Interpretable Graph Convolutional Network for Multi-view Semi-supervised Learning

Zhihao Wu, Xincan Lin, Zhenghong Lin, Zhaoliang Chen, Yang Bai and Shiping Wang

***Abstract*—As real-world data become increasingly heterogeneous, multi-view semi-supervised learning has garnered widespread attention. Although existing studies have made efforts towards this and achieved decent performance, they are restricted to shallow models and how to mine deeper information from multiple views remains to be investigated. As a recently emerged neural network, Graph Convolutional Network (GCN) exploits graph structure to propagate label signals and has achieved encouraging performance, and it has been widely employed in various fields. Nonetheless, research on solving multi-view learning problems via GCN is limited and lacks interpretability. To address this gap, in this paper we propose a framework termed Interpretable Multi-view Graph Convolutional Network (IMvGCN[1]). We first combine the reconstruction error and Laplacian embedding to formulate a multi-view learning problem that explores the original space from feature and topology perspectives. In light of a series of derivations, we establish a potential connection between GCN and multi-view learning, which holds significance for both domains. Furthermore, we propose an orthogonal normalization method to guarantee the mathematical connection, which solves the intractable problem of orthogonal constraints in deep learning. In addition, the proposed framework is applied to the multi-view semi-supervised learning task. Comprehensive experiments demonstrate the superiority of our proposed method over other state-of-the-art methods.**

***Index Terms*—Graph convolutional network, interpretable deep learning, orthogonal normalization, multi-view semi-supervised classification.**

I. Introduction

With the development of multimedia technology, the complexity and diversity of data are keeping growing. Meanwhile, extensive data collection pathways are available in many real-world scenarios. For example, the news is usually reported in different sources and with different media; an object can be illustrated by features of multiple mediums such as image, text and audio, etc. These types of data, which contain representations of the same instance from different sources or various feature extractors, are known as multi-view data. Actually, multi-view data have been extensively applied to a multitude of practical applications, such as machine learning [1], [2], [3], data mining [4], [5], [6] and computer vision [7], [8], [9]. Although each view only carries partial information of a sample, they implicitly express the same semantics. Due to the potential consistency and complementarity among views, multi-view data usually lead to encouraging performance over single-view data, which has motivated the research on multi-view learning [10], [11], [12].

Despite the growing variety of data collection ways, labeling data is still time-consuming and labor-intensive. And due to the two characteristics of multi-view data: richer data sources and fewer labels, this issue is even severer for multi-view learning. Therefore, it requires more efficient utilization of unlabeled samples to handle multi-view data. Many studies [13], [14], [15] have indicated that encouraging performance can be obtained by utilizing unlabeled multi-view data, which suggests the multi-view semi-supervised learning. Among numerous multi-view semi-supervised learning approaches, the graph-based methods exhibit several advantages, notably the ability to represent arbitrarily distributed data. The graph-based multi-view learning paradigms are usually based on graphs constructed by the similarity between paired samples, and various approaches have been developed on this common foundation. Nie et al. [16] proposed a model that can automatically allocate a weight to each view. Li et al. [17] adopted a straightforward strategy coalescing multi-view graphs to learn the weights and the common graph. Jia et al. [18] introduced orthogonality and adversarial similarity constraints to learn representations with less redundancy. The majority of these approaches are limited to shallow models, thus there is a wide expectation to leverage the representative capacity of deep learning to achieve more satisfactory effects.

Recently, graph-based deep learning methods, like graph neural network [19], graph autoencoder [20] and graph LSTM [21], [22] etc., have been increasingly investigated. Among them, Graph Convolutional Network (GCN) [23] has drawn considerable attention owing to its outstanding performance in graph learning. Based on rigorous theoretical foundations, GCN can efficiently propagate label signals in non-Euclidean space. Numerous variants of GCN [24], [25], [26] have been widely applied in diverse fields, while their applications to multi-view learning are still limited. Zhang et al. [27] designed a multi-view GCN with a fusion module capturing the spatial correlations between nodes. Cheng et al. [28] designed two-pathway encoders to map graph embedding features and learn cross-view consistency information. Although these approaches have attempted to apply GCN to multi-view learning

This work is in part supported by the National Natural Science Foundation of China under Grants U21A20472 and 62276065, and the National Key Research and Development Plan of China under Grant 2021YFB3600503. Corresponding author: Shiping Wang.

Zhihao Wu, Xincan Lin, Zhenghong Lin, Zhaoliang Chen and Shiping Wang are with the College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China and also with the Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China (E-mail: zhihaowu1999@gmail.com, xincanlinms@gmail.com, hongzhenglin970323@gmail.com, chenzl23@outlook.com, shipingwang-phd@163.com). Yang Bai is with the School of Cyberspace Security, Chengdu University of Information Technology, Chengdu 610054, China (E-mail: alicepub@163.com)

[1]Code is available at https://github.com/ZhihaoWu99/IMvGCN.

and demonstrated their superior performance, they did not explicitly establish a theoretical connection between GCN and multi-view learning. In other words, although these approaches have overcome the drawbacks of traditional shallow models, the lack of interpretability ensues. There have been several studies [29], [30], [31], [32] making efforts in interpretable neural networks. Nonetheless, only a limited number of studies have investigated how GCNs are connected to optimization problems theoretically [33]. Beyond that, some recent research has indicated shortcomings of GCN. For instance, Zhang et al. [34] sought to find high-order connectivity patterns to capture semantic information. Wang et al. [35] revealed that GCN did not well integrate information from neighbors. Li et al. [36] argued that GCN suffered from over-smoothing problem, that is, the performance of GCN decreases as the number of layers increases. And these problems may damage performance even more when handling complex multi-view data. In summary, it is vital to improve GCN and explore the interpretability of GCN-based methods for multi-view learning.

For the purpose of tackling the aforementioned issues, this paper proposes an effective framework termed Interpretable Multi-view Graph Convolutional Network (IMvGCN), which optimizes a reasonable multi-perspective learning problem and establishes its theoretic connection to GCN. Considering the latent consistency and complementarity among multifarious views, we first formulate a multi-view learning problem. More specifically, the reconstruction error is introduced and extended to the multi-view case, which captures the independence features and consistency semantics across views. Further, the multi-view Laplacian embedding is presented to maintain the local invariance on manifold structures for each view and to enable the framework to adapt to distinct distributed data. Crucially, based on a series of derivations, we propose a GCN-based framework with a flexible filter inspired by the optimization. Beyond that, a differentiable orthogonal normalization method is presented to satisfy the orthogonal constraint. The orthogonal normalization guarantees the connection between the multi-view problem and IMvGCN while allowing the weight matrices to be updated in a data-driven manner, and a detailed analysis is provided to demonstrate its feasibility. It can also adapt to various neural networks with promising generalizability. The optimization-inspired framework IMvGCN can not only retain the interpretability but also benefit from the deep network. Figure 1 briefly illustrates the proposed IMvGCN. The main contributions of this paper are summarized as follows:

1) Introduce the multi-view reconstruction error paired with Laplacian embedding to capture the independence and consistency, providing the basis for bridging multi-view learning and graph convolutional network.
2) Propose an end-to-end framework through a series of theoretical derivations, which achieves adjustable Laplacian smoothing by constructing a flexible graph filter.
3) A differentiable orthogonal normalization method is proposed, which furnishes a way to achieve strict orthogonality in deep learning with broad applicability.
4) Experimental results indicate the superiority of IMvGCN in multi-view semi-supervised classification tasks, especially when the labeled samples are scarce (e.g. 1%).

II. Related Work

In this section, we review some works relevant to our research, starting with an overview of GCN and then focusing on various approaches for multi-view semi-supervised learning.

A. Graph Convolutional Network

Graph convolutional network (GCN) was first proposed by Kipf et al. [23] and applied for improving semi-supervised classification performance. Generally, the propagation rule of GCN can be formulated as:

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

where $\mathbf{H}^{(l)}$ and $\mathbf{H}^{(l+1)}$ denote the input and the output of the l-th graph convolutional layer with σ denoting an activation function, and $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the refined adjacency matrix with self-connection. Besides, $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$, and $\mathbf{W}^{(l)}$ is a trainable weight matrix. GCN is capable of simultaneously aggregating the feature and spatial information, which facilitates the exploitation of the latent connections between nodes so as to learn a discriminative representation for a specific task. Due to the effectiveness of GCN, various extensions and variants of GCN have achieved desirable performance. Li et al. [36] pointed out that the graph convolution of the GCN was indeed a special form of Laplacian smoothing and then provided solutions to improve the GCN model for semi-supervised learning, which brought a deeper insight to understand GCN. Jie et al. [28] interpreted graph convolution as an integral transform of embedding functions under probability measures and utilized Monte Carlo approaches to consistently estimate integrals, leading to a batched training scheme named FastGCN. To alleviate the unnecessary complexity and redundant computation, Wu et al. [37] simplified GCN as SGCN by successively removing nonlinearities and collapsing weight matrices between consecutive layers, which yielded up to two orders of magnitude speedup. Chiang et al. [38] further presented a new GCN training algorithm for graph-based models, which led to significantly improved memory and computational efficiency. For employing the convolutional neural networks directly, Gao et al. [39] proposed a learnable graph convolutional layer to automatically transform generic graphs to data of grid-like structures, which is conducted through a k-largest node selection process. Although existing GCN models have demonstrated satisfactory performance in single-view graph data, there is still a great challenge to extend GCN on multi-view data.

B. Multi-view Semi-Supervised Classification

As the rapid development of multimedia, data exist in more complex and diverse forms, attributed to which multi-view learning has become a hot topic. In the past decades, plenty of multi-view methods have been devoted to semi-supervised learning [16], [28], [29], [35], [40]. Nie et al. [41] proposed a parameter-free multi-view learning algorithm with adaptive

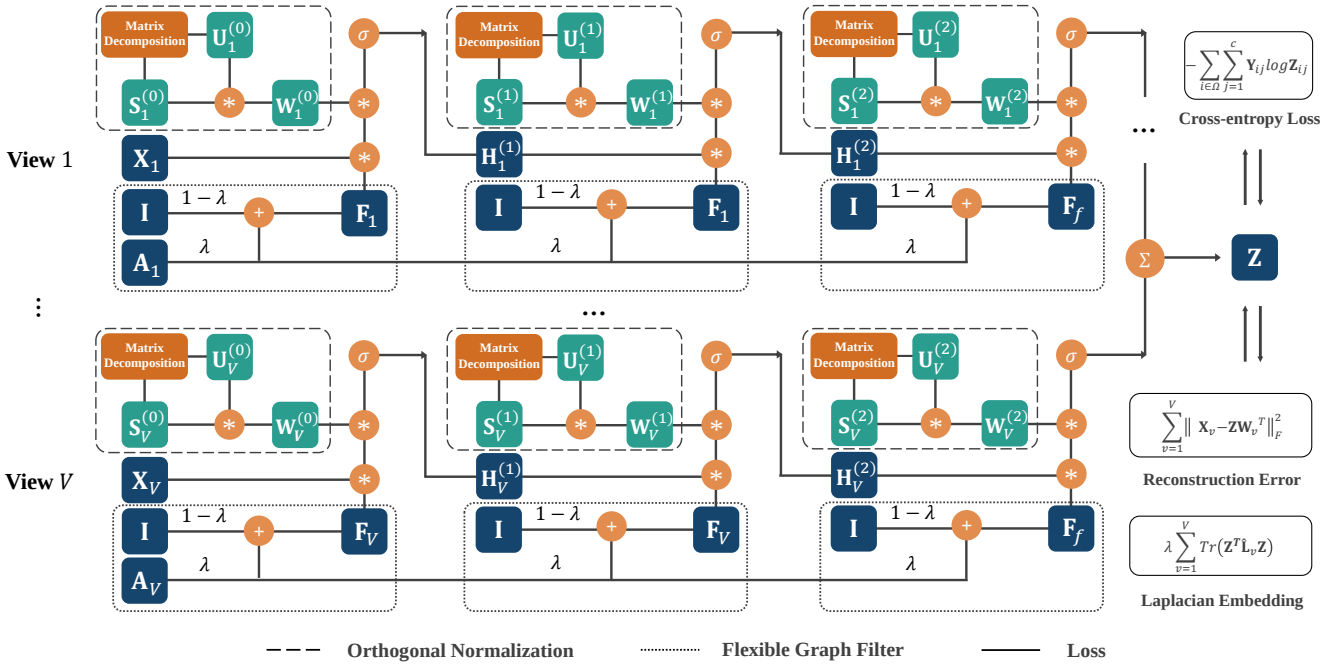


Fig. 1: The overview of the proposed IMvGCN, which establishes the connection between GCN and multi-view learning. The constructed flexible graph filter achieves adjustable Laplacian smoothing, and orthogonal normalization constrains the weights to avoid trivial solutions, allowing the network to benefit from orthogonality with its differentiability and generalization capability.

neighbors to jointly perform multi-view clustering and semi-supervised classification and local manifold structure learning, and further presented an auto-weight graph-oriented method to automatically allocate ideal weight for each view in [16]. To simultaneously utilize the consensus and complementary properties of multi-view data, Jia et al. [42] established a semi-supervised multi-view deep discriminant representation learning model, and incorporated the orthogonality and adversarial similarity constraints to reduce the redundancy of learned representations. Besides, Wang et al. [29] found that sparse regularizer learning is equivalent to learning a parameterized activation function and built a differentiable and reusable neural network to learn data-driven sparse regularizers adaptively, thus to learn a sparse representation for multi-view clustering and semi-supervised classification. To take advantage of the effectiveness of GCN, Li et al. [40] proposed a multi-view semi-supervised learning model by adaptively exploiting the graph information from the multiple views with combined Laplacian matrices, which simultaneously unified co-training, spectral graph information and the expressive power of neural network into one framework.

III. The Proposed Method

In this section, we first define the optimization problem and then further formulate the proposed method. For better understanding, we start with the introduction of some basic notations used in this paper. Denote given multi-view data as $\mathcal{X} = \{\mathbf{X}_v\}_{v=1}^V$, where $\mathbf{X}_v = [\mathbf{x}_1; \ldots; \mathbf{x}_n] \in \mathbb{R}^{n \times m_v}$ is the data from v-th view for any $v \in \{1, \ldots, V\}$, n is the number of samples and V is the number of views.

A. Problem Formulation

Dimensionality reduction is a series of classical methods mapping input samples onto a low-dimensional space. We first consider the projection of $\mathbf{X}$ onto some low-dimensional space and generate $\mathbf{Z}$. However, directly mapping $\mathbf{X}$ onto a low-dimensional space may lack generalization capability in many realistic scenarios. In order to obtain an effective low-dimensional representation, we introduce the reconstruction error between the projected data $\mathbf{Z}$ and original data $\mathbf{X}$, as $\mathbf{E} = \mathbf{X} - \mathbf{Z}\mathbf{W}^T$. To minimize the reconstruction error $\mathbf{E}$, the objective function can be defined as

$$\min_{\mathbf{Z},\mathbf{W}} \left\|\mathbf{X} - \mathbf{Z}\mathbf{W}^T\right\|_F^2 \quad \text{s.t. } \mathbf{W}^T\mathbf{W} = \mathbf{I}, \tag{2}$$

where $\mathbf{Z} = [\mathbf{z}_1; \ldots; \mathbf{z}_n] \in \mathbb{R}^{n \times k}$ is the learned representation of input data $\mathbf{X} \in \mathbb{R}^{n \times m}$ with $\mathbf{W} \in \mathbb{R}^{m \times k}$ being a standard orthonormal basis. Equation (2) is for the single-view case, as to the multi-view input $\mathcal{X}$, the target is to learn a common representation $\mathbf{Z}$ which is capable of extracting consistency and independence information among views. We first consider Equation (2) in each view, that is, calculate the reconstruction loss with respect to the v-th view input $\mathbf{X}_v$ and $\mathbf{Z}$. Then the sum of the reconstruction error of each view is minimized for the sake of capturing the consistent features across views. Consequently, Equation (2) is extended to a multi-view case:

$$\begin{aligned} &\min_{\mathbf{Z},\mathbf{W}_v} \sum_{v=1}^{V} \left\|\mathbf{X}_v - \mathbf{Z}\mathbf{W}_v^T\right\|_F^2 \\ &\text{s.t. } \mathbf{W}_v^T\mathbf{W}_v = \mathbf{I}, v \in \{1, 2, \ldots, V\}, \end{aligned} \tag{3}$$

where $\mathbf{W}_v \in \mathbb{R}^{m_v \times k}$ is the orthonormal basis of the v-th view. Equation (3) explores a common latent space of various

views, and projects the obtained representation $\mathbf{Z}$ onto the original feature space with $\mathbf{W}_v$. Although this method is efficient for many real-world applications, it perceives the transformation from the original space to the hidden space as a linear projection. Namely, the method ignores the spatial structure of input data, and it is inevitable to lose the inherent nonlinear characteristics.

Building a graph on the input data, Laplacian embedding preserves the intrinsic structure between data points in the low-dimensional space. Herein, $\mathcal{G} = \{\mathbf{X}, \mathbf{A}\}$ denotes an undirected graph, and $\mathbf{A}$ represents the adjacency matrix constructed using k-nearest neighbors (KNN) algorithm. Therefore, the (i, j)-th element of $\mathbf{A}$ can be formulated as

$$\mathbf{A}_{ij} = \begin{cases} 1, \ \mathbf{x}_i \in \text{KNN}(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \text{KNN}(\mathbf{x}_i), \\ 0, \ \text{otherwise}, \end{cases} \tag{4}$$

where $\mathbf{x}_i$ is the i-th row vector of $\mathbf{X}$ and $\text{KNN}(\mathbf{x}_i)$ denotes the set of k nearest neighbors of $\mathbf{x}_i$. With renormalized $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{I} + \mathbf{A})\mathbf{D}^{-\frac{1}{2}}$, the Laplacian matrix can be calculated as $\hat{\mathbf{L}} = \mathbf{I} - \hat{\mathbf{A}}$, and the objective function of Laplacian embedding can be specified by

$$\min_{\mathbf{Z}} \frac{1}{2} \sum_{i,j=1}^{n} \|\mathbf{z}_i - \mathbf{z}_j\|_2^2 \hat{\mathbf{A}}_{ij} = \min_{\mathbf{Z}} \ Tr\left(\mathbf{Z}^T \hat{\mathbf{L}} \mathbf{Z}\right), \tag{5}$$

where $\mathbf{z}_i$ denotes the representation of the i-th sample. By adopting Equation (5), we aim to keep the connected nodes in the predefined graph $\mathcal{G}$ close in the latent space. In other words, Equation (5) preserves the local invariance on the manifold structure. To overcome the aforementioned problems of Equation (2), we consider Equation (5) while solving Equation (2) to learn a better representation $\mathbf{Z}$. Similar to Equation (3), we first consider within-view invariance criterion $Tr\left(\mathbf{Z}^T \hat{\mathbf{L}} \mathbf{Z}\right)$ to maintain the nonlinear structure in the feature space, then capture the intrinsic manifold structure of each view to learn a hidden embedding with consistent spatial structure across views:

$$\min_{\mathbf{Z}} \sum_{v=1}^{V} Tr\left(\mathbf{Z}^T \hat{\mathbf{L}}_v \mathbf{Z}\right), \tag{6}$$

where $\hat{\mathbf{L}}_v$ is the Laplacian matrix constructed on the v-th view input $\mathbf{X}_v$. In fact, Equation (6) also suffers from the problem of being overstrict. Meanwhile, as mentioned before, a sample has diverse spatial structures in different views in the case of multi-view data, which is ignored in Equation (3). Accordingly, the two considerations are combined into a framework addressing the aforementioned issues:

$$\begin{aligned} \min_{\mathbf{Z}, \mathbf{W}_v} \ & \sum_{v=1}^{V} \left\|\mathbf{X}_v - \mathbf{Z}\mathbf{W}_v^T\right\|_F^2 + \lambda \sum_{v=1}^{V} Tr\left(\mathbf{Z}^T \hat{\mathbf{L}}_v \mathbf{Z}\right) \\ & \text{s.t. } \mathbf{W}_v^T \mathbf{W}_v = \mathbf{I}, v \in \{1, 2, \ldots, V\}, \end{aligned} \tag{7}$$

where λ is a hyperparameter that balances the importance of two terms, and $\mathbf{Z}$ denotes the learned common representation for all views. By considering the reconstruction error along with local invariance on manifold structure of multi-view data $\{\mathbf{X}_v\}_{v=1}^V$, Equation (7) can be well adapted to multi-view data containing different distributions in real scenarios. The following subsection focuses on the optimization of this problem and shows how it meets GCN.

B. Interpretable Multi-view GCN

In Problem (7), there are several optimization variables $\mathbf{Z}$ and $\{\mathbf{W}_v\}_{v=1}^V$, hence an alternating method is adopted to update them one by one. By fixing $\mathbf{W}_v$, all $\mathbf{W}_v$ are regarded as constants during the process of optimizing variable $\mathbf{Z}$, and Problem (7) is minimized over $\mathbf{Z}$. Denoting Problem (7) as $\mathcal{L}$ with fixed $\mathbf{W}_v$, the gradient of $\mathcal{L}$ w.r.t. $\mathbf{Z}$ can be computed as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{Z}} = 2V \left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right) \mathbf{Z} - 2 \sum_{v=1}^{V} \mathbf{X}_v \mathbf{W}_v. \tag{8}$$

Letting Equation (8) equal to 0, we have

$$\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right) \mathbf{Z} = \frac{1}{V} \sum_{v=1}^{V} \mathbf{X}_v \mathbf{W}_v. \tag{9}$$

Before proceeding to the next step of the derivation, we analyze $\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)$. For the symmetric adjacency matrix $\mathbf{A}$, its renormalized adjacency matrix is computed as $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{I} + \mathbf{A})\mathbf{D}^{-\frac{1}{2}}$, and the symmetric normalized Laplacian matrix $\hat{\mathbf{L}} = \mathbf{I} - \hat{\mathbf{A}}$ satisfies: $(\mathbf{I} + \lambda \hat{\mathbf{L}})$ is positive definite when $\lambda > -\frac{1}{2}$. For detailed derivation, please refer to Appendix A. Thus, the factor $\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)$ on the left side of the Equation (9) can be transformed to

$$\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right) = \frac{1}{V} \sum_{v=1}^{V} (\mathbf{I}_v + \lambda \hat{\mathbf{L}}_v), \tag{10}$$

where $\mathbf{I}_v$ is the v-th identity matrix. According to the analysis above and the properties of positive definite matrices, each $(\mathbf{I}_v + \lambda \hat{\mathbf{L}}_v)$ is positive definite and thus $\sum_{v=1}^{V} (\mathbf{I}_v + \lambda \hat{\mathbf{L}}_v)$ is also positive definite. Further, we multiply the inverse matrix of positive definite $\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)$ on both sides of Equation (9) simultaneously:

$$\mathbf{Z} = \left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)^{-1} \frac{1}{V} \sum_{v=1}^{V} \mathbf{X}_v \mathbf{W}_v. \tag{11}$$

Here we obtain the optimal solution to $\mathbf{Z}$. Then, the K-order Taylor expansion of $\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)^{-1}$ is

$$\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)^{-1} \approx \sum_{i=0}^{K} \left(-\frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)^i, \tag{12}$$

where K is the order of this Taylor series. When $K = 1$, we have the first-order truncated of this Taylor expansion:

$$\left(\mathbf{I} + \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v\right)^{-1} \approx \ \mathbf{I} - \frac{\lambda}{V} \sum_{v=1}^{V} \hat{\mathbf{L}}_v. \tag{13}$$

Here we set $\lambda \in (-\frac{1}{2}, \frac{1}{2}]$ to guarantee that $\sum_{v=1}^{V} (\mathbf{I}_v + \lambda \hat{\mathbf{L}}_v)$ is positive definite and that the Taylor series is convergent

simultaneously (refer to Appendix A). Moreover, Equation (13) can be expanded to

$$\begin{aligned}\left(\mathbf{I}+\frac{\lambda}{V}\sum_{v=1}^{V}\hat{\mathbf{L}}_v\right)^{-1} &\approx \mathbf{I}-\lambda(\mathbf{I}-\frac{1}{V}\sum_{v=1}^{V}\hat{\mathbf{A}}_v)\\ &=(1-\lambda)\mathbf{I}+\frac{\lambda}{V}\sum_{v=1}^{V}\hat{\mathbf{A}}_v,\end{aligned} \tag{14}$$

where $\hat{\mathbf{A}}_v$ is the renormalized adjacency matrix of the v-th view. By using $\hat{\mathbf{A}}_f=\frac{1}{V}\sum_{v=1}^{V}\hat{\mathbf{A}}_v$, we obtain the solution to $\mathbf{Z}$ alternating $\left(\mathbf{I}+\frac{\lambda}{V}\sum_{v=1}^{V}\hat{\mathbf{L}}_v\right)^{-1}$ with its first-order Taylor approximation:

$$\mathbf{Z}=\left((1-\lambda)\mathbf{I}+\lambda\hat{\mathbf{A}}_f\right)\frac{1}{V}\sum_{v=1}^{V}\mathbf{X}_v\mathbf{W}_v. \tag{15}$$

Supported by the above derivations, we further establish a theoretical connection between multi-view representation learning and GCN. When λ is close to 1, Equation (15) becomes

$$\mathbf{Z}=\frac{1}{V}\sum_{v=1}^{V}\hat{\mathbf{A}}_f\mathbf{X}_v\mathbf{W}_v. \tag{16}$$

With $\hat{\mathbf{A}}_f$ being the sum of renormalized adjacency matrix, we find that the simplified Equation (16) is the weighted average formulation of forward propagation of GCN. In other words, considering constructing a multi-view GCN, Equation (16) is indeed a simple form of the forward propagation equation for this model. In summary, the multi-view representation learning defined in Problem (7) can then be solved by designing a multi-view GCN. Following this, we return to the discussion of Equation (15). Li et al. [36] have revealed that the filter of GCN is actually a special form of Laplacian smoothing aggregating features from neighbors of a node. Thus, with the identity matrix adding an extra self-loop to $\hat{\mathbf{A}}_f$, hyperparameter λ controls the balance between strengthening self features and collecting information from neighbor nodes. And similar to the original GCN, truncating the expansion of Taylor series to the first-order guarantees that a one-step Laplacian smoothing only aggregates information from first-order neighbors of a node. Accordingly, a filter is proposed with flexible Laplacian smoothing, that is,

$$\mathbf{F}_f=(1-\lambda)\mathbf{I}+\lambda\hat{\mathbf{A}}_f. \tag{17}$$

Here, we have $\mathbf{F}_f=\mathbf{U}g_\lambda(\Sigma)\mathbf{U}^{\mathrm{T}}$ with $g_\lambda(\Sigma)=\mathbf{I}-\lambda\Sigma$, where $\mathbf{U}$ and Σ are obtained by $\frac{1}{V}\sum_{v=1}^{V}\hat{\mathbf{L}}_v=\mathbf{U}\Sigma\mathbf{U}^{\mathrm{T}}$. It can be observed that the response function $g_\lambda(\Sigma)$ changes with λ. The common embedding $\mathbf{Z}$ is generated by the forward computation of multi-view GCN, and naturally $\{\mathbf{W}_v\}_{v=1}^{V}$ can be updated by the backward propagation. For the scalability, we allow the scheme to be performed recursively within views, and replace $\mathbf{X}_v$ with $\mathbf{H}_v$. The multi-view module is constructed with the following propagation rule:

$$\mathbf{Z}=\frac{1}{V}\sum_{v=1}^{V}\mathbf{F}_f\mathbf{H}_v\mathbf{W}_v, \tag{18}$$

where $\mathbf{Z}$ is the learned embedding. By extending the flexible Laplacian smoothing filter $\mathbf{F}_f$ to view-wise as

$$\mathbf{F}_v=(1-\lambda)\mathbf{I}+\lambda\hat{\mathbf{A}}_v, \tag{19}$$

the forward propagation in the v-th view can be described as

$$\mathbf{H}_v^{(l+1)}=\sigma\left(\mathbf{F}_v\mathbf{H}_v^{(l)}\mathbf{W}_v^{(l)}\right), \tag{20}$$

where $\mathbf{H}_v^{(l)}$ denotes the hidden feature, $\mathbf{W}_v^{(l)}$ is the learnable parameter of the v-th view in the l-th layer and σ is an activation function. As a simple example, the learned embedding $\mathbf{Z}$ of a 3-layer model is calculated by

$$\mathbf{Z}=\frac{1}{V}\sum_{v=1}^{V}\mathbf{F}_f\sigma\left(\mathbf{F}_v\sigma\left(\mathbf{F}_v\mathbf{X}_v\mathbf{W}_v^{(0)}\right)\mathbf{W}_v^{(1)}\right)\mathbf{W}_v^{(2)}. \tag{21}$$

After completing the forward computation, $\mathbf{Z}$ is replaced with the computed $\mathbf{Z}^*$ and we consider obtaining the weights $\{\mathbf{W}_v^{(0)},\ldots,\mathbf{W}_v^{(L-1)}\}_{v=1}^{V}$. Since we construct a multi-layer model, $\mathbf{W}_v$ in the original objective function is replaced as $\mathbf{W}_v=\mathbf{W}_v^{(0)}\mathbf{W}_v^{(1)}\mathbf{W}_v^{(2)}$ for approximation. It is obvious that $\mathbf{W}_v$ is still orthogonal with $\mathbf{W}_v^T\mathbf{W}_v=\mathbf{I}$ if each $\mathbf{W}_v^{(l)}$ satisfies the orthonal constraint. Owing to the nonlinearity of the activation function, this approach is adopted to approximate the original problem, and the following optimization is considered:

$$\begin{aligned}(\mathbf{W}_v^{(l)})^*=\arg\min_{\mathbf{W}_v^{(l)}}&\sum_{v=1}^{V}\left\|\mathbf{X}_v-\mathbf{Z}^*\mathbf{W}_v^T\right\|_F^2\\ +\lambda\sum_{v=1}^{V}&Tr\left((\mathbf{Z}^*)^T\hat{\mathbf{L}}_v\mathbf{Z}^*\right)\ \text{s.t.}\ \left(\mathbf{W}_v^{(l)}\right)^T\mathbf{W}_v^{(l)}=\mathbf{I},\\ &v\in\{1,2,\ldots,V\},l\in\{0,1,\ldots,L-1\},\end{aligned} \tag{22}$$

where $\mathbf{W}_v=\prod_{l=0}^{L-1}\mathbf{W}_v^{(l)}$, and for the Equation (21) it is $\mathbf{W}_v=\mathbf{W}_v^{(0)}\mathbf{W}_v^{(1)}\mathbf{W}_v^{(2)}$ here. As mentioned before, the process of updating $\{\mathbf{W}_v^{(0)},\ldots,\mathbf{W}_v^{(L-1)}\}_{v=1}^{V}$ is performed by backward propagation, and the difficulty is how to preserve orthogonality of $\mathbf{W}_v^{(l)}$. Therefore, we introduce a differentiable orthogonal normalization method in Section III-C to achieve strict orthogonality of $\mathbf{W}_v^{(l)}$. Here we default that $\mathbf{W}_v^{(l)}$ satisfies the orthogonal constraint, and the reconstruction error with Laplacian embedding loss is formulated as

$$\mathcal{L}_{rl}=\sum_{v=1}^{V}\left\|\mathbf{X}_v-\mathbf{Z}\mathbf{W}_v^T\right\|_F^2+\lambda\sum_{v=1}^{V}Tr\left(\mathbf{Z}^T\hat{\mathbf{L}}_v\mathbf{Z}\right). \tag{23}$$

Up to here, Problem (7) is solved by establishing a GCN-based multi-view neural network. Nonetheless, from another perspective, the capabilities of GCN are not fully explored. Many studies have found that the power of the GCN is owed to that it can efficiently propagate label signals by utilizing topological information. Relying on only the above two losses will substantially damage this ability of the proposed framework. Therein, the cross-entropy function is introduced to motivate the flow of label information:

$$\mathcal{L}_c=-\sum_{i\in\Omega}\sum_{j=1}^{c}\mathbf{Y}_{ij}log\hat{\mathbf{Y}}_{ij}, \tag{24}$$

where $\mathbf{Y} \in \mathbb{R}^{|\Omega| \times c}$ is the label matrix generated from the set Ω which is a tiny part of the entire label space, and $\hat{\mathbf{Y}} = softmax(\mathbf{Z})$ denotes the predicted labels. It is highlighted that, induced by the unsupervised multi-view problem, the model structurally approximates the optimization, enhancing its capacity in scenarios with extremely limited labels.

C. Orthogonal Normalization

In this subsection, we focus on the orthogonal constraint of $\{\mathbf{W}_v\}_{v=1}^V$, which is a key point of the proposed method. Although the solution for $\{\mathbf{W}_v\}_{v=1}^V$ can be obtained using the aforementioned multi-view framework, it will deviate the designed framework from the original multi-view learning problem. To be specific, the forward computation can be regarded as performing the propagation $\mathbf{F}_{(v)}\mathbf{H}_{(v)}$ and the transformation $\mathbf{F}_{(v)}\mathbf{H}_{(v)}\mathbf{W}_{(v)}$. With the orthogonal constraint on $\{\mathbf{W}_v\}_{v=1}^V$, the transformation is indeed projecting the feature onto a low-dimensional space under the guidance of the original objective. The advantages are two-fold: Firstly, the network retains the interpretability, which originates from the traditional multi-view problem; meanwhile, the learning of $\{\mathbf{W}_v\}_{v=1}^V$ is not over-restricted by the problem and is updated in a data-driven way where the task-oriented loss can be introduced, which originates from the deep learning. As a result, keeping the connection with the original unsupervised objective helps to capture consistent and complementary information from multiple views. Consequently, the designed framework can learn a more discriminative consistent representation when the labeled samples are scarce, which is the essential goal of multi-view semi-supervised learning.

Inherently, how to achieve this is a well-known dilemma in deep learning due to the difficulty of orthogonality preservation, and the relaxed orthogonal constraints were used in many previous works. For the sake of theoretical strictness and avoiding trivial solutions, we do not take any method to relax the constraint, such as transforming it into reconstruction loss. In the following part, a normalization approach is proposed to ensure strict orthogonality and provide theoretical guarantees for its feasibility. Note that the proposed orthogonal normalization method can be plugged into other neural networks rather than limited to the present framework.

Theorem 1: Given matrix $\mathbf{A} \in \mathbb{R}^{d_1 \times d_2}$, suppose that $\mathbf{A}^T\mathbf{A}$ can be decomposed as $\mathbf{A}^T\mathbf{A} = \mathbf{U}\mathbf{U}^T$ where $\mathbf{U} \in \mathbb{R}^{d_2 \times d_2}$ is an invertible matrix, we have the constructed matrix $\mathbf{B} = \mathbf{A}\left(\mathbf{U}^{-1}\right)^T$ satisfing $\mathbf{B}^T\mathbf{B} = \mathbf{I}$.

Proof: For matrix decomposition $\mathbf{A}^T\mathbf{A} = \mathbf{U}\mathbf{U}^T$, we have

$$\mathbf{U}^{-1}\mathbf{A}^T\mathbf{A}\left(\mathbf{U}^{-1}\right)^T = \mathbf{I}, \tag{25}$$

as $\mathbf{U}$ is invertible. Then, by constructing $\mathbf{B} = \mathbf{A}\left(\mathbf{U}^{-1}\right)^T$, it can be concluded that $\mathbf{B}$ satisfies the orthogonal constraint, as shown below:

$$\begin{aligned}\mathbf{B}^T\mathbf{B} &= (\mathbf{A}(\mathbf{U}^{-1})^T)^T(\mathbf{A}\left(\mathbf{U}^{-1}\right)^T) \\ &= \mathbf{U}^{-1}\mathbf{U}\mathbf{U}^T\left(\mathbf{U}^{-1}\right)^T = \mathbf{I}.\end{aligned} \tag{26}$$

□

According to Theorem 1, we can conduct different types of matrix decomposition on $\mathbf{A}^T\mathbf{A}$ to obtain an orthogonal matrix. For example, we have $\mathbf{U} = \mathbf{V}\mathbf{\Sigma}^{-\frac{1}{2}}$ when performing eigenvalue decomposition $\mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T$. For Cholesky decomposition $\mathbf{A}^T\mathbf{A} = \mathbf{L}\mathbf{L}^T$, we have $\mathbf{U} = \mathbf{L}$. In this paper, we adopt Cholesky decomposition. To be specific, suppose that $\mathbf{A}^T\mathbf{A}$ is full rank with the given matrix $\mathbf{A} \in \mathbb{R}^{d_1 \times d_2}$, then the Cholesky decomposition can be performed on $\mathbf{A}^T\mathbf{A}$ as $\mathbf{A}^T\mathbf{A} = \mathbf{U}\mathbf{U}^T$, where $\mathbf{U} \in \mathbb{R}^{d_2 \times d_2}$ is a lower triangular matrix and $\mathbf{U}^T$ is an upper triangular. In practice, the full rank of $\mathbf{A}^T\mathbf{A}$ can be achieved by adding small numbers to its diagonal elements. In summary, we can set a variable $\mathbf{A}$ called proxy variable, and perform matrix decomposition on it and obtain projection matrix $\mathbf{U}$, then $\mathbf{A}$ is projected to $\mathbf{B}$, which is the true variable employed to conduct forward computation. Now we have the method to solve orthogonal constraints in deep learning, but there are still some problems.

Theorem 2: Given $\mathbf{B}$ satisfying $\mathbf{B}^T\mathbf{B} = \mathbf{I}$, the construction is not unique and there are infinite matrices $\mathbf{C} = \mathbf{B}\mathbf{R}$ that satisfy $\mathbf{C}^T\mathbf{C} = \mathbf{I}$, where $\mathbf{R}$ is any arbitrary matrix.

Proof: Setting $\mathbf{C} = \mathbf{B}\mathbf{R}$ where $\mathbf{R} \in \mathbb{R}^{d_2 \times d_2}$ is an arbitrary orthogonal matrix, $\mathbf{C}^T\mathbf{C} = \mathbf{I}$ is supported by

$$\mathbf{C}^T\mathbf{C} = \mathbf{R}^T\mathbf{B}^T\mathbf{B}\mathbf{R} = \mathbf{R}^T\mathbf{R} = \mathbf{I}, \tag{27}$$

with the premise that $\mathbf{B}^T\mathbf{B} = \mathbf{I}$ and $\mathbf{R}^T\mathbf{R} = \mathbf{I}$. □

According to Theorem 2, the construction method is more generally defined as $\mathbf{C} = \mathbf{A}\left(\mathbf{U}^{-1}\right)^T\mathbf{R}$ with $\mathbf{R}$ being an arbitrary orthogonal matrix. It can be clearly observed that the construction of $\mathbf{C}$ is indeed a linear projection on $\mathbf{A}$. Thus, to find the optimal $\mathbf{C}$, we naturally want to optimize

$$\begin{aligned}&\min_{\mathbf{R}} \ \|\mathbf{A} - \mathbf{C}\|_F^2 \\ &\text{s.t. } \mathbf{C} = \mathbf{A}\left(\mathbf{U}^{-1}\right)^T\mathbf{R}, \mathbf{C}^T\mathbf{C} = \mathbf{I}.\end{aligned} \tag{28}$$

By a series of derivations (refer to Appendix B), Equation (28) can be transformed into

$$\max_{\mathbf{R}} \ Tr\left(\mathbf{U}\mathbf{R}\right) = \sum_{i=1}^{d_2} \mathbf{U}_{ii}\mathbf{R}_{ii}, \ \text{s.t. } \mathbf{R}^T\mathbf{R} = \mathbf{I} \tag{29}$$

where $\mathbf{U}_{ii}$ and $\mathbf{R}_{ii}$ are the elements on the diagonals of $\mathbf{U}_{ii}$ and $\mathbf{R}_{ii}$ respectively. The norm of each row or column vector of $\mathbf{R}$ is 1 because $\mathbf{R}$ is an orthogonal matrix, thereby it is evident that the solution of $\mathbf{R}$ is that $\mathbf{R}^* = \mathbf{I}$. Replacing $\mathbf{R}$ with $\mathbf{R}^*$, the optimal solution of $\mathbf{C}$ is

$$\mathbf{C}^* = \mathbf{A}\left(\mathbf{U}^{-1}\right)^T. \tag{30}$$

Analysis above theoretically proves the feasibility and reasonableness of the proposed method, and we apply the orthogonal normalization formulated by Equation (30) to preserve orthogonality of each weight matrix $\mathbf{W}_v^{(l)}$. Specifically, we define a parameter matrix $\mathbf{S}_v^{(l)}$ and construct $\mathbf{W}_v^{(l)}$ as

$$\mathbf{W}_v^{(l)} = \mathbf{S}_v^{(l)}((\mathbf{U}_v^{(l)})^{-1})^T, \tag{31}$$

where $((\mathbf{U}_v^{(l)})^{-1})^T$ is obtained by performing Cholesky decomposition on $(\mathbf{S}_v^{(l)})^T\mathbf{S}_v^{(l)}$:

$$(\mathbf{S}_v^{(l)})^T\mathbf{S}_v^{(l)} = \mathbf{U}_v^{(l)}(\mathbf{U}_v^{(l)})^T. \tag{32}$$

Note that, backward propagation requires calculating the derivative of projection $\mathbf{W}_v^{(l)} = \mathbf{S}_v^{(l)}((\mathbf{U}_v^{(l)})^{-1})^T$ during the

Algorithm 1 IMvGCN

Input: Multi-view data $\{\mathbf{X}_1, \ldots, \mathbf{X}_V\}$, semi-supervised information $\mathbf{Y}$, hyperparameters λ and α.
Output: The fused embedding $\mathbf{Z}$.

```
Generate adjacency matrices {Â_1, ..., Â_V};
while not converge or the framework does not meet the early stop condition do
    for v = 0 → V do
        for l = 1 → L do
            Compute F_v or F_f with (19) or (17);
            Perform Cholesky decomposition on (S_v^(l))^T S_v^(l) to obtain ((U_v^(l))^-1)^T;
            Calculate W_v^(l) with (31);
            Compute H_v^(l+1) with (20);
        end for
    end for
    Calculate the common embedding Z with (18);
    Update parameter matrices {S_1^(l), ..., S_V^(l)} with backward propagation;
end while
return The fused representation Z.
```

training process. The derivative of $\mathbf{W}_v^{(l)}$ with respect to $\mathbf{S}_v^{(l)}$ can be computed since the gradient information can be transferred through Cholesky decomposition [43], and the same conclusion holds for eigenvalue decomposition [30]. Therefore, when using a differentiable matrix decomposition method, the proposed orthogonal normalization module is differentiable, and the derivative of the loss function with respect to $\mathbf{S}_v^{(l)}$ can be calculated by the chain rule, as $\frac{\partial \mathcal{L}}{\partial \mathbf{S}_v^{(l)}} = \left(\frac{\partial \mathbf{W}_v^{(l)}}{\partial \mathbf{S}_v^{(l)}}\right)^T \frac{\partial \mathcal{L}}{\partial \mathbf{W}_v^{(l)}}$. As discussed above, orthogonal normalization ensures a stronger link between original multi-view learning problem and the proposed IMvGCN, thereby further bolstering the performance in label-scare scenarios, which is soon validated by the experimental results. Finally, training details and the computational complexity of the proposed framework are described in the following subsection.

D. Training Details

Combining Equations (23) and (24), the proposed IMvGCN is trained using the following objective function:

$$\mathcal{L} = \mathcal{L}_c + \alpha \mathcal{L}_{rl}, \tag{33}$$

where α is a hyperparameter balancing the two terms. Algorithm 1 depicts the procedure of IMvGCN. Generally, the procedure contains within-view propagation and the fusion of the embeddings obtained from each view.

We also analyze the complexity of the forward calculation of IMvGCN. Firstly, we consider the computational complexity of IMvGCN without the Taylor approximation. In this case, the forward computation includes three main parts, that is, matrix inversion, Cholesky decomposition and matrix multiplications. For $\left(\mathbf{I} + \frac{\lambda}{V}\sum_{v=1}^{V}\hat{\mathbf{L}}_v\right)^{-1} \in \mathbb{R}^{n\times n}$, the computational complexity of matrix inversion is $\mathcal{O}(n^3)$. Then the complexity of Cholesky decomposition performed on $(\mathbf{W}_v^{(l)})^T\mathbf{W}_v^{(l)} \in \mathbb{R}^{d_v^{(l+1)}\times d_v^{(l+1)}}$ is $\mathcal{O}\left(\left(d_v^{(l+1)}\right)^3\right)$, and it is $\mathcal{O}\left(n^2 d_v^{(l)} + n d_v^{(l)} d_v^{(l+1)}\right)$ for matrix multiplications. Thus, for the l-th layer in the v-th view, it requires $\mathcal{O}\left(n^3 + n^2 d_v^{(l)} + n d_v^{(l)} d_v^{(l+1)} + \left(d_v^{(l+1)}\right)^3\right)$. Denoting the maximum value of hidden dimension set $\{d_v^{(1)}, d_v^{(2)}, \ldots, d_v^{(L)}\}_{v=1}^{V}$ as d, we have $\mathcal{O}(n^3 + n^2 d + nd^2 + d^3)$ for all layers. In particular, the complexity is $\mathcal{O}\left(n^3 + n^2 m + nmd + d^3\right)$ for the first layer of the v-th view, where m is the maximum input feature dimension among all views. Owing to $d \ll min\{m, n\}$, the total computation requires $\mathcal{O}(Vn^2(Ln + m))$ when we build an L-layer network for each of the V views. Subsequently, we calculate the computational complexity of IMvGCN using Taylor approximation. As we do not need to perform the matrix inversion, the computational complexity for each layer is reduced to $\mathcal{O}(n^2 d + nd^2 + d^3)$ and it can be simplified as $\mathcal{O}(n^2 m + nmd + d^3)$ for the first layer. Consequently, the forward computation complexity of IMvGCN consumes $\mathcal{O}(Vn^2(Ld + m))$.

TABLE I: Detailed statistics of all datasets.

Datasets	# Samples	# Views	Feature Distributions	# Classes
3Sources	169	3	3,560/3,631/3,068	6
Animals	10,158	2	4,096/4,096	50
Caltech101	9,144	6	48/40/254/1,984/512/928	101
Citeseer	3,312	2	3,703/3,312	6
GRAZ02	1,476	6	512/32/256/500/500/680	4
NGs	500	3	2,000/2,000/2,000	5
MNIST	2,000	3	30/9/9	10
Out-Scene	2,688	4	512/432/256/45	8
NoisyMNIST	30,000	2	784/784	10
Reuters	18,758	5	21,531/24,892/34,251/15,506/11,547	6

IV. Experiment

In this section, the proposed framework IMvGCN is compared with nine methods on eight real-world benchmark datasets. We first describe the datasets and experimental settings, then demonstrate the performance of IMvGCN. Moreover, comprehensive experiments are conducted to analyze its convergence and parameter sensitivity.

A. Experimental Settings

Eight public benchmark datasets with multiple descriptions of samples are utilized to perform multi-view semi-supervised classification tasks: 3Sources[2], Animals, Caltech101[3], GRAZ02[4], NGs[5], MNIST, Out-Scene, NoisyMNIST[6] and Reuters[7]. A summary of statistics about these datasets is recorded in Table I. The detailed description is given as follows.

3Sources is a textual dataset containing news from three sources: BBC, Reuters and Guardian. We select 169 stories reported by three companies to constitute three-view data.

[2]http://mlg.ucd.ie/datasets/3sources.html
[3]http://www.vision.caltech.edu/ImageDatasets/Caltech101/Caltech101.html
[4]http://www.emt.tugraz.at/~pinz/data/GRAZ_02
[5]http://www.qwone.com/~jason/20Newsgroups
[6]https://github.com/nineleven/NoisyMNISTDetection
[7]https://www.kaggle.com/datasets/nltkdata/reuter

TABLE II: Comparison results (mean% and standard deviation%) of all compared methods with 1% and 5% labeled samples, where the best results are highlighted in red and the second best results are highlighted in blue. AMGL fails to work on some datasets, which are marked with "—" in the table, and OOM denotes the out-of-memory error.

Ratio	Datasets	Metrics	KNN	AMGL	MVAR	MLAN	AWDR	HLR-M^{2}VS	GCN-Fusion	Co-GCN	DSRL	Ours
1%	3Sources	ACC	35.7 (10.9)	52.0 (8.9)	23.2 (11.1)	32.8 (4.0)	22.3 (13.1)	22.9 (10.9)	**53.9 (10.4)**	45.4 (5.1)	33.2 (0.2)	**84.1 (0.1)**
		F1	16.7 (6.6)	**48.9 (6.8)**	12.4 (8.4)	13.8 (8.3)	5.8 (3.1)	6.0 (2.5)	23.2 (4.2)	21.1 (4.8)	8.3 (0.0)	**81.9 (0.6)**
	Animals	ACC	55.3 (1.0)	52.4 (1.3)	43.2 (2.9)	51.7 (6.8)	55.0 (4.0)	46.5 (2.0)	**60.4 (2.0)**	55.9 (6.1)	53.4 (2.9)	**70.4 (0.1)**
		F1	42.5 (0.8)	48.0 (1.7)	38.5 (2.8)	46.4 (4.9)	43.3 (3.4)	42.3 (1.7)	**53.3 (2.4)**	42.4 (5.3)	43.3 (3.9)	**62.0 (0.1)**
	Caltech101	ACC	18.8 (0.9)	33.3 (0.6)	23.5 (2.2)	17.1 (2.6)	16.8 (1.3)	29.9 (1.9)	**38.0 (1.1)**	24.7 (3.2)	36.1 (1.3)	**40.1 (0.3)**
		F1	5.7 (0.6)	**15.7 (1.5)**	5.2 (0.7)	1.9 (0.2)	3.4 (0.5)	8.3 (1.1)	14.0 (1.4)	7.2 (1.4)	10.6 (1.3)	**15.9 (0.2)**
	Citeseer	ACC	20.2 (3.4)	—	20.5 (1.4)	43.9 (6.5)	35.1 (4.3)	36.5 (3.2)	46.3 (2.9)	35.1 (2.7)	**49.3 (2.5)**	**61.0 (0.3)**
		F1	27.8 (1.0)	—	26.5 (0.8)	36.9 (5.1)	37.6 (2.3)	38.6 (5.0)	38.1 (2.8)	29.1 (2.2)	**43.2 (1.7)**	**55.8 (0.3)**
	GRAZ02	ACC	30.6 (4.5)	38.5 (2.6)	34.6 (3.6)	30.2 (2.0)	37.7 (5.2)	**49.7 (12.4)**	45.3 (3.8)	37.8 (7.3)	41.2 (1.4)	**51.5 (0.1)**
		F1	31.2 (5.3)	40.3 (2.6)	28.9 (7.3)	34.7 (4.0)	37.5 (5.3)	43.9 (14.6)	**44.9 (5.1)**	34.2 (7.7)	39.9 (2.9)	**49.7 (0.1)**
	NGs	ACC	23.7 (2.1)	—	20.1 (0.3)	61.1 (13.9)	23.0 (2.5)	26.1 (5.0)	**83.1 (8.1)**	55.8 (9.5)	56.9 (6.3)	**90.5 (0.0)**
		F1	27.3 (6.4)	—	20.2 (10.2)	68.4 (10.1)	29.3 (4.8)	28.6 (6.5)	**81.8 (10.2)**	47.5 (10.5)	49.8 (10.2)	**90.5 (0.0)**
	MNIST	ACC	37.2 (2.9)	45.9 (4.3)	52.3 (8.7)	57.4 (3.8)	46.7 (4.1)	33.1 (9.6)	77.4 (1.7)	**80.5 (0.9)**	67.1 (8.3)	**88.4 (0.3)**
		F1	31.7 (4.5)	46.7 (6.0)	48.2 (4.8)	56.8 (2.9)	42.8 (4.3)	35.8 (8.4)	74.3 (2.3)	**78.0 (0.7)**	56.0 (12.5)	**85.7 (0.7)**
	Out-Scene	ACC	38.6 (3.3)	47.0 (2.2)	28.9 (4.8)	23.9 (1.5)	35.2 (1.3)	29.4 (5.5)	**59.3 (2.1)**	55.2 (2.3)	49.7 (3.1)	**65.0 (0.6)**
		F1	38.4 (3.4)	51.6 (2.2)	31.6 (3.4)	22.8 (9.3)	32.6 (3.6)	38.0 (5.5)	**60.2 (2.5)**	54.9 (4.1)	47.0 (3.9)	**64.6 (0.6)**
	NoisyMNIST	ACC	75.1 (1.0)	**89.8 (1.7)**	59.1 (1.9)	OOM	70.0 (0.9)	OOM	79.0 (1.3)	74.6 (3.6)	OOM	**90.4 (0.3)**
		F1	75.4 (1.0)	**89.8 (1.8)**	58.5 (1.5)	OOM	69.7 (0.5)	OOM	79.0 (1.3)	72.3 (4.2)	OOM	**90.2 (0.4)**
	Reuters	ACC	37.0 (0.0)	—	72.3 (0.0)	OOM	OOM	OOM	74.4 (1.0)	**75.9 (2.5)**	OOM	**79.9 (0.5)**
		F1	40.9 (0.0)	—	68.7 (0.0)	OOM	OOM	OOM	**70.1 (0.8)**	62.1 (4.9)	OOM	**76.1 (0.7)**
5%	3Sources	ACC	41.5 (8.7)	58.8 (10.3)	36.4 (10.9)	62.2 (11.0)	27.6 (10.3)	63.4 (14.0)	**85.8 (5.1)**	71.3 (0.9)	63.6 (5.3)	**92.5 (0.2)**
		F1	31.1 (12.8)	48.1 (7.8)	17.8 (7.1)	41.9 (12.4)	8.5 (4.6)	50.8 (22.8)	**76.6 (9.8)**	53.5 (2.6)	41.5 (3.5)	**89.7 (0.5)**
	Animals	ACC	69.5 (1.0)	66.3 (0.7)	78.0 (1.1)	78.8 (0.7)	**79.3 (0.7)**	66.3 (1.2)	75.8 (0.7)	79.0 (0.4)	73.8 (0.7)	**81.6 (0.1)**
		F1	64.6 (1.5)	60.5 (0.8)	**73.0 (1.1)**	72.9 (1.0)	72.6 (0.9)	62.2 (1.4)	67.7 (1.0)	72.2 (0.5)	67.0 (1.7)	**74.8 (0.2)**
	Caltech101	ACC	26.7 (0.8)	42.0 (0.5)	44.3 (1.0)	35.6 (0.1)	37.2 (0.7)	41.5 (1.7)	47.8 (0.7)	29.3 (1.0)	**48.8 (0.6)**	**51.7 (0.2)**
		F1	11.2 (0.6)	25.1 (0.7)	25.1 (1.4)	16.9 (0.7)	16.6 (0.7)	24.6 (1.6)	27.4 (0.5)	14.6 (0.6)	**27.8 (1.2)**	**30.7 (0.2)**
	Citeseer	ACC	24.9 (6.3)	—	**63.1 (1.2)**	62.3 (4.5)	62.0 (1.4)	52.0 (2.9)	60.0 (1.1)	49.9 (0.4)	58.8 (1.8)	**70.9 (0.1)**
		F1	32.1 (2.3)	—	**59.2 (1.9)**	59.2 (2.3)	57.5 (0.9)	50.5 (1.7)	52.4 (0.5)	44.5 (0.3)	53.8 (1.7)	**65.6 (0.3)**
	GRAZ02	ACC	41.4 (2.2)	50.9 (1.5)	49.0 (3.8)	50.2 (6.3)	40.9 (1.2)	45.2 (5.5)	**53.1 (2.6)**	38.0 (0.7)	46.9 (3.5)	**58.6 (0.6)**
		F1	41.6 (2.1)	51.7 (1.6)	50.5 (3.6)	52.4 (2.7)	40.5 (1.5)	48.7 (4.4)	**53.6 (2.7)**	37.6 (0.6)	47.1 (3.7)	**58.7 (0.7)**
	NGs	ACC	34.6 (8.1)	—	29.1 (9.1)	**93.8 (1.1)**	54.1 (5.4)	59.9 (11.1)	92.7 (1.4)	83.9 (0.3)	71.4 (2.8)	**97.5 (0.1)**
		F1	45.7 (7.0)	—	40.5 (9.4)	**93.9 (1.1)**	62.9 (5.7)	65.6 (9.1)	92.7 (1.4)	83.7 (0.3)	71.7 (2.8)	**97.5 (0.1)**
	MNIST	ACC	57.7 (1.9)	65.9 (2.3)	78.2 (3.9)	85.6 (1.2)	66.4 (4.8)	67.0 (6.4)	83.5 (1.2)	85.4 (2.3)	**86.7 (1.2)**	**90.8 (0.3)**
		F1	54.3 (1.7)	67.1 (1.9)	76.3 (4.7)	82.7 (1.4)	65.3 (3.6)	68.4 (4.0)	81.1 (1.2)	83.5 (2.3)	**83.0 (3.1)**	**88.6 (0.2)**
	Out-Scene	ACC	46.4 (1.8)	64.2 (1.5)	42.5 (1.5)	69.0 (2.0)	52.9 (1.6)	63.0 (4.0)	**69.6 (1.2)**	69.0 (0.5)	61.9 (2.0)	**74.3 (0.3)**
		F1	47.6 (1.8)	66.0 (1.4)	44.5 (1.9)	72.5 (1.6)	54.3 (1.4)	66.7 (2.4)	**69.9 (1.4)**	69.4 (0.5)	62.5 (2.0)	**74.5 (0.3)**
	NoisyMNIST	ACC	85.3 (0.5)	**93.9 (1.1)**	73.3 (0.8)	OOM	76.2 (0.5)	OOM	90.2 (1.8)	88.9 (2.3)	OOM	**94.6 (0.4)**
		F1	85.4 (0.4)	**93.9 (1.1)**	72.7 (0.7)	OOM	75.9 (0.5)	OOM	90.1 (1.5)	88.2 (1.8)	OOM	**94.4 (0.4)**
	Reuters	ACC	41.2 (0.0)	—	82.3 (0.0)	OOM	OOM	OOM	**81.2 (0.4)**	81.2 (2.2)	OOM	**84.2 (0.0)**
		F1	46.7 (0.0)	—	80.0 (0.0)	OOM	OOM	OOM	**79.0 (0.4)**	72.0 (4.4)	OOM	**81.7 (0.0)**

Animals consists of 30,475 animal images divided into 50 categories. We generate a subset containing 10,158 samples, that is, 1/3 samples per class selected from Animals. And two types of features are extracted from these images using DECAF and VGG19.

Caltech101 is a widely used object recognition dataset containing 101 classes of images. Six features extracted from the original images constitute the data from six views: 48-D Gabor feature, 40-D wavelet moments (WN), 254-D CENTRIST feature, 1,984-D HOG feature, 512-D GIST feature, and 928-D LBP feature.

Citeseer is a research paper dataset with 3,312 samples that are classified into 6 categories. Adopting the setup of [44], we generate 3,703-D bag-of-words vectors as the first view and 3,312-D vectors representing the citation relationships between documents as the second view.

GRAZ02 is a dataset for object categorization with four classes of images: bicycles, people, cars, and a class that does not contain these objects. The following 6 types of features are extracted: SIFT, SURF, GIST, LBP, PHOG and WT.

NGs is a subset of the 20 Newsgroup datasets that is a collection of approximately 20,000 newsgroup documents. NGs consists of 500 newsgroup documents in 5 classes, and each document is pre-processed with three different methods as three views.

MNIST is a well-known handwritten digit dataset, where three types of features are extracted: 30-D IsoProjection, 9-D Linear Discriminant Analysis and 9-D Neighborhood Preserving Embedding features.

Out-Scene has 2,688 images in 8 groups. For each image, we extract 512-D GIST feature, 432-D color moment feature, 256-D HOG feature and 48-D LBP feature.

NoisyMNIST contains 30,000 images. It is generated by adding the white Gaussian noise, the motion blur and a combination of additive white Gaussian noise and reduced contrast to MNIST dataset.

Reuters consists of 18,758 documents in 6 categories. Each document is written in five different languages, including English, French, German, Italian and Spanish.

For validation of the proposed framework, we compare our approach with several classical and state-of-the-art algorithms in the multi-view semi-supervised classification task, including

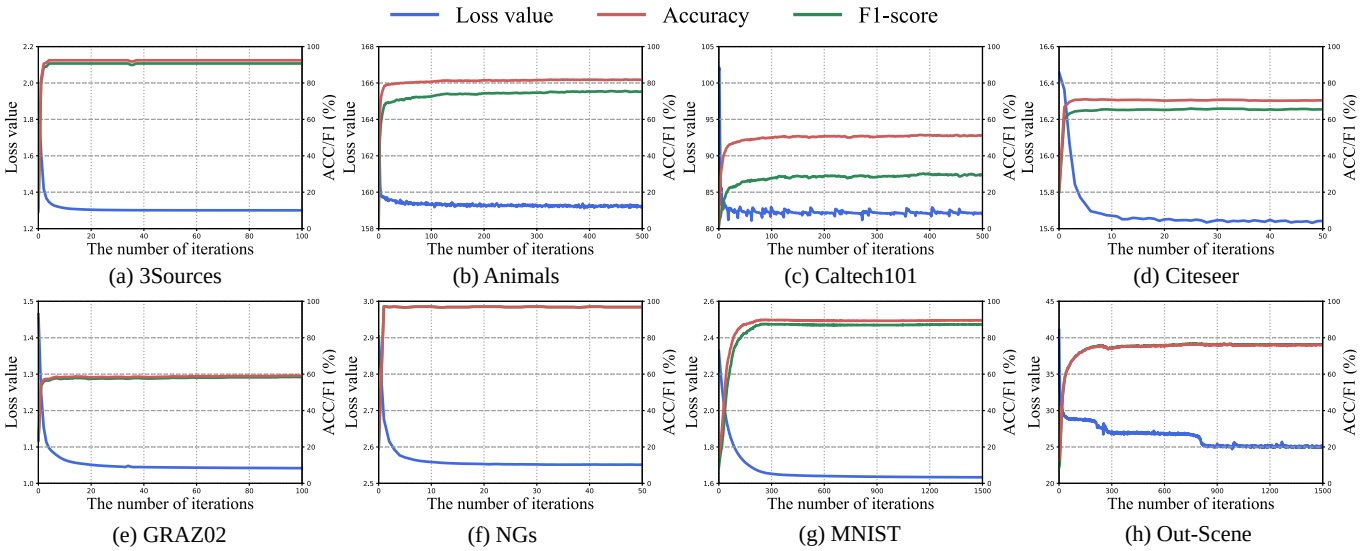


Fig. 2: Curves of train loss value (blue), testing accuracy (red) and F1-score (green) on all datasets (with 5% labeled samples).

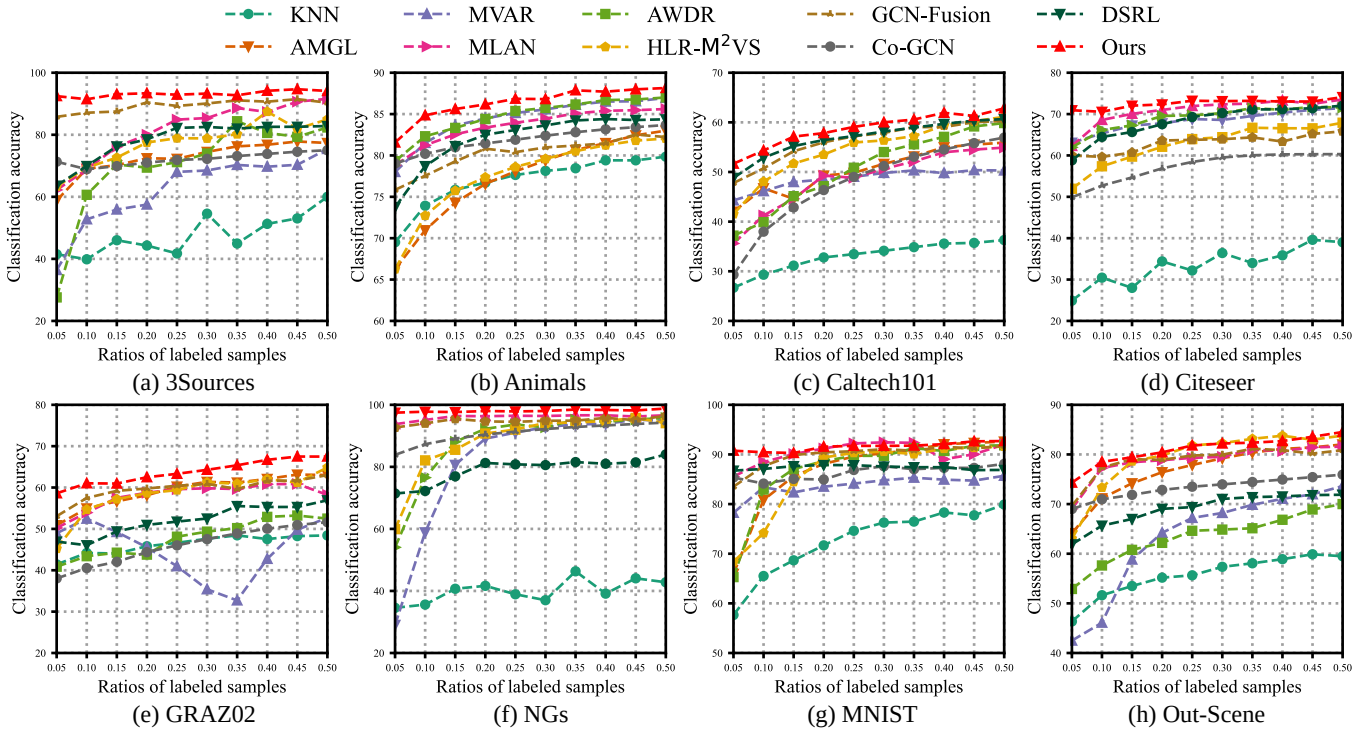


Fig. 3: The various performance (accuracy) of all compared methods on all test datasets. The ratio of labeled samples ranges in $\{0.05, 0.10, \ldots, 0.50\}$.

KNN, AMGL [44], MVAR [45], MLAN [41], AWDR [46], HLR-M^{2}VS [47], GCN-Fusion [23], Co-GCN [40] and DSRL [29]. The description of these methods and some detailed settings are given below.

KNN is a classical algorithm that utilizes k nearest training samples to conduct classification. The number of neighbors k is selected from the set $\{1, 3, 5, 7, 9\}$.

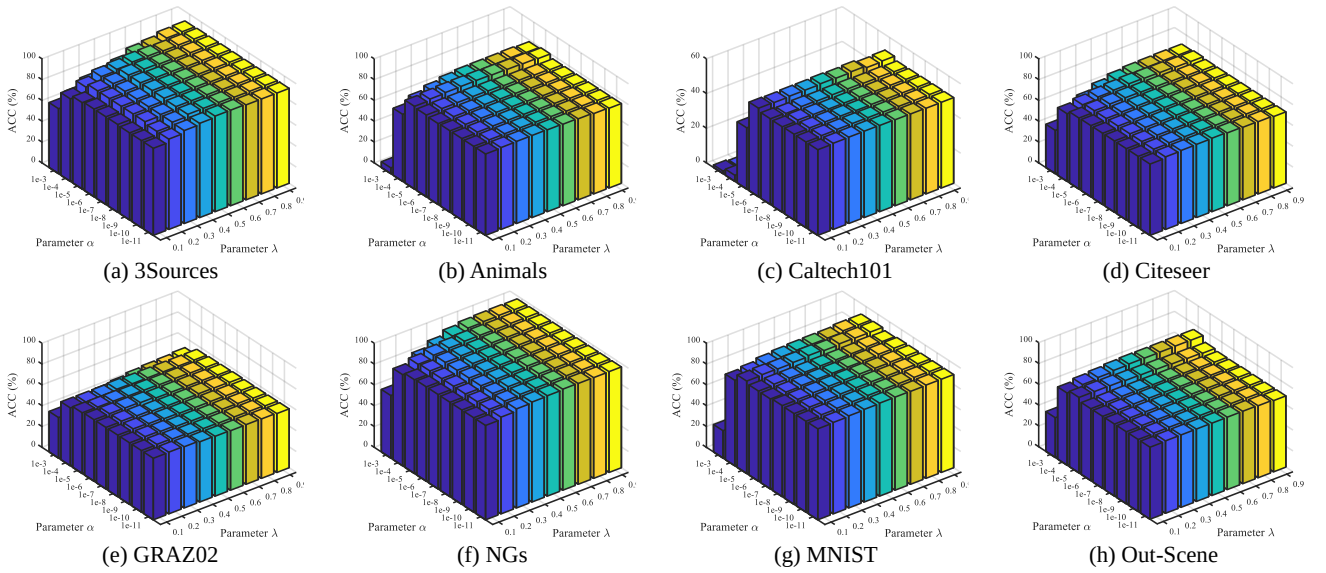
AMGL is a framework that automatically learns an optimal weight for each view. Note that it is parameter-free.

MVAR combines weighted regression based losses computed by $\ell_{2,1}$ norm in all views. We tune the trade-off weight for each view as $\lambda = 1000$, and the redistribution parameter r over views is fixed as 2.

MLAN integrates clustering and semi-supervised classification into a unified framework by learning local structure representations. The number of adaptive neighbors is tuned in $[1, 10]$ to find the best performance on diverse datasets.

AWDR is a supervised multi-view learning algorithm assigning the learned optimal weights to features from various views. The trade-off parameter λ is set as 1.0.

HLR-M^{2}VS explores the global consensus constraint and local geometrical regularization for nonlinear subspace learn-



(a) 3Sources (b) Animals (c) Caltech101 (d) Citeseer

(e) GRAZ02 (f) NGs (g) MNIST (h) Out-Scene

Fig. 4: Parameter sensitivity demonstration of the proposed method on datasets with accuracy.

TABLE III: Ablation study (mean% and standard deviation%) of the proposed method on all datasets, where ON is Orthogonal Normalization and the experiment is conducted with 5% labeled samples.

Datasets	$\mathcal{L}_{rl}$	ON	ACC	F1	Datasets	$\mathcal{L}_{rl}$	ON	ACC	F1
3Sources			87.0 (0.9)	79.6 (0.9)	GRAZ02			56.5 (0.9)	56.7 (0.9)
	✓		87.9 (1.8)	84.5 (2.7)		✓		57.4 (0.8)	57.2 (0.7)
		✓	91.8 (0.2)	84.1 (1.7)			✓	57.2 (1.1)	57.5 (1.0)
	✓	✓	**92.5 (0.2)**	**89.7 (0.5)**		✓	✓	**58.6 (0.6)**	**58.7 (0.7)**
Animals			74.1 (0.2)	68.8 (0.3)	NGs			96.4 (0.2)	96.4 (0.2)
	✓		79.7 (0.2)	71.7 (0.2)		✓		96.9 (0.1)	96.9 (0.1)
		✓	78.6 (0.0)	73.8 (0.0)			✓	96.6 (0.1)	96.5 (0.1)
	✓	✓	**81.6 (0.1)**	**74.8 (0.2)**		✓	✓	**97.5 (0.1)**	**97.5 (0.1)**
Caltech101			40.4 (0.6)	21.2 (1.3)	MNIST			82.9 (3.3)	78.9 (3.2)
	✓		48.1 (0.2)	25.6 (0.6)		✓		88.3 (0.7)	86.4 (1.1)
		✓	48.3 (0.0)	26.6 (0.1)			✓	89.2 (0.2)	86.9 (0.2)
	✓	✓	**51.7 (0.2)**	**30.7 (0.2)**		✓	✓	**90.8 (0.3)**	**88.6 (0.2)**
Citeseer			65.3 (0.6)	60.7 (0.7)	Out-Scene			69.8 (2.5)	69.7 (4.1)
	✓		66.4 (1.7)	60.5 (1.2)		✓		71.1 (1.4)	70.6 (1.5)
		✓	70.4 (0.0)	65.5 (0.0)			✓	70.4 (0.2)	69.8 (0.2)
	✓	✓	**70.9 (0.1)**	**65.6 (0.3)**		✓	✓	**74.3 (0.3)**	**74.5 (0.3)**

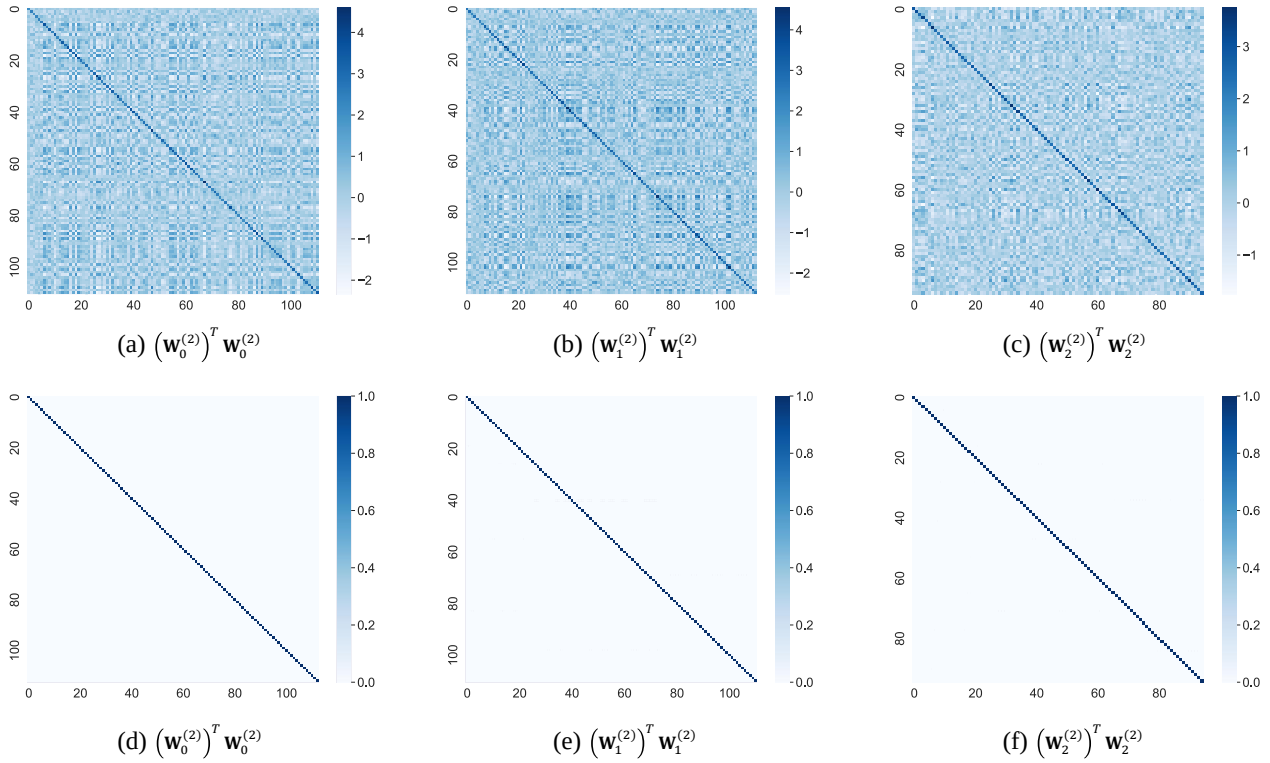
ing simultaneously. We select the weighted factors as $\lambda_1 = 0.2$ and $\lambda_2 = 0.4$.

GCN-Fusion is based on the GCN model that copes with multi-view data. We construct the average adjacency matrix for model initialization and utilize a 2-layer GCN in our experiments, with the learning rate as 0.001.

Co-GCN is a GCN-based method exploiting the graph information from multiple views through an adaptive combination of graph Laplacian matrices. Note that the settings of the graph convolutional layers and learning rate are the same as those in GCN fusion.

DSRL is an end-to-end framework, which builds a neural network with multiple blocks containing learnable activation functions. The number of layers is fixed as 10.

As for the proposed IMvGCN, Adam optimizer is applied to update the learnable parameters, and the learning rate is set to 1×10^{-2}. A 3-layer neural network is employed in the framework. As we stop training when the loss converges, the maximum number of iterations of each dataset varies. The sizes of the hidden units are determined by the feature dimensions of datasets, and the dimensions of output representations are divided by 4 layer by layer. The hyperparameter λ is fixed as 0.5 for all the datasets. Due to the fact that the loss term $\mathcal{L}_{rl}$ is usually much larger than the cross-entropy loss, the balance hyperparameter α is selected in the range $\{1 \times 10^{-4}, 1 \times 10^{-5}, 1 \times 10^{-6}\}$. Besides, we adopt $tanh(\cdot)$ as the activation function of IMvGCN in all the experiments. For the KNN algorithm used to construct the adjacency matrices, hyperparameter k takes values ranging from 5 to 15. In this paper, the proposed framework is implemented by PyTorch platform and runs on the computer with AMD R9-5900X CPU, Nvidia RTX 3060 GPU, Nvidia RTX 3090 GPU (NoisyMNIST and Reuters) and 72G RAM.



(a) $\left(\mathbf{w}_0^{(2)}\right)^T \mathbf{w}_0^{(2)}$ (b) $\left(\mathbf{w}_1^{(2)}\right)^T \mathbf{w}_1^{(2)}$ (c) $\left(\mathbf{w}_2^{(2)}\right)^T \mathbf{w}_2^{(2)}$

(d) $\left(\mathbf{w}_0^{(2)}\right)^T \mathbf{w}_0^{(2)}$ (e) $\left(\mathbf{w}_1^{(2)}\right)^T \mathbf{w}_1^{(2)}$ (f) $\left(\mathbf{w}_2^{(2)}\right)^T \mathbf{w}_2^{(2)}$

Fig. 5: Visualization of $(\mathbf{W}_v^{(l)})^T \mathbf{W}_v^{(l)}$ in some layers of the framework, where each $\mathbf{W}_v^{(l)}$ is learned on 3Sources. (a), (b) and (c) are without the orthogonal normalization, while (d), (e) and (f) are with it.

B. Multi-view Semi-supervised Classification

Performance Comparison: We conduct substantial experiments comparing IMvGCN with the selected nine approaches in multi-view semi-supervised classification tasks. Figure 2 illustrates the curves of IMvGCN in terms of loss values, accuracy and F1-score during the training procedure. It points out that the loss values of IMvGCN converge rapidly, and both accuracy and F1-score maintain stable after convergence. Performance of all the methods with 1% and 5% labeled samples is reported in Table II, which means only 1% or 5% labels are used to calculate the loss during training, and the rest 99% or 95% samples are used to verify the performance. For ease of observation, we highlight the best and the second best results in red and blue, respectively.

As can be seen, the shallow and deep models show advantages on different datasets respectively, while IMvGCN gains a large advantage over the compared algorithms on all datasets. Most of the selected methods are graph-oriented methods, where AMGL, MLAN and HLR-M^{2}VS are shallow models while GCN-Fusion and Co-GCN are both GCN-based methods, and they outperform the other methods on most of the datasets, suggesting the superiority of the graph-oriented methods. In particular, the two GCN-based methods exhibit remarkable performance. Nevertheless, the proposed framework still significantly surpasses these GCN-based methods, which reveals that the proposed flexible filter is more effective and gains stronger generalization capability than the original filter. It can also be noticed that IMvGCN performs stably with only 1% labeled samples, even surpasses the second-place GCN-Fusion by more than 30% on 3Sources, which can be attributed to the reconstruction error with Laplacian embedding loss. When the labels are rare, the cross-entropy loss can only rely on limited supervised information, while the reconstruction error and Laplacian embedding loss do not require supervision and can explore the feature space well to compensate for the deficiency of cross-entropy loss. Similar conclusions can be drawn from the observations of Figure 3, where we illustrate the accuracy of IMvGCN with various ratios of labeled samples (the figure with respect to F1-score is provided in Appendix C). Besides, the performance of all methods varies considerably at low ratios but tends to be comparable as the ratio increases. And IMvGCN gains a remarkable edge over other methods when the supervised information is scarce, which is quite worthwhile. In a nutshell, the experimental results indicate that the established connection between the multi-view learning problem and GCN endows IMvGCN with strong robustness.

Parameter Sensitivity: The effects of hyperparameters α and λ are illustrated in Figure 4, and the results with respect to F1-score is provided in Appendix C. According to the observations, it can be concluded that the selection of these two hyperparameters has a remarkable influence on classification performance. More specifically, accuracy and F1-score are at lower levels when the parameter α is too large, while they are more stable when parameter α is less than

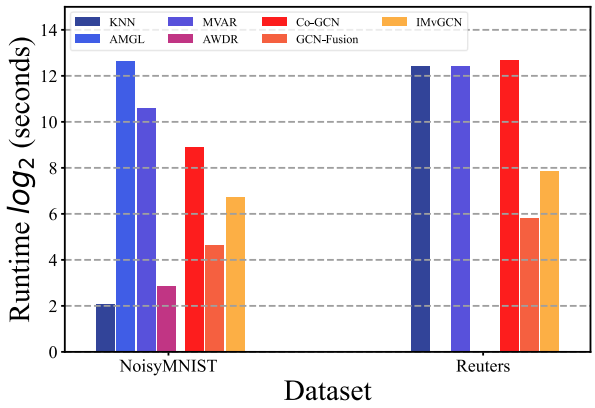


Fig. 6: All methods' runtime on two large-scale datasets (KNN, AMGL, MVAR, MLAN, AWDR and HLR-M^{2}VS are run on CPU, and the other four are run on GPU). Some methods encounter errors and the runtimes can not be plotted.

1×10^{-4}. Nonetheless, a small value of parameter α also leads to performance degradation, thus the suggested value for α is between 1×10^{-6} and 1×10^{-4}. In addition, the performance of the model is relatively stable with different values of parameter λ. In general, smaller values of λ result in poorer performance, and peak performance is usually achieved with moderate or large values of λ in $[0.5, 0.8]$.

Ablation Study: We also validate the effectiveness of the proposed orthogonal normalization approach and loss term $\mathcal{L}_{rl}$ on all datasets. Table III records the experimental results of ablation study. It can be observed that by simultaneously using orthogonal normalization and loss $\mathcal{L}_{rl}$, the proposed method achieves the best performance in terms of both accuracy and F1-score. Besides, using one of the two individually results in a smaller performance gain. Observations indicate that orthogonal normalization and $\mathcal{L}_{rl}$ improve the performance and robustness of the framework substantially.

Further, we visualize $(\mathbf{W}_v^{(l)})^T\mathbf{W}_v^{(l)}$ with and without orthogonal normalization of the weight $\mathbf{W}_v^{(l)}$. Figure 5 illustrates that the diagonal elements of the matrix $(\mathbf{W}_v^{(l)})^T\mathbf{W}_v^{(l)}$ with normalized $\mathbf{W}_v^{(l)}$ are approximately 1 while the values at other positions are approximately 0. In other words, the experimental results validate the effectiveness of the proposed orthogonal normalization approach. Moreover, the observations also indicate that the orthogonal $\mathbf{W}_v^{(l)}$ satisfies $\left\|(\mathbf{W}_v^{(l)})_i\right\| = 1$ where $(\mathbf{w}_v^{(l)})_i$ is the i-th row vector of $\mathbf{W}_v^{(l)}$, so it stabilizes the forward and backward propagation of neural networks. As all weights trained on all datasets can achieve similar results, we simply exhibit an example of $\mathbf{W}_0^{(2)}$, $\mathbf{W}_1^{(2)}$ and $\mathbf{W}_2^{(2)}$ trained by the proposed framework on 3Sources dataset, which are relatively small and appropriate for visualization.

Runtime Comparison: Fig. 6 collects the training time (i.e., runtime) of all methods on two selected large datasets, i.e., NoisyMNIST (large sample size) and Reuters (large feature size). Note that, the first six methods are implemented by Matlab and run on CPU, called shallow methods, and the other four are implemented by Pytorch and run on GPU, called deep methods. So the comparison between the two groups is only for reference. Besides, some methods fail to work with the out-of-memory error so we can not plot the columns corresponding to them. As exhibited in the figure, AMGL and MVAR run more slowly than other shallow methods, and KNN is greatly influenced by the dimensionality. Among deep methods, DSRL suffers from out-of-time error, taking over 24 hours, and Co-GCN's runtime is similar to the shallow method. Considering it is accelerated by GPU, the result is undesired. We find that IMvGCN requires much less time than Co-GCN and DSRL. It has moderately low time complexity while performing better than competitors.

V. Conclusion

In this paper, we propose an end-to-end framework dubbed IMvGCN, which constructs a flexible graph filter and introduces the orthogonal normalization to further improve the interpretability of neural networks. We first design a multi-view learning problem, where reconstruction error and Laplacian embedding loss are integrated and extended to the multi-view case. Inspired by this problem, a series of derivations are conducted to establish the mathematical connection between multi-view learning and GCN. In addition, the absence or relaxation of constraints often limits the interpretability of deep model, and we propose an orthogonal normalization method to achieve strict orthogonality in deep learning, which is general enough to be adopted in more other models rather than just IMvGCN. Finally, we apply IMvGCN to multi-view semi-supervised classification tasks and validate its effectiveness through comprehensive experiments.

Several potential topics about multi-view learning and GCN remain to be explored. Most studies focus on feature fusion, but it is critical to aggregate topological information across views to obtain optimal consistent connectivity relationships, especially for GCN-based models. In the future, we will devote ourselves to further extending IMvGCN and investigating a multi-view learning framework with learnable topology.

References

[1] Y. Chen, X. Xiao, and Y. Zhou, "Jointly learning kernel representation tensor and affinity matrix for multi-view clustering," *IEEE Transactions on Multimedia*, vol. 22, no. 8, pp. 1985–1997, 2020.

[2] C. Chen, J. Xin, Y. Wang, L. Chen, and M. K. Ng, "A semisupervised classification approach for multidomain networks with domain selection," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 1, pp. 269–283, 2019.

[3] S. Huang, Y. Zhang, L. Fu, and S. Wang, "Learnable multi-view matrix factorization with graph embedding and flexible loss," *IEEE Transactions on Multimedia*, 2022. doi:10.1109/TMM.2022.3157997.

[4] X. Liu, M. Li, C. Tang, J. Xia, J. Xiong, L. Liu, M. Kloft, and E. Zhu, "Efficient and effective regularized incomplete multi-view clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2634–2646, 2021.

[5] C. Tang, X. Liu, X. Zheng, W. Li, J. Xiong, L. Wang, A. Y. Zomaya, and A. Longo, "Defusionnet: Defocus blur detection via recurrently fusing and refining discriminative multi-scale deep features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 2, pp. 955–968, 2022.

[6] C. Tang, X. Liu, P. Wang, C. Zhang, M. Li, and L. Wang, "Adaptive hypergraph embedded semi-supervised multi-label image annotation," *IEEE Transactions on Multimedia*, vol. 21, no. 11, pp. 2837–2849, 2019.

[7] J. Huang, W. Yan, T. H. Li, S. Liu, and G. Li, "Learning the global descriptor for 3-d object recognition based on multiple views decomposition," *IEEE Transactions on Multimedia*, vol. 24, pp. 188–201, 2022.

[8] J. Xu, X. Zhang, W. Li, X. Liu, and J. Han, "Joint multi-view 2d convolutional neural networks for 3d object classification," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pp. 3202–3208, 2020.

[9] B. Jiang, Z. Zhou, X. Wang, J. Tang, and B. Luo, "cmsalgan: RGB-D salient object detection with cross-view generative adversarial networks," *IEEE Transactions on Multimedia*, vol. 23, pp. 1343–1353, 2021.

[10] S. Du, Z. Liu, Z. Chen, W. Yang, and S. Wang, "Differentiable bi-sparse multi-view co-clustering," *IEEE Transactions on Signal Processing*, vol. 69, pp. 4623–4636, 2021.

[11] C. Tang, X. Liu, X. Zhu, E. Zhu, Z. Luo, L. Wang, and W. Gao, "CGD: multi-view clustering via cross-view graph diffusion," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pp. 5924–5931, 2020.

[12] Y. Chen, S. Wang, C. Peng, Z. Hua, and Y. Zhou, "Generalized nonconvex low-rank tensor approximation for multi-view subspace clustering," *IEEE Transactions on Image Processing*, vol. 30, pp. 4022–4035, 2021.

[13] L. Fu, Z. Chen, Y. Chen, and S. Wang, "Unified low-rank tensor learning and spectral embedding for multi-view subspace clustering," *IEEE Transactions on Multimedia*, pp. 1–14, 2022. doi=10.1109/TMM.2022.3185886.

[14] Z. Wan, C. Zhang, P. Zhu, and Q. Hu, "Multi-view information-bottleneck representation learning," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pp. 10085–10092, 2021.

[15] J. Liu, X. Liu, Y. Zhang, P. Zhang, W. Tu, S. Wang, S. Zhou, W. Liang, S. Wang, and Y. Yang, "Self-representation subspace clustering for incomplete multi-view data," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 2726–2734, 2021.

[16] F. Nie, G. Cai, J. Li, and X. Li, "Auto-weighted multi-view learning for image clustering and semi-supervised classification," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1501–1511, 2018.

[17] X. Li, H. Zhang, R. Wang, and F. Nie, "Multiview clustering: A scalable and parameter-free bipartite graph fusion method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 330–344, 2022.

[18] X. Jia, X. Jing, X. Zhu, S. Chen, B. Du, Z. Cai, Z. He, and D. Yue, "Semi-supervised multi-view deep discriminant representation learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 7, pp. 2496–2509, 2021.

[19] S. Wang, Z. Wu, Y. Chen, and Y. Chen, "Beyond graph convolutional network: An interpretable regularizer-centered optimization framework," *arXiv preprint arXiv:2301.04318*, 2023.

[20] P. Sánchez-Martín, M. Rateike, and I. Valera, "VACA: designing variational graph autoencoders for causal queries," in *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, pp. 8159–8168, 2022.

[21] J. Tang, X. Shu, R. Yan, and L. Zhang, "Coherence constrained graph LSTM for group activity recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 2, pp. 636–647, 2022.

[22] X. Shu, L. Zhang, Y. Sun, and J. Tang, "Host-parasite: Graph lstm-in-lstm for group activity recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 663–674, 2021.

[23] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the 5th International Conference on Learning Representations*, pp. 1–14, 2017.

[24] Y. Gao, Y. Feng, S. Ji, and R. Ji, "Hgnn+: General hypergraph neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3181–3199, 2023.

[25] K. Sun, Z. Lin, and Z. Zhu, "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*, pp. 5892–5899, 2020.

[26] D. Bo, X. Wang, C. Shi, and H. Shen, "Beyond low-frequency information in graph convolutional networks," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pp. 3950–3957, 2021.

[27] J. Sun, J. Zhang, Q. Li, X. Yi, Y. Liang, and Y. Zheng, "Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 5, pp. 2348–2359, 2022.

[28] J. Cheng, Q. Wang, Z. Tao, D. Xie, and Q. Gao, "Multi-view attribute graph convolution networks for clustering," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pp. 2973–2979, 2020.

[29] S. Wang, Z. Chen, S. Du, and Z. Lin, "Learning deep sparse regularizers with applications to multi-view clustering and semi-supervised classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5042–5055, 2022.

[30] L. Huang, X. Liu, B. Lang, A. W. Yu, Y. Wang, and B. Li, "Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pp. 3271–3278, 2018.

[31] Z. Chen, J. Yao, G. Xiao, and S. Wang, "Efficient and differentiable low-rank matrix completion with back propagation," *IEEE Transactions on Multimedia*, pp. 1–14, 2021.

[32] Z. Huang, J. T. Zhou, H. Zhu, C. Zhang, J. Lv, and X. Peng, "Deep spectral representation learning from multi-view data," *IEEE Transactions on Image Processing*, vol. 30, pp. 5352–5362, 2021.

[33] M. Zhu, X. Wang, C. Shi, H. Ji, and P. Cui, "Interpreting and unifying graph neural networks with an optimization framework," in *Proceedings of the Web Conference*, pp. 1215–1226, 2021.

[34] Z. Zhang, C. Chen, Y. Chang, W. Hu, X. Xing, Y. Zhou, and Z. Zheng, "Shne: Semantics and homophily preserving network embedding," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2021.

[35] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, and J. Pei, "Am-gcn: Adaptive multi-channel graph convolutional networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1243–1253, 2020.

[36] Q. Li, Z. Han, and X. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pp. 3538–3545, 2018.

[37] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proceedings of the 36th International Conference on Machine Learning*, pp. 6861–6871, 2019.

[38] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 257–266, 2019.

[39] H. Gao, Z. Wang, and S. Ji, "Large-scale learnable graph convolutional networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1416–1424, 2018.

[40] S. Li, W. Li, and W. Wang, "Co-gcn for multi-view semi-supervised learning," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pp. 4691–4698, 2020.

[41] F. Nie, G. Cai, and X. Li, "Multi-view clustering and semi-supervised classification with adaptive neighbours," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pp. 2408–2414, 2017.

[42] X. Jia, X.-Y. Jing, X. Zhu, S. Chen, B. Du, Z. Cai, Z. He, and D. Yue, "Semi-supervised multi-view deep discriminant representation learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 7, pp. 2496–2509, 2020.

[43] S. P. Smith, "Differentiation of the cholesky algorithm," *Journal of Computational and Graphical Statistics*, vol. 4, no. 2, pp. 134–147, 1995.

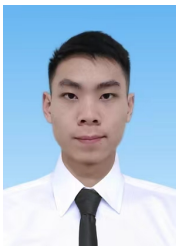
[44] F. Nie, J. Li, and X. Li, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pp. 1881–1887, 2016.

[45] H. Tao, C. Hou, F. Nie, J. Zhu, and D. Yi, "Scalable multi-view semi-supervised classification via adaptive regression," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4283–4296, 2017.

[46] M. Yang, C. Deng, and F. Nie, "Adaptive-weighting discriminative regression for multi-view classification," *Pattern Recognition*, vol. 88, pp. 236–245, 2019.

[47] Y. Xie, W. Zhang, Y. Qu, L. Dai, and D. Tao, "Hyper-laplacian regularized multilinear multiview self-representations for clustering and semisupervised learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 2, pp. 572–586, 2020.

Zhihao Wu received his B.S. degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China in 2021. He is currently pursuing the M.S. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. He is also currently visiting Shenzhen Research Institute of Big Data and Chinese University of Hong Kong (Shenzhen), Shenzhen, China. His current research interests include machine learning, multi-view learning and graph neural networks.



Xincan Lin received his B.S. degree from the College of Computer Science and Data Science, Fuzhou University, Fuzhou, China in 2021. He is currently pursuing the M.S. degree with the College of Computer Science and Data Science, Fuzhou University, Fuzhou, China. His current research interests include machine learning, multiview learning and ccontrastive learning.

Zhenghong Lin received his B.S. degree from the College of Computer and Data Science, Fujian Agriculture and Forestry University, Fuzhou, China in 2019. He is currently pursuing the Ph.D. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His current research interests include active learning, machine learning, deep learning, graph neural networks, computer vision and recommender systems.

Zhaoliang Chen received his B.S. degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China in 2019. He is currently pursuing the Ph.D. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. He is also currently visiting Faculty of Computer Science, University of Vienna, Vienna, Austria. His current research interests include machine learning, deep learning, graph neural networks and recommender systems.

Yang Bai received the Ph.D. degree from University of Electronic Science and Technology of China, Chengdu, China, in 2022. She is currently an Associate Researcher with the School of Cybersecurity, Chengdu University of Information Technology. Her research interests include machine learning, information security and privacy-preserving.

Shiping Wang received his Ph.D. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China in 2014. He worked as a research fellow in Nanyang Technological University, Singapore from August 2015 to August 2016. He is currently a professor with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His research interests include machine learning, computer vision and granular computing.