}$) and the classification accuracy of our method in the case $k = 5$. Based on the experimental results, the accuracy of our method increased with the varied values of k from 5 to 15 and decreased with the varied values of k from 15 to 30. In addition, our method achieved the best experimental results, while the values of k are either 15 or 20. This demonstrates that: 1) the small values of k cannot fully describe the neighborhoods of the data and large values of k will increase the chance of wrong neighborhoods, which makes the relationship among samples less discriminative, and 2) our proposed local structure learning method is sensitive to the value of k.

V. CONCLUSION AND FUTURE WORKS

In this article, we have designed a novel dynamic GCN by proposing a multigraph fusion method to improve the quality of the graph in the GCN method. To do this, our method first learned the unified local graph preserving the local structure from multiple initial graphs and the global graph preserving the global structure from the low-dimensional space of the original high-dimensional data and then proposed a novel fusion method to integrate their complementary information and common information, aiming at correctly capturing the intrinsic graph structure inherent in the data. Experimental results on 12 real datasets showed the effectiveness of our method, compared to state-of-the-art comparison methods.

Our proposed method was set to deal with the scenario of semisupervised learning. In the future work, we will extend it to conduct supervised learning. In particular, we plan to combine our proposed method of local and global structure learning with contrastive learning for self-supervised learning.

APPENDIX

A. Details the Derivation of (10)

The purpose of (10) is to learn the similarity between two nodes. In this article, we use the Euclidean distance to measure the similarity between data points

$$e_{ij} = \left\| \mathbf{x}_i \Theta - \mathbf{x}_j \Theta \right\|. \tag{14}$$

To make coefficients easily comparable across different nodes, we normalize them across all choices of j using the softmax function

$$s_{ij} = \operatorname{softmax}\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{j=1}^{n} \exp\left(e_{ij}\right)}. \tag{15}$$

In this article, the graph learning layer is parameterized by a weight vector $\mathbf{p}$ and applying the ReLU nonlinearity. Fully expanded out, the similarity computed may then be expressed as

$$\hat{a}_{ij}^{(v')} = \frac{\exp\left(\operatorname{ReLU}\left(\mathbf{p}^T \left\| \mathbf{x}_i \Theta - \mathbf{x}_j \Theta \right\|\right)\right)}{\sum_{j=1}^{n} \exp\left(\operatorname{ReLU}\left(\mathbf{p}^T \left\| \mathbf{x}_i \Theta - \mathbf{x}_j \Theta \right\|\right)\right)}. \tag{16}$$

B. Details the Mathematical Transformation of (9)

We first perform the following mathematical transformation on (7):

$$\begin{aligned}
&\sum_{i,j=1}^{n}\left\|\mathbf{x}_i\Theta-\mathbf{x}_j\Theta\right\|_F^2\hat{a}_{ij}^{(v)}+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=\sum_{i,j=1}^{n}\left\|\tilde{\mathbf{x}}_i-\tilde{\mathbf{x}}_j\right\|_F^2\hat{a}_{ij}^{(v)}+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=\sum_{i,j=1}^{n}\left(\tilde{\mathbf{x}}_i^T\tilde{\mathbf{x}}_i-2\tilde{\mathbf{x}}_i^T\tilde{\mathbf{x}}_j+\tilde{\mathbf{x}}_j^T\tilde{\mathbf{x}}_j\right)\hat{a}_{ij}^{(v)}+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=\sum_{i=1}^{n}\left(\sum_{j=1}^{n}\hat{a}_{ij}^{(v)}\right)\tilde{\mathbf{x}}_i^T\tilde{\mathbf{x}}_i+\sum_{j=1}^{n}\left(\sum_{i=1}^{n}\hat{a}_{ij}^{(v)}\right)\tilde{\mathbf{x}}_j^T\tilde{\mathbf{x}}_j\\
&\quad-2\sum_{i=1}^{n}\sum_{j=1}^{n}\tilde{\mathbf{x}}_i^T\tilde{\mathbf{x}}_j\hat{a}_{ij}^{(v)}+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=2\sum_{i=1}^{n}d_{ii}^{(v)}\tilde{\mathbf{x}}_i^T\tilde{\mathbf{x}}_i-2\sum_{i=1}^{n}\sum_{j=1}^{n}\tilde{\mathbf{x}}_i^T\tilde{\mathbf{x}}_j\hat{a}_{ij}^{(v)}+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=2\sum_{i=1}^{n}\left(\sqrt{d_{ii}^{(v)}}\tilde{\mathbf{x}}_i\right)^T\left(\sqrt{d_{ii}^{(v)}}\tilde{\mathbf{x}}_i\right)^T-2\sum_{i=1}^{n}\tilde{\mathbf{x}}_j^T\left(\sum_{j=1}^{n}\tilde{\mathbf{x}}_j\hat{a}_{ij}^{(v)}\right)\\
&\quad+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=2\text{trace}\left(\tilde{\mathbf{X}}^T\mathbf{D}^{(v)}\tilde{\mathbf{X}}\right))-2\sum_{i=1}^{n}\tilde{\mathbf{x}}_j^T\left(\tilde{\mathbf{X}}\hat{\mathbf{A}}^{(v)}\right)_i+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=2\text{trace}\left(\tilde{\mathbf{X}}^T\mathbf{D}^{(v)}\tilde{\mathbf{X}}\right))-2\text{trace}\left(\tilde{\mathbf{X}}^T\hat{\mathbf{A}}^{(v)}\tilde{\mathbf{X}}\right))+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=2\text{trace}\left(\tilde{\mathbf{X}}^T\left(\mathbf{D}^{(v)}-\hat{\mathbf{A}}^{(v)}\right)\tilde{\mathbf{X}}\right)+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2\\
&=2\text{trace}\left(\tilde{\mathbf{X}}^T\mathbf{L}^{(v)}\tilde{\mathbf{X}}\right)+\lambda_1\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2.
\end{aligned}$$

Combining (7) with (8), we can obtain

$$\begin{aligned}
&\sum_{v=1}^{V}\text{tr}\left(\Theta^T\mathbf{X}^T\mathbf{L}^{(v)}\mathbf{X}\Theta\right)+\lambda_1\left\|\mathbf{X}\Theta-\hat{\mathbf{A}}^{(0)}\mathbf{X}\Theta\right\|_F^2\\
&\qquad+\lambda_2\sum_{v=1}^{V}\left\|\hat{\mathbf{A}}^{(v)}-\mathbf{A}^{(v)}\right\|_F^2+\lambda_3\left\|\hat{\mathbf{A}}^{(0)}\right\|_1.
\end{aligned}$$

REFERENCES

[1] S. Gadgil, Q. Zhao, A. Pfefferbaum, E. V. Sullivan, E. Adeli, and K. M. Pohl, "Spatio-temporal graph convolution for resting-state FMRI analysis," in *Proc. MICCAI*, 2020, pp. 528–538.

[2] X. Jiang, R. Zhu, S. Li, and P. Ji, "Co-embedding of nodes and edges with graph neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Oct. 14, 2020, doi: 10.1109/TPAMI.2020.3029762.

[3] L. Peng *et al.*, "Reverse graph learning for graph neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 5, 2022, doi: 10.1109/TNNLS.2022.3161030.

[4] Y. Zhu, J. Ma, C. Yuan, and X. Zhu, "Interpretable learning based dynamic graph convolutional networks for Alzheimer's disease analysis," *Inf. Fusion*, vol. 77, pp. 53–61, Jan. 2022.

[5] J. Cheng, Q. Wang, Z. Tao, D. Xie, and Q. Gao, "Multi-view attribute graph convolution networks for clustering," in *Proc. IJCAI*, Jul. 2020.

[6] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[7] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *Proc. ICML*, 2020, pp. 1725–1735.

[8] X. Jiang, P. Ji, and S. Li, "CensNet: Convolution with edge-node switching in graph neural networks," in *Proc. IJCAI*, Aug. 2019, pp. 2656–2662.

[9] M. Wu, S. Pan, C. Zhou, X. Chang, and X. Zhu, "Unsupervised domain adaptive graph convolutional networks," in *Proc. Web Conf.*, Apr. 2020, pp. 1457–1467.

[10] R. Zhou, X. Chang, L. Shi, Y.-D. Shen, Y. Yang, and F. Nie, "Person reidentification via multi-feature fusion with adaptive graph learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 5, pp. 1592–1601, May 2020.

[11] X. Zhu, S. Zhang, Y. Zhu, P. Zhu, and Y. Gao, "Unsupervised spectral feature selection with dynamic hyper-graph learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 6, pp. 3016–3028, Jun. 2022, doi: 10.1109/TKDE.2020.3017250.

[12] L. Sun *et al.*, "Adaptive feature selection guided deep forest for COVID-19 classification with chest CT," *IEEE J. Biomed. Health Informat.*, vol. 24, no. 10, pp. 2798–2805, Oct. 2020.

[13] N. Salem and S. Hussein, "Data dimensional reduction and principal components analysis," *Proc. Comput. Sci.*, vol. 163, pp. 292–299, Jan. 2019.

[14] X. Wang, J. He, and L. Ma, "Exploiting local and global structure for point cloud semantic segmentation with contextual point representations," in *Proc. NIPS*, 2019, pp. 4571–4581.

[15] J. Gan, Z. Peng, X. Zhu, R. Hu, J. Ma, and G. Wu, "Brain functional connectivity analysis based on multi-graph fusion," *Med. Image Anal.*, vol. 71, Jul. 2021, Art. no. 102057, doi: 10.1016/j.media.2021.102057.

[16] C. Yuan, Z. Zhong, C. Lei, X. Zhu, and R. Hu, "Adaptive reverse graph learning for robust subspace learning," *Inf. Process. Manage.*, vol. 58, no. 6, Nov. 2021, Art. no. 102733, doi: 10.1016/j.ipm.2021.102733.

[17] R. Hu *et al.*, "Multi-band brain network analysis for functional neuroimaging biomarker identification," *IEEE Trans. Med. Imag.*, vol. 40, no. 12, pp. 3843–3855, Dec. 2021, doi: 10.1109/TMI.2021.3099641.

[18] Z. Ren and Q. Sun, "Simultaneous global and local graph structure preserving for multiple kernel clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 1839–1851, May 2021.

[19] X. Liu, L. Wang, J. Zhang, J. Yin, and H. Liu, "Global and local structure preservation for feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 6, pp. 1083–1095, Jun. 2013.

[20] Y. Takai, A. Miyauchi, M. Ikeda, and Y. Yoshida, "Hypergraph clustering based on pagerank," in *Proc. KDD*, 2020, pp. 1970–1978.

[21] K. Xiong, F. Nie, and J. Han, "Linear manifold regularization with adaptive graph for semi-supervised dimensionality reduction," in *Proc. IJCAI*, Aug. 2017, pp. 3147–3153.

[22] Z. Li, F. Nie, X. Chang, Y. Yang, C. Zhang, and N. Sebe, "Dynamic affinity graph construction for spectral clustering using multiple features," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6323–6332, Dec. 2018.

[23] M. Luo, X. Chang, L. Nie, Y. Yang, A. G. Hauptmann, and Q. Zheng, "An adaptive semisupervised feature analysis for video semantic recognition," *IEEE Trans. Cybern.*, vol. 48, no. 2, pp. 648–660, Feb. 2018.

[24] T. Wang *et al.*, "Label group diffusion for image and image pair segmentation," *Pattern Recognit.*, vol. 112, Apr. 2021, Art. no. 107789.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[26] B. Jiang, Z. Zhang, D. Lin, J. Tang, and B. Luo, "Semi-supervised learning with graph learning-convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11313–11320.

[27] K. Zhang, T. Li, S. Shen, B. Liu, J. Chen, and Q. Liu, "Adaptive graph convolutional network with attention graph clustering for co-saliency detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9050–9059.

[28] F. Nie, G. Cai, and X. Li, "Multi-view clustering and semi-supervised classification with adaptive neighbours," in *Proc. AAAI*, vol. 31, no. 1, 2017.

[29] H. Wang, Y. Yang, and B. Liu, "GMC: Graph-based multi-view clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 6, pp. 1116–1129, May 2019.

[30] X. Zhu, S. Zhang, W. He, R. Hu, C. Lei, and P. Zhu, "One-step multi-view spectral clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 10, pp. 2022–2034, Oct. 2019.

[31] C. Tang *et al.*, "Cross-view locality preserved diversity and consensus learning for multi-view unsupervised feature selection," *IEEE Trans. Knowl. Data Eng.*, early access, Jan. 1, 2021, doi: 10.1109/TKDE.2020.3048678.

[32] T. Tong, K. Gray, Q. Gao, L. Chen, and D. Rueckert, "Multi-modal classification of Alzheimer's disease using nonlinear graph fusion," *Pattern Recognit.*, vol. 63, pp. 171–181, Mar. 2017.
[33] C. Tang *et al.*, "CGD: Multi-view clustering via cross-view graph diffusion," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 5924–5931.
[34] O. Lindenbaum, A. Yeredor, M. Salhov, and A. Averbuch, "Multi-view diffusion maps," *Inf. Fusion*, vol. 55, pp. 127–149, Mar. 2020.
[35] C. Zhuang and Q. Ma, "Dual graph convolutional networks for graph-based semi-supervised classification," in *Proc. WWW*, 2018, pp. 499–508.
[36] B. Jiang, X. Jiang, A. Zhou, J. Tang, and B. Luo, "A unified multiple graph learning and convolutional network model for co-saliency estimation," in *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 1375–1382.
[37] F. Nie, X. Wang, M. I. Jordan, and H. Huang, "The constrained Laplacian rank algorithm for graph-based clustering," in *Proc. AAAI*, 2016, pp. 1969–1976.
[38] X. Gao, W. Hu, and Z. Guo, "Exploring structure-adaptive graph learning for robust semi-supervised classification," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2020, pp. 1–6.
[39] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. G. Hauptmann, "A comprehensive survey of scene graphs: Generation and application," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Dec. 23, 2022, doi: 10.1109/TPAMI.2021.3137605.
[40] Y. Chen, L. Wu, and M. J. Zaki, "Deep iterative and adaptive learning for graph neural networks," 2019, *arXiv:1912.07832*.
[41] C. Yan, X. Chang, M. Luo, H. Liu, X. Zhang, and Q. Zheng, "Semantics-guided contrastive network for zero-shot object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jan. 4, 2022, doi: 10.1109/TPAMI.2021.3140070.
[42] J. Chen, J. Ye, and Q. Li, "Integrating global and local structures: A least squares framework for dimensionality reduction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[43] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2002.
[44] C. Tang *et al.*, "Feature selective projection with low-rank embedding and dual Laplacian regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 9, pp. 1747–1760, Sep. 2020.
[45] Z. Jiang, Z. Lin, and L. S. Davis, "Label consistent K-SVD: Learning a discriminative dictionary for recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2651–2664, Nov. 2013.
[46] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.
[47] J. Tang, W. Hu, X. Gao, and Z. Guo, "Joint learning of graph representation and node features in graph convolutional neural networks," 2019, *arXiv:1909.04931*.
[48] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 66–74.
[49] J. Zhao *et al.*, "Adaptive learning of local semantic and global structure representations for text classification," in *Proc. ICCL*, 2018, pp. 2033–2043.

Jiangzhang Gan received the M.S. degree in computer science from Guangxi Normal University, Guilin, China, in 2019. He is currently pursuing the Ph.D. degree with the School of Mathematical and Computational Science, Massey University Auckland, Auckland, New Zealand.

His current research interests include medical image analysis, feature selection, graph representation learning, and sparse learning.

Rongyao Hu received the M.S. degree in computer science from Guangxi Normal University, Guilin, China. He is currently pursuing the Ph.D. degree with Massey University Auckland, Auckland, New Zealand.

He has published 25 articles in English, including one ESI Hotspot article and four ESI Highly Cited articles. His research interests include feature extraction, graph representation learning, and image detection.

Mr. Hu has served as a Reviewer for several top journals and international conferences, such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON IMAGE PROCESSING, and *Information Fusion.*

Yujie Mo received the B.S. degree in computer science and technology from Northeastern University, Shenyang, China, in 2020. He is currently pursuing the M.S. degree with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China.

His research interests include graph representation learning and related applications.

Zhao Kang received the Ph.D. degree in computer science from Southern Illinois University, Carbondale, IL, USA, in 2017.

He is currently an Associate Professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China. He has published over 80 research papers in top-tier conferences and journals, including NeurIPS, AAAI, IJCAI, ICDE, CVPR, MM, SIGKDD, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON CYBERNETICS, and IEEE TRANSACTIONS ON IMAGE PROCESSING. His research interests are machine learning, pattern recognition, and deep learning.

Dr. Kang has been an AC/SPC/PC Member at a number of top conferences, such as NeurIPS, AAAI, IJCAI, CVPR, ICCV, ICLR, and MM. He regularly servers as a Reviewer for *Journal of Machine Learning Research*, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and IEEE TRANSACTIONS ON IMAGE PROCESSING.

Liang Peng received the B.S. degree in software from the University of Electronic Science and Technology of China, Chengdu, China, in 2018, and the master's degree from the Intelligent Information Technologies and Applications Laboratory, University of Electronic Science and Technology of China, in 2019, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering.

He worked with NVIDIA, Shanghai, China, in 2017. His current research interests include graph representation learning and medical image analysis.

Yonghua Zhu received the M.S. degree from the School of Computer, Electronics and Information Science, Guangxi University, Nanning, China, in 2019. He is currently pursuing the Ph.D. degree with the School of Computer Science, The University of Auckland, Auckland, New Zealand.

His current research interests include dimensional reduction, clustering, and reasoning with natural language.

Xiaofeng Zhu (Senior Member, IEEE) received the Ph.D. degree in computer science from The University of Queensland, Brisbane, QLD, Australia, in 2013.

He is currently with the University of Electronic Science and Technology of China, Chengdu, China. His current research interests include big data, artificial intelligence, and medical image analysis.