

Sparsity-induced Graph Convolutional Network for Semi-supervised Learning

Jianhang Zhou, *Student Member, IEEE*, Shaoning Zeng, *Member, IEEE*, and Bob Zhang, *Senior Member, IEEE*

Abstract—The graph representation (GR) in a data space reveals the intrinsic information as well as the natural relationships of data, which is regarded as a powerful means of representation for solving the semi-supervised learning problem. To effectively learn on a pre-defined graph with both labeled data and unlabeled data, the graph convolutional network (GCN) was proposed and has attracted a lot of attention due to its high-performance graph-based feature extraction along with its low computational complexity. Nevertheless, the performance of GCNs is highly sensitive to the quality of the graph, meaning with high probability the GCNs will achieve poor performances on a badly-defined graphs. In numerous real-world semi-supervised learning problems, the graph connecting each entity in the data space implicitly exists so that there is no naturally pre-defined graph in these problems. To overcome the issues, in this paper, we apply unified graph representation (GR) techniques and graph convolutional (GC) networks in a framework that can be implemented in semi-supervised learning problems. To achieve this framework, we propose sparsity-induced graph convolutional network (SIGCN) for semi-supervised learning. SIGCN introduces the sparsity to formulate significant relationships between instances by constructing a newly-proposed L_0 -based graph (termed as the sparsity-induced graph), before applying graph convolution to capture the high-quality features based on this graph for label propagation. We prove and demonstrate the feasibility of the unified framework as well as effectiveness in capturing features. Extensive experiments and comparisons were performed to show the proposed SIGCN obtains a state-of-the-art performance in the semi-supervised learning problem.

Impact Statement—Semi-supervised learning is a widely-studied and popular learning paradigm in the Artificial Intelligence (AI) area, which learns from both labelled data and unlabelled data. Furthermore, graph representation is a classical way to perform semi-supervised learning by building relationships in the data space. At present, there are no pervasive graph learning frameworks in the sample-level for semi-supervised tasks, since in many cases the sample-level relationship does not naturally exist. To address this issue, this paper proposes the Sparsity-Induced Graph Convolutional Network (SIGCN), establishing an effective and efficient graph-based semi-supervised classifier. SIGCN achieves a state-of-the-art performance in semi-supervised learning when compared with 15 leading classifiers from different categories. The proposed methodology provides a direct drop-in solution for tackling different AI applications under the semi-supervised scenario.

Index Terms—Graph representation, Graph Convolutional Networks, Semi-supervised learning, L_0 -norm, Sparsity.

I. INTRODUCTION

Corresponding author: Bob Zhang (e-mail: bobzhang@um.edu.mo).

J. Zhou, S. Zeng, and B. Zhang, are with the Pattern Analysis and Machine Intelligence Research Group, Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mails: yc07424, bobzhang@um.edu.mo).

S. Zeng is with the School of Computer Science and Engineering, Huizhou University, Guangdong 516007, China (e-mail: zsn@outlook.com).

SEMI-supervised learning is a machine learning paradigm that learns jointly on both labelled data and unlabelled data [1]. As a broadly-speaking consensus in the artificial intelligence community, few accessible data from the natural world instinctively comes with labels [2], [3]. To deal with the laborious data annotation work, the semi-supervised learning attracts much attention recently [4], [5]. In the conventional supervised learning and un-supervised learning, the learners absorb knowledge only from the labelled data or unlabelled data [6], [7]. The semi-supervised learning leverages the knowledge from both sides so as to exceed the performance of supervised and unsupervised learning paradigm. The graph-based methods are a popular family of semi-supervised learning methods, which assumes that the connected nodes in a graph share the same class label [1]. In this case, the knowledge from the labelled data is able to be propagated to the unlabelled data. Based on this framework, various classification methods are boosted [8], [9], reflecting the information from the graph structure and the way of label propagation truly benefit the classification. Although the graph-based methods provide a theoretically-sound solution to the semi-supervised learning, the typically Euclidean-based similarity is hard to measure the neighborliness among instances in the high-dimensional data (e.g., image, voice-print, and other multi-media data, etc.) [1].

The Graph Convolutional Network (GCN) is a neural network structure to learn the representation of nodes in the given graph [10] by performing graph convolution operation. Different from the traditional graph-based semi-supervised learning method, the GCN learns through a single supervised loss without the explicit graph-based Laplacian regularization term, and propagates knowledge of nodes via the gradient information distributed on graph convolution filters of multiple layers. Recent works and applications show the GCN has promising performance on processing the relational data, such as social networks [11], recommendation system [12], biomedical prediction [13]. Noted that in the above application scenarios, the correlation among data samples is explicit, which means there is a graph naturally established. However, the graph implicitly exists in plenty of pattern recognition tasks, making the graph representation on the raw data a under-determined problem. Many related works construct the graph for pattern recognition in terms of the Euclidean-based similarities [8]. And they may not be able to exactly measure the similarity in the high-dimension data [1]. To deal with this problem, we introduce the sparsity for the graph representation to construct a robust and explainable graph while preserving the homogenous relationship between data samples.

The sparsity is a critical property that is capable of digging

out the good intrinsic structures from the complex data [14]. From the perspective of neural science, the natural vision system always aims to represent the visual information via minimal redundancy [15]. By imposing sparsity in the visual sensory signal, the information representation becomes compact, which concentrates on the significant components that are informative to reconstruct the intrinsic information from the observation. With this point of view, the sparse representation makes big success in the pattern recognition and computer vision field [16], [17]. The sparsity in this linear representation will enforce far greater weights to the samples from the correct class and zero weight to the samples from the incorrect classes [17]. This reflects the sparsity helps to exploit the intrinsic information from the homogeneous samples and prohibit heterogeneous samples from the representation. To perform classification, effective classifier sparse representation-based classifier SRC [16] is proposed, which utilizes the nearest subspace classification strategy [18] to decide the label of test sample based on the sparse representation reconstruction. Besides this, the SARC [19] adopted the sparsity to the L_2 -based representation and achieve the performance enhancement. These convey the message that the sparsity is able to express homogeneous knowledge in a low-dimensional space from the high-dimensional data. To obtain the sparsest solution in a linear combination, the L_0 -minimization is applied as it should, while usually L_1 -minimization is utilized in practice due to its convexity property [16]. The surrogate L_1 -norm also contributes to construct robust graph for pattern analysis [17], [20]. Even though the l_1 -graph shows robustness and datum-adaptive characteristic in image analysis [20], it is an approximation of sparsest L_0 minimization, which may lose property of sparsity to some extent [21].

To deal with the under-determined graph representation problem in different data spaces, in this paper, we propose the Sparsity-Induced Graph Convolutional Network to perform semi-supervised learning, termed as SIGCN. In SIGCN, the property of sparsity is introduced to the graph learning in order to construct a robust and explainable graph for various kinds of data. Since the premise condition of GCN is a well-established graph, where all nodes share the same class label and are connected, the sparsity has the ability to enforce the homogenous nodes in the learned graph to be connected even in the high-dimensional data space. We denote the learned graph as the Sparsity-Induced graph (SIgraph), because the learned graph leverages both homogeneous knowledge from the sparsity information and neighborly information in the neighborliness between data samples. Different from previously proposed L_1 -graphs, the SIgraph imposes a stronger sparsity to maintain the edges between the data samples by utilizing L_0 -minimization. To implement this, we establish the objective function of SIGCN with two components: supervised loss and L_0 -penalty. We prove the feasibility of implementing a SIgraph in the graph convolutional learning framework, while demonstrating the effectiveness of the proposed SIgraph in an intuitive way. The main contributions of this paper lay in the following three aspects:

1) **Pervasive graph learning framework.** Our proposed

method can be considered as a graph learning framework with the ability to learn the graph embedding through graph convolution for different data without a naturally-defined graph. We prove and demonstrate that the proposed methodology can effectively learn the graph embedding.

- 2) **Sparsity-induced graph.** We proposed a novel sparsity-induced graph to establish a robust and explainable graph to reveal the relationship between data samples. The established graph leverages homogenous information as well as the neighborliness of each data sample.
- 3) **Effective and efficient graph-based semi-supervised classifier.** We performed extensive experiments on 12 datasets and compared with 15 methods to show the state-of-the-art classification performance in semi-supervised learning using the proposed SIGCN. Besides this, we also demonstrated that the proposed method has a high computational-efficiency.

The remainder of this paper is organized as follows. In section II, we provide a review of the related works involved in this paper and proposed method. In section III, we present the methodology proposed in this paper and provide its theoretical proof of feasibility as well as an explanation for the rationale of the method. Following this, we show the simulation, experimental results, comparisons, and provide a discussion with respect to the proposed method in section IV. Finally, in section V we summarize this paper.

II. RELATED WORK

A. Semi-supervised Learning

The semi-supervised learning is a machine learning task that learns from both labelled data and unlabelled data [1]. Given a dataset $X = \{x_1, x_2, \dots, x_n\}$ contains a subset of labelled data $X^l = \{x_1^l, x_2^l, \dots, x_n^l\}$ and unlabelled data $X^u = \{x_1^u, x_2^u, \dots, x_n^u\}$, the semi-supervised learning aims to learn a more effective learner $f : X \rightarrow Y$ by collectively utilizing labelled data X^l and unlabelled data X^u , which outperforms the learners $f^l : X^l \rightarrow Y^l$ of only using X^l or $f^u : X^u \rightarrow Y^u$ of only using X^u . There are two main branches for addressing the semi-supervised learning problem: inductive learning [22] and transductive learning [23]. The inductive learning trains a model $f^l : X \rightarrow Y$ according to the labelled data X^l that has an ability to predict the unobserved unlabelled data. The transductive learning learns the label of unlabelled data when training the model $f : X \rightarrow Y$. The graph-based learning method [24] is a representative semi-supervised learning method, which has implementation in inductive learning paradigm as well as transductive learning paradigm. The main idea of the graph-based learning method is to construct a graph that represents relationships among instances in the dataset X for predicting unlabelled data X^u with the following loss function:

$$\begin{aligned} \mathcal{L} &= \sum_{i=1}^p \mathcal{L}_{\infty}(y_i, f(x_i)) + \gamma \sum_{i=1}^n \sum_{j=1}^n \eta_{ij} \|f(x_i) - f(x_j)\|^2 \\ &= \sum_{i=1}^{n_l} \mathcal{L}_{\infty}(y_i, f(x_i)) + \gamma f(X)^T \Delta f(X) \end{aligned} \quad (1)$$

where $\mathcal{L}_\infty(\cdot, \cdot)$ is the supervised loss function, γ is the weighting factor, η is the similarity matrix whose entry η_{ij} represents the similarity between instance x_i and x_j , $\Delta = \eta - D$ is the graph Laplacian matrix, $D_{ii} = \sum_{j=1}^n \eta_{ij}$. The label propagation [25] learns by propagating label information through the edges starting from the labelled data while clamping the ground truth label of labelled data. The local and global consistency (LGC) method [23] makes the prediction under the assumption of local and global consistency in order to alleviate the influence of noisy labels, which means the both spatial and structural neighbors should have a high probability to be assigned the same label.

B. Graph Convolutional Network

The Graph Convolutional Network (GCN) [10] is a graph-based neural network model that learns representation of nodes while considering features of nodes and graph structure. For a given graph signal $x_G \in \mathbb{R}^N$ and pre-defined graph $G = (\varrho, \varsigma)$, the GCN adopts graph convolution via filter g to generate representation of each node:

$$\begin{aligned} g * x_G &= F^{-1}(F(g) \odot F(x_G)) \\ &= U(U^T g \odot U^T x_G) \\ &= U g_w U^T x_G \end{aligned} \quad (2)$$

where $g_w = \text{diag}(U^T g)$ is set as the graph convolutional filter, \odot is the hadamard product, U is the matrix of the eigenvectors of graph Laplacian matrix $L = I - D^{-\frac{1}{2}} \eta D^{-\frac{1}{2}}$ containing graph information of G . When the $x_G \in \mathbb{R}^{n \times r}$ has r channels, the $g_w = W$ becomes a matrix of parameters that trained in each layer of the network. By performing layer-wise propagation on the graph structure with the following rule:

$$T^{m+1} = \sigma \left(\hat{D}^{-\frac{1}{2}} \hat{\eta} \hat{D}^{-\frac{1}{2}} T^m W^m \right) \quad (3)$$

where T^m is the activation matrix of m^{th} hidden layer of the GCN, W^m is the weight matrix of the m^{th} layer, $\sigma(\cdot)$ is the activate function, $\hat{\eta} = \eta + I_N$, $\hat{D}_{ii} = \sum_{j=1}^n \hat{\eta}_{ij}$ is the diagonal matrix containing degrees of each node. The loss function of the GCN is a simplified version of Eq. 1, which doesn't contain the explicit graph Laplacian regularization term. In [10], the GCN applies cross-entropy error as the loss function:

$$\mathcal{L} = - \sum_{i=1}^{n_l} \sum_{j=1}^c \pi_j(y_i) \ln Z_{ij} \quad (4)$$

C. Sparsity

The sparsity is widely discussed and applied in different fields since it helps to exploit the good intrinsic structures from the complex data [14]. For example, the sparse coding is adopted in the neuroscience to interpret the nature of biological vision system [26]. In the information theory area, the compressed sensing theory was built on the sparsity, assuming that the sparse representation is capable of reconstructing a signal economically [27]. For the signal processing [28], the sparsity of signal is beneficial to error correction. The quality of sparse approximation for representing images determines the compression ratio of image compression [29]. In pattern

recognition, linear representation-based classifiers become robust and powerful due to the property of sparsity [16], [30]. Considering a linear representation problem as follows:

$$\begin{aligned} \min \quad & \|\alpha\|_0 \\ \text{subject to} \quad & t = X\alpha + \epsilon \end{aligned} \quad (5)$$

where $t \in \mathbb{R}^N$ is a sample to be represented by all instances in the dataset X , $\alpha \in \mathbb{R}^n$ is the coefficient vector of linear representation, ϵ is the modeling errors. In the above Eq. 5, all the samples from dataset X are utilized to linearly represent the given sample t . Here the sparsity appears when the most of entries in the vector α to be zero. It is a common case that the dimensionality N of the sample to be recovered is smaller than the number of observations n , which makes the above Eq. 5 to be an under-determined linear system. Therefore, the sparse solution of coefficient vector is such an appropriate choice to capture the naturally significant component from the observations while exactly recover the given sample t [29]. This problem can be solved by exhaustive search strategy, however, the computational complexity of this strategy is $O(n^k)$, which is computationally prohibitive. To alternatively address the above discussed NP-hard problem, the greedy strategy algorithms [21] are popular methods to generate sparse solutions that approximate the L_0 -minimization (e.g., Orthogonal Matching Pursuit [31], etc.).

III. SPARSITY-INDUCED GRAPH CONVOLUTIONAL NETWORK

A. Problem definition and objective function

If we introduce the sparsity to the graph convolution operation on a graph signal x_g , it is equivalent to perform graph convolution in the homogenous sub-graph. First, we define the homogenous sub-graph in the following Definition 1:

Definition 1 (Homogenous subgraph). Given a graph $G = \langle V, E \rangle$ organized by all instances of the dataset $V \in \mathbb{R}^{m \times n}$ and their corresponding edge $E \in \mathbb{R}^{n \times n}$, for any vertex V_i , the homogenous subgraph is defined by $G_H^i = \langle \hat{V}_i, \hat{E}_i \rangle$, where $\hat{V}_i \in \mathbb{R}^{m \times \hat{n}_i}$ denotes the vertices from the homogenous class and $\hat{E}_i \in \mathbb{R}^{\hat{n}_i \times \hat{n}_i}$ denotes their corresponding edges. Here, the full homogenous graph G_H is defined as the union set of the homogenous subgraphs of all vertices in the dataset $G_H = G_H^1 \cup G_H^2 \cup \dots \cup G_H^n$.

We made the following Assumption 1 based on the nature of sparsity according to the prior knowledge present in [16]:

Assumption 1 (Property of sparsity in classification). Given an instance x_i , the instances with non-zero coefficients in the sparse linear representation (subgraph linear representation) have a higher probability to be from the homogenous class of x_i .

Accordingly, we define the concept of sparsity-induced adjacency matrix Ω^s in Definition 2:

Definition 2 (Sparse adjacency matrix). Given a collection of data $X \in \mathbb{R}^{m \times n}$, the sparse adjacency matrix $\hat{\Omega}^s \in \mathbb{R}^{n \times n}$ is

defined according to the following formulation:

$$\hat{\Omega}_{ij}^s = \begin{cases} 1, & x_j \in SLR(x_i) \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

where $SLR(x_i)$ represents the function that outputs a set consisting of the instances associated with non-zero coefficients in the subgraph linear representation with respect to each instance x_i in the dataset $X \in \mathbb{R}^{m \times n}$.

We are motivated by the inference of sparsity imposed to the graph embedding via the graph convolution technique summarized in the following Theorem 1:

Theorem 1 (Graph convolution on homogenous subgraph). Considering the adjacency matrix $\Omega^n \in \mathbb{R}^{n \times n}$ where $\Omega_{ij}^n = 1$ when the j^{th} instance $x_j \in neighbors\{x_i\}$, and sparsity-induced adjacency matrix $\hat{\Omega}^s \in \mathbb{R}^{n \times n}$ where $\hat{\Omega}_{ij}^s = 1$ when the j^{th} instance $x_j \in SLR\{x_i\}$ (The $SLR\{\cdot\}$ is the sample set of subgraph linear representation), the graph convolution on a signal x_G of graph $\mathcal{G}_{\hat{\Omega}} = \langle V_{\hat{\Omega}}, E_{\hat{\Omega}} \rangle$ defined by adjacency matrix $\hat{\Omega} = \tilde{\Omega}^n \odot \tilde{\Omega}^s$ is equivalent to the graph convolution on the 1^{st} -order differential signal x_H of full homogenous subgraph \mathcal{G}_H with respect to the dataset.

Proof.

Let $\Omega^n \in \mathbb{R}^{n \times n}$ be the adjacency matrix where $\Omega_{ij}^n = 1$ when the j^{th} instance $x_j \in neighbors\{x_i\}$:

$$\Omega_{ij}^n = \begin{cases} 1, & x_j \in neighbors(x_i, k) \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

where $neighbors(x_i, k)$ is the function that outputs a set consisting of k nearest neighbors of the instance x_i .

Let $\hat{\Omega}^s \in \mathbb{R}^{n \times n}$ be the sparse adjacency matrix as defined in Definition 2 where $\hat{\Omega}_{ij}^s = 1$ when the j^{th} instance $x_j \in SLR(x_i)$. Noticeably, the instance $x_i \in X = [X^l, X^u]$ is the i^{th} sample from the dataset X . Therefore, for the unlabeled data X^u in X , we perform subgraph linear representation on it as normal to connect them with other data samples in X . The $SLR(x_i)$ can be expressed as follows:

$$SLR(x_i) = \{x_j | \alpha_j \neq 0\}. \quad (8)$$

where α is calculated using the following formulation:

$$\begin{aligned} \min \quad & \|\alpha\|_0 \\ \text{subject to} \quad & x_i = \{X \setminus x_i\} \alpha \end{aligned} \quad (9)$$

We denote \mathcal{G} as the graph defined by the adjacency matrix $\hat{\Omega}$ as follows:

$$\begin{aligned} \hat{\Omega} &= (\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n) \\ &= \tilde{\Omega}^n \odot \tilde{\Omega}^s. \end{aligned} \quad (10)$$

Since we would like to perform the graph convolution on the homogenous subgraphs with respect to each sample in the dataset, in the SIGCN, we should make sure each node in the graph is connected to more nodes from the same class as possible. However, when introducing the sparsity for constructing the graph (with Eq. 6), all connected nodes with respect to each single node may not belong to the

same class due to the existing noise or outliers sometimes. According to the analysis in [32] and [33], the neighbors in the sparse linear representation tend to have a higher posterior probability of being from the same class. Therefore, we take the intersection set between the neighbor sets and the sparse linear representation sets to ensure the connected nodes belong to the same class.

Suppose x_G is a graph signal. According to the Eq.2 shown in the section II, the graph convolution on the graph \mathcal{G} can be described as follows:

$$\begin{aligned} g * x_G &= F^{-1}(F(g) \odot F(x_G)) \\ &= U(U^T g \odot U^T x_G) \\ &= U g \odot U^T x_G \\ &\approx \Theta D^{-\frac{1}{2}} \tilde{\Omega} D^{-\frac{1}{2}} x_G \\ &= \Theta D^{-\frac{1}{2}} \tilde{\Omega}^n \odot \tilde{\Omega}^s D^{-\frac{1}{2}} x_G \\ &= \Theta D_n^{-\frac{1}{2}} \tilde{\Omega}^n D_n^{-\frac{1}{2}} D_s^{-\frac{1}{2}} \tilde{\Omega}^s D_s^{-\frac{1}{2}} x_G \\ &= \Theta D_n^{-\frac{1}{2}} \tilde{\Omega}^n D_n^{-\frac{1}{2}} L_s x_G \\ &= \Theta D_n^{-\frac{1}{2}} \tilde{\Omega}^n D_n^{-\frac{1}{2}} x_{G_H} \end{aligned} \quad (11)$$

where the Chebyshev polynomial approximation is used in the 4^{th} row of above equation, $D_n^{ii} = \sum_j \tilde{\Omega}_{ij}^n$, $D_s^{ii} = \sum_j \tilde{\Omega}_{ij}^s$, L_s is the graph Laplacian matrix with respect to the graph \mathcal{G}_s defined by the sparse adjacency matrix Ω^s , $x_{G_H} = L_s x_G$ represents the 1^{st} -order differential signal. Besides this, the Laplacian projection of graph signal x_G makes the node representation of graph compact and separable [34], which means the nodes from the same class tend to have similar embedding.

In terms of Assumption 1, only the coefficients of the homogenous instances in the subgraph linear representation coefficients with respect to the instance x_i will be non-zero. We can find that L_s is an operator that calculates the first-order difference signal of each node on the homogenous graph. Therefore, the $L_s x_G$ can be represented as follows:

$$L_H x_G = \begin{bmatrix} \sum_{i \in G_H^1} (x_1 - x_i) \\ \sum_{i \in G_H^2} (x_2 - x_i) \\ \vdots \\ \sum_{i \in G_H^n} (x_n - x_i) \end{bmatrix} \quad (12)$$

Completed.

The inference from Theorem 1 informs us that the sparsity-introduced graph convolution can make the graph embedding operation focus on the differences between the homogenous instances. The overview of proposed SIGCN is shown in the Figure 1. First, the samples from the raw data are connected according to the adjacency matrix Ω^n and adjacency matrix Ω^s separately. Then, the sparsity-induced graph (SIgraph) \mathcal{G}_{SI} is obtained according to Ω^n and Ω^s . Next, the multi-layer graph convolution is performed on the sparsity-induced graph and it is equivalent to graph-convolve on the full homogenous subgraph on the SIgraph. Finally, the predicted labels are achieved for each unlabelled data as the architecture output.

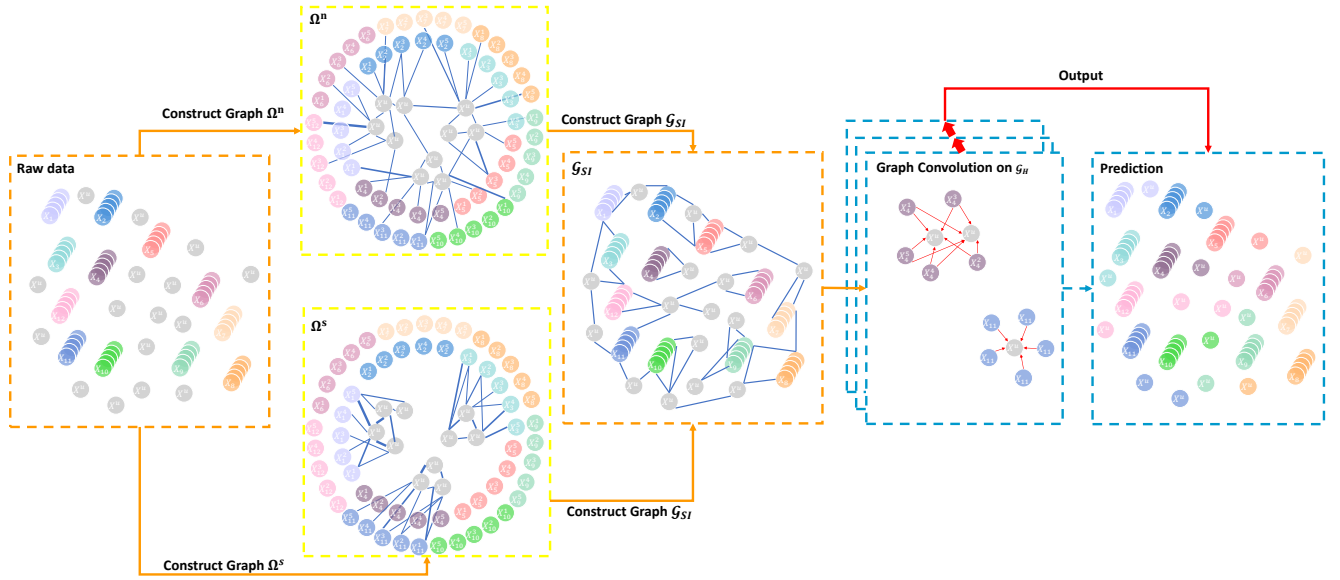


Fig. 1: The overview of the proposed Sparsity-Induced Graph Convolutional Network (SIGCN). The circles in different colors represent the labelled data of different classes X_i . The labelled samples stacked in the same color belong to the same class. The grey circles X'' signify the unlabelled data. Ω^n is the adjacency matrix defined by the neighbors and Ω^s is the adjacency matrix defined by a subgraph linear representation. G_H is the full homogenous subgraph.

We propose the following objective function for the Sparsity-Induced Graph Convolutional Network (SIGCN):

$$\begin{aligned} \min_{W, \hat{\Omega}} \mathcal{L} &= \min_{W, \hat{\Omega}} \mathcal{L}_1 + \mathcal{L}_2 \\ &= - \sum_{i=1}^{n_l} \sum_{j=1}^c \pi_j(y_i) \ln Z_{ij}^o + \sum_{i=1}^n \left\{ \|x_i - X_{x_i \notin X} \Omega_{:,i}^s\|_2 + \|\Omega_{:,i}^s\|_0 \right\} \\ s.t. \quad \hat{\Omega} &= \xi((\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n)) \\ \Omega_{ij}^n &= \begin{cases} \|x_j - x_i\|_2, & x_j \in \text{neighbors}(x_i, k) \\ 0, & \text{otherwise} \end{cases} \\ Z^o &= \sigma(\hat{D}^{-\frac{1}{2}} \hat{\Omega} \hat{D}^{-\frac{1}{2}} \delta(Z^{o-1}) W^{o-1}) \end{aligned} \quad (13)$$

where Z^o is the output result from the last layer of SIGCN, $\hat{D}_{ii} = \sum_j \hat{\Omega}_{ij}$, $\Omega_{:,i}^s$ represents the i^{th} column of the coefficient matrix Ω^s . \odot represents the Hadamard product between two matrices, $\text{neighbors}(\cdot)$ represents the function that returns k neighbors of the i^{th} sample x_i , $\hat{\Omega}^s$ is the sparse adjacency matrix as defined in Definition 2, $\sigma(\cdot)$ represents the softmax function, $\xi(\cdot)$ represents the filter function to remove edges with small weights. The first term \mathcal{L}_1 in Eq. 13 is the supervised loss function of SIGCN, and the second term \mathcal{L}_2 is the L_0 penalty that learns the sparsity-induced graph. In SIGCN, we minimize the above Eq. 13 to achieve an optimized weight matrix for each layer $W^{(i)}$ and the adjacency matrix of sparsity-induced graph $\hat{\Omega}$ as defined in the Definition 3.

B. Sparsity-induced Graph Learning

We obtain the Sparsity-Induced graph (SIgraph) as follows:

$$\begin{aligned} \min_{\hat{\Omega}} \sum_{i=1}^n \left\{ \|x_i - X_{x_i \notin X} \Omega_{:,i}^s\|_2 + \|\Omega_{:,i}^s\|_0 \right\} \\ s.t. \quad \hat{\Omega} &= \xi((\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n)) \\ \Omega_{ij}^n &= \begin{cases} \|x_j - x_i\|_2, & x_j \in \text{neighbors}(x_i, k) \\ 0, & \text{otherwise} \end{cases} \end{aligned} \quad (14)$$

where $\hat{\Omega}$ represents the adjacency matrix of the Sparsity-Induced graph (SIgraph), $\hat{\Omega}^s \in \mathbb{R}^{n \times n}$ represents the sparse adjacency matrix. To solve the above problem, we first solve a series of L_0 -minimization sub-problems with respect to each instance x_i to obtain the optimal sparse linear representation coefficient vector $\Omega_{:,i}^s \in \mathbb{R}^{n-1}$. Then, concatenating $\Omega_{:,i}^s$ with complement zero in the diagonal line to produce the $\hat{\Omega}^s$. Finally, calculating the adjacency matrix of sparsity-induced graph: $\hat{\Omega} = (\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n)$. Here we provide the definition of Sparsity-Induced graph (SIgraph) in the following Definition 3:

Definition 3 (Sparsity-induced graph). Given the dataset $X \in \mathbb{R}^{m \times n}$, the Sparsity-induced graph (SIgraph) is defined as the adjacency matrix like: $\hat{\Omega} = \xi((\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n)) \in \mathbb{R}^{n \times n}$ where $\Omega^n \in \mathbb{R}^{n \times n}$ represents the adjacency matrix of neighbors, and $\hat{\Omega}^s \in \mathbb{R}^{n \times n}$ is the sparse adjacency matrix, $\xi(\cdot)$ represents the filter function to remove edges with small weights.

Given an instance x_i in the dataset X , we intuitively depict the SIgraph construction procedure with respect to this instance in Figure 2. In the first step, we construct two types

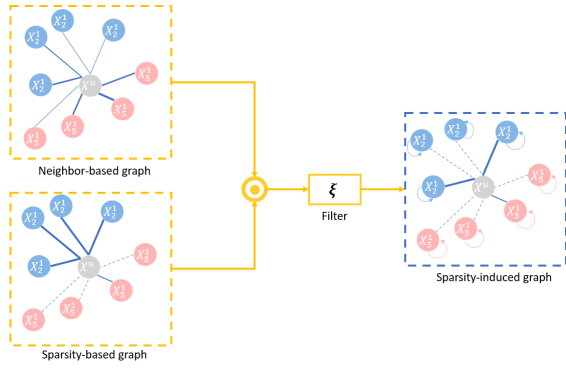


Fig. 2: The Sparsity-Induced graph (SIgraph) construction procedure with respect to one instance. The node in gray color X^u represents the unlabeled sample, the node in blue or pink color X_j^i represents the labeled samples. Two graphs are initially constructed: neighbor-based graph and sparsity-based graph. The neighbor-based graph connects the instance with k neighbors, while the sparsity-based graph connects the instances based on the subgraph linear representation. The thickness of the solid line between instances signifies the weight values. The dashed line represents the weight value as being close to zero. ξ in the orange box represents the filter function. The rightmost graph is the output Sparsity-Induced graph (SIgraph).

of graphs: the neighbor-based graph and the sparsity-based graph. The neighbor-based graph selects k neighbors of the given instance x_i :

$$\mathcal{G}^n = \langle V_{\omega^n}, E_{\omega^n} \rangle \quad (15)$$

where $\omega_j^n = 1$ when x_j belongs to the k neighbors of the given instance x_i . The sparsity-based graph. The neighbor-based graph adjacency matrix can be represent as follows:

$$\Omega_{ij}^n = \begin{cases} \|x_j - x_i\|_2, & x_j \in \text{neighbors}(x_i, k) \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

To construct the sparsity-based graph, we calculate the subgraph linear representation coefficients by using all instances in the dataset except for the given instance $X_{x_i \notin X}$. The subproblem with respect to instance x_i can be written as follows:

$$\omega_i = \min_{\omega} \|x_i - X_{x_i \notin X} \omega\|_2 + \|\omega\|_0 \quad (17)$$

where $\omega_i \in \mathbb{R}^{n-1}$ represents a column of the matrix Ω^s . We utilized orthogonal matching pursuit (OMP) [31] to solve the optimization problem in Eq.17. The OMP uses the greedy strategy to iteratively produce the approximation solution to the problem of Eq.17. We concatenate ω_i to generate the subgraph linear representation coefficient matrix $\Omega^s \in \mathbb{R}^{(n-1) \times (n-1)}$:

$$\Omega^s = [\omega_1, \omega_2, \dots, \omega_n] \quad (18)$$

Then, we complement zero on the diagonal line of Ω^s to

align with the size of Ω^n as follows:

$$\hat{\Omega}^s = \begin{bmatrix} 0 & \omega_{21} & \cdots & \omega_{n1} \\ \omega_{11} & 0 & \cdots & \omega_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{1n} & \omega_{2n} & \cdots & 0 \end{bmatrix} \quad (19)$$

where ω_{ij} represents the j^{th} elements in the subgraph linear representation coefficient vector of the i^{th} instance ω_i . Hence, the sparsity-based graph is constructed according to $\hat{\Omega}^s$:

$$\mathcal{G}^s = \langle V_{\hat{\Omega}^s}, E_{\hat{\Omega}^s} \rangle \quad (20)$$

After both graphs are ready, we make hadamard product between the adjacency matrix of \mathcal{G}^n and \mathcal{G}^s and add self-loops to all nodes in both graphs:

$$\hat{\Omega} = \xi((\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n), \tau) \quad (21)$$

where the filter function $\xi(\cdot)$ can be described as:

$$\xi(x, \tau) = \begin{cases} x, & \text{when } x \geq \tau \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

where τ is the small-value threshold set in advance.

We present Algorithm 1 to summarize the steps of constructing the sparsity-induced graph.

Algorithm 1 Construction of sparsity-induced graph

Require: Dataset X , threshold τ , number of neighbors k .

1. $\Omega^n \leftarrow \{\}, \Omega^s \leftarrow \{\}$;
 2. Normalize the columns of X ;
 3. Calculate the neighbor-based adjacency matrix Ω^n according to the Eq.16;
 - For** $i \leftarrow 1$ to n
 4. Calculate the subgraph linear representation coefficient vector ω_i of the instance x_i according to the Eq. 20:

$$\omega_i = \min_{\omega} \|x_i - X_{x_i \notin X} \omega\|_2 + \|\omega\|_0$$
 5. $\Omega^s \leftarrow \Omega^s \cup \omega_i$;
 - End For**
 6. Generate $\hat{\Omega}^s$ from Ω^s according to the Eq. 19;
 7. Construct the adjacency matrix of sparse-induced graph according to the Eq. 14:

$$\hat{\Omega} = \xi((\Omega^n + I_n) \odot (\hat{\Omega}^s + I_n), \tau);$$
 8. Construct the sparsity induced-graph:

$$\mathcal{G}^s = \langle V_{\hat{\Omega}^s}, E_{\hat{\Omega}^s} \rangle;$$
 - return** Graph \mathcal{G}^s ;
-

C. Semi-supervised learning with Sparsity-induced Graph

The proposed Sparsity-Induced Graph Convolutional Network learns the trainable weights $W^{(i)}$ between layers in terms of the supervised loss function shown as the \mathcal{L}_1 in Eq.13:

$$\min_W - \sum_{i=1}^{n_l} \sum_{j=1}^c \pi_j(y_i) \ln Z_{ij}^o \quad (23)$$

s.t. $Z^o = \sigma(\hat{D}^{-\frac{1}{2}} \hat{\Omega} \hat{D}^{-\frac{1}{2}} \delta(Z^{o-1}) W^{o-1})$

where $\hat{\Omega}$ is the SIgraph constructed using Algorithm 1, $\hat{D}_{ii} = \sum_j \hat{\Omega}_{ij}$. The problem of Eq.23 can be minimized by using ADAM optimizer to achieve the optimal weighting matrix $W^{(i)}$ for each layer. Suppose we have a two-layer SIGCN, the final output of the SIGCN can be described as follows:

$$Z^o = \sigma(\hat{D}^{-\frac{1}{2}} \hat{\Omega} \hat{D}^{-\frac{1}{2}} \delta(\hat{D}^{-\frac{1}{2}} \hat{\Omega} \hat{D}^{-\frac{1}{2}} X W^{(1)}) W^{(2)}) \quad (24)$$

where $W^{(1)}$ and $W^{(2)}$ represents the weighting matrix of the first layer and the second layer, $\sigma(\cdot)$ is the softmax function, $\delta(\cdot)$ is the ReLU activation function. The SIGCN for semi-supervised learning is summarized in the Algorithm 3.

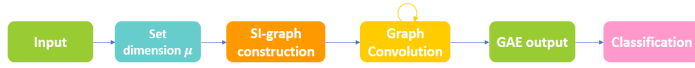


Fig. 3: The workflow of Sparsity-Induced Graph Representation Encoder. After we input the data to the SIGRE, the expected dimension μ of the SIGAE output is set. Then, the normalization of each column in the given data is performed. Next, the SIgraph of the given data is constructed. Followed by this, the graph convolution is repeatedly implemented on the SIgraph until the number of iterations set previously is satisfied. Finally, SIGAE outputs the result. The outputs will be sent for classification.

Besides predicting the labels directly, we can consider the SIGCN as the Sparsity-Induced Graph Representation Encoder (SIGRE). The main purpose of SIGRE is to generate d-dimensional features of each sample simultaneously encoded with SIgraph-based representation. According to Theorem 1, the first-order difference signal is on the homogenous subgraphs with respect to each sample. Therefore, samples from the same class tend to have similar features (small intra-class variation), while those from different classes will have dissimilar features (large inter-class variation) in the encoding procedure, which produces discriminative feature learning [35] and dimensionality reduction. In this case, different classifiers can well perform classification based on the encoded feature [36]. We define SIGRE as shown in Definition 4. The workflow of SIGRE is shown in Figure 3, where there are three procedures: set dimension of feature μ , SIgraph construction, and graph convolution. The expected dimension of feature is set as the dimension of feature map of the SIGRE. The structure figure of SIGRE is shown in Figure 4.

Definition 4 (Sparsity-induced Graph Representation Encoder). Given a collection of data containing both labeled samples $X^l = [X_1^l, X_2^l, \dots, X_n^l] \in \mathbb{R}^{m \times n_l}$ and unlabeled samples $X^u \in \mathbb{R}^{m \times n_u}$, the sparsity-induced graph representation encoder $SGE(X^l, X^u, p, \mu) : \mathbb{R}^{m \times (n_l + n_u)} \rightarrow \mathbb{R}^{\mu \times (n_l + n_u)}$ is a feature encoder that encodes the feature representation of each sample in X^l and X^u from the original space \mathbb{R}^m to the target space \mathbb{R}^μ with the following formulation:

$$SGE(X^l, X^u, p, \mu) = \pi(h^{(p)}), \quad (25)$$

where $\pi(\cdot)$ represents the activation function, $h^{(p)} = \begin{cases} \pi(\Psi \mathcal{F}^{(p-1)} W^{(p-1)}), & p \geq 2 \\ \pi(\Psi X W^{(0)}), & p = 1 \end{cases}$, $\Psi = \hat{D}^{-\frac{1}{2}} \hat{\Omega} \hat{D}^{-\frac{1}{2}}$, $X =$

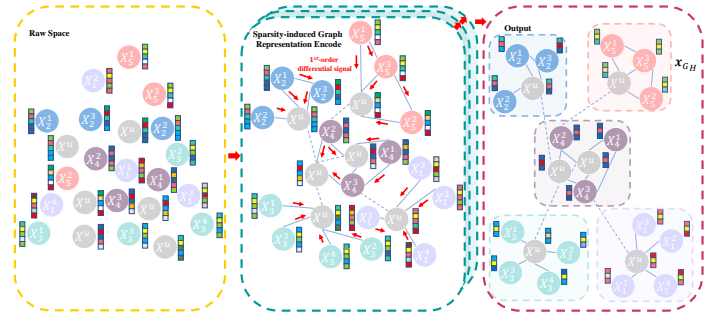


Fig. 4: Structural figure of the Sparsity-induced Graph Representation Encoder (SIGRE). The left area (enclosed in the yellow dashed line) shows the raw space containing input with both labeled samples (nodes) and unlabeled samples (nodes). The color bar besides each node represents the features. In the center area (enclosed in the green dashed line), the SIGRE has multiple layers for graph convolution on the full homogenous graph. The 1st-order differential signals are convolved for feature encoding. In the right area (enclosed in the purple dashed line), the nodes from the same homogenous subgraph show similar patterns (feature).

$[X^l, X^u]$, $\mathcal{F}^{(i)}$ is the feature map of the i^{th} layer of SAGRE, p is the number of layers of SIGRE.

In addition, we provide Theorem 2 to show that SIGRE is able to encode discriminative features.

Theorem 2 (Discriminative feature learning property of SIGRE). Given a collection of data $X \in \mathbb{R}^{m \times n}$, let $S = \{S_i\}_c^1$ be the SIGRE output, where $S_i = \{S_i^j\}_{n_c}^1 \in \mathbb{R}^{\mu \times n_c}$. The feature of S_i^j from the same class i will be similar, while feature of S_i^j will be different from the feature of $S_{p \neq i}^{r \neq j}$.

Proof.

According to Theorem 1, we rewrite the m-dimensional graph signal in the homogenous graph convolution shown in Eq. 12:

$$x_{GH} = \begin{bmatrix} \sum_{i \in G_H^1} (x_1^1 - x_i^1) & \dots & \sum_{i \in G_H^1} (x_1^m - x_i^m) \\ \sum_{i \in G_H^2} (x_2^1 - x_i^1) & \dots & \sum_{i \in G_H^2} (x_2^m - x_i^m) \\ \vdots & \ddots & \vdots \\ \sum_{i \in G_H^n} (x_n^1 - x_i^1) & \dots & \sum_{i \in G_H^n} (x_n^m - x_i^m) \end{bmatrix}. \quad (26)$$

Suppose both the two samples x_i and x_j are from the same class S_q , we measure the similarity between ζ_i of the graph signal x_i and ζ_j of the graph signal x_j as follows:

$$\begin{aligned} s(\zeta_i, \zeta_j) &= \left\| \sum_{r \in G_H^q} (x_i - x_r) - \sum_{r \in G_H^q} (x_j - x_r) \right\|_2^2 \\ &= \left\| \sum x_i - \sum x_r - \sum x_j + \sum x_k \right\|_2^2 \\ &= \left\| nx_i - nx_j - \sum x_r + \sum x_k \right\|_2^2 \\ &= \left\| n(x_i - x_j) \right\|_2^2 \end{aligned} \quad (27)$$

If x_i and x_j are samples from a different class: $x_i \in S_q$, $x_j \in S_v$ ($S_q \neq S_v$), the similarity between ζ_i and ζ_j is

calculated as follows:

$$\begin{aligned}
 s(\zeta'_i, \zeta'_j) &= \left\| \sum_{r \in G_H^q} (x'_i - x'_r) - \sum_{r \in G_H^v} (x'_j - x'_r) \right\|_2^2 \\
 &= \left\| \sum x'_i - \sum x'_r - \sum x'_j + \sum x'_k \right\|_2^2 \\
 &= \left\| n(x'_i - nx'_j) + (\sum x'_k - \sum x'_r) \right\|_2^2 \\
 &= \left\| n(x'_i - nx'_j) + (n\mu'_v - n\mu'_q) \right\|_2^2 \\
 &= \sum n^2 (x'_i - x'_j)^2 + \sum n^2 (\mu'_v - \mu'_q)^2 \\
 &\quad + 2n (x'_i - x'_j) (\mu'_v - \mu'_q) \\
 &= \left\| n(x'_i - x'_j) \right\|_2^2 + \left\| n(\mu'_v - \mu'_q) \right\|_2^2 + 2n(x'_i - x'_j)(\mu'_v - \mu'_q),
 \end{aligned} \tag{28}$$

since the $x_i, x_j \in \mathcal{S}_q$, and $x_{t_i} \in \mathcal{S}_q, x_{t_j} \in \mathcal{S}_v (\mathcal{S}_q \neq \mathcal{S}_v)$, then,

$$\left\| n(x'_i - x'_j) \right\|_2^2 \geq \left\| n(x_i - x_j) \right\|_2^2 \tag{29}$$

We can infer that $\left\| n(\mu'_v - \mu'_q) \right\|_2^2 \geq 0$, $2n(x'_i - x'_j)(\mu'_v - \mu'_q) \geq 0$. Therefore,

$$s(\zeta_i, \zeta_j) \leq s(\zeta'_i, \zeta'_j), \tag{30}$$

where the equality establishes only when $\mu'_v = \mu'_q$.

The above Eq. 30 depicts that the similarity of the graph signals will be smaller when their samples are in the same homogenous graph, while the similarity will be larger when they do not belong to the same homogenous graph. This indicates that the feature of \mathcal{S}_i^j from the same class i will be similar, and feature of \mathcal{S}_i^j will be different from the feature of $\mathcal{S}_i^{r \neq j}$.

Completed.

We summarize the SIGRE in the following Algorithm 2.

Algorithm 2 Sparsity-induced graph representation encoder

Require: Labeled samples X^l , unlabeled samples X^u , output dimension μ , number of layers p , weight matrices W^p .

1. Construct a sparsity-induced graph using Algorithm 1 and get G^s ;

2. Set the dimension of the output feature μ ;

For $i \leftarrow 1$ to p

3. Compute output of each layer using:

$$h^{(p)} = \begin{cases} \pi(\Psi \mathcal{F}^{(i-1)} W^{(i-1)}), & i \geq 2 \\ \pi(\Psi X W^{(0)}), & i = 1 \end{cases}$$

End For

4. Extract the encoded feature:

$$E = \pi(h^{(p)});$$

return E ;

D. Asymptotic time complexity analysis

We analyzed the computational complexity of SIGCN. Suppose there are n instances with N dimensions in the dataset X . The computational complexity of steps 3 and 4 in constructing

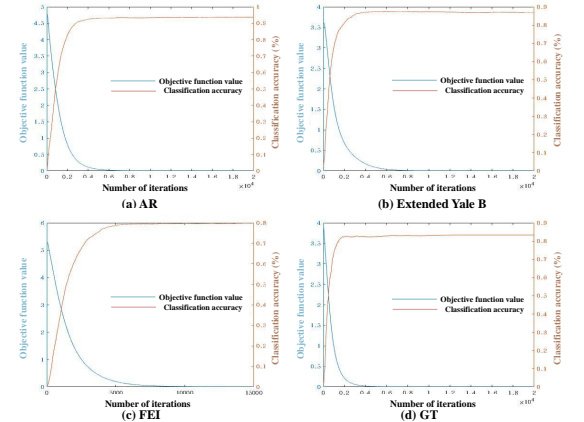


Fig. 5: Objective function value (left) and classification accuracy (right) versus the number of iterations for SIGCN on four datasets: (a) AR, (b) Extended Yale B, (c) FEI, and (d) GT.

Algorithm 3 Sparsity-induced graph convolutional network

Require: Dataset X , threshold τ , number of neighbors k , labels of labelled data Y^l , maximum number of iteration max_iter .

1. Construct sparsity-induced graph using Algorithm 1 and get G^s ;

For $iter=1, \dots, max_iter$ **do**

2. Compute $\psi \leftarrow \nabla L_{G^s}$ according to Eq. 23;

3. Perform Adam to update W according to the gradient estimator ψ ;

End For

4. Calculate the final output according to Eq. 24;









return Labels of unlabelled data Y^u ;

a Sigraph shown in Algorithm 1 is $O(n^2 \times N) + O(n^3)$. Step 7 in constructing a Sigraph is $O(n^2)$. This means the computational complexity of Eq. 23 is $O(\varsigma \times \mathcal{E} \times N \times H \times C)$, where ς is the number of iterations, \mathcal{E} is the number of edges in the Sigraph, and H is the size of the feature map in the hidden layer. Therefore, the overall computational complexity of the proposed SIGCN is $O(n^3 + n^2 \times (N+1) + \varsigma \times \mathcal{E} \times N \times H \times C)$.

E. Convergence analysis

Since the applied ADAM optimizer for solving the Eq.23 is an iterative method, we prove the convergence property of the proposed method with the experiments displayed in the Figure 5, which includes the objective function value and the classification accuracy versus the number of iterations on four datasets (AR, Extended Yale B, FEI, and GT). The maximum number of iterations is set to 20000 for the AR, Extended Yale B, and GT datasets with 15000 for the FEI dataset. It is noted that the objective function's value decreased sharply within the first 6000 iterations in these four sub-figures, before smoothing and leveling out afterwards. Correspondingly, the classification accuracy steadily increases within the first 6000 iterations and tends to flatten for the remaining number of iterations.

TABLE I: Detailed configuration of the eight image databases.

Face dataset	Classes	Labeled	Unlabeled	Size	Images
GT [37]	50	3-5	10-12	40×30	
AR [38]	120	4-16	10-22	40×32	
FEI [39]	200	3-5	9-11	24×96	
Ex.YaleB [40]	38	9-30	34-55	32×32	
Object dataset	Classes	Training	Testing	Size	Images
COIL20 [41]	20	5-20	10-25	32×32	
Flavia [42]	32	20-35	25-40	40×30	
MNIST [43]	10	20-100	100-180	32×32	
FMNIST [44]	10	20-100	100-180	32×32	

IV. EXPERIMENTS AND ANALYSIS

A. Dataset description and experimental settings

In the experiments, we adopted 8 image datasets (summarized in Table I) and made 4 synthetic datasets for evaluation. We categorized these 12 datasets into following 5 categories:

- **Faces with occlusive interference.** The AR face database [38] contains 3120 face images from 120 persons. For each person, there are approximately 26 images covering facial expression changes, lighting condition changes, and occlusive parts changes (sun glasses, scarf). Each image is resized to 40×30 pixels in the experiment.
- **Faces from different views.** The GT face database [37] contains 750 face images from 50 persons. For each person, there are 15 images covering facial expression changes and lighting condition changes. The FEI face database [39] contains 2800 images from 200 persons. Each person has 14 images with different orientations. The Extended Yale B face database contains 2432 images from 38 persons. Each person has 64 images including those taken under different lighting conditions and face poses.
- **Random views in objects.** The COIL20 dataset [41] contains 600 images from 20 objects. Each object was captured 30 images with different orientations. The MNIST dataset [43] is a hand-writing digits recognition dataset containing 10 classes of digits (0-9). In the experiment, we used a subset of 2000 images evenly distributed from the 10 classes for the evaluation in a semi-supervised learning scenario.
- **Changeable shapes in objects.** The Flavia dataset [42] contains 1907 images from 32 leaf species. Each type of leaf has different shapes, lengths, and widths. There are about 60 images for each species. The Fashion-MNIST dataset [44] is a fashionable dress recognition dataset containing 10 classes of fashionable dresses. In the experiment, we applied a subset of 2000 images, which are evenly distributed into 10 classes for the evaluation in a semi-supervised learning scenario.
- **Synthetic data.** We generate four datasets from two manifold structures: two-moon data and seven-ring data. The two-moon data contains 1012 two-dimensional sam-

ples from two classes, data from one class is distributed in a single moon-like shape. The local parts from two classes intersects each other. We construct 3 subtypes of two-moon data (2Moon_v1, 2Moon_v2, 2Moon_v3) shown in the Figure 10a, 10b, 10c. The seven-ring data (7Ring) contains 3500 two-dimensional samples from seven classes. Data samples from one class are distributed on one of the seven concentric circles shown as Figure 10d.

To show the superiority of the proposed method, we applied 15 methods in total for comparison: KNN [46], SVM [47], SRC [16], CRC [48], ProCRC [49], SARC [19], AwnLRR [50], CNN [51], LRRADP [52], PMSSL [53], TS-STSS [33], READ [30], DeepWalk [54], GCN [10], GAT [55]. The ProCRC [49], AwnLRR [50], TS-STSS [33], and READ [30] are the state-of-the-art linear representation-based classifiers. The GCN [10], GAT [55], and DeepWalk [54] are graph-based methods for semi-supervised learning. LRRADP [52] and PMSSL [53] are state-of-the-art semi-supervised learning methods. We set the number of neighbors as 10 in KNN. For SVM, we adopted the RBF kernel with $\gamma = 0.2$. In the LRC methods, we set scaling factor $\lambda = 0.0001$. We set $\lambda_1 = 10$, $\lambda_2 = 0.001$, and $\lambda_3 = 0.02$ for AwnLRR. In LRRADP, the $\lambda_1 = 50$ and $\lambda_2 = 11$. We set parameter $\lambda = 0.1$ for PMSSL. In the READ method, we generated the virtual samples using image flipping, brightening, and masking as mentioned in [30]. We built an 8-layer CNN by simulating the AlexNet [56] architecture as the CNN-8 in the comparisons. We applied the k-nearest neighbor graph in DeepWalk, GCN, and GAT and set the same number of neighbors as the proposed SIGCN. We present the parameter settings of the proposed SIGCN in section IV-B. Here, the the average accuracy over 30 times of iterations was calculated as the results in the following section IV-B, IV-D, and IV-E. In addition, we showed the standard deviation in the comparison of section IV-D

All experiments were performed on a PC equipped with an Intel Core i7-6700 CPU and 16GB RAM. The software platform is PyTorch 1.7.1 [57]. We also applied a deep graph library (DGL) [58] to implement the SIGCN. We trained the deep learning architecture on a GEFORCE GTX 1070 Ti graphics card.

B. Parameters Sensitivity Analysis

We show the performance of the proposed SIGCN under different parameter settings: k (number of neighbors), M^H (size of the feature map in the hidden layer), and the learning rate (lr). The results are presented in Figure 6. For these parameters mentioned above, we set $k \in \{30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90\}$, $M^H \in \{300, 400, 500, 600, 700, 800, 900\}$, and $lr \in \{10^{-4}, 2.5 \times 10^{-4}, 5 \times 10^{-4}, 10^{-3}, 2.5 \times 10^{-3}, 5 \times 10^{-3}\}$. We selected the highest accuracies when fixing two parameters as the results in Figure 6. Here, we chose 5, 8, 5, 21, 20, and 66 samples per class as the training set in GT, AR, FEI, Extended Yale B, Flavia, and Fashion-MNIST, respectively. The remaining samples were taken as the test samples.

From Figure 6, we can find in most cases (except for Fashion-MNIST) when the number of neighbors k increases,

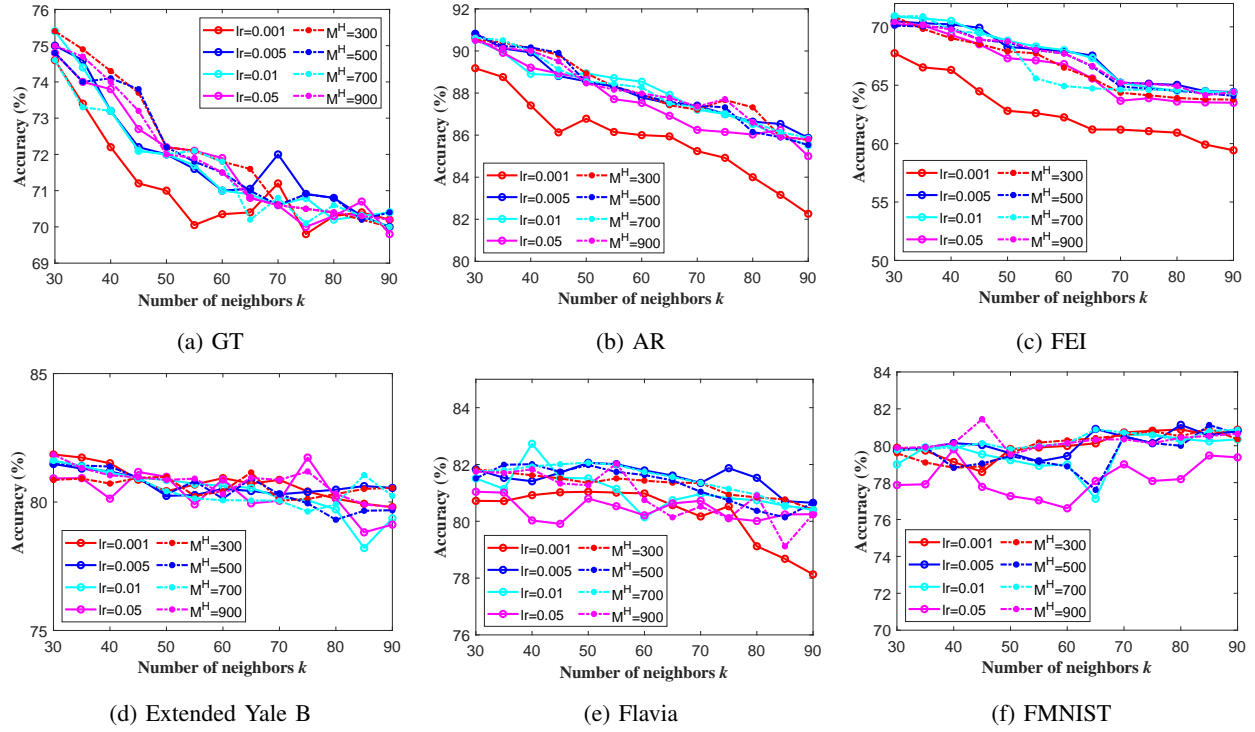


Fig. 6: Parameter sensitivity analysis of the proposed method. (a) GT, (b) AR, (c) FEI, (d) Extended Yale B, (e) Flavia, and (f) Fashion-MNIST.

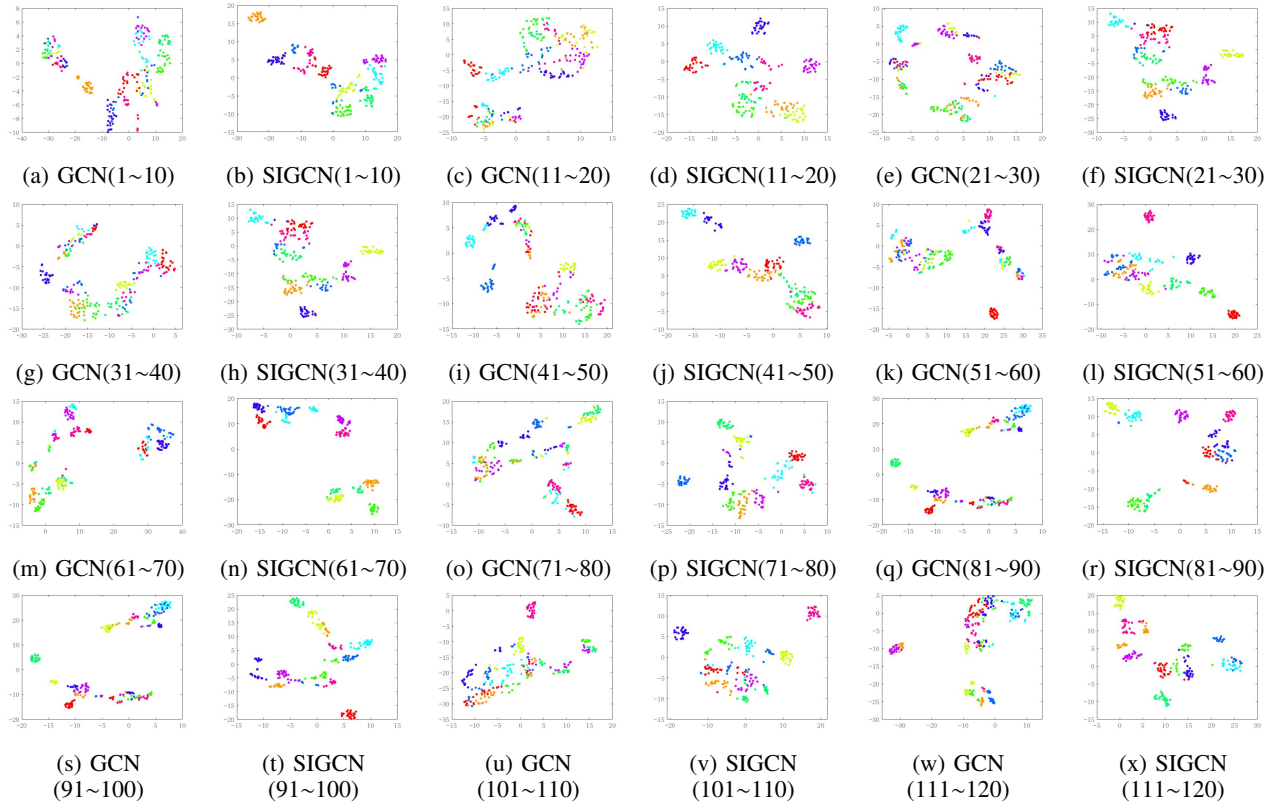


Fig. 7: The 2D visualizations using the t-SNE [45] algorithm from the output of the feature map in GCN [10] (left) and the proposed SIGCN (right) on the AR dataset. Each sub-figure shows the feature map of 10 classes.

TABLE II: Comparison results between SIGCN and classical, popular, and classical, popular, and state-of-the-art methods.

Dataset	GT	AR	FEI	Ext. YaleB	COIL20	Flavia	MNIST	FMNIST
KNN	39.60±1.15	61.57±3.12	35.56±1.05	67.24±0.24	53.52±2.17	59.44±3.11	80.06±1.22	66.44±0.73
SVM	49.00±5.18	47.22±0.20	35.67±3.12	73.41±0.08	83.32±4.37	72.31±3.62	76.61±1.79	64.56±0.27
SRC	50.91±2.38	63.75±2.11	52.78±0.74	43.27±2.37	55.67±0.31	56.60±1.13	84.94±2.01	73.39±1.85
CRC	59.55±1.32	77.56±2.87	39.83±1.12	67.06±0.97	57.33±3.07	50.05±0.72	77.11±1.46	73.56±1.19
ProCRC	50.45±2.93	76.55±2.01	41.11±3.03	80.41±1.96	58.33±2.15	52.48±2.32	77.44±2.17	73.17±1.44
SARC	54.00±2.53	75.36±2.11	27.33±0.97	40.11±2.67	62.48±1.98	23.65±2.09	80.72±1.90	47.83±1.28
AWNLRR	61.00±0.79	78.27±1.66	41.22±1.68	80.37±2.33	54.62±3.17	58.00±2.51	81.11±1.22	74.22±1.77
CNN-8	67.40±2.13	62.55±2.11	60.65±2.13	65.78±4.41	86.12±4.17	75.39±2.57	<u>85.75±3.41</u>	70.11±1.77
LRRADP	52.46±2.05	65.37±1.99	50.22±1.83	69.37±3.88	94.39±0.85	49.70±3.99	84.73±0.54	64.41±1.14
PMSSL	54.43±1.29	45.80±0.74	44.98±1.32	45.56±0.24	80.71±1.66	49.39±0.49	80.02±1.13	70.32±1.32
DeepWalk	43.20±4.73	40.13±5.12	50.45±3.01	25.28±5.87	73.50±4.17	71.16±2.77	80.40±2.89	63.87±0.98
GCN	64.80±2.36	64.43±0.31	<u>66.77±2.31</u>	72.24±3.13	96.75±1.10	<u>81.53±0.57</u>	85.59±0.52	73.43±2.23
GAT	66.08±2.61	37.52±3.18	63.75±2.56	79.54±2.39	<u>97.33±1.17</u>	80.05±0.79	84.72±0.94	75.70±1.14
TS-STSS	65.20±3.35	72.73±2.72	57.33±0.67	57.88±1.90	71.12±1.80	67.48±0.93	80.78±1.52	74.67±1.09
READ	65.20±0.72	<u>78.27±1.31</u>	53.22±1.38	<u>79.61±1.41</u>	72.14±0.96	67.00±0.91	83.50±1.51	<u>77.17±0.86</u>
SIGCN	75.40±2.13	91.98±3.34	70.90±2.32	83.59±1.13	99.33±0.24	82.89±2.53	89.00±1.38	81.84±0.74
Impr	8.00%	13.71%	4.13%	4.13%	2%	1.36%	3.25%	4.67%
Impr _{GCN}	10.6%	27.55%	4.13%	11.35%	2.58%	1.36%	3.41%	8.41%

1: The highest values in each column are marked in **Bold** font, and the second highest values in each column are underlined.

2: The **Impr** row demonstrates the improvements brought by SIGCN compared with the second highest accuracy in each dataset.

3: The number of labelled samples per class in each dataset: GT-5, AR-12, COIL20-15, FEI-5, Extended YaleB-30, Flavia-30, MNIST-20, and FMNIST-20.

the accuracies will go downward. This may be due to the fact that with more neighbors connected, the more nodes from different classes will be included in the constructed Sigraph with respect a single node when k is larger than the number of samples per class n_k , which will cause a performance degradation. In other words, this indicates the construction of Sigraph will not involve large numbers of neighbors to reach an ideal performance. In some cases, the learning rate lr will have an impact on the overall performance, especially on the GT, AR, and FEI datasets. When $lr = 10^{-4}$, the impact is significant. This is because the model may not fully converge. We can see that when it ranges from 5×10^{-4} to 5×10^{-3} , the changes are minor, except for the Fashion-MNIST dataset. For the Fashion-MNIST dataset, the accuracy is influenced by the problem of over-fitting. In addition, the model will not be heavily influenced by the size of the hidden layer M^H . The largest gap between the best and the worst accuracies shown in Figure 6 with different settings of M^H is 2.8%. Overall, we can draw three conclusions from this parameter analysis: (1) SIGCN is sensitive to the learning rate lr . (2) The performance will be degraded when $k > n_k$. (3) TSIGCN is not sensitive to the size of hidden layer M^H . Here, we select the optimal parameter settings in each dataset according to the parameter sensitivity analysis for the following comparisons.

C. Visualization of the graph embedding via SIGCN

We utilized the t-SNE [45] algorithm to visualize the output of feature map of both SIGCN and GCN [10] in the AR dataset [38]. We first adopted the PCA algorithm to reduce the dimensionality of the output feature map to 60, and then utilized the t-SNE [45] algorithm to project the data representation to a 2-D plane. The visualization outcomes are shown in the Figure 7. The different colors represent different classes. Each sub-figure in Figure 7 contains 10 classes of feature maps in the AR dataset. Generally speaking, we can clearly see that the output feature map of the proposed SIGCN has more compact distribution compared with the feature map of GCN [10]. First,

The embedded points of SIGCN from the same class have stronger tendency to distribute together (see Figure 7d, 7l, and 7r) Conversely, in Figure 7d, 7l, and 7r, the distribution of embedded points of GCN [10] have elongated shapes, which have higher probability of overlapping with distributions of points from other classes. Second, the classification boundaries of SIGCN are overtly clearer than those of GCN's (see Figure 7b, 7d, 7r 7t, and 7x).

D. Results and comparisons

1) *Experiments on image datasets:* We show the experimental results and comparisons with 15 classical, popular, and state-of-the-art methods on 8 datasets. The accuracy comparisons are shown in the Table II. The number of training samples per class of each dataset are marked at the bottom of Table II. The remaining samples in the datasets are collected as the test sets. Figure 8 presents their performances by increasing the number of labelled samples per class in the 8 datasets. We divided the methods involved in the comparison into 4 groups: LRC-based method, graph learning-based methods, semi-supervised learning-based methods, and convolutional neural networks. Generally speaking, it can be observed that the proposed SIGCN has achieved the highest accuracies in 8 datasets as shown in Table II. However, the second-highest accuracies (see the results underlined) distributed in the different methods, shows a comprehensive better classification ability of the proposed method. Here, the improvements of SIGCN compared with the second-highest accuracies is large in some datasets (GT, AR, and Ext. Yale B). Lastly, SIGCN is able to attain a high accuracy when using a small number of labelled samples per class (see COIL20 and AR).

Compared with the state-of-the-art methods of the LRC-based methods (ProCRC [49], AWNLRR [50], TS-STSS [33], and READ [30]), the proposed SIGCN has a better recognition performance. Specifically speaking, the SIGCN is 10.2%, 13.71%, and 15.66%, 17.68%, 3.98%, 15.41%, 2.98%, 1.08% higher than READ (the latest LRC classifier) in the eight datasets shown in Table II, respectively.

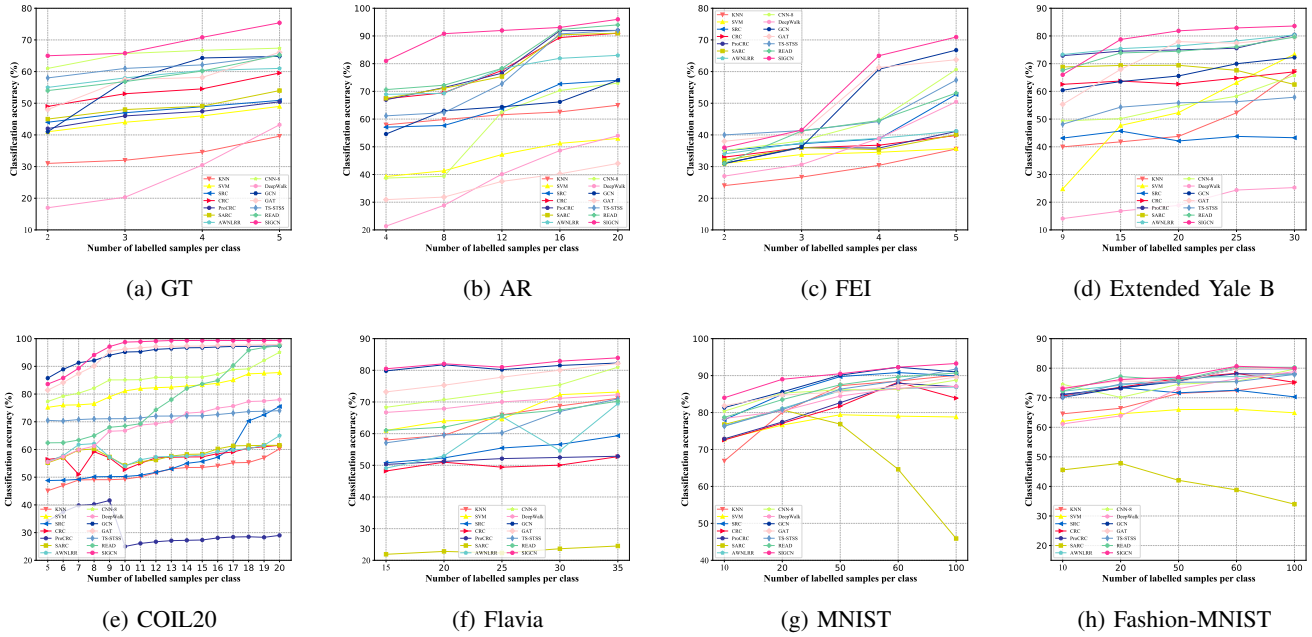


Fig. 8: Performance and comparison with the state-of-the-art methods whereby we increase the labelled samples per class. (a) GT, (b) AR, (c) FEI, (d) Extended Yale B, (e) COIL20, (f) Flavia, (g) MNIST, and (h) Fashion-MNIST.

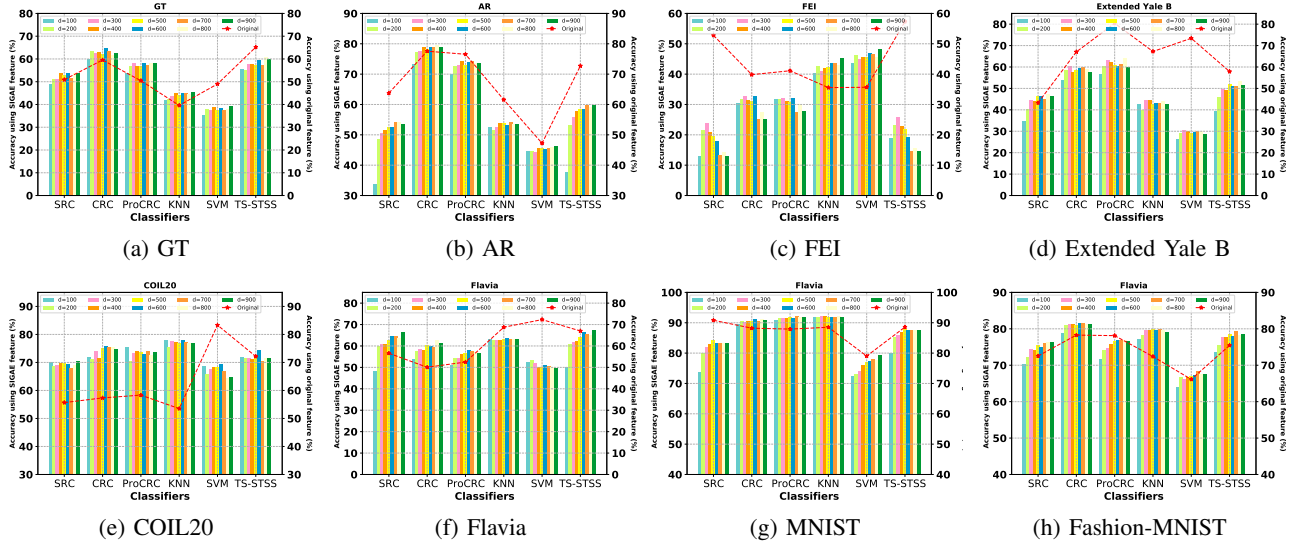


Fig. 9: Performance of five methods using features extracted from SIGAE and original feature. (a) GT, (b) AR, (c) FEI, (d) Extended Yale B, (e) COIL20, (f) Flavia, (g) MNIST, and (h) Fashion-MNIST.

In comparison to the graph learning-based methods (DeepWalk [54], GCN [10], and GAT [55]), the proposed SIGCN also surpasses these methods. Compared with the original GCN [10], the proposed method is 10.6%, 27.55%, 4.13%, 11.35%, 2.58%, 1.36%, and 8.41% higher. This indicates that SIGCN has superiority over conventional methods. For GAT [55], which is GCN with an attention mechanism, SIGCN also outperformed it on the eight datasets. Regarding the state-of-the-art methods of semi-supervised learning, other than graph learning (LRRADP [52] and PMSL [53]), SIGCN showed a better recognition performance in all datasets. Clearly, SIGCN is also better than the 8-layer convolutional neural network,

showing the effect of the graph convolutional network.

As shown in Figure 8, with the increasing number of labelled samples per class in 8 datasets, we can clearly find that the proposed SIGCN attained the best performance in most cases. In some cases, although the proposed method hasn't obtained the best performance in the beginning, it will become the best classifier with the increasing number of labelled data (e.g., Figure 8(c) and (d)).

2) *Experiments on synthetic datasets:* We tested the proposed method with two groups of synthetic data: two-moons data and seven-rings data. For the comparison, we applied 13 methods including the state-of-the-art methods. As shown in

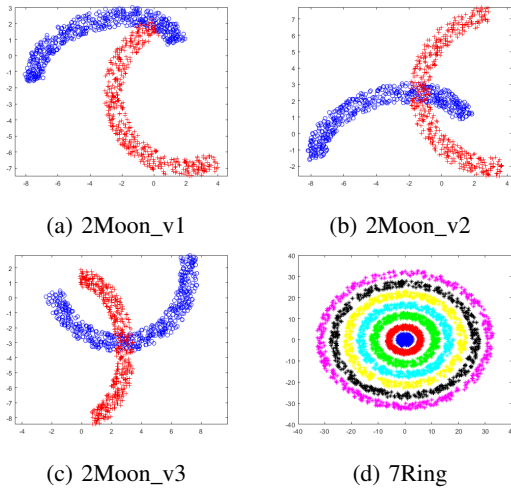


Fig. 10: Illustration of two-moons data and seven-rings data. (1) 2Moon_v1, (2) 2Moon_v2, (3) 2Moon_v3, and (4) 7Ring.

Dataset	2Moons_v1	2Moons_v2	2Moons_v3	7Rings
KNN	68.68±1.13	72.34±1.05	66.71±1.01	31.82±0.58
SVM	66.51±0.93	70.58±0.87	67.10±1.07	34.37±0.91
SRC	67.80±1.21	70.88±1.10	67.34±0.98	37.19±0.97
CRC	33.48±1.93	27.67±2.19	32.60±2.12	36.60±1.42
ProCRC	68.47±0.87	72.44±0.75	67.21±0.53	24.88±0.91
SARC	92.63±2.41	75.79±1.74	72.76±1.66	35.25±1.72
AWNLRR	91.75±0.12	75.25±0.21	69.60±0.32	43.97±0.22
LRRADP	91.58±0.36	73.60±0.74	69.76±1.03	45.51±1.70
PMSSL	50.00±0.49	94.20±0.51	87.22±0.74	65.55±2.53
DeepWalk	95.11±1.51	79.56±1.21	79.86±1.37	68.97±1.24
GCN	94.39±0.54	74.00±0.24	85.00±0.52	100.00±0
GAT	94.47±0.11	78.00±0.13	85.53±0.55	97.72±0.12
TS-STSS	68.68±2.21	72.54±2.02	69.77±1.13	38.82±1.98
SIGCN	99.56±0.07	94.97±0.51	96.56±0.62	100.00±0
Impr	5.09%	0.77%	9.34%	0%
Impr _{GCN}	5.17%	20.97%	11.56%	0%

TABLE III: Performance on the synthetic datasets between SIGCN and 13 other methods.

Table III the proposed SIGCN has promising performances on both two-moons data and seven-rings data. We took 10% of the samples as the training set and applied the remaining 90% of the samples as the test set. The accuracies are above 95% except for 2Moon_v2 (94.97%). This indicates SIGCN is able to adaptively learn the manifold structures and has a strong generalization ability. In the comparison, SIGCN outperformed LRC-based methods by 30.88%, 19.18%, 23.8%, and 60.19% on the four datasets, implying a stronger classification ability than the state-of-the-art linear classifiers. Regarding the graph learning-based methods, SIGCN is still the best, and we can find the highest gap between SIGCN and these methods are 20.97%. The large gap may be due to the fact that the conventional graph learning-based methods fail to correctly classify samples in the intersection area of the two classes. Although the semi-supervised learning-based methods show powerful abilities on these datasets (e.g., PMSSL [53] in 2Moon_v2 and 2Moon_v3), the proposed SIGCN has improvements in the comparison. This indicates the effectiveness of integrating the Sgraph and feature embedding through graph convolution. In the 7Ring dataset, SIGCN achieved 100% accuracy, which

proves that it can perfectly learn the manifold structure of the rings-distributed data. Except for GCN, the proposed method outperforms all other methods, showing its superiority in performance. Despite GCN having a competitive performance with SIGCN, it costs much more computational time than SIGCN (see section IV-F).

E. Evaluation of feature extraction using the Sparsity-induced Graph Auto-Encoder

We evaluated the Sparsity-Induced Graph Representation-Encoder (SIGRE) in this subsection. We first extracted multiple-dimensional features from different dataset, and then evaluate the classification performance using five methods on eight image dataset mentioned in section IV-A. The five methods used in this experiment are: SRC [16], CRC [48], ProCRC [49], KNN [46], SVM [47], and TS-STSS [33]. We extracted features with dimensions ranging from 100 to 900 for each dataset. The detailed results are shown in the Figure 9. From the Figure 9, we can observe that the SIGCN is able to extract effective features, which improves the performance of the classifiers. The highest improved accuracies in each dataset are: 3.09% ($d = 800$ and $d = 900$, GT), 4.29% ($d = 800$, Ext. Yale B), 15% ($d = 900$, COIL20), 10.03% ($d = 900$, Flavia), and 3.88% ($d = 800$ and $d = 900$, FMNIST).

F. Computational time analysis

In this subsection, we made the computational time analysis by comparing the time consumed of the proposed method with the graph learning-based methods: DeepWalk [54], GCN [10], and GAT [55]. For GCN, GAT, and SIGCN, we ran 2000 iterations and recorded the consumed time during training and prediction. The details of the computational time analysis are shown in Table IV.

TABLE IV: Computational time comparison between SIGCN, DeepWalk, GCN, and GAT.

Methods	DeepWalk	GCN	GAT	SIGCN	Speed up
GT	136.58	17.89	609.31	12.49	5.4 ↓
AR	239.12	41.82	1952.41	33.85	7.97 ↓
FEI	671.43	56.07	2142.73	42.21	13.86 ↓
YaleB	208.91	31.62	1003.87	23.58	8.04 ↓
COIL20	117.36	14.33	553.19	10.19	4.14 ↓
Flavia	246.58	29.6	804.56	21.64	7.96 ↓
MNIST	143.79	24.51	702.9	17.81	6.7 ↓
FMNIST	160.04	25.06	718.54	17.19	7.87 ↓

1: The smallest values in each row are marked in **Bold** font.

2: The **Speed up** column demonstrates the computational time reduction.

3: The unit of measurement is second (s).

The SIGCN costs the least computational time compared with other graph learning-based methods when performing supervised learning. Compared with GCN, the SIGCN made the largest acceleration with 13.86s in the FEI dataset, and smallest acceleration with 5.4s in the GT dataset.

G. Discussion

We can summarize the property of the proposed method as follows:

The graph embedding distribution of SIGCN is compact and clear. Clearly illustrated in section IV-C, the output feature map of the proposed SIGCN is compact and clear. Compared with the GCN [10], the SIGCN has a more compact distribution and clearer classification boundaries. This means the feature outputs of the samples from the same class will be similar, while feature outputs of samples from the different class will be distinctive.

SIGCN has a state-of-the-art performance in semi-supervised learning. In section IV-D, SIGCN achieved a state-of-the-art performance for semi-supervised learning (see Table II). The proposed SIGCN achieved the highest accuracies on all 8 image datasets as well as 4 synthetic datasets, showing the promising classification performance and generalization. The advantage of SIGCN can also be reflected in the Figure 8. In most cases, SIGCN attained the highest accuracy with the changing number of labelled samples per class.

SIGRE makes effectiveness feature extraction. It clearly illustrates that the features extracted by SIGRE can help classifiers to improve their classification performance compared with using the original feature. Besides this, the dimensionality of the extracted features is smaller than the original features, which means SIGRE can also be used as a dimension reduction method.

SIGCN has high efficiency for implementation. The SIGCN consumed less time than other graph learning-based methods in the second level. This is due to the fact that the connections inside SIGCN are sparse compared with other methods. The high efficiency of SIGCN makes itself feasible to implement in many different real-world applications.

We have the challenges as well as the research direction of the future work in the following two points:

- 1) SIGCN will suffer from the setting of the number of neighbors, it is necessary to propose a method that adaptively determines the number of neighbors in different cases.
- 2) The stability of SIGCN in different datasets is still a problem that affects the degree of improvement brought by the method. In the future, we will enhance the stability by modeling the relationship between data distribution and the sparsity-induced graph construction.

V. CONCLUSION

In this paper, we proposed the Sparsity-Induced Graph Convolutional Network (SIGCN) to perform semi-supervised learning. SIGCN first constructs the Sparsity-Induced graph and then performs the graph convolution on the whole graph to generate a node representation for classification. We proved the graph convolution on the Sgraph is equivalent to the graph convolution on the homogenous sub-graph, where it is feasible to learn the label information of an unlabelled node from the labelled node in the Sgraph. Furthermore, we established a Sparsity-Induced Graph Representation-Encoder based on the SIGCN architecture to extract effective features and perform dimension reduction. According to the extensive experiments and comparisons on 12 datasets, the proposed SIGCN achieves a state-of-the-art classification performance in semi-supervised learning.

ACKNOWLEDGMENT

This work was supported in part by the University of Macau (File no. MYRG2018-00053-FST), in part by the National Natural Science Foundation of China under Grant 61602540, and in part by the Open Research Fund of the Beijing Key Laboratory of Big Data Technology for Food Safety (Project No. BTBD-2021KF05). The authors would like to express their appreciation for the reviewers' constructive suggestions.

REFERENCES

- [1] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.
- [2] R. Agerri and G. Rigau, "Robust multilingual named entity recognition with shallow semi-supervised features," *Artificial Intelligence*, vol. 238, pp. 63–82, 2016.
- [3] B. Hui, P. Zhu, and Q. Hu, "Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 4215–4222.
- [4] J. Jeong, S. Lee, J. Kim, and N. Kwak, "Consistency-based semi-supervised learning for object detection," in *Advances in neural information processing systems*, 2019, pp. 10 759–10 768.
- [5] S. Zhao and B. Zhang, "Joint constrained least-square regression with deep convolutional feature for palmprint recognition," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.
- [6] S. Liu, P. Reviriego, X. Tang, W. Tang, and F. Lombardi, "Result-based re-computation for error-tolerant classification by a support vector machine," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 1, pp. 62–73, 2020.
- [7] U. K. Dutta, M. Harandi, and C. C. Sekhar, "Unsupervised deep metric learning via orthogonality based probabilistic loss," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 1, pp. 74–84, 2020.
- [8] J. Wen, X. Fang, Y. Xu, C. Tian, and L. Fei, "Low-rank representation with adaptive graph regularization," *Neural Networks*, vol. 108, pp. 83–96, 2018.
- [9] F. Dornaika, A. Baradaaji, and Y. El Traoulsi, "Semi-supervised classification via simultaneous label and discriminant embedding estimation," *Information Sciences*, vol. 546, pp. 146–165.
- [10] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
- [11] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang, "Deepinf: Modeling influence locality in large social networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18)*, 2018.
- [12] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 974–983.
- [13] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinformatics*, vol. 34, no. 13, pp. i457–i466, 2018.
- [14] B. A. Olshausen and D. J. Field, "Natural image statistics and efficient coding," *Network: computation in neural systems*, vol. 7, no. 2, pp. 333–339, 1996.
- [15] D. J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *Josa a*, vol. 4, no. 12, pp. 2379–2394, 1987.
- [16] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 2, pp. 210–227, 2008.
- [17] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.
- [18] K.-C. Lee, J. Ho, and D. J. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 5, pp. 684–698, 2005.
- [19] N. Akhtar, F. Shafait, and A. Mian, "Efficient classification with sparsity augmented collaborative representation," *Pattern Recognition*, vol. 65, pp. 136–145, 2017.
- [20] C. Bin, Y. Jianchao, Y. Shuicheng, F. Yun, and T. Huang, "Learning with 11-graph for image analysis," *IEEE Transactions on Image Processing*, vol. 19, no. 4, pp. 858–866, 2010.

- [21] H. Ren, H. Pan, S. I. Olsen, and T. B. Moeslund, "Greedy vs. 11 convex optimization in sparse coding: comparative study in abnormal event detection," in *International Conference on Machine Learning 2015*, vol. 37, 2015.
- [22] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *Journal of machine learning research*, vol. 7, no. Nov, pp. 2399–2434, 2006.
- [23] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," *Advances in neural information processing systems*, vol. 16, pp. 321–328, 2003.
- [24] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the 20th International conference on Machine learning (ICML-03)*, 2003, pp. 912–919.
- [25] X. Zhu and Z. Ghahramani, "Learning from labels and unlabeled data with label propagation," *Carnegie Mellon University: School of Computer Science*, 2002.
- [26] B. A. Olshausen and D. J. Field, "Sparse coding of sensory inputs," *Current opinion in neurobiology*, vol. 14, no. 4, pp. 481–487, 2004.
- [27] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [28] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on signal processing*, vol. 41, no. 12, pp. 3397–3415, 1993.
- [29] D. S. Taubman, "Image compression fundamentals, standards and practice," *JPEG-2000*, 2002.
- [30] S. Zeng, B. Zhang, J. Gou, and Y. Xu, "Regularization on augmented data to diversify sparse representation for robust image classification," *IEEE Transactions on Cybernetics*, 2020.
- [31] Y. C. Pati, R. Rezaeiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of 27th Asilomar conference on signals, systems and computers*. IEEE, 1993, pp. 40–44.
- [32] Y. Xu, D. Zhang, J. Yang, and J.-Y. Yang, "A two-phase test sample sparse representation method for use with face recognition," *IEEE Transactions on circuits and systems for video technology*, vol. 21, no. 9, pp. 1255–1262, 2011.
- [33] J. Zhou, S. Zeng, and B. Zhang, "Two-stage knowledge transfer framework for image classification," *Pattern Recognition*, vol. 107, p. 107529, 2020.
- [34] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.
- [35] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European conference on computer vision*. Springer, 2016, pp. 499–515.
- [36] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.
- [37] L. Chen, H. Man, and A. V. Nefian, "Face recognition based on multi-class mapping of fisher scores," *Pattern Recognition*, vol. 38, no. 6, pp. 799–811, 2005.
- [38] A. M. Martinez, "The ar face database," *CVC Technical Report24*, 1998.
- [39] C. E. Thomaz and G. A. Giraldi, "A new ranking method for principal components analysis and its application to face image analysis," *Image and vision computing*, vol. 28, no. 6, pp. 902–913, 2010.
- [40] A. S. Georgiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 643–660, 2001.
- [41] S. Nane, S. Nayar, and H. Murase, "Columbia object image library: Coil-20," *Dept. Comp. Sci., Columbia University, New York, Tech. Rep.*, 1996.
- [42] S. G. Wu, F. S. Bao, E. Y. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang, "A leaf recognition algorithm for plant classification using probabilistic neural network," in *2007 IEEE international symposium on signal processing and information technology*. IEEE, 2007, pp. 11–16.
- [43] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [44] H. Xiao, K. Rasul, and R. Vollgraf, "(2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.
- [45] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [46] N. S. Altman, "An introduction to kernel and nearest-neighbor non-parametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.
- [47] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [48] L. Zhang, M. Yang, and X. Feng, "Sparse representation or collaborative representation: Which helps face recognition?" in *2011 International conference on computer vision*. IEEE, 2011, pp. 471–478.
- [49] S. Cai, L. Zhang, W. Zuo, and X. Feng, "A probabilistic collaborative representation based approach for pattern classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2950–2959.
- [50] J. Wen, B. Zhang, Y. Xu, J. Yang, and N. Han, "Adaptive weighted nonnegative low-rank representation," *Pattern Recognition*, vol. 81, pp. 326–340, 2018.
- [51] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1, no. 2.
- [52] L. Fei, Y. Xu, X. Fang, and J. Yang, "Low rank representation with adaptive distance penalty for semi-supervised subspace classification," *Pattern Recognition*, vol. 67, pp. 252–262, 2017.
- [53] P. Mercado, F. Tudisco, and M. Hein, "Generalized matrix means for semi-supervised learning with multilayer graphs," in *Advances in neural information processing systems*, 2019, pp. 14 877–14 886.
- [54] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.
- [55] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.
- [56] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [57] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, 2019, pp. 8026–8037.
- [58] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," *arXiv preprint arXiv:1909.01315*, 2019.

Jianhang Zhou (Graduate Student Member, IEEE) received the B.S. degree from Nanjing Forestry University (NJFU), in 2018, a M.S. degree in Computer Science from University of Macau in 2020. He is currently pursuing the Ph.D. degree in computer science in Faculty of Science and Technology at the University of Macau. His research interest includes pattern recognition, subspace learning, and medical biometrics.



Shaoning Zeng (M'18) received his B.S. and M.S. from Beihang University, Beijing, China, in 2004 and 2007, respectively. He received the Ph.D. degree in Computer Science from the University of Macau in 2020. Now, he is an Associate Professor in the Yangtze Delta Region Institute (Huzhou) of University of Electronic Science and Technology of China. His research interests include computer vision, multimedia analysis and deep learning.



Bob Zhang (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2011. He is currently an Associate Professor in the Department of Computer and Information Science, University of Macau, Taipa, Macau. His research interests focus on biometrics, pattern recognition, and image/video processing. Dr. Zhang is a Technical Committee Member of the IEEE Systems, Man, and Cybernetics Society and Associate Editors of Artificial Intelligence Review and IET



Computer Vision.