Multi-View Graph Convolutional Networks with Differentiable Node Selection

ZHAOLIANG CHEN, College of Computer and Data Science, Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, China
LELE FU, School of Systems Science and Engineering, Sun Yat-sen University, China
SHUNXIN XIAO and SHIPING WANG, College of Computer and Data Science, Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, China
CLAUDIA PLANT, Faculty of Computer Science, ds:UniVie, University of Vienna, Austria
WENZHONG GUO, College of Computer and Data Science, Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, China

Multi-view data containing complementary and consensus information can facilitate representation learning by exploiting the intact integration of multi-view features. Because most objects in the real world often have underlying connections, organizing multi-view data as heterogeneous graphs is beneficial to extracting latent information among different objects. Due to the powerful capability to gather information of neighborhood nodes, in this article, we apply Graph Convolutional Network (GCN) to cope with heterogeneous graph data originating from multi-view data, which is still under-explored in the field of GCN. In order to improve the quality of network topology and alleviate the interference of noises yielded by graph fusion, some methods undertake sorting operations before the graph convolution procedure. These GCN-based methods generally sort and select the most confident neighborhood nodes for each vertex, such as picking the top-k nodes according to pre-defined confidence values. Nonetheless, this is problematic due to the non-differentiable sorting operators and inflexible graph embedding learning, which may result in blocked gradient computations and undesired performance. To cope with these issues, we propose a joint framework dubbed Multi-view Graph Convolutional Network with Differentiable Node Selection (MGCN-DNS), which is constituted of an adaptive graph fusion layer, a graph learning module, and a differentiable node selection schema. MGCN-DNS accepts multi-channel graph-structural data as inputs and aims to learn more robust graph fusion through a differentiable neural network. The effectiveness of the proposed method is verified by rigorous comparisons with considerable state-of-the-art approaches in terms of multi-view semi-supervised classification tasks, and the experimental results indicate that MGCN-DNS achieves pleasurable performance on several benchmark multi-view datasets.

CCS Concepts: • **Computer systems organization** → **Neural networks**; • **Computing methodologies** → **Supervised learning by classification;**

This work is in part supported by the National Natural Science Foundation of China under Grant U21A20472 and 62276065, and the National Key Research and Development Plan of China under Grant 2021YFB3600503.
Authors' addresses: Z. Chen, S. Xiao, S. Wang, and W. Guo (corresponding author), College of Computer and Data Science, Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China; emails: chenzl23@outlook.com, xiaoshunxin.tj@gmail.com, shipingwangphd@163.com, guowenzhong@fzu.edu.cn; L. Fu, School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou 511400, China; email: lawrencefzu@gmail.com; C. Plant, Faculty of Computer Science, ds:UniVie, University of Vienna, Vienna 1090, Austria; email: claudia.plant@univie.ac.at.

1556-4681/2023/08-ART6 $15.00
https://doi.org/10.1145/3608954

Additional Key Words and Phrases: Multi-view learning, graph convolutional network, semi-supervised classification, differentiable node selection

ACM Reference format:
Zhaoliang Chen, Lele Fu, Shunxin Xiao, Shiping Wang, Claudia Plant, and Wenzhong Guo. 2023. Multi-View Graph Convolutional Networks with Differentiable Node Selection. *ACM Trans. Knowl. Discov. Data.* 18, 1, Article 6 (August 2023), 21 pages.
https://doi.org/10.1145/3608954

1 INTRODUCTION

Multi-view data extensively exist in a great number of practical applications, owing to the fact that most objects in the real world can be described from multifarious perspectives [6, 21, 32, 38]. A large amount of research has verified that multi-view data can lead to encouraging performance compared with single-view data, such as the semi-supervised classification task that only has scarce supervision information. Because objects in the real world often lead to latent connections and can be regarded as nodes, we can construct heterogeneous graphs from multi-view data which depict objects more comprehensively. Accordingly, graph-based learning models can be adapted to multi-view learning tasks. Graph learning is a critical technique for many machine learning tasks like clustering and classification [3, 16, 36, 37]. In particular, **Graph Convolutional Network (GCN)** is a crucial technique of graph-based representation learning and has been universally applied to a multitude of machine learning tasks. GCN-based methods have been well-established in computer vision [19, 40, 43], classification tasks [5, 13, 50], and topological graph structure analyses [1, 33, 35]. In a nutshell, graph convolutions are performed in the non-Euclidean domain instead of extensively employed standard convolutions for the Euclidean domain, which have a powerful ability to integrate the connectivity patterns and feature attributions of graphs in a Fourier space. Actually, GCN is a typical network that aggregates topological neighbors via convolutional layers, so that node and feature representations propagate over the network topology. Attributed to these factors, frameworks based on GCN have achieved noticeable performance for extracting robust numeric representations.

One challenge of the GCN-based multi-view learning is the estimation and fusion of node connections, because these data often have no natural graph structure. Most existing methods estimated graphs from node feature similarities. However, these estimated node relationships sometimes are not accurate and contain noises, attributed to which it is pivotal to develop the graph refining strategy, such as neighborhood node selection and edge denoising. In addition, a traditional GCN-based method only enables single-view data, meaning that further study on frameworks addressing multi-graph-structural data should be conducted. Some existing methods have endeavored to tackle this deficiency with a linear weighted sum of heterogeneous graphs [4, 17, 47], but it may not be ideal because such a strategy cannot extract deeper representation and results may be interfered by new noises encountered. Consequently, a study on multi-channel GCN-based models learning more robust and generalized feature graphs should be further explored.

Node sorting and selection strategies have been utilized in numerous works related to GCN to pick top-k confident nodes, which prompt the sparsity and lessen the interference of noises [12, 26, 34]. In addition, it is helpful for sustained explorations of relationships between every two vertices and augmenting the edges with higher importance, because of which the time consumption declines and the performance is improved. Figure 1 briefly illustrates the node selection procedure, which selects neighborhood nodes with high confidence or importance for each vertex. The confidence or importance values can be measured by the weights of edges or feedback from downstream tasks. Nevertheless, such a strategy is generally non-differentiable, thereby

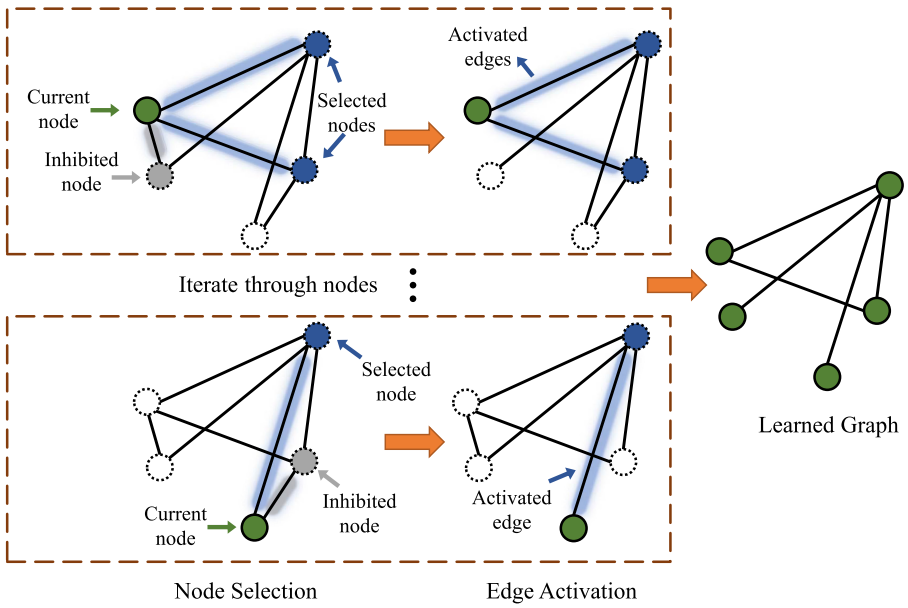

Fig. 1. Node selection for a graph-structural network. For each node (green), only other connected nodes with higher importance or confidence (blue) are selected. Namely, salient edges (blue) connecting to important nodes are activated and other edges (gray) are inhibited. Thus, a sparser graph is learned due to the restraint of some edges.

often leading to ineffective gradient propagation, which is problematic for neural networks. Consequently, it becomes pivotal to delve into a differentiable network allowing gradient descent and back propagation everywhere. Although Grover et al. presented a differentiable operator generating permutation matrix [7], the sorting and selection operations are still performed with arg max operator which is non-differentiable and prohibits the gradient propagation. Therefore, we devote ourselves to establishing a differentiable learning schema that automatically selects confident nodes and maintains the differentiability of the neural network.

In pursuit of solving these mentioned issues, this article proposes a GCN-based framework dubbed **Multi-view Graph Convolutional Network with Differentiable Node Selection** (**MGCN-DNS**) to undertake multi-view semi-supervised classification tasks. The proposed method addresses multi-graph-structural data via a graph fusion layer and extracts numerical representations of node relationships through a graph learning module and a novel differentiable node selection schema. In detail, adjacency matrices from heterogeneous views serve as the inputs for each channel in the graph fusion layer, which explores the underlying complementary information from all channels and outputs the combined adjacency matrix via a two-stage graph fusion process. Next, the intact graph is refined by a graph learning module and a differentiable node selection module for each vertex. To our best knowledge, this work is the first attempt to integrate differentiable node selection schema into the adjacency matrix learning of GCN-based models. Compared with existing non-differentiable node selection strategies, this allows us to build an end-to-end framework where gradients can be propagated to all modules, owing to which the network parameters can be fully optimized. Finally, the refined adjacency matrix is employed by graph convolutions to yield node embeddings. In a nutshell, the main contributions of this article are outlined as follows:

(1) Devise a two-stage graph fusion process that exploits the complementary information from adjacency matrices, where the first-stage fusion enriches the node connections in each view,

and the second-stage fusion adaptively merges these enriched adjacency matrices following the instructions provided by the previous fusion stage.

(2) Develop a differentiable node selection procedure to choose nodes with higher confidence from the merged graph. The differentiable procedure converts the continuous probabilistic permutation matrix to confidence values measuring the importance of distinct nodes, which avoids computationally intractable back propagation caused by non-differentiable sorting and selecting operations.

(3) Construct an end-to-end GCN to deal with multi-channel topological graphs, which extracts neighborhood relationships via graph learning modules and differentiable node selection.

(4) The proposed framework is applied to undertaking the multi-view semi-supervised classification task and achieves pleasurable performance in comparison with classical and state-of-the-art methods.

The rest of this article is organized as follows. Related works including multi-view learning, multi-graph learning, and GCN are discussed in Section 2. We elaborate on the proposed MGCN-DNS and relevant theoretical analysis in Section 3. In Section 4, substantial experiments and comparisons are conducted to show the superiority of the model. Finally, we conclude our work and discuss future directions in Section 5.

2 RELATED WORK

In this section, we discuss some relevant works w.r.t. multi-view learning and GCN. Before that, the primary notations and explanation of mathematical symbols used in this article are clarified in Table 1.

2.1 Multi-View Learning

In real-world practical applications, a considerable number of data exist as heterogeneous forms, leading to a rapidly increasing attention on multi-modality fusion. Consequently, multi-view learning has been extensively studied in recent years. Liu et al. improved the performance of multi-view clustering by considering information from multiple incomplete views [20]. Chen et al. proposed a partial tubal nuclear norm regularized multi-view learning algorithm, which minimized the partial accumulation of the smaller tubal singular values to keep the low-rank property of the self-representation tensor [2]. Han et al. performed multi-view clustering via establishing a sparse membership matrix over heterogeneous views and studied the centroid matrix and its corresponding weight of each view [8]. Wang et al. put forward a parameter-free multi-view subspace clustering model with consensus anchor guidance, which had linear computational complexity [30]. Huang et al. established a framework integrating diversity, sparsity and consensus to flexibly process multi-view data, which projected a linear regression model onto deduce view-specific embedding regularizers and automatically updates weights of various perspectives [11]. Han et al. explored multi-human association on multi-view images taken by various cameras at the same moment [9]. Yan et al. proposed a bipartite graph-based discriminative feature learning algorithm to perform multi-view clustering, which was a unified framework to consider bipartite graph learning and discriminative feature learning [42]. Zhang et al. constructed anchor-based affinity graphs to address the multi-view semi-supervised problem [48].

2.2 Multi-Graph Fusion

Owing to the fact that most multi-view data can be represented as feature attributes and node relationships incorporated in distinct network topologies, multi-graph fusion has become a popular issue. Some of these methods merged graph information from natural graph data. Sadikaj

Table 1. A Summary of Primary Notations in This Article

Notations	Explanations
$\mathbf{X}^{(v)}$	Feature matrix of the vth view.
$\mathbf{A}^{(v)}$	Adjacency matrix of the vth view.
$w^{(i,j)}$	Adaptive weight of the adjacency matrix.
$\mathbf{A}_w^{(v)}$	Complementary graph of the vth view.
$\alpha^{(i)}$	Weight of the ith complementary graph.
$\mathbf{A}_f$	Fused intact adjacency matrix.
$\hat{\mathbf{A}}$	Refined graph via graph learning module.
$\mathbf{S}_1, \mathbf{S}_2$	Weight matrices of graph learning module.
$\mathbf{P}$	Continuous permutation matrix.
Softmax$(\cdot)$	Softmax function.
γ	Hyperparameter in graph learning module.
τ	Temperature parameter in the construction of $\mathbf{P}$.
$\mathcal{I}_i$	Node confidence value.
$\mathbf{C}$	Coefficient matrix for node selection.
θ	Learnable thresholder.
$\mathbf{A}_{select}$	Adjacency matrix after differentiable node selection.
c	Number of classes.
$\sigma(\cdot)$	Optional activation function.
$\mathbf{A}_s$	Adaptively learned adjacency matrix fusion.
$\mathbf{W}^{(l)}$	Trainable weight matrix in graph convolutions.
$\mathbf{H}^{(l)}$	Output of GCN in the lth layer.
$\mathbf{Z}$	Learned graph embedding.
$\mathbf{Y}$	Groundtruth of labeled samples.
$\mathcal{L}$	Loss function.

et al. designed a SpectralMix algorithm which integrated all information from different node attributes, relationships, and graph structures to cope with multi-relational graphs [25]. Tang et al. built a multi-graph-fusion-based spectral clustering method via proposing a hyperspectral band selection algorithm [27]. Xiao et al. proposed a dual fusion-propagation graph neural network to capture the consistency and complementarity information for multi-view clustering [39]. Khan et al. come up with a GCN-based framework for conducting multi-channel classification tasks through multi-graph fusion and salient edge augment [14]. Yao et al. applied multiple GCNs to yield various node embeddings and fused these embeddings via the attention mechanism [45]. A self-supervised framework was constructed by Hassani et al. to learn node and graph level representations simultaneously [10]. Some other works also integrated heterogeneous graphs generated from multi-view data which are more common in real-world applications. For example, a co-training strategy was exploited to explore a unified GCN-based framework, where the graph information embedded in sundry views was trained automatically [17]. Graphs of multiple views were incorporated into a consistent global graph, whose Laplacian matrix was subjected to multiple strongly connected components [24]. Nie et al. designed a parameter-free framework that automatically assigned weights to each view and explored a label indicator matrix by collecting loss functions of all views [23]. Xie et al. constructed a unified tensor space to jointly explore multi-view correlations by local geometrical structures, where a low-rank tensor regularization was imposed to guarantee that all views come to an agreement [41]. Wang et al. learned sparse graphs from heterogeneous views via a deep sparse regularization network to conduct multi-view

semi-supervised classification and clustering [29]. Most of these prior works concentrated on exploring an effective paradigm to merge graphs with linear combinations or attention mechanisms. In this article, we further build a two-stage graph fusion operation for all initial adjacency matrices, which merges robust neighborhood correlations from all views and explores complementary information via trainable weights.

2.3 Graph Convolutional Network

A spectral graph convolution is performed by a signal $x \in \mathbb{R}^m$ and a filter $g_\theta = diag(\theta)$, represented as

$$g_\theta \star x = \mathbf{U} g_\theta \mathbf{U}^\top x, \tag{1}$$

where $\mathbf{U}$ is the matrix of eigenvalues. Kipf et al. [15] conducted the first-order approximation of truncated Chebyshev polynomial to simplify the spectral graph convolution, and applied it to the semi-supervised classification, where data were organized as network topology. In particular, a convolutional layer of GCN is formally defined as

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

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix considering the self-connections, $\mathbf{H}^{(l)}$ is the learned graph embedding in the lth layer, and $[\tilde{\mathbf{D}}]_{ii} = \sum_j [\tilde{\mathbf{A}}]_{ij}$. To simplify notations, in this article, we directly replace the renormalized $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ with $\mathbf{A}$. Numerous studies have paid attention to GCN-based frameworks, exploiting the capability of gathering node attributes to promote the performance of varying machine learning tasks. Zhu et al. proposed a variant of GCN derived via a modified Markov diffusion kernel, which explored the global and local contexts of nodes [51]. Ye et al. established a context-encoding network with GCN to study skeleton topology adaptively [46]. Liu et al. integrated GCN with hidden conditional random fields to reserve the skeleton structure information of people during the classification stage [18]. Wang et al. mined the specific and common embeddings from node features, topological structures, and their combinations simultaneously via a multi-channel GCN, whose adaptive importance weights were trained via the attention mechanism [31].

For the purpose of promoting the sparsity and performance of graph-based models, a population of methods has attempted to select connective nodes with higher confidence for each vertex. Wu et al. chose the top-k largest values of the adjacency matrix to explore more robust node relationships and improve the accuracy of traffic flow estimation [34]. Nodes with low confidence values were randomly selected to build subgraphs for the GCN framework [49]. Hui et al. selected labeled nodes with top-k confidence scores during training iterations [12]. Sun et al. sorted the confidence values of vertices to improve the quality of graphs in GCN-based methods [26]. In light of these prior works, node selection is generally conducted by a sorting operator or top-k operator denoted by arg $topk$. Algorithm 1 illustrates a common node selection process with a top-k operator in most GCN-based frameworks. These methods generally preserved the top-k most important or confident neighborhood nodes of each vertex, according to the weights of edges, which resulted in a sparser adjacency matrix after the node selection operation. Nevertheless, these models face a problematic issue that they are not differentiable everywhere owing to the non-differentiable node sorting and selection procedure, which may result in the gradient vanishing and blocking of gradient propagation during training. For the sake of tackling this problem, in this article, we construct a differentiable learning progress that figures out the confidence score of each node and automatically selects vertices with higher confidence.

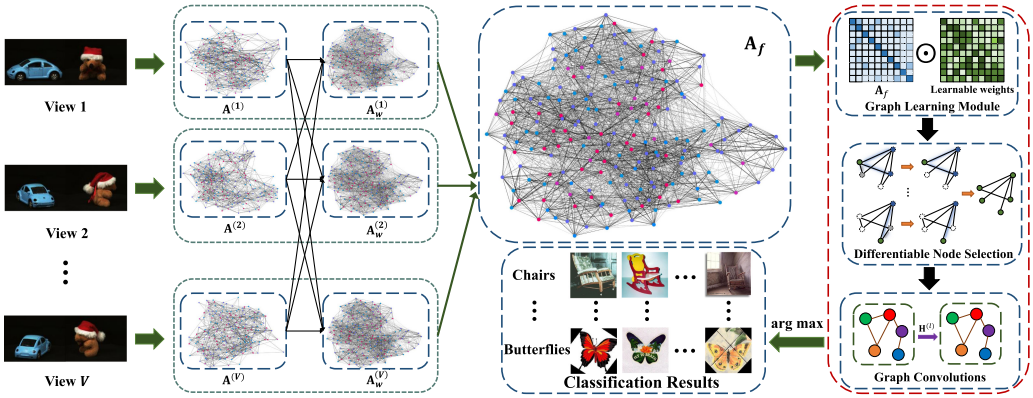

Fig. 2. Structure of the proposed MGCN-DNS, which endeavors to solve multi-view semi-supervised classification problems. First of all, multiple graphs recording node connections are estimated from sample feature similarities. After the two-stage adaptive fusion of all graphs, the learned intact adjacency matrix is dexterously manipulated by the graph learning module and differentiable node selection process, respectively. Eventually, the refined adjacency matrix is exploited in GCNs to produce node embeddings that indicate the classification results.

ALGORITHM 1: A Common Algorithm for Node Selection

Input: Adjacency matrix $\mathbf{A} \in \mathbb{R}^{m \times m}$.
Output: Adjacency matrix $\mathbf{A}_{select}$ after node selection.
Initialize k of top-k ranking;
Initialize $\mathbf{A}_{select}$ as a null matrix;
for $i = 1 \rightarrow m$ **do**
Generate the node connectivity patterns of the i-th node by $\mathbf{a}_i = [a_{i1}, \ldots, a_{im}]$, where $a_{ij} = [\mathbf{A}]_{ij}$;
Find the index of the top-k largest (or the most important) nodes by $idx = \arg topk(\mathbf{a}_i)$;
Conduct $\mathbf{A}_{select}[i, idx] = \mathbf{a}_i[idx]$;
end for
return Selected adjacency matrix $\mathbf{A}_{select}$.

3 MULTI-VIEW GRAPH CONVOLUTIONAL NETWORK WITH DIFFERENTIABLE NODE SELECTION

Multi-view semi-supervised classification is a widely investigated problem in recent years, which conduct classification tasks with very limited labeled samples. It aims to overcome the scarce semi-supervised signals via making full use of information from heterogeneous views. Given multi-view data $\mathcal{X} = \{\mathbf{X}^{(v)}\}_{v=1}^{V}$, where $\mathbf{X}^{(v)} \in \mathbb{R}^{m \times n}$ represents the feature matrix of the vth view with totally V views, the proposed MGCN-DNS is supposed to tackle the multi-view semi-supervised classification problems with renormalized adjacency matrices $\mathcal{A} = \{\mathbf{A}^{(v)}\}_{v=1}^{V}$ generated from $\mathcal{X}$, where $\mathbf{A}^{(v)} \in \mathbb{R}^{m \times m}$. Adjacency matrices from distinct channels exploit complementary information by a two-stage adaptive graph fusion layer, after which MGCN-DNS refines the fused adjacency matrix via a graph learning module and a differentiable node selection procedure. Finally, the refined intact adjacency matrix is applied to conducting graph convolution operations to yield node embeddings. Figure 2 briefly illustrates the structure of the entire framework.

3.1 Multi-Channel Graph Fusion

First of all, the adjacency matrices are initialized via the k-**Nearest Neighbors** (**KNN**) method, which extracts the top-k most similar samples as the neighbors of each node. This procedure can be defined as

$$\mathbf{A}_{\mathcal{S}} = \begin{cases} 1 \ or \ sim_{ij}, \ x_i \in TopK(x_j) \ or \ x_j \in TopK(x_i), \\ 0, \ otherwise, \end{cases} \tag{3}$$

where sim_{ij} is the similarity between samples i and j. Herein, $TopK(x_i)$ is the set of the top-k most similar samples of the node i.

We first perform an adaptive weighted sum of initial adjacency matrices from all views to leverage complementary information from other perspectives. Namely,

$$\mathbf{A}_w^{(v)} = \sum_{i=1}^{V} w^{(v,i)} \mathbf{A}^{(i)}, \tag{4}$$

where $w^{(v,i)}$ is the trainable weight which indicates the importance of the ith view graph to the vth view graph. Graph $\mathbf{A}_w^{(v)}$ of the vth view can be regarded as a complementary graph because it is enriched by information from other views. We also utilize the softmax renormalization defined in Equation (5) for each channel

$$w^{(v,i)} \leftarrow \frac{\exp\left(w^{(v,i)}\right)}{\sum_{i=1}^{V} \exp\left(w^{(v,i)}\right)} \tag{5}$$

to guarantee that $\sum_{i=1}^{V} w^{(v,i)} = 1$ during training. Consequently, a larger $w^{(v,i)}$ trained automatically indicates that the corresponding adjacency matrix in the ith view should make up more for missing information in the vth view. In other words, if $\sum_{v=1}^{V} w^{(v,i)}$ is larger, the graph in the ith view should be more important because it provides more complementary information to all views. Therefore, we compute $\alpha^{(i)}$ measuring the contribution of the ith view to all views via

$$\alpha^{(i)} = \sum_{v=1}^{V} w^{(v,i)}, \tag{6}$$

which is renormalized by

$$\alpha^{(i)} \leftarrow \frac{\alpha^{(i)}}{\sum_{i=1}^{V} \alpha^{(i)}}. \tag{7}$$

Because $w^{(v,i)}$ measures how much additional information is provided by the ith view, an adjacency matrix with a higher value of corresponding α should contribute more to the final fused node relationship information, thereby containing more complementary information. Actually, if the value of α is tiny, it indicates that the adjacency matrix in this view has little complementary information and is redundant for the other views. Thus, we should consider reducing the impact of the corresponding adjacency matrix. In light of the previous analysis, the unique merged adjacency matrix is obtained by considering this factor:

$$\mathbf{A}_f = \sum_{i=1}^{V} \alpha^{(i)} \mathbf{A}_w^{(i)}. \tag{8}$$

3.2 Graph Learning Module

After combining network topologies via the two-stage graph fusion, we construct a graph learning module to further refine the integrated adjacency matrix with the given downstream task. Because GCN assumes that the graph information should be symmetric and bi-directional, the outputs of the learnable layer also ought to hold this assumption. Denoting learnable weight matrices as $\mathbf{S}_1, \mathbf{S}_2 \in \mathbb{R}^{m \times m}$, the graph learning module is formulated as

$$\hat{\mathbf{A}} = \mathbf{A}_f \odot \text{Sigmoid}\left(\gamma \left|\mathbf{S}_1 {\mathbf{S}_2}^T - \mathbf{S}_2 {\mathbf{S}_1}^T\right|\right), \tag{9}$$

where $\odot$ is the Hadamard product and $\mathbf{S}_1, \mathbf{S}_2$ are randomly initialized. Hyperparameter γ controls the saturation rate of the activation function. Thus, the learned $\hat{\mathbf{A}}$ keeps properties that edges of topological graphs should be non-negative and symmetric, which is achieved by the trainable matrix $\text{Sigmoid}(\gamma|\mathbf{S}_1 {\mathbf{S}_2}^T - \mathbf{S}_2 {\mathbf{S}_1}^T|)$ serving as an adaptive shrinkage weight matrix. Herein, $\text{Sigmoid}(\cdot)$ performs the projection to make sure that all entries range in $(0, 1)$, which dexterously avoids excessive values and inhibits unimportant connections in a data-driven way during training. Therefore, an adjacency matrix is refined via learnable weights $\mathbf{S}_1, \mathbf{S}_2$ in the graph learning module. Because the trainable weights $\mathbf{S}_1, \mathbf{S}_2$ are automatically optimized according to the cross entropy loss defined in (20), it approximates the optimal graph tailored for the downstream tasks at the best.

3.3 Differentiable Node Selection

So as to further alleviate the influence of noises and make full use of pivotal neighborhood relationships, the proposed method conducts differentiable node selection in each channel to pick essential nodes and promote the sparsity of graphs, on the basis of the probabilistic permutation matrix $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_m]^T \in \mathbb{R}^{m \times m}$. In particular, $\mathbf{p}_i = [p_{i1}, \ldots, p_{im}]^T$ is the ith row of $\mathbf{P}$, where p_{ij} denotes the chance that the node i is the jth important node among all m nodes. As an example, given $\mathbf{p}_i = [0.1, 0.8, 0.1]$, we can consider the node i as the second most important node because $p_{i2} = 0.8$.

For the purpose of obtaining the node confidence suggesting the importance of a vertex, we first convert the learned $\hat{\mathbf{A}}$ to the probabilistic permutation matrix $\mathbf{P}$. Particularly, the ith row of $\mathbf{P}$ is computed by

$$\mathbf{p}_i(\mathbf{a}_s, \tau) = \text{Softmax}\left[\left((m + 1 - 2i)\mathbf{a}_s - \Delta\mathbf{1}\right)/\tau\right], \tag{10}$$

where $\tau > 0$ is a temperature parameter and $\Delta \in \mathbb{R}^{m \times m}$ is a pairwise difference matrix calculated by

$$[\Delta]_{ij} = \left|[\mathbf{a}_s]_i - [\mathbf{a}_s]_j\right|. \tag{11}$$

Vector $\mathbf{a}_s \in \mathbb{R}^m$ denotes the mean of nonzero entries in each column of the matrix $\hat{\mathbf{A}}$, measuring the average indegree of a vertex. Namely,

$$[\mathbf{a}_s]_j = \frac{\sum_{i=1}^{m} [\hat{\mathbf{A}}]_{ij}}{\delta}, \tag{12}$$

where δ is the number of nonzero entries in each column. Therefore, Equation (10) is differentiable everywhere, which ensures the gradient computation and the back propagation during network training. It was proved in [7] that $\mathbf{P}$ is a unimodal row stochastic matrix where the sum of each row equals one, and the permutation can be obtained by a row-wise arg max operator. Note that entries in $\mathbf{P}$ stand for the chances of permutations, which can promote the flexibility of the model and avoid the negative effect of noises to some extent. Compared with binary permutation matrices or a direct top-k operator on entries of adjacency matrices, a probabilistic continuous permutation matrix enables more flexible optimization of graph-structural data with specific tasks.

Therefore, the permutation matrix $\mathbf{P}$ may provide better robustness and generalization during training iterations.

Next, we introduce a metric to evaluate the confidence of each node with the probabilistic permutation matrix. It is notable that the probability distribution of the node i should be higher in the front of vector $\mathbf{p}_i$, if the node i has a higher chance corresponding to a larger confidence value. Inspired by the **Discounted Cumulative Gain** (**DCG**) that is commonly employed in recommender systems and search algorithms, we calculate the confidence of the node i by

$$\mathcal{I}_i = \sum_{j=1}^{m} \frac{2^{p_{ij}} - 1}{\log_2(j+1)}, \tag{13}$$

and renormalize it with

$$\bar{\mathcal{I}}_i = \frac{\mathcal{I}_i - \mathcal{I}_{min}}{\mathcal{I}_{max} - \mathcal{I}_{min}}, \tag{14}$$

where $\mathcal{I}_{max}$ and $\mathcal{I}_{min}$ are the largest and smallest values of node confidence computed by Equation (13), respectively. As a matter of fact, Equations (13) and (14) transfer the density near each node into a probability value ranging in $[0, 1]$. Theoretically, a higher $\bar{\mathcal{I}}_i$ indicates a higher confidence of the node i, because a larger p_{ij} with smaller i contributes more to the accumulation computations in Equation (13). On the basis of this, the differentiable node selection procedure is performed via a coefficient matrix $\mathbf{C} \in \mathbb{R}^{m \times m}$, where each entry is computed by

$$[\mathbf{C}]_{ij} = \frac{\bar{\mathcal{I}}_i + \bar{\mathcal{I}}_j}{2}. \tag{15}$$

Consequently, we develop a learnable node selection schema according to previous analyses, formulated as

$$\mathbf{C}_{select} = \mathrm{ReLU}\left(\mathbf{C} - \theta\right), \tag{16}$$

$$\mathbf{A}_{select} = \hat{\mathbf{A}} \odot \frac{\mathbf{C}_{select}}{\max\left(\mathbf{C}_{select}\right)}, \tag{17}$$

where $\theta > 0$ serves as a learnable thresholder and $\max\left(\mathbf{C}_{select}\right)$ is the largest value of $\mathbf{C}_{select}$. To ensure that $0 \leq \theta \leq 1$ holds, we employ a sigmoid projection of θ before the computation, that is, $\theta = \mathrm{Sigmoid}\left(\theta\right)$. With Equation (17), the sparsity of $\mathbf{C}_{select}$ is promoted due to the zero entries generated by the ReLU activation. Compared with universally adopted fixed activation functions in neural networks, we formulate it as a trainable one. This makes the differentiable node selection schema learn a refined sparse graph in a data-driven paradigm. Figure 3 summarizes the differentiable node selection procedure with a simple graph. A new refined graph is obtained by the Hadamard product of the original adjacency matrix and a coefficient matrix generated from a permutation matrix. Because Equations (16) and (17) choose confident neighbors for each specific node, they can also be regarded as a differentiable edge activation procedure to reserve the salient edges. As an example in Figure 3, it can be seen that an insignificant edge connecting nodes 2 and 3 is removed by the differentiable node selection procedure. Actually, these differentiable functions implicitly perform the non-differentiable top-k operator during training, where the trainable parameter θ can be regarded as a surrogate of KNN that keeps the largest k values of node connections.

With the refined adjacency matrix, the proposed framework conducts graph convolution of the lth layer by

$$\mathbf{H}^{(l)} = \sigma\left(\mathbf{A}_{select}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}\right). \tag{18}$$

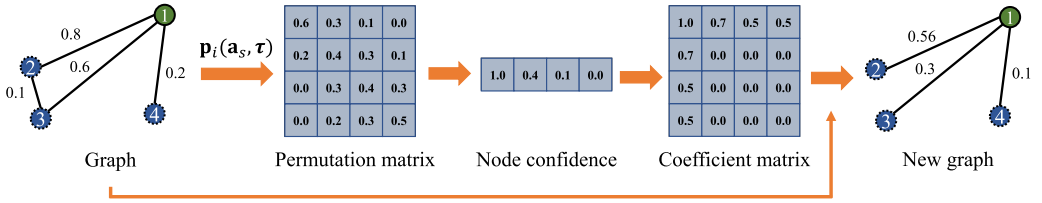

Fig. 3. Differentiable node selection (edge activation) procedure. In this example, node 2 only selects node 1 as the most important neighborhood node, resulting in the inhibition of the edge between node 2 and node 3.

As a straightforward example, the node embedding $\mathbf{Z}$ of a 2-layer GCN is computed by

$$\mathbf{Z} = \text{softmax}\left(\mathbf{A}_{select}\sigma\left(\mathbf{A}_{select}\mathbf{H}\mathbf{W}^{(1)}\right)\mathbf{W}^{(2)}\right), \tag{19}$$

where $[\mathbf{Z}]_{ij}$ denotes the probability that the node i belongs to the jth class. In terms of a semi-supervised classification task, the loss is measured by the cross-entropy function:

$$\mathcal{L} = -\sum_{i\in\Omega}\sum_{j=1}^{c}\mathbf{Y}_{ij}\ln\mathbf{Z}_{ij}, \tag{20}$$

where $\mathbf{Y} \in \mathbb{R}^{m\times c}$ is the ground truth of the dataset. The loss values are calculated through the samples from the partially labeled sample set Ω accounting for a small portion of the entire dataset.

ALGORITHM 2: Multi-View Graph Convolutional Network with Differentiable Node Selection (MGCN-DNS)

Input: Multi-view data $\mathcal{X} = \{\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(V)}\}$ and partially labeled matrix $\mathbf{Y} \in \mathbb{R}^{m\times c}$.
Output: Node embedding $\mathbf{Z}$.
Initialize trainable weights $\{w^{(v,i)}\}_{v,i=1}^{V}$;
Initialize hyperparameter γ and temperature parameter τ;
Estimate adjacency matrices $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(V)}$ via KNN;
Initialize learnable matrices $\mathbf{S}_1, \mathbf{S}_2$ and thresholder θ;
Initialize learnable weight matrices of GCN;
while not convergent **do**
  Conduct learnable multi-channel graph fusion with Equations (4), (5), (6), (7) and (8);
  Compute the refined $\hat{\mathbf{A}}$ with Equation (9);
  Generate probabilistic permutation matrix $\mathbf{P}$ with Equations (10) and (11);
  Obtain the normalized confidence vector $\tilde{I}$ with Equations (13) and (14);
  Conduct differentiable node selection via $\mathbf{C}_{select}$, according to Equations (16) and (17);
  Calculate $\mathbf{Z} = \mathbf{H}^{(L)}$ of GCN with Equation (18);
  Evaluate the loss of the framework with Equation (20);
  Update learnable parameters $\{\mathbf{S}_1\}_{v=1}^{V}$, $\{\mathbf{S}_2\}_{v=1}^{V}$, $\{\theta\}_{v=1}^{V}$, $\{\mathbf{W}^{(l)}\}_{l=1}^{L}$ and $\{w^{(i,j)}\}_{i,j=1}^{V}$ with back propagation;
end while
return Node embedding $\mathbf{Z}$.

3.4 Computational Complexity

Algorithm 2 provides an elaborate description of the proposed MGCN-DNS. In general, the forward propagation of MGCN-DNS framework can be divided into the following steps: adaptive graph

fusion, graph learning, differentiable node selection, and graph convolutions. The graph fusion process consumes $O(mn)$ and the graph learning module costs $O(m^2 + m^3)$. The computational complexity of sorting operator and node confidence computations is $O(m^2)$. Although the classical sorting algorithm only consumes an overall complexity of $O(m \log m)$, the proposed node selection calculation can run efficiently on GPU hardware parallelly. In addition, given d hidden units and $d \ll n$, the computational consumption of graph convolution is $O(mnd)$. Consequently, the upper bound on time consumption of forward calculations in an iteration is $O(m(m^2 + nd + m + n))$.

4 EXPERIMENTAL RESULTS

In this section, the proposed MGCN-DNS is evaluated by substantial experiments to verify its effectiveness. Several benchmark datasets and state-of-the-art methods are selected to assess the performance of MGCN-DNS. The framework is implemented with Pytorch platform and run on the computer with AMD R9-5900X CPU, RTX 3060 GPU and 32G RAM.

4.1 Dataset Descriptions

Eight publicly available multi-view datasets are selected for performance comparison: Caltech-20, Caltech-all,[1] BBCnews,[2] BBCsports,[3] Citeseer,[4] MNIST-10K,[5] NUS-WIDE,[6] and Youtube,[7] which include multiple descriptions of images or documents. The features of each view in these datasets are extracted by some well-known feature extraction algorithms. Here, we provide a more detailed introduction to these datasets:

(1) **Caltech-20** and **Caltech-all:** Caltech101 is a well-known object recognition dataset including 101 classes of images. Six multifarious features are extracted: 48-dimensional Gabor, 40-dimensional wavelet moments, 254-dimensional CENTRIST, 1,984-dimensional histogram of oriented gradients, 512-dimensional GIST, and 928-dimensional LBP features. We follow prior work [24] where 20 classes are selected to generate the Caltech-20 subset.
(2) **BBCnews:** It is a collection of news reports which covers politics, entertainment, business, sport, and technology fields. The dataset is constituted of four views via splitting documents into four related segments.
(3) **BBCsports:** Different from BBCnews, it is a dataset consisting of diverse areas from BBC sports websites, including football, athletics, cricket, rugby, and tennis news, which are illustrated from two distinct views.
(4) **Citeseer:** This is a dataset for research papers, where papers can be divided into six classes. Citeseer has two feature views, including 3,703-dimensional content features and 3,312-dimensional citation features.
(5) **MNIST-10K:** It is a well-known dataset of handwritten digits, where three types of features are extracted: 30-dimensional IsoProjection, 9-dimensional linear discriminant analysis features, and 9-dimensional neighborhood preserving embedding features.
(6) **NUS-WIDE:** This is a popular dataset for object recognition tasks. We randomly choose eight classes and six feature sets: 64-dimensional color histogram, 144-dimensional color correlogram, 73-dimensional edge direction histogram, 128-dimensional wavelet texture,

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html
[2] http://mlg.ucd.ie/datasets/segment.html
[3] http://mlg.ucd.ie/datasets/bbc.html
[4] https://lig-membres.imag.fr/grimal/data.html
[5] http://yann.lecun.com/exdb/mnist/
[6] https://lms.comp.nus.edu.sg/wp-content/uploads/2019/research/nuswide/NUS-WIDE.html
[7] http://archive.ics.uci.edu/ml/machine-learning-databases/00269/

Table 2. A Brief Statistics of All Test Multi-View Datasets

Datasets	# Samples	# Views	# Features	# Classes	Data Types
Caltech-20	1,474	6	48/40/254/1,984/512/928	7	Object images
Caltech-all	9,144	6	48/40/254/1,984/512/928	20	Object images
BBCnews	685	4	4,659/4,633/4,665/4,684	5	Documents
BBCsports	544	2	3,183/3,203	5	Documents
Citeseer	3,312	2	3,703/3,312	6	Documents
MNIST-10K	10,000	3	30/9/9	10	Digit images
NUS-WIDE	1,600	6	64/144/73/128/225/500	8	Web images
Youtube	2,000	6	2,000/64/1,024/512/64/647	10	Video data

225-dimensional block-wise color moments, and 500-dimensional bag of words from SIFT descriptors.

(7) **Youtube:** This dataset contains various audio and visual features on video data, including 2,000-dimensional mfcc, 64-dimensional volume stream, 1,024-dimensional spectrogram stream, 512-dimensional cuboids histogram, 64-dimensional hist motion estimate, and 647-dimensional HOG features.

A summary of all datasets is presented in Table 2, including statistics of numbers of samples, views, classes, features, and data types.

4.2 Compared Methods

So as to validate the effectiveness of the proposed MGCN-DNS, we compare the performance of semi-supervised classification tasks with several traditional and state-of-the-art methods, including KNN, SVM, AMGL [23], MVAR [28], MLAN [22], AWDR [44], HLR-$\mathbf{M}^2$VS [41], ELR-MVSC [11], DSRL [29], GCN [15], Co-GCN [17], and SSGCN [51]. For the sake of reproducibility and providing a fair test bed, the configurations are set empirically during experiments or defined by default settings as original papers.

It is noteworthy that there are only three GCN-based baselines (GCN, Co-GCN, and SSGCN), attributed to the fact that limited work has focused on GCN conducting downstream classification tasks with multi-view data. Because GCN and SSGCN can only manipulate single-view data, we adopt average weighted adjacency matrices as inputs. Nevertheless, most selected approaches are graph-based models so that we are able to conduct a convincing comparison.

4.3 Evaluation Metrics

The performance of each individual algorithm is evaluated via classification accuracy, that is, the percentage of samples that are correctly classified. Following some prior works [11, 17, 41], we conduct semi-supervised classification experiments by splitting data into the training set and testing set, where the number of training samples is smaller than the testing samples. We terminate all gradient-based methods including the proposed MGCN-DNS when they reach the maximum iterations or their losses converge. All approaches are executed five times with randomly selected labeled samples at each time. Accordingly, we record the average classification accuracy and its standard deviation in all comparative experiments.

4.4 Experimental Results

4.4.1 Performance Comparison. First of all, we perform a substantial comparison of MGCN-DNS and other classical and state-of-the-art methods in terms of semi-supervised classification, measured by classification accuracy. With regard to experimental settings of the proposed

Table 3. Classification Accuracy (Mean% and Standard Deviation%) of All Compared Methods with 1% and 10% Labeled Samples, where the Best Performance is Highlighted in Red and the Second-Best Result is Highlighted in Blue

Ratios	Methods \ Datasets	Caltech-20	Caltech-all	BBCnews	BBCsports	Citeseer	MNIST-10K	NUS-WIDE	Youtube
1%	KNN	51.2 (0.4)	19.5 (1.9)	26.9 (9.7)	28.1 (9.6)	24.3 (5.5)	74.8 (1.0)	23.6 (3.1)	7.75 (1.0)
	SVM	53.9 (1.9)	20.3 (2.2)	32.2 (9.3)	29.5 (8.0)	38.8 (5.2)	77.3 (2.6)	26.3 (1.6)	21.0 (6.1)
	AMGL [23]	27.2 (2.7)	11.2 (3.1)	32.5 (3.4)	35.1 (4.3)	-	85.9 (5.2)	29.9 (4.3)	5.31 (0.9)
	MVAR [28]	45.8 (9.6)	23.0 (0.2)	28.1 (6.5)	35.5 (5.9)	20.3 (5.4)	82.3 (0.1)	15.3 (2.7)	4.72 (1.3)
	MLAN [22]	21.2 (2.4)	11.5 (2.2)	29.6 (8.1)	41.0 (7.9)	24.5 (4.2)	85.6 (3.5)	23.0 (9.1)	2.50 (0.2)
	AWDR [44]	47.7 (6.5)	16.5 (0.4)	32.2 (9.9)	20.0 (5.1)	41.0 (2.1)	66.8 (2.3)	22.3 (4.0)	31.6 (8.5)
	HLR-M^{2}VS [41]	60.7 (9.3)	-	50.0 (9.3)	40.9 (9.1)	26.8 (3.4)	-	18.8 (3.6)	8.03 (1.1)
	ERL-MVSC [11]	57.2 (1.9)	21.5 (0.4)	57.7 (4.0)	66.7 (7.5)	48.0 (4.8)	83.4 (0.3)	22.0 (2.2)	26.9 (2.1)
	DSRL [29]	65.5 (2.6)	22.5 (0.6)	59.4 (9.7)	66.5 (9.5)	44.4 (7.4)	72.8 (7.3)	31.4 (2.9)	26.7 (1.7)
	GCN fusion [15]	66.0 (1.2)	24.4 (0.7)	72.9 (3.0)	81.2 (1.1)	50.2 (4.3)	81.8 (1.2)	**34.2 (2.4)**	29.5 (2.5)
	Co-GCN [17]	**70.3 (1.2)**	25.1 (0.5)	73.0 (1.0)	**81.4 (9.1)**	**51.3 (2.4)**	83.3 (0.9)	32.0 (2.3)	**31.2 (3.1)**
	SSGCN [51]	66.4 (2.4)	**25.8 (0.8)**	**74.9 (3.3)**	80.4 (2.1)	49.8 (3.1)	**83.9 (0.7)**	33.9 (1.3)	28.9 (4.3)
	MGCN-DNS	**69.3 (1.3)**	**26.2 (1.4)**	**77.8 (5.4)**	**83.5 (9.6)**	**52.4 (2.9)**	**88.5 (0.8)**	**44.2 (3.9)**	**33.4 (6.6)**
10%	KNN	67.0 (0.4)	29.6 (0.3)	38.3 (9.6)	38.3 (9.5)	26.9 (2.3)	89.8 (0.4)	32.7 (2.9)	36.8 (1.6)
	SVM	71.1 (1.0)	35.3 (0.4)	72.9 (7.0)	70.9 (9.1)	**65.1 (0.5)**	88.6 (0.5)	43.1 (1.4)	45.6 (1.0)
	AMGL [23]	45.0 (3.1)	24.4 (0.2)	52.3 (5.5)	55.6 (1.4)	-	88.5 (0.2)	41.9 (0.2)	44.2 (0.8)
	MVAR [28]	68.9 (4.6)	43.5 (0.3)	75.3 (5.5)	61.9 (1.2)	61.9 (0.8)	85.3 (0.8)	33.3 (2.6)	37.0 (1.3)
	MLAN [22]	45.9 (1.7)	31.2 (0.5)	74.1 (0.9)	62.6 (2.2)	62.1 (0.8)	88.6 (0.3)	27.0 (1.4)	36.4 (1.0)
	AWDR [44]	49.2 (1.7)	45.0 (1.3)	85.7 (1.4)	81.3 (3.3)	64.2 (0.6)	78.1 (0.3)	40.6 (1.9)	36.1 (2.8)
	HLR-M^{2}VS [41]	80.5 (1.8)	-	78.1 (2.8)	84.6 (0.4)	56.2 (0.9)	-	35.3 (1.7)	51.2 (4.1)
	ERL-MVSC [11]	83.2 (1.6)	46.1 (0.3)	85.9 (2.2)	90.3 (1.9)	57.8 (0.9)	90.9 (0.3)	**44.4 (2.1)**	57.4 (1.8)
	DSRL [29]	**84.1 (2.0)**	**51.1 (0.2)**	88.5 (1.2)	91.6 (5.4)	63.9 (2.7)	89.3 (0.4)	44.3 (2.4)	48.0 (1.4)
	GCN fusion [15]	79.9 (0.3)	44.2 (0.3)	69.6 (1.8)	87.0 (2.0)	64.1 (0.6)	89.3 (0.5)	42.9 (2.1)	57.8 (1.4)
	Co-GCN [17]	78.2 (0.8)	46.2 (0.6)	81.9 (1.5)	84.8 (1.4)	63.5 (0.4)	**91.2 (0.4)**	43.5 (0.4)	**58.0 (0.5)**
	SSGCN [51]	82.5 (0.7)	47.1 (0.1)	**90.7 (0.5)**	**94.1 (0.1)**	63.1 (0.4)	89.9 (0.1)	41.2 (0.4)	57.1 (1.1)
	MGCN-DNS	**84.3 (1.2)**	**53.3 (0.3)**	**93.6 (0.6)**	**97.4 (0.6)**	**66.4 (0.5)**	**92.2 (0.2)**	**69.2 (3.1)**	**60.5 (1.3)**

We cannot obtain results of HLR-M^{2}VS and AMGL on some datasets due to its high computational complexity or SVD computation error, and mark the results with "-".

MGCN-DNS, the learning rate is fixed as 0.1 and a 2-layer graph convolution structure is applied. The updates of all trainable parameters are conducted by the Adam optimizer. The experimental results of semi-supervised classification are reported in Table 3, recording the performance when 1% and 10% data are randomly labeled, respectively. The table reveals that the proposed MGCN-DNS achieves encouraging performance with varying ratios of labeled data, leading in all compared methods on almost all datasets. Overall, the performance leading of GCN-based models is more remarkable when only 1% samples are labeled, and the proposed MGCN-DNS gains competitive classification accuracy in this case. Figure 4 further provides the classification accuracy of MGCN-DNS with a larger proportion of labeled samples. It points out that the performance leading is more considerable with a small number of labeled data (e.g., 5% known labels), and the accuracy gaps among multifarious methods narrow as the ratio of labeled instances rises. These observations indicate that MGCN-DNS succeeds in exploiting the properties of graph-structural data with extremely limited supervised information. It is noted that MGCN-DNS also obtains pleasurable results compared with other state-of-the-art GCN-based frameworks, which verifies that the proposed model promotes the performance of GCN in terms of multi-view semi-supervised classification.

4.4.2 Ablation Study. We also examine the effectiveness of the graph learning module and the differentiable node selection operation by conducting ablation experiments on all datasets, as demonstrated in Table 4. The experimental results indicate that either of two modules succeeds in improving the performance of the entire framework, and the accuracy improvement is more

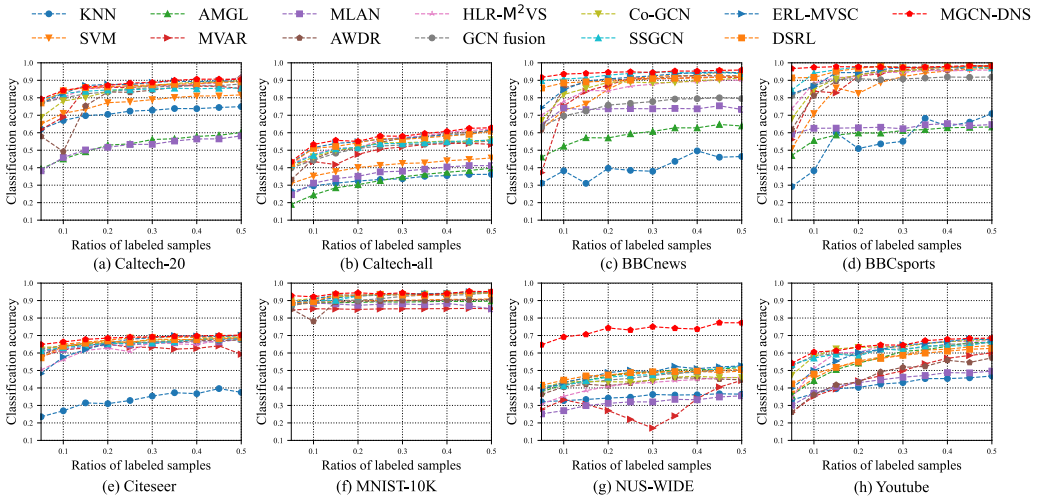

Fig. 4. The varied performance (classification accuracy) of all compared methods as the ratio of labeled data ranges in $\{0.05, 0.10, \ldots, 0.50\}$ on all test datasets.

Table 4. Ablation Study (Mean Classification Accuracy (std%)) of MGCN-DNS on All Datasets

Datasets/Modules	GLM	DNS	Accuracy (std%)	Datasets/Modules	GLM	DNS	Accuracy (std%)
Caltech-20			82.2 (1.2)	Caltech-all			44.2 (0.3)
	✓		83.9 (1.2)		✓		46.2 (0.4)
		✓	82.8 (1.8)			✓	45.7 (0.5)
	✓	✓	**84.3 (1.2)**		✓	✓	**48.6 (0.3)**
Citeseer			64.2 (0.3)	MNIST-10K			90.3 (0.5)
	✓		65.4 (0.5)		✓		90.5 (0.8)
		✓	65.2 (0.2)			✓	91.1 (0.3)
	✓	✓	**66.4 (0.5)**		✓	✓	**92.2 (0.2)**
BBCnews			90.8 (1.2)	BBCsports			95.6 (0.5)
	✓		92.9 (1.0)		✓		96.8 (0.6)
		✓	93.2 (0.5)			✓	96.2 (0.9)
	✓	✓	**93.6 (0.6)**		✓	✓	**97.4 (0.6)**
NUS-WIDE			61.9 (3.9)	Youtube			57.5 (0.9)
	✓		65.3 (2.4)		✓		59.5 (0.9)
		✓	66.1 (2.9)			✓	59.0 (0.3)
	✓	✓	**69.2 (3.1)**		✓	✓	**60.5 (1.3)**

GLM: Graph Learning Module. DNS: Differentiable Node Selection.

favorable when both operations are conducted. These observations validate that MGCN-DNS framework executes sustained explorations of node representation through the graph learning module and the differentiable node selection process, which facilitates improving the accuracy of downstream classification tasks.

In addition, we visualize partial learned adjacency matrices $\mathbf{A}_f$, $\hat{\mathbf{A}}$, and $\mathbf{A}_{select}$ in the network, as exhibited in Figure 5. These figures suggest that $\hat{\mathbf{A}}$ learned by the graph learning module is distinct from the automatically fused adjacency matrix $\mathbf{A}_f$, which inhibits some node relationships owing to the adaptive shrinkage weight matrix (e.g., brown boxes in Figure 5). Nevertheless, most distinctive connections are retained, and the refined adjacency matrix is still symmetric. Due to the

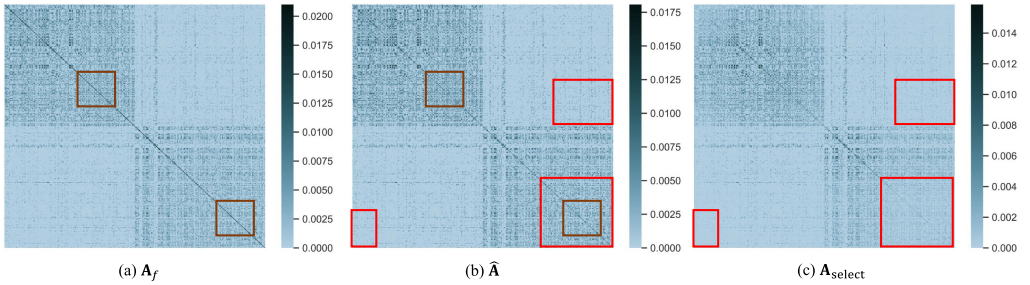

Fig. 5. Visualization of partial adjacency matrices $\mathbf{A}_f$, $\hat{\mathbf{A}}$, and $\mathbf{A}_{select}$ on NUS-WIDE dataset.

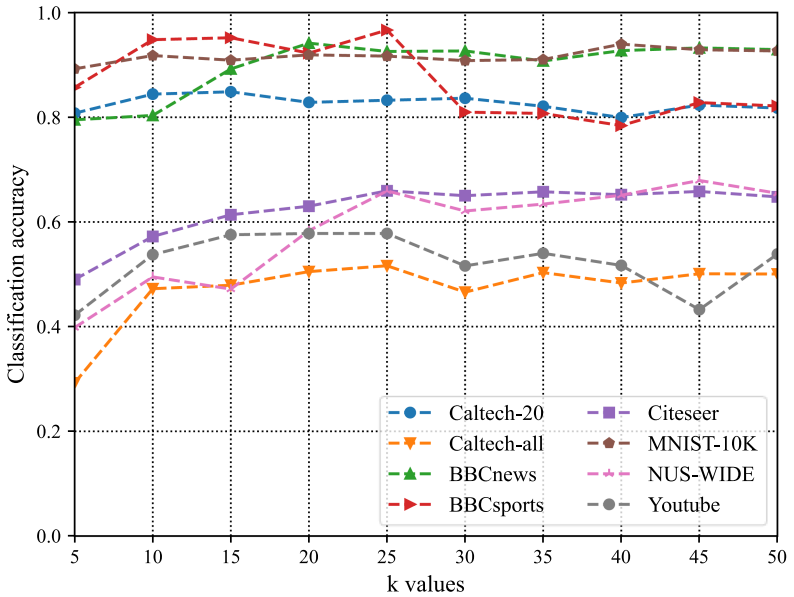

Fig. 6. Parameter sensitivity (Accuracy) w.r.t. neighbor numbers k on all test datasets.

proposed differentiable node selection operator, most salient edges are highlighted and some connective information is not selected in $\mathbf{A}_{select}$, as shown in the red boxes in Figure 5. Consequently, $\mathbf{A}_{select}$ is sparser than previous adjacency matrices. In summary, the ablation study validates that the learned sparse $\mathbf{A}_{select}$ is beneficial to the information propagation over the graph and the performance improvement of GCN.

4.4.3 Parameter Sensitivity. First, we analyze the effect of neighbor numbers k when constructing multi-view graphs, shown in Figure 6. The experimental analysis points out that the performance of MGCN-DNS is sensitive to the initialization of adjacency matrices, which is tightly relevant to the neighbor number k. Overall, a small k (e.g., $k = 5$) often results in poor classification performance, and the accuracy rises as the number of neighbors increases. Nonetheless, on some datasets like BBCsports and Youtube, constructing node relationships with excessive neighbors lead to significant performance decline. This may be due to the fact that too many neighbors may introduce unexpected noises. In conclusion, the optimal values of k for constructing adjacency matrices are quite different on various datasets. Because k is directly related to the performance

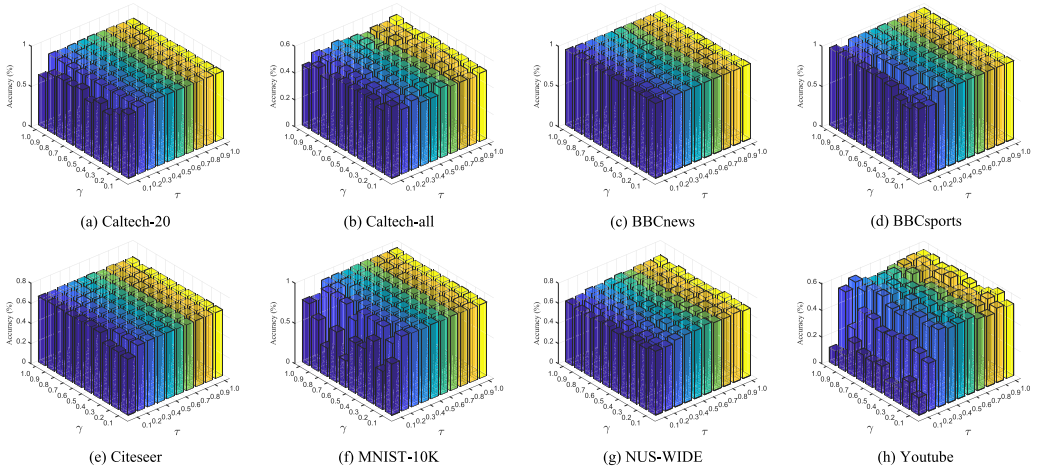

Fig. 7. Parameter sensitivity (Accuracy) w.r.t. hyperparameters γ and τ on all test datasets.

of graph-based models, we ought to set a suitable k for each dataset when establishing topological networks.

Furthermore, Figure 7 explores the effect of hyperparameters γ and τ on selected datasets. From this figure, we can draw the following conclusions. It can be observed that the selection of these two hyperparameters has a noticeable influence on classification performance. In general, small values of γ and τ often lead to poor accuracy on most datasets, which is especially significant for τ. On all datasets except BBCnews, MGCN-DNS behaves unsatisfactorily when $\tau < 0.3$. The reason for the unfavorable performance may be that the permutation matrix $\mathbf{P}$ becomes an exact one rather than an approximate and continuous one, when a small τ is adopted. In most cases, a probabilistic and continuous permutation matrix is helpful to the dexterous optimization of the model. In general, the classification accuracy is robust to the change of hyperparameters when γ and τ are large enough on most datasets.

4.4.4 Visualization of Weights. We visualize the learnable weights $\{w^{(i,j)}\}_{i,j=1}^{V}$ and $\{\alpha^{(i)}\}_{i=1}^{V}$ in Figure 8. It can be validated that KNN graphs from some views are essential for providing complementary information, attributed to which these views are more important. Accordingly, MGCN-DNS adaptively assigns higher α values to these views for the sake of fully leveraging complementary information. The model also assigns tiny values to some views, which means that these views have little complementary information. Besides, a direct connection of initial KNN may be problematic because varying types of features in multi-view data have assorted meanings. This may result in the dissonance of information from different views. Thus, we develop a two-stage fusion process which generates the combined $\mathbf{A}_f$ via a linear weighted accumulation on the basis of learned $\{\alpha^{(i)}\}_{i=1}^{V}$.

4.4.5 Training Details. Furthermore, we analyze the convergence of the proposed framework, and look into the classification accuracy of train sets and test sets during the training procedure, recorded in Figure 9. The experimental results demonstrate that the loss values of MGCN-DNS plunge and converge rapidly, swelling to the lowest point within 200 iterations, suggesting the high efficiency of the framework. The classification accuracy of the train set soars and stabilizes at a peak during training. Although the predictive accuracy of unlabeled data shoots up as training continues, it reaches a lower plateau compared with training accuracy, and begins to fluctuate or

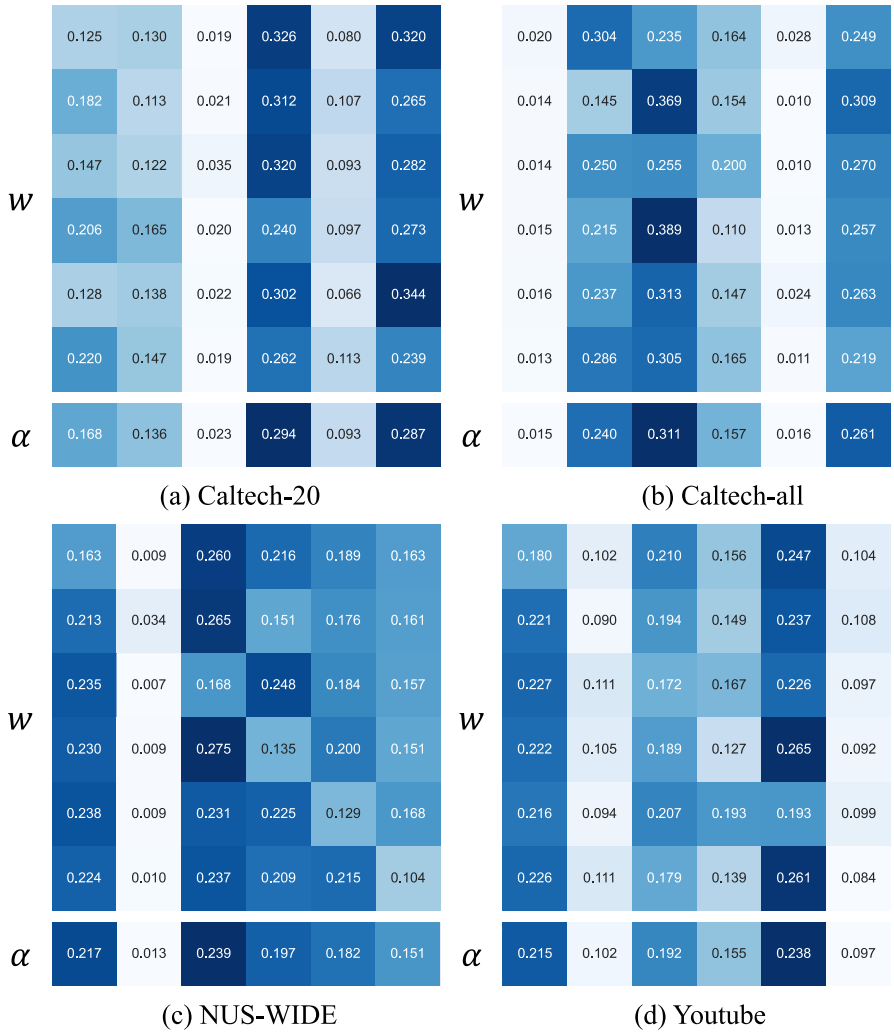

Fig. 8. Visualization of trainable weights w and α on Caltech-20, Caltech-all, NUS-WIDE, and youtube datasets.

even dwindle when the training is not finished, indicating that the phenomenon of overfitting may take place with an excessive number of iterations.

5 CONCLUSION

In this work, we proposed a multi-channel GCN-based model with a differentiable node selection schema. The proposed method applied multi-graph-structural data for initialization and aimed to learn topological information with better robustness and generalization. In each channel, different initialized graphs were integrated by a two-stage adaptive graph fusion procedure. The fused graph was refined by a graph learning module and the differentiable operation of node selection, which explored graph embeddings with higher quality to promote the performance of graph convolutions. Eventually, the proposed framework was applied to carrying out multi-view semi-supervised

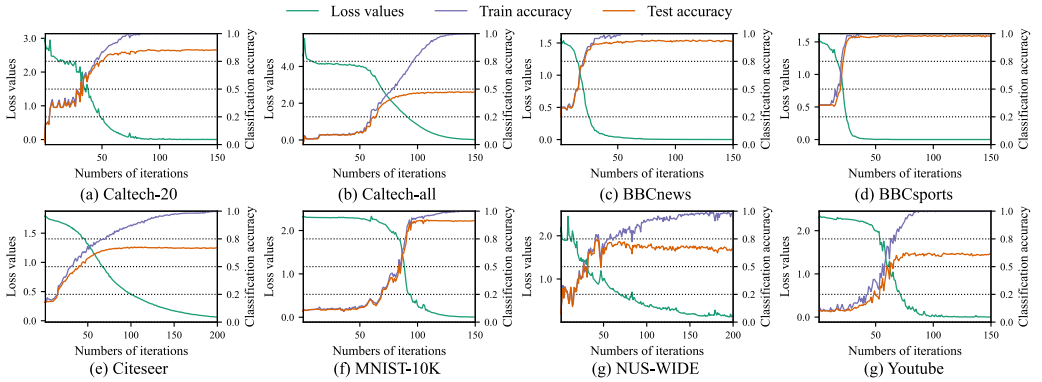

Fig. 9. Curves of loss values (green), train accuracy (purple), and test accuracy (orange) with 10% labeled samples on all datasets.

classification tasks. Comprehensive experimental results validated the effectiveness of MGCN-DNS and revealed that it gained encouraging performance improvement.

In the future, we will devote ourselves to further study of multi-channel GCN-based models and exploit insightful models to conduct sustained explorations of graph embeddings with better robustness and explainable semantic information. A more dexterous strategy to fuse graphs from heterogeneous sources is also our next direction.

REFERENCES

[1] Ling Chen, Xing Tang, Weiqi Chen, Yuntao Qian, Yansheng Li, and Yongjun Zhang. 2021. DACHA: A dual graph convolution based temporal knowledge graph representation learning method using historical relation. *ACM Transactions on Knowledge Discovery from Data* 16, 3 (2021), 1–18.

[2] Yongyong Chen, Shuqin Wang, Chong Peng, Guangming Lu, and Yicong Zhou. 2021. Partial tubal nuclear norm regularized multi-view learning. In *Proceedings of the 29th ACM International Conference on Multimedia*. 1341–1349.

[3] Yan-Hui Chen, Ling Huang, Chang-Dong Wang, and Jian-Huang Lai. 2022. Hybrid-order gated graph neural network for session-based recommendation. *IEEE Transactions on Industrial Informatics* 18, 3 (2022), 1458–1467.

[4] Zhaoliang Chen, Lele Fu, Jie Yao, Wenzhong Guo, Claudia Plant, and Shiping Wang. 2023. Learnable graph convolutional network and feature fusion for multi-view learning. *Information Fusion* 95 (2023), 109–119.

[5] Yanni Dong, Quanwei Liu, Bo Du, and Liangpei Zhang. 2022. Weighted feature fusion of convolutional neural network and graph attention network for hyperspectral image classification. *IEEE Transactions on Image Processing* 31 (2022), 1559–1572.

[6] Lele Fu, Zhaoliang Chen, Yongyong Chen, and Shiping Wang. 2022. Unified low-rank tensor learning and spectral embedding for multi-view subspace clustering. *IEEE Transactions on Multimedia* (2022). DOI: https://doi.org/10.1109/TMM.2022.3185886

[7] Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. 2019. Stochastic optimization of sorting networks via continuous relaxations. In *Proceedings of the 7th International Conference on Learning Representations*.

[8] Junwei Han, Jinglin Xu, Feiping Nie, and Xuelong Li. 2022. Multi-view k-means clustering with adaptive sparse memberships and weight allocation. *IEEE Transactions on Knowledge and Data Engineering* 34, 2 (2022), 816–827.

[9] Ruize Han, Yun Wang, Haomin Yan, Wei Feng, and Song Wang. 2022. Multi-view multi-human association with deep assignment network. *IEEE Transactions on Image Processing* 31 (2022), 1830–1840.

[10] Kaveh Hassani and Amir Hosein Khas Ahmadi. 2020. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning*, Vol. 119. 4116–4126.

[11] Aiping Huang, Zheng Wang, Yannan Zheng, Tiesong Zhao, and Chia-Wen Lin. 2021. Embedding regularizer learning for multi-view semi-supervised classification. *IEEE Transactions on Image Processing* 30 (2021), 6997–7011.

[12] Binyuan Hui, Pengfei Zhu, and Qinghua Hu. 2020. Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, Vol. 34. 4215–4222.

[13] Ziyu Jia, Youfang Lin, Jing Wang, Ronghao Zhou, Xiaojun Ning, Yuanlai He, and Yaoshuai Zhao. 2020. Graphsleepnet: Adaptive spatial-temporal graph convolutional networks for sleep stage classification. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*. 1324–1330.

[14] Muhammad Raza Khan and Joshua E. Blumenstock. 2019. Multi-GCN: Graph convolutional networks for multi-view networks, with applications to global poverty. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, Vol. 33. 606–613.

[15] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*.

[16] Juan-Hui Li, Ling Huang, Chang-Dong Wang, Dong Huang, Jian-Huang Lai, and Pei Chen. 2021. Attributed network embedding with micro-meso structure. *ACM Transactions on Knowledge Discovery from Data* 15, 4 (2021), 72:1–72:26.

[17] Shu Li, Wen-Tao Li, and Wei Wang. 2020. Co-GCN for multi-view semi-supervised learning. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. 4691–4698.

[18] Kai Liu, Lei Gao, Naimul Mefraz Khan, Lin Qi, and Ling Guan. 2021. A multi-stream graph convolutional networks-hidden conditional random field model for skeleton-based action recognition. *IEEE Transactions on Multimedia* 23 (2021), 64–76.

[19] Xiyao Liu, Zhong Ji, Yanwei Pang, Jungong Han, and Xuelong Li. 2021. DGIG-Net: Dynamic graph-in-graph networks for few-shot human-object interaction. *IEEE Transactions on Cybernetics* 52, 8 (2021), 7852–7864.

[20] Xinwang Liu, Miaomiao Li, Chang Tang, Jingyuan Xia, Jian Xiong, Li Liu, Marius Kloft, and En Zhu. 2021. Efficient and effective regularized incomplete multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 8 (2021), 2634–2646.

[21] Zhoumin Lu, Feiping Nie, Rong Wang, and Xuelong Li. 2022. A differentiable perspective for multi-view spectral clustering with flexible extension. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 45, 6 (2022), 7087–7098. DOI: https://doi.org/10.1109/TPAMI.2022.3224978

[22] Feiping Nie, Guohao Cai, and Xuelong Li. 2017. Multi-view clustering and semi-supervised classification with adaptive neighbours. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 2408–2414.

[23] Feiping Nie, Jing Li, and Xuelong Li. 2016. Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 1881–1887.

[24] Feiping Nie, Jing Li, and Xuelong Li. 2017. Self-weighted multiview clustering with multiple graphs. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2564–2570.

[25] Ylli Sadikaj, Yllka Velaj, Sahar Behzadi, and Claudia Plant. 2021. Spectral clustering of attributed multi-relational graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1431–1440.

[26] Ke Sun, Zhouchen Lin, and Zhanxing Zhu. 2020. Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. 5892–5899.

[27] Chang Tang, Xinwang Liu, En Zhu, Lizhe Wang, and Albert Y. Zomaya. 2021. Hyperspectral band selection via spatial-spectral weighted region-wise multiple graph fusion-based spectral clustering. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*. 3038–3044.

[28] Hong Tao, Chenping Hou, Feiping Nie, Jubo Zhu, and Dongyun Yi. 2017. Scalable multi-view semi-supervised classification via adaptive regression. *IEEE Transactions on Image Processing* 26, 9 (2017), 4283–4296.

[29] Shiping Wang, Zhaoliang Chen, Shide Du, and Zhouchen Lin. 2022. Learning deep sparse regularizers with applications to multi-view clustering and semi-supervised classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 9 (2022), 5042–5055.

[30] Siwei Wang, Xinwang Liu, Xinzhong Zhu, Pei Zhang, Yi Zhang, Feng Gao, and En Zhu. 2022. Fast parameter-free multi-view subspace clustering with consensus anchor guidance. *IEEE Transactions on Image Processing* 31 (2022), 556–568.

[31] Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. AM-GCN: Adaptive multi-channel graph convolutional networks. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1243–1253.

[32] Yu Wang, Chuan Chen, Jinrong Lai, Lele Fu, Yuren Zhou, and Zibin Zheng. 2022. A self-representation method with local similarity preserving for fast multi-view outlier detection. *ACM Transactions on Knowledge Discovery from Data* 17, 1 (2022), 1–20. DOI: https://doi.org/10.1145/3532191

[33] Xuemei Wei, Yezheng Liu, Jianshan Sun, Yuanchun Jiang, Qifeng Tang, and Kun Yuan. 2022. Dual subgraph-based graph neural network for friendship prediction in location-based social networks. *ACM Transactions on Knowledge Discovery from Data* 17, 3 (2022), 1–28. DOI: https://doi.org/10.1145/3554981

[34] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 753–763.

[35] Tong Xia, Junjie Lin, Yong Li, Jie Feng, Pan Hui, Funing Sun, Diansheng Guo, and Depeng Jin. 2021. 3DGCN: 3-dimensional dynamic graph convolutional network for citywide crowd flow prediction. *ACM Transactions on Knowledge Discovery from Data* 15, 6 (2021), 1–21.

[36] Wei Xia, Quanxue Gao, Qianqian Wang, Xinbo Gao, Chris Ding, and Dacheng Tao. 2022. Tensorized bipartite graph learning for multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 4 (2022), 5187–5202. DOI: https://doi.org/10.1109/TPAMI.2022.3187976

[37] Wei Xia, Qianqian Wang, Quanxue Gao, Ming Yang, and Xinbo Gao. 2022. Self-consistent contrastive attributed graph clustering with pseudo-label prompt. *IEEE Transactions on Multimedia* (2022). DOI: https://doi.org/10.1109/TMM.2022.3213208

[38] Wei Xia, Qianqian Wang, Quanxue Gao, Xiangdong Zhang, and Xinbo Gao. 2022. Self-supervised graph convolutional network for multi-view clustering. *IEEE Transactions on Multimedia* 24 (2022), 3182–3192.

[39] Shunxin Xiao, Shide Du, Zhaoliang Chen, Yunhe Zhang, and Shiping Wang. 2023. Dual fusion-propagation graph neural network for multi-view clustering. *IEEE Transactions on Multimedia* (2023), 1–13. DOI: https://doi.org/10.1109/TMM.2023.3248173

[40] Hong-Xia Xie, Ling Lo, Hong-Han Shuai, and Wen-Huang Cheng. 2020. AU-assisted graph attention convolutional network for micro-expression recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*. 2871–2880.

[41] Yuan Xie, Wensheng Zhang, Yanyun Qu, Longquan Dai, and Dacheng Tao. 2020. Hyper-Laplacian regularized multilinear multiview self-representations for clustering and semisupervised learning. *IEEE Transactions on Cybernetics* 50, 2 (2020), 572–586.

[42] Weiqing Yan, Jindong Xu, Jinglei Liu, Guanghui Yue, and Chang Tang. 2022. Bipartite graph-based discriminative feature learning for multi-view clustering. In *Proceedings of the 30th ACM International Conference on Multimedia*. ACM, 3403–3411.

[43] Hao Yang, Dan Yan, Li Zhang, Yunda Sun, Dong Li, and Stephen J. Maybank. 2022. Feedback graph convolutional network for skeleton-based action recognition. *IEEE Transactions on Image Processing* 31 (2022), 164–175.

[44] Muli Yang, Cheng Deng, and Feiping Nie. 2019. Adaptive-weighting discriminative regression for multi-view classification. *Pattern Recognition* 88 (2019), 236–245.

[45] Kaixuan Yao, Jiye Liang, Jianqing Liang, Ming Li, and Feilong Cao. 2022. Multi-view graph convolutional networks with attention mechanism. *Artificial Intelligence* 307 (2022), 103708.

[46] Fanfan Ye, Shiliang Pu, Qiaoyong Zhong, Chao Li, Di Xie, and Huiming Tang. 2020. Dynamic GCN: Context-enriched topology learning for skeleton-based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*. 55–63.

[47] Bin Zhang, Qianyao Qiang, Fei Wang, and Feiping Nie. 2021. Flexible multi-view unsupervised graph embedding. *IEEE Transactions on Image Processing* 30 (2021), 4143–4156.

[48] Bin Zhang, Qianyao Qiang, Fei Wang, and Feiping Nie. 2022. Fast multi-view semi-supervised learning with learned graph. *IEEE Transactions on Knowledge and Data Engineering* 34, 1 (2022), 286–299.

[49] Yaobin Zhang, Weihong Deng, Mei Wang, Jiani Hu, Xian Li, Dongyue Zhao, and Dongchao Wen. 2020. Global-local GCN: Large-scale label noise cleansing for face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7731–7740.

[50] Ziyang Zhang, Chuan Chen, Yaomin Chang, Weibo Hu, Xingxing Xing, Yuren Zhou, and Zibin Zheng. 2021. SHNE: Semantics and homophily preserving network embedding. *IEEE Transactions on Neural Networks and Learning Systems* (2021). DOI: https://doi.org/10.1109/TNNLS.2021.3116936

[51] Hao Zhu and Piotr Koniusz. 2021. Simple spectral graph convolution. In *Proceedings of the 9th International Conference on Learning Representations*.

Received 5 December 2022; revised 27 March 2023; accepted 5 July 2023