Information Fusion 95 (2023) 109–119

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full length article

Learnable graph convolutional network and feature fusion for multi-view learning

Zhaoliang Chen [a,b], Lele Fu [c], Jie Yao [a,b], Wenzhong Guo [a,b], Claudia Plant [d,e], Shiping Wang [a,b,*]

[a] College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China
[b] Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China
[c] School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou 510260, China
[d] Faculty of Computer Science, University of Vienna, Vienna 1090, Austria
[e] ds:UniVie, Vienna 1090, Austria

ARTICLE INFO

Keywords:
Information fusion with deep learning
Multi-view learning
Graph convolutional network
Semi-supervised classification

ABSTRACT

In practical applications, multi-view data depicting objects from assorted perspectives can facilitate the accuracy increase of learning algorithms. However, given multi-view data, there is limited work for learning discriminative node relationships and graph information simultaneously via graph convolutional network that has drawn the attention from considerable researchers in recent years. Most of existing methods only consider the weighted sum of adjacency matrices, yet a joint neural network of both feature and graph fusion is still under-explored. To cope with these issues, this paper proposes a joint deep learning framework called Learnable Graph Convolutional Network and Feature Fusion (LGCN-FF), consisting of two modules: feature fusion network and learnable graph convolutional network. The former aims to learn an underlying feature representation from heterogeneous views, while the latter explores a more discriminative graph fusion via learnable weights and a parametric activation function dubbed Differentiable Shrinkage Activation (DSA) function. The proposed LGCN-FF is validated to be superior to various state-of-the-art methods in multi-view semi-supervised classification.

1. Introduction

In real-world applications, a large amount of information exists in varied forms, because an object can be described from heterogeneous data sources. For example, streaming media can be illustrated by features of frames, audio, and textual descriptions, which come into being multi-view data. This motivates researchers to discover the latent consistent information across diverse views [1–3]. Instead of directly exploiting features from heterogeneous sources, it should be helpful to extract node relationships among samples and propagate supervision signals across nodes, which motivates us to conduct graph learning on multi-view data. Graph learning is a crucial field of machine learning in decades and has been extensively applied to a multitude of practical applications, such as node classification [4–6], social network analysis [7–9] and computer vision [10–12]. In recent years, Graph Convolutional Network (GCN) has been widely explored for its powerful ability to integrate connectivity patterns and feature attributes with given graph-structured data [5]. Owing to the ability to propagate node features to their neighbors based on the renormalized adjacency matrix, GCN can make full use of existing node information and scarce supervision signals. Besides, GCN also has a faster training speed, because the graph convolution operation is an approximation of a truncated Chebyshev polynomial. A large number of studies have revealed the remarkable performance boosting of GCN in various learning tasks [13–15], especially in semi-supervised learning tasks that only have a small number of labeled nodes.

Although most multi-view datasets do not naturally contain the topological structure like traditional citation and link prediction datasets, samples in real-world applications often have implicit connections that can be extracted. In light of this, we can mine the hidden relationships among samples from existing features and generate multifarious graphs. These estimated graphs generally describe node relationships from various perspectives with their complementarity. As shown on the left side of Fig. 1, most existing methods utilize feature fusion methods or graph fusion models before applying GCN, both of which are critical for the performance of downstream multi-view learning tasks.

A multi-view learning framework should get more benefits if it can learn an intact node representation from both feature fusion and graph fusion problems. To our best knowledge, there is very limited work on

* Corresponding author at: College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China.
E-mail address: shipingwangphd@163.com (S. Wang).

https://doi.org/10.1016/j.inffus.2023.02.013
Received 3 June 2022; Received in revised form 13 January 2023; Accepted 10 February 2023
Available online 16 February 2023

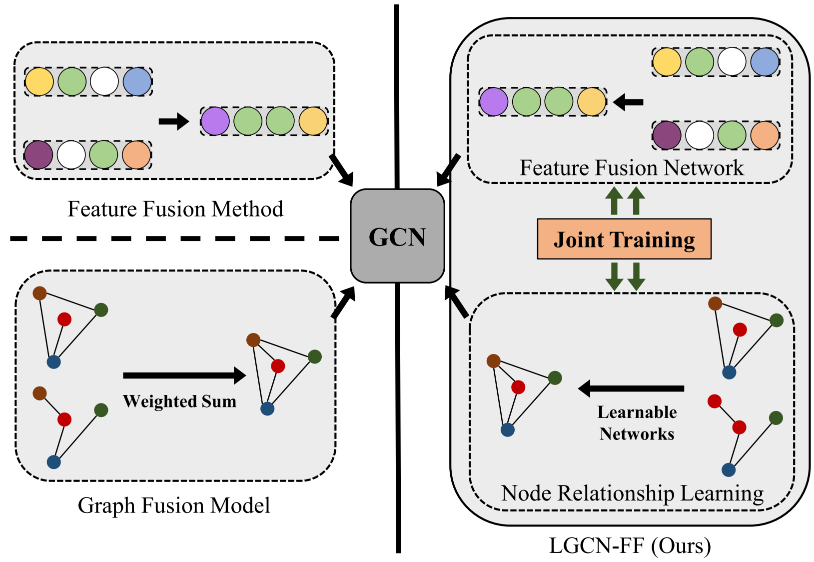


Fig. 1. A brief comparison of the proposed LGCN-FF and other methods. Compared with most multi-view learning models (left) that only concentrated on either feature fusion approaches or graph fusion approaches (e.g., [16,17]), LGCN-FF (right) is a joint framework that simultaneously trains a feature fusion network and a learnable graph fusion network with a multi-stage training strategy.

the joint training of feature and graph fusion learning, which is beneficial to exploring co-optimal solutions to both two problems. Thus, one challenge is how to solve feature and graph fusion problems that are difficult to be optimized jointly in an end-to-end framework, which is the primary problem we need to address in this paper. Besides, as we have discussed, multi-view data in the real world generally do not exist as the network topology, attributed to which most algorithms preprocessed the original data and converted them to graph-structured data. Because k-Nearest Neighbor (KNN) can estimate edges among nodes via calculating similarities of samples and exploring nearest neighbors, most of the existing methods generated graphs via KNN. However, some studies have pointed out that this is sometimes inaccurate and may yield undesired links between samples [18]. Previous works generally applied these adjacency matrices without training or refining neighborhood relationships [16,19,20], which possibly resulted in the performance decline of multi-view learning. Although some researchers have successfully leveraged GCN to deal with multi-view data [16,17], they only considered a weighted combination of different adjacency matrices. This is problematic because a linear weighted sum of adjacency matrices may amplify and aggregate incongruous noises from distinct graphs estimated by KNN. Therefore, a well-established graph refining procedure should be conducted after numerous graphs are integrated into a unique one, so that the negative impact of undesired connections is mitigated.

Consequently, we propose an end-to-end framework dubbed Learnable Graph Convolutional Network and Feature Fusion (LGCN-FF). A brief description is shown on the right side of Fig. 1. LGCN-FF is comprised of two fundamental components: a feature fusion network and a learnable GCN network. The former aims to resolve the feature fusion problem with given multi-view data, and the latter is to learn adjacency matrix fusion with multiple graphs generated from distinct features. Feature fusion is realized by multiple sparse autoencoders and a fully-connected network that is responsible for incorporating all features. Adjacency matrix fusion is first conducted by a weighted sum of all graphs, where all weights are learned automatically. For the purpose of learning a more discriminative graph representation, we present a learnable function termed as Differentiable Shrinkage Activation (DSA) to further explore adjacency matrix fusion, which adaptively refines feasible and robust node relationships during training. It can be regarded as an analogous pattern to soft thresholding operator in Iterative Shrinkage Thresholding Algorithm (ISTA) [21] and Singular Value Thresholding (SVT) [22], which are applied to address sparse coding and low-rank approximation problems, respectively. Each iteration of the proposed LGCN-FF contains four optimization steps in light of their own loss functions. Therefore, LGCN-FF is a joint framework that learns features and node relationships simultaneously.

The main contributions of this paper are as follows:

(1) Propose an end-to-end neural network framework for multi-view semi-supervised classification, which integrates sparse autoencoders and a learnable GCN to jointly learn intact representations of multiple features and graphs.

(2) Construct a learnable GCN framework with adaptive weights and a parametric DSA function, both of which mine more discriminative and robust representations of graphs from heterogeneous views automatically.

(3) Develop a multi-step optimization strategy for LGCN-FF via back propagation, each of which updates corresponding parameters while fixing other learnable parameters.

(4) The proposed framework is leveraged to conduct multi-view semi-supervised classification tasks, and achieves superior performance compared with other state-of-the-art graph-based algorithms.

The rest of this paper is organized as follows. Related works on GCN, multi-view learning, feature and graph fusion are reviewed in Section 2. We elaborate the proposed LGCN-FF in Section 3, including the detailed introduction of each component and algorithm analyses. Finally, the effectiveness of the proposed framework is verified via substantial experiments in Section 4, and our work is concluded in Section 5.

2. Related work

2.1. Graph convolutional network

In this subsection, we first review recent works on GCN. A spectral graph convolution operation is conducted by a signal $x \in \mathbb{R}^m$ and a filter $g_\theta = diag(\theta)$, formulated as

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

where $\mathbf{U}$ denotes the matrix of eigenvalues of the normalized graph Laplacian matrix. For the purpose of saving computational resources, Kipf et al. [5] performed the first-order approximation of truncated Chebyshev polynomial and imposed it on the node classification tasks with topological networks. Specifically, the lth layer of a spectral GCN is formally defined as

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

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ denotes the adjacency matrix considering the self-connections, and $[\tilde{\mathbf{D}}]_{ii} = \sum_j [\tilde{\mathbf{A}}]_{ij}$. Layer-specific weight matrix is denoted by $\mathbf{W}^{(l)}$. The graph convolution operation can be regarded as a special form of Laplacian smoothing [23], which propagates neighborhood features across the whole graph. Due to the encouraging performance of GCN, many variant algorithms have been explored. For example, Xu et al. put forward an innovative answer-centric approach dubbed radial graph convolutional networks to cope with the visual question generation tasks [24]. Liu et al. integrated GCN with a hidden conditional random field to reserve the skeleton structure information during the classification stage [25]. Bo et al. investigated the low-frequency and high-frequency signals in a graph, and proposed a model that adaptively integrated different signals during message passing [26]. A variant of GCN was derived through a modified Markov diffusion kernel, which explored the global and local contexts of nodes [27]. Guo et al. exploited GCN to propagate features over the relationship affinity matrix, generating relationship-regularized representations of objects to produce the scene graph [28]. A convolution operator on the multi-relational graph was developed, based on which the proposed multi-dimensional convolution operator achieved the eigenvalue decomposition of a Laplacian tensor [29]. Lei et al. established the graph receptive fields according to diffusion paths and applied them to build a compact graph convolutional network [30]. A multi-stage GCN-based framework was presented with the self-supervised learning to improve the generalization performance on the graph with limited supervised information [31]. These GCN-based works have significantly promoted the performance of various learning tasks in both Euclidean and non-Euclidean domains.

2.2. Multi-view learning

Multi-view learning that leverages assorted types of features from heterogeneous views has promoted the performance of various machine learning tasks [32]. Traditional methods including Supporting Vector Machine (SVM), random forest and AdaBoost algorithm were applied to learn robust multi-view representations [33,34]. For example, Sun et al. proposed a multi-view semi-supervised learning framework based on SVMs, which integrated the manifold regularization and the multi-view regularization [35]. Xu and Sun adopted the Adaboost algorithm in the multi-view learning scenario, which combined multiple learners to output the hypothesis [36]. In recent years, many multi-view learning models with complex objectives were further proposed. A multi-view and multi-feature learning framework was constructed to simultaneously consider the fusion of features and views, which refined a discriminant representation from distinctive classes [37]. Chen et al. proposed a joint framework for multi-view spectral clustering by learning an adaptive transition probability matrix [38]. The nuclear-norm-based optimization method was proposed to conduct multi-view image data fusion via a joint learning framework [39]. Late fusing incomplete multi-view clustering was proposed to learn a cluster assignment from distinct views to exploit a consensus clustering matrix [40]. AE^2-Nets utilized inner autoencoders to perform view-specific representation learning, and adopted the outer autoencoders to implement multi-view information encoding [41]. Wen et al. presented an effective incomplete multi-view clustering framework to make full use of the local geometric information and the unbalanced discriminating powers of incomplete multi-view observations [42]. Sparsity-based optimization methods are also essential in multi-view learning [43,44]. For instance, Sun et al. proposed a sparse semi-supervised learning framework, which adopted scarce unlabeled data and a few labeled data in objective functions to accelerate function evaluations [45]. Du et al. presented a differentiable bi-sparse multi-view co-clustering algorithm, which transformed sparse constraints into equivalent deep networks [46]. In this paper, we also apply sparse autoencoders to get sparse outputs in the proposed framework. Differing from these previous works, we need to generate an overcomplete underlying representation from multi-view features that have different dimensions, which contains intact node characteristics of all views.

2.3. Feature or graph fusion

Effective feature or graph fusion is universally applied to the multi-view data processing to achieve desired learning performance, which takes advantage of full observations from multi-view representations. Zhou et al. fused information from multiple kernels to improve the performance of multi-kernel clustering [47]. Tang et al. proposed a deep neural network that recurrently fused and refined multi-scale deep features [48]. Graphs of multiple views can be integrated into a consistent global graph, whose Laplacian matrix is constrained with multiple strongly connected components [49]. Huang et al. put forward a unified multi-view image data fusion model on the basis of nuclear norm optimizations [50]. [51] paid attention to preserving the local structure of data while conducting the graph fusion. A graph neural network-based fusion mechanism was designed to extract the complementary information across views [52]. [53] learned graph matrices of heterogeneous views, and a unified graph matrix is recovered via a mutual reinforcement manner. A unified framework was proposed by introducing a co-training strategy into the GCN framework, where the graph information embedded in multiple views is explored adaptively [16]. Nonetheless, most of them only concentrated on either graph fusion or feature fusion, both of which influenced the performance of GCN considerably. It is pivotal to develop a framework with a co-training pattern that can conduct feature fusion and graph fusion simultaneously. In the following section, we will elaborate the proposed method to solve this issue, being the main contribution of this paper.

3. The proposed method

For the purpose of jointly learning feature fusion and graph fusion, we develop an end-to-end unified neural network framework consisting of two primary components: feature fusion network and learnable GCN. The optimization procedure of one iteration in this framework is divided into multiple steps, inspired by the Alternating Direction Minimization (ADM) [54] strategy. In particular, each independent optimization step has its own loss function, all of which make up a complete training iteration of the whole network. Given m samples with n features, the proposed LGCN-FF aims to solve semi-supervised classification problems with given multi-view data $\mathcal{X} = \{\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(V)}\}$ and the scarce supervision signal $\mathbf{Y} \in \mathbb{R}^{|\Omega| \times c}$, where $\mathbf{X}^{(v)} \in \mathbb{R}^{m \times n_v}$ denotes the features of the vth view with totally V views, $\mathbf{Y}$ is the incomplete binary label matrix generated from the labeled sample set Ω that satisfies $|\Omega| \ll m$. Namely, only a small proportion of samples provide labels for supervision. Fig. 2 provides a detailed illustration of the proposed LGCN-FF. Feature fusion aims to integrate multi-view features with varying dimensions into an intact representation with the same dimension, exploring the underlying features. Learnable GCN is supposed to merge multiple adjacency matrices, generating a unique graph with better robustness and generalization.

3.1. Feature fusion network

In order to tackle multi-view data consisting of various features with varied dimensions and explore the underlying information across multiple views, the proposed LGCN-FF firstly projects original multi-view representations onto a shared latent space. Considering that feature dimensions of multiple perspectives may extremely differ, it is not applicable to mapping these features onto the same compressed latent

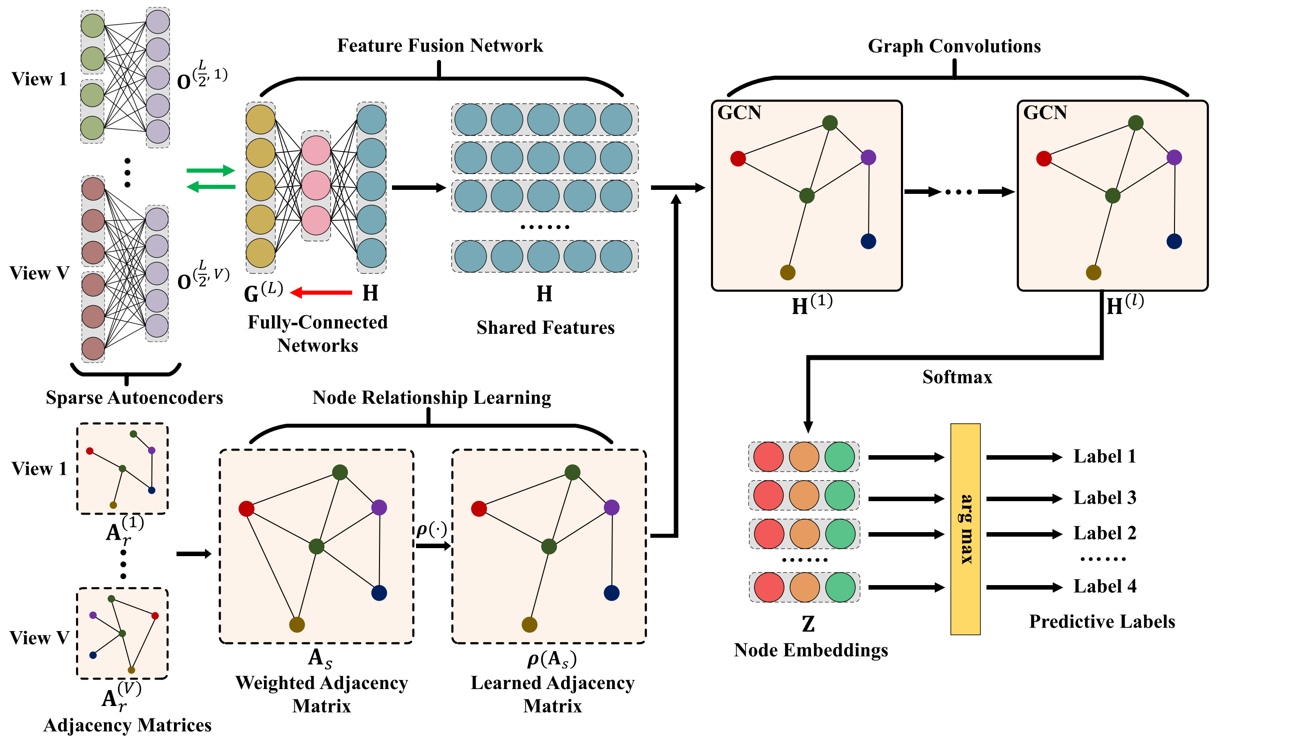


Fig. 2. Structure of the proposed LGCN-FF, which consists of a feature fusion network and a learnable GCN, where nodes with different colors indicate nodes belonging to different classes. LGCN-FF is an end-to-end framework whose learnable parameters are updated by a multi-step optimization strategy. Intact representations of both multi-view features and graphs are learned simultaneously to promote the performance of GCN.

space. Therefore, we employ sparse autoencoders to explore overcomplete underlying representations for all views. Each view corresponds to a view-specific sparse autoencoder, and these sparse autoencoders map the original features onto the sparse space with the same dimension. Formally, the output $\mathbf{O}^{(l,v)} \in \mathbb{R}^{m\times d_l}$ of the lth layer for the vth sparse autoencoder is defined as

$$\mathbf{O}^{(l,v)} = \sigma\left(\mathbf{O}^{(l-1,v)}\mathbf{W}_{sa}^{(l,v)} + \mathbf{b}_{sa}^{(l,v)}\right), \tag{3}$$

where $\mathbf{W}_{sa}^{(l,v)} \in \mathbb{R}^{d_{l-1}\times d_l}$ and $\mathbf{b}_{sa}^{(l,v)} \in \mathbb{R}^{d_l}$ are layer-specific weight and bias, respectively. The input of the vth sparse autoencoder is $\mathbf{X}^{(v)}$, i.e., $\mathbf{O}^{(0,v)} = \mathbf{X}^{(v)}$. Here, $\sigma(\cdot)$ is the layer-specific activation function. The loss function of the sparse autoencoder in the vth view is

$$\mathcal{L}_{sa}^{(v)} = \frac{1}{2}\left\|\mathbf{O}^{(L,v)} - \mathbf{X}^{(v)}\right\|_2^2 + \beta\mathcal{D}_{KL}(\varrho\|\hat{\varrho}), \tag{4}$$

where $\mathcal{D}_{KL}$ is the Kullback–Leibler divergence and β controls the sparsity penalty degree. Hyperparameter ϱ is the value maintaining the sparsity, and $\hat{\varrho}$ is the average of the distribution of latent neuron activations. Kullback–Leibler divergence is calculated by

$$\mathcal{D}_{KL}(\varrho\|\hat{\varrho}) = \varrho\log\frac{\varrho}{\hat{\varrho}} + (1-\varrho)\log\frac{(1-\varrho)}{(1-\hat{\varrho})}. \tag{5}$$

In order to fuse the hidden features from diverse perspectives into a shared feature vector, we further utilize a fully-connected neural network to carry out the feature fusion task. Assuming that there are a total of L layers in the fully-connected neural network, the forward propagation in the lth layer is computed by

$$\mathbf{G}^{(l)} = \sigma\left(\mathbf{G}^{(l-1)}\mathbf{W}_{fc}^{(l)} + \mathbf{b}_{fc}^{(l)}\right), \tag{6}$$

where $\mathbf{G}^{(0)} = \mathbf{H}$. The matrix $\mathbf{H}$ is also a learnable input updated by gradient descent and back propagation techniques. We project the learned $\mathbf{H}$ onto various view-specific latent features $\{\mathbf{O}^{(\frac{L}{2},v)}\}_{v=1}^{V}$ via a trainable fully-connected network. In fact, the trainable $\mathbf{H}$ also serves as the shared node representation in the learnable GCN. A two-step optimization strategy is employed for updating $\{\mathbf{W}_{fc}^{(l)}, \mathbf{b}_{fc}^{(l)}\}_{l=1}^{L}$ and $\mathbf{H}$. Both two steps share the same reconstruction loss function as defined below:

$$\mathcal{L}_{fc} = \frac{1}{2}\sum_{v=1}^{V}\left\|\mathbf{G}^{(L)} - \mathbf{O}^{(\frac{L}{2},v)}\right\|_2^2. \tag{7}$$

With the assumption that features of each single view can be rebuilt from the intact common representation $\mathbf{H}$ by trainable weights and biases in the fully-connected network, Eq. (7) is regarded as the trade-off of reconstruction errors among heterogeneous views and explores the shared underlying features. We present the details of optimization steps in Section 3.3.

3.2. Learnable graph convolutional network

In this subsection, we present a learnable GCN which automatically integrates the adjacency matrices generated by multiple views and learns a graph containing more discriminative node relationships. Firstly, the adaptive weighted sum of adjacency matrices is obtained by

$$\mathbf{A}_s = \sum_{v=1}^{V}\pi^{(v)}\mathbf{A}_r^{(v)}, \tag{8}$$

where $\mathbf{A}_r^{(v)} = (\tilde{\mathbf{D}}^{(v)})^{-\frac{1}{2}}\tilde{\mathbf{A}}^{(v)}(\tilde{\mathbf{D}}^{(v)})^{-\frac{1}{2}}$ is the initial renormalization adjacency matrix of the vth view, and $\pi^{(v)}$ is the automatically learned view-specific weight coefficient. The initialization of adjacency matrices can be conducted via the KNN method. Namely, we evaluate the node similarities according to their features with Gaussian heat kernel, and then select the top-k similar samples for each node as its neighbors, on the basis of which an initial graph is constructed. Owing to the multi-view features, we can generate V different initial graphs. Because $\{\pi^{(v)}\}_{v=1}^{V}$ is constrained with $\sum_{v=1}^{V}\pi^{(v)} = 1$, we employ the softmax renormalization at each epoch as

$$\pi^{(v)} \leftarrow \frac{\exp\left(\pi^{(v)}\right)}{\sum_{v=1}^{V}\exp\left(\pi^{(v)}\right)} \tag{9}$$

for $v = 1, \ldots, V$.

Nevertheless, a straightforward weighted sum of adjacency matrices may not be sufficiently feasible for multi-view graph learning, because a linear weighted sum of all graphs may yield undesired connections between nodes in the fused graph. Besides, attributed to the fact that the neighborhood relationships are estimated by KNN, which may be not accurate enough, a data-driven refining process should be taken to explore a more comprehensive graph fusion without corrupting the structure and characteristic information of the original graphs. In order to achieve an optimal adjacency matrix fusion for the given task, we propose the Differentiable Shrinkage Activation (DSA) function denoted by $\rho(\cdot)$ to refine the weighted adjacency matrix. Because GCN is developed with the precondition that the graph should be undirected, we require that the output of $\rho(\mathbf{A}_s)$ should also be symmetric. To this end, the learnable DSA function $\rho(\cdot)$ is defined as

$$\rho(\mathbf{A}_s) = \mathbf{A}_s \odot \mathrm{ReLU}\,(\mathbf{S} - \boldsymbol{\Theta})\,, \tag{10}$$

where $\odot$ is the Hadamard product (entry-wise product), $\mathbf{S} \in \mathbb{R}^{m\times m}$ denotes the learnable coefficient matrix and $\boldsymbol{\Theta} \in \mathbb{R}^{m\times m}$ controls the thresholds of node relationship activations. For the sake of theoretic strictness and better interpretation, it is required that $\mathbf{S}$ and $\boldsymbol{\Theta}$ should be symmetric. Therefore, we define the coefficient matrix $\mathbf{S}$ as

$$\mathbf{S} = \mathrm{Sigmoid}\left(\frac{1}{2}\left(\bar{\mathbf{S}} + \bar{\mathbf{S}}^{\mathrm{T}}\right)\right), \tag{11}$$

which is parameterized by a learnable matrix $\bar{\mathbf{S}} \in \mathbb{R}^{m\times m}$. On the basis of Eq. (11), the proposed method can learn an edge-specific coefficient for each edge of the undirected graph, which automatically shrinks node relationships with coefficient values ranging in $[0, 1]$.

In order to reduce local data noises and construct a sparser graph, the learnable matrix $\boldsymbol{\Theta}$ in Eq. (10) is considered as a thresholding matrix controlling the edge activation. For simplicity and theoretical rigor, we define the entry of the thresholding matrix as

$$[\boldsymbol{\Theta}]_{ij} = [\boldsymbol{\Theta}]_{ji} = \mathrm{Sigmoid}(\theta_i),\ \forall i \leqslant j \leqslant m \tag{12}$$

with $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_m]$, where $\boldsymbol{\theta}$ is a learnable vector and $\mathrm{Sigmoid}(\cdot)$ admits the non-negativity of thresholders. Consequently, $\boldsymbol{\Theta}$ is symmetric and promotes the sparseness of outputs calculated by Eq. (10), which can also be regarded as trainable biases of the coefficient matrix. It is noted that only the node relationship information whose coefficient is greater than its corresponding thresholding value can be activated. DSA function is beneficial for improving the performance of GCN, due to the ability of automatical feature learning via coefficient matrix and thresholding values. Actually, it is an analogous pattern as shrinkage function widely employed in the proximal optimization which promotes the sparse or low-rank property, e.g., the soft thresholding operator in ISTA [21] or SVT [22] algorithms. However, the classical thresholders are generally hyperparameters that should be predefined, and all signals share the same fixed thresholders. Thus we transform soft thresholding operators into a trainable activation function so that the neural networks can learn a tailored thresholder matrix by back propagation with given tasks and datasets. We initialize $\bar{\mathbf{S}}$ randomly to compute $\mathbf{S}$, and initialize $\boldsymbol{\theta}$ as a vector to generate $\boldsymbol{\Theta}$ in the beginning of training. With these previous analyses, the lth layer of the learnable GCN is formulated by

$$\mathbf{H}^{(l)} = \sigma\left(\rho(\mathbf{A}_s)\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}_{lgcn}\right), \tag{13}$$

where $\mathbf{H}^{(0)} = \mathbf{H}$. Namely, the trainable $\mathbf{H}$ obtained in the previous module becomes the unique common representation of multiple views and is regarded as the input node features in GCN. We use a widely employed 2-layer learnable spectral GCN as an example, which computes the node embedding $\mathbf{Z}$ with

$$\mathbf{Z} = \mathrm{softmax}\left(\rho(\mathbf{A}_s)\sigma\left(\rho(\mathbf{A}_s)\mathbf{H}\mathbf{W}^{(1)}_{lgcn}\right)\mathbf{W}^{(2)}_{lgcn}\right). \tag{14}$$

For a semi-supervised classification task, the loss function of the learnable GCN is defined by the cross-entropy error over semi-supervised information generated from the labeled sample set Ω, as shown below:

$$\mathcal{L}_{lgcn} = -\sum_{i\in\Omega}\sum_{j=1}^{c}\mathbf{Y}_{ij}\ln\mathbf{Z}_{ij}, \tag{15}$$

where $\mathbf{Y} \in \mathbb{R}^{|\Omega|\times c}$ is the incomplete label matrix generated from Ω satisfying $|\Omega| \ll m$.

Algorithm 1 Training Framework of LGCN-FF

Input: Multi-view data $\mathcal{X} = \{\mathbf{X}^{(1)}, \cdots, \mathbf{X}^{(V)}\}$ and semi-supervised information $\mathbf{Y} \in \mathbb{R}^{|\Omega|\times c}$.
Output: Node embedding $\mathbf{Z}$.

```
Initialize weights and biases of sparse autoencoders;
Initialize weights, biases and learnable input H of fully-connected networks;
Initialize learnable weights {π^(v) = 1/V}_{v=1}^V, S ∈ R^{m×m} and θ = [θ_1, …, θ_m] of learnable GCN;
Initialize adjacency matrices A^(1), ⋯, A^(V) via KNN;
while not convergent do
  for v = 1 → V do
    Compute O^(L/2, v) and O^(L,v) of the vth sparse autoencoder with Equation (3);
    Update {W_sa^(l,v), b_sa^(l,v)}_{l=1}^L with back propagation;
  end for
  for v = 1 → V do
    Compute G^(L) of the fully-connected network with Equation (6);
  end for
  Update {W_fc^(l), b_fc^(l)}_{l=1}^L with back propagation;
  for v = 1 → V do
    Compute G^(L) of the fully-connected network with Equation (6);
  end for
  Update H with back propagation;
  Compute Z = H^(L) of the learnable GCN with Equation (13);
  Update {W_lgcn^(l)}_{l=1}^L, {π^(v)}_{v=1}^V, S and Θ with back propagation;
end while
return Node embedding Z.
```

3.3. Training strategy

The proposed LGCN-FF is an end-to-end neural network framework with a multi-step optimization method, as described in Algorithm 1. The proposed model aims to solve graph fusion problems, feature fusion problems and downstream classification problems jointly. However, a unified objective target of these problems is complex and not jointly convex for all trainable variables. If we directly optimize the whole framework with a loss function that includes all optimization targets, it is difficult for the model to get the optimal variables, and the loss may even not converge. Consequently, we follow the ADM strategy [54] which optimizes variables in each subproblem when fixing other irrelevant variables. Theoretically, this is beneficial to the model to obtain the optimal solutions during training, because each subproblem is fully optimized. The optimization of this framework is divided into the following four steps: optimizing trainable weights and biases of sparse autoencoders, optimizing trainable weights and biases of the fully-connected network, optimizing the trainable input $\mathbf{H}$, and optimizing trainable parameters in learnable GCN. In an independent training iteration, each step performs one-step forward propagation, and then conducts back propagation with fixed uncorrelated variables. It is noticed that each step optimizes step-specific variables via its own loss function, i.e., all sparse autoencoders in the first step employ Eq. (4), the second and the third steps share the same loss function defined in Eq. (7), and the final step applies Eq. (15). Although the

Table 1
A brief description of all test multi-view datasets.

Datasets	# Samples	# Views	# Features	# Classes	Data types
ALOI	1,079	4	64/64/77/13	10	Object images
BBCnews	685	4	4,659/4,633/4,665/4,684	5	Textual documents
BBCsports	544	2	3,183/3,203	5	Textual documents
MNIST	10,000	3	30/9/9	10	Digit images
Wikipedia	693	2	128/10	10	Textual documents
MSRC-v1	210	5	24/576/512/256/254	7	Object images
Reuters	18,758	5	21,531/24,892/34,251/15,506/11,547	6	Textual documents

Table 2
Classification accuracy (mean% and standard deviation%) of all compared semi-supervised classification methods with 10% labeled samples as supervision, where the best performance is highlighted in bold and the second best result is underlined. Limited by the computational complexity of algorithms and machine resources, some models encounter out-of-time or out-of-memory error on MNIST and Reuters datasets, marked with "–".

Datasets\Methods	ALOI	BBCnews	BBCsports	MNIST	Wikipedia	MSRC-v1	Reuters
KNN	45.7 (3.1)	38.3 (9.6)	38.3 (9.5)	88.8 (0.4)	58.7 (3.0)	52.9 (8.8)	34.4 (0.5)
SVM	41.5 (6.6)	76.7 (7.6)	73.4 (4.9)	87.2 (1.3)	62.4 (4.2)	58.9 (5.9)	48.7 (0.3)
AdaBoost	69.9 (9.5)	49.6 (6.6)	55.1 (9.8)	63.3 (6.0)	53.1 (3.4)	33.5 (5.2)	54.0 (0.2)
AMGL [55]	82.4 (3.3)	52.3 (5.5)	55.6 (1.4)	88.5 (0.2)	10.1 (0.8)	<u>85.9 (1.8)</u>	–
MVAR [56]	72.9 (5.5)	75.3 (5.5)	83.7 (3.8)	85.3 (0.8)	61.2 (3.4)	54.8 (7.5)	<u>64.6 (0.3)</u>
MLAN [51]	87.6 (1.6)	74.1 (0.9)	62.6 (2.2)	88.6 (0.3)	10.2 (0.8)	82.2 (5.4)	–
AWDR [57]	93.6 (1.9)	85.7 (1.4)	81.3 (3.3)	78.1 (0.3)	<u>62.5 (5.8)</u>	57.7 (7.0)	61.3 (0.6)
HLR-M^{2}VS [58]	87.7 (1.7)	78.1 (2.8)	84.6 (0.4)	–	36.5 (3.4)	79.6 (8.4)	–
ERL-MVSC [59]	90.5 (2.7)	85.9 (2.2)	90.3 (1.9)	89.5 (0.3)	51.6 (2.3)	73.3 (3.7)	–
GCN fusion [5]	92.6 (1.2)	89.6 (1.8)	87.0 (2.0)	89.2 (0.5)	60.1 (0.6)	68.6 (7.2)	55.4 (0.3)
SSGCN fusion [27]	93.4 (1.0)	<u>89.7 (5.5)</u>	<u>94.1 (1.2)</u>	89.3 (0.1)	62.1 (0.4)	70.3 (4.9)	56.8 (0.2)
Co-GCN [16]	<u>96.5 (0.4)</u>	81.9 (1.5)	84.8 (1.4)	<u>89.9 (0.4)</u>	57.9 (0.7)	62.9 (4.4)	60.2 (0.6)
LGCN-FF	**97.1 (0.6)**	**91.2 (0.9)**	**98.2 (0.6)**	**90.2 (0.3)**	**67.4 (1.5)**	**90.4 (2.0)**	**67.3 (0.5)**

formulated problem is optimized separately in the same iteration with the output of the former optimization becoming the input of the latter one, the whole framework is organized by ADM strategy so that each convex subproblem can be solved effectively. The trainable parameters in the whole framework are listed as follows.

1. Feature fusion network: $\{\mathbf{W}_{sa}^{(l,v)}, \mathbf{b}_{sa}^{(l,v)}\}_{l=1,v=1}^{L,V}$ and $\{\mathbf{W}_{fc}^{(l)}, \mathbf{b}_{fc}^{(l)}\}_{l=1}^{L}$.
2. Learnable GCN: $\{\pi^{(v)}\}_{v=1}^{V}$, $\mathbf{S}$, $\theta = [\theta_1, \dots, \theta_m]$ and $\{\mathbf{W}_{lgcn}^{(l)}\}_{l=1}^{2}$.

At each iteration, given multi-view data $\{\mathbf{X}^{(v)} \in \mathbb{R}^{m\times n}\}_{v=1}^{V}$ with V views, the computational complexity for sparse autoencoders and the feature fusion network is $\mathcal{O}(2Vmnd + md^2)$ if all embeddings are projected onto a d-dimension vector with $d \ll n$. The forward propagation of learnable GCN costs $\mathcal{O}(mn + md^2)$.

4. Experimental analyses

4.1. Experimental settings

4.1.1. Datasets description

The proposed LGCN-FF framework is utilized to perform semi-supervised classification tasks on several real-world multi-view datasets. Seven publicly available multi-view datasets are selected for the performance evaluation, as listed below:

- **ALOI**[1]**:** This is an image dataset which contains objects that are taken under varied light conditions or rotation angles. Multi-view features including 64-D RGB color histograms, 64-D HSV color histograms, 77-D color similarities and 13-D Haralick features are involved.
- **BBCnews**[2]**:** It is a collection of news reports which covers politics, entertainment, business, sport and technology fields. There are totally four different textual features extracted from various segments to describe the news.
- **BBCsports**[3]**:** Different from BBCnews, it is a dataset consisting of five different areas from BBC sport websites, including football, athletics, cricket, rugby and tennis news, illustrated from two distinct views.
- **MNIST**[4]**:** It is a well-known dataset of handwritten digits, where three types of features are extracted: 30-dimension IsoProjection, 9-dimension Linear Discriminant Analysis (LDA) and 9-dimension Neighborhood Preserving Embedding (NPE) features.
- **Wikipedia**[5]**:** It is an article dataset that consists of 693 documents with 10 categories, which was crawled from Wikipedia website. Each entry is represented as two textual feature representations.
- **MSRC-v1**[6]**:** It is a well-known image dataset with totally eight classes. Following previous work, a subset of this dataset with seven classes is applied. There are five visual features extracted from each image in sum: 24-D color moment, 576-D Histogram of Oriented Gradients (HOG), 512-D GIST, 256-D local binary pattern and 256-D CENTRIST features.
- **Reuters**[7]**:** This is a subset of Reuters dataset containing feature representations of documents that were written in five different languages: English, French, German, Spanish and Italian. Each language corresponds to an independent view.

A statistical summary of these datasets is presented in Table 1, including the numbers of views, features and classes.

4.1.2. Compared methods

We compare the performance of the proposed LGCN-FF with the following baselines: KNN, SVM, AdaBoost, AMGL [55], MVAR [56], MLAN [51], AWDR [57], HLR-**M**^{2}VS [58], ERL-MVSC [59], GCN

1 http://aloi.science.uva.nl.
2 http://mlg.ucd.ie/datasets/segment.html.
3 http://mlg.ucd.ie/datasets/bbc.html.
4 http://yann.lecun.com/exdb/mnist/.
5 http://www.svcl.ucsd.edu/projects/crossmodal/.
6 http://riemenschneider.hayko.at/vision/dataset/task.php?did=35.
7 http://archive.ics.uci.edu/ml/machine-learning-databases/00259/.

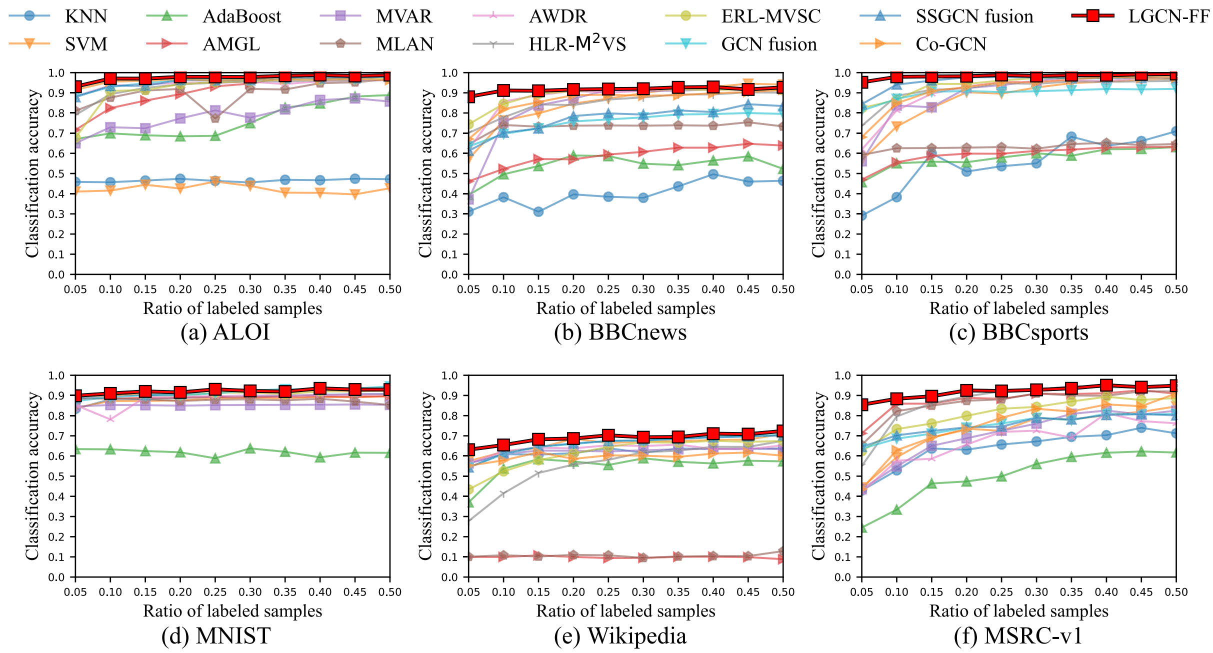
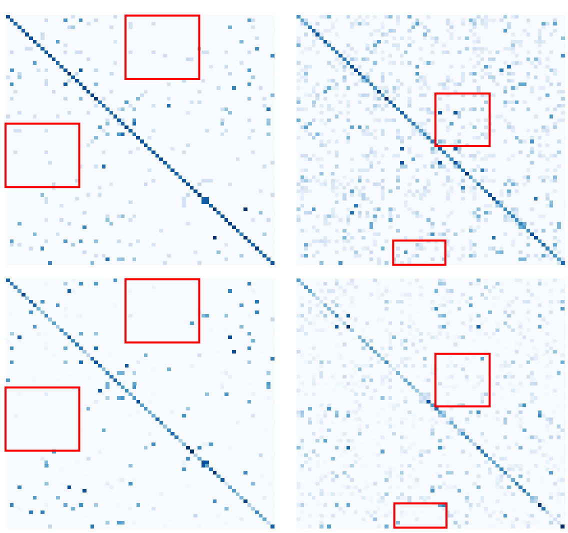


Fig. 3. The varied performance of all compared methods as the ratio of labeled data ranges in $\{0.05, 0.10, \ldots, 0.50\}$ on ALOI, BBCnews, BBCsports, MNIST, Wikipedia and MSRC-v1 datasets.



(a) ALOI (b) MSRC-v1

Fig. 4. Visualization of weighted adjacency matrices (the first row) and the adjacency matrices learned by LGCN-FF (the second row) with selected datasets, where darker colors indicate higher element values of matrices. This figure only exhibits partial adjacency matrices for a better presentation, where red boxes highlight node connections that diminish or disappear.

fusion [5], SSGCN fusion [27] and Co-GCN [16]. Most of these compared methods are graph-oriented algorithms, among which GCN fusion, SSGCN fusion and Co-GCN are based on GCN. Because the original GCN and SSGCN models are not able to directly process multi-view data, we compute the average adjacency matrix before graph convolutions. It is noted that Co-GCN is a state-of-the-art GCN-based framework for multi-view learning. Actually, only three GCN-based methods are involved in experiments, attributed to the fact that limited work has focused on GCN conducting downstream classification tasks with multi-view data. This also amplifies the contribution of this work.

4.1.3. Parameter settings

For most parameter settings, we follow the original settings of compared methods if feasible. Note that AMGL is a parameter-free framework thus we do not need to predefine extra hyperparameters. In particular, some parameter settings for compared methods are empirically set for better performance, as follows:

- **MVAR**: the trade-off weight for each view is tuned as $\lambda = 1000$, and the redistribution parameter over views is set as $r = 2$;
- **MLAN**: the number of adaptive neighbors is tuned in $[1, 10]$;
- **AWDR**: the trade-off parameter is fixed as $\lambda = 1.0$;
- **HLR-M^{2}VS**: weighted factors are set as $\lambda_1 = 0.2$ and $\lambda_2 = 0.4$;
- **ERL-MVSC**: hyperparameters are set as $\alpha = 2$ and $\beta = \gamma = 1$.
- **GCN fusion** and **SSGCN fusion**: a 2-layer GCN is employed and the learning rate is set as 0.001;
- **Co-GCN**: the settings of the convolutional layers and learning rate are the same as those in GCN fusion.

As to LGCN-FF, we empirically adopt sparse autoencoders with the dimensions of latent representations selected from $\{256, 512, 1024, 2048\}$. Adam optimizer is employed to update all learnable parameters with learning rate $lr = 0.01$ for the feature fusion network and learnable GCN. For all sparse autoencoders the learning rate is set to $lr = 0.001$. We utilize ℓ_2-norm as regularization for all learnable parameters and set weight decay as 0.01. Activation functions of the learnable GCN and the fully-connected network are set as $\mathrm{ReLU}(\cdot)$. Sigmoid function is adopted as the activation function for sparse autoencoders. Initial adjacency matrices are constructed by KNN. The dropout rate of learnable GCN is set as 0.3. The default setting for hyperparameter controlling the sparsity penalty degree is $\beta = 1$. The maximum number of iterations is set as 500. In this paper, the proposed LGCN-FF framework is implemented by PyTorch platform and run on the machine with R9-5900X CPU, Nvidia RTX 3060 GPU and 32G RAM.

4.2. Semi-supervised classification

Classification Results: The performance of all compared methods with 10% randomly labeled data is presented in Table 2, where the

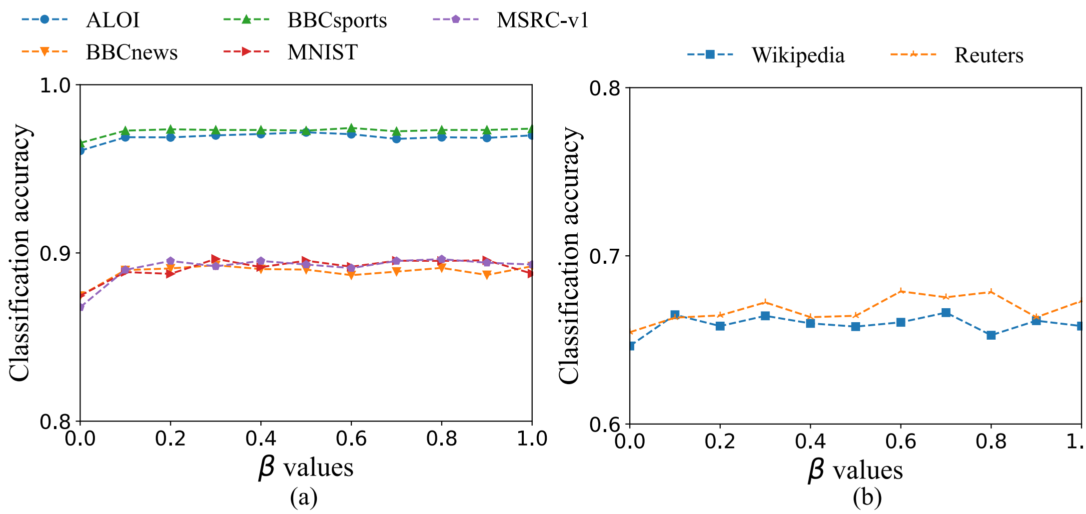


Fig. 5. Classification accuracy of LGCN-FF with varying β values on (a) ALOI, BBCsports, BBCnews, MSRC-v1 and MNIST, (b) Wikipedia and Reuters datasets.

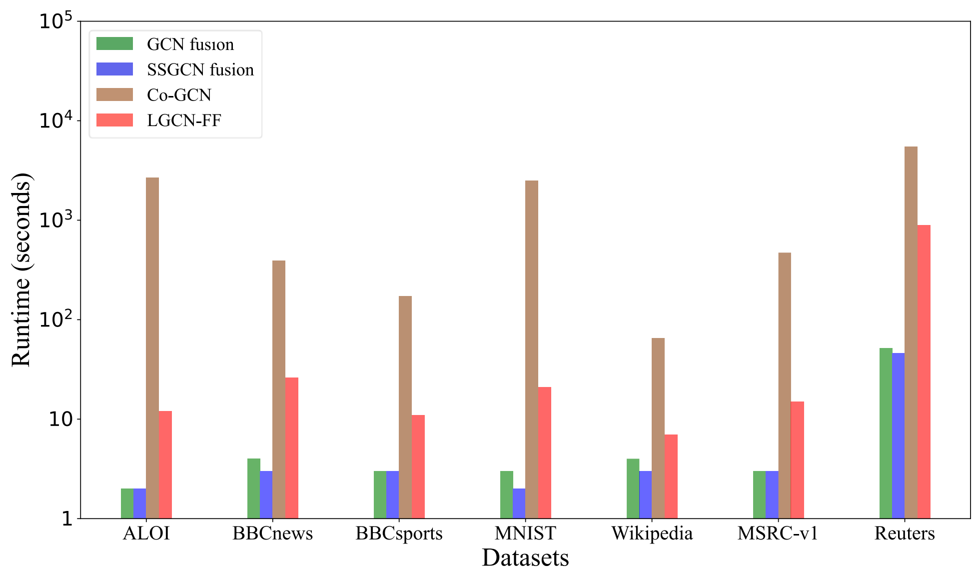


Fig. 6. Training time comparison between LGCN-FF and other GCN-based models.

classification accuracy is used as an evaluation metric. All methods are run 5 times and we record their average results and standard deviations. We only compute cross-entropy errors $\mathcal{L}_{lgcn}$ of the learnable GCN under the supervision of 10% labeled samples and evaluate the prediction performance with the rest 90% unlabeled data. The experimental results reveal that LGCN-FF reaches remarkable performance on all test datasets. Compared with GCN-based methods, the performance improvement is more considerable on BBCnews, BBCsports, MSRC-v1 and Reuters datasets. This observation suggests that the proposed LGCN-FF has stronger capacity of propagating node attributes among samples and extracting feature representations on relatively small datasets. Besides, Fig. 3 demonstrates the performance of all compared methods with various ratios of labeled samples. The experimental results show that LGCN-FF performs satisfactorily with relatively small supervision ratios (e.g., 5% or 10% labeled samples) on all datasets, and other algorithms generally require more supervision information to achieve comparable accuracy. The performance improvement is more significant on BBCnews, BBCsports and MSRC-v1 datasets. LGCN-FF also gains competitive accuracy with 5% labeled samples on MNIST dataset, and outperforms other methods with more labels. This indicates that LGCN-FF is more in line with the intention of semi-supervised classification. In a nutshell, the proposed framework gains superior performance compared with these state-of-the-art approaches.

Refined Adjacency Matrices: Fig. 4 presents the visualization of partial average weighted adjacency matrices and adjacency matrices learned by LGCN-FF. Compared with a direct weighted sum strategy, the adjacency matrices refined by the DSA function are relatively pure. It can be seen that some entries in learned adjacency matrices diminish or disappear, thereby resulting in sparser and more robust graphs. The learned adjacency matrix makes critical node relationships more pronounced, which is beneficial for the node embedding learning. The pleasurable performance of LGCN-FF also favors the superiority of the proposed framework.

Ablation Studies: In order to verify the effectiveness of the learnable GCN component, we also test the classification accuracy of the Weighted GCN-FF (WGCN-FF) that simply employs an average weighted adjacency matrix across all views. Besides, the performance of Adaptive WGCN-FF (AWGCN-FF) is also recorded, where it learns weights of different adjacency matrices automatically and then directly utilizes graph convolution operations without further refining. Actually, LGCN-FF is constructed based on AWGCN-FF, and adds a learnable DSA function $\rho(\cdot)$. Results of the ablation study are presented in Table 3. It is worth mentioning that the performance of GCN fusion in Table 2

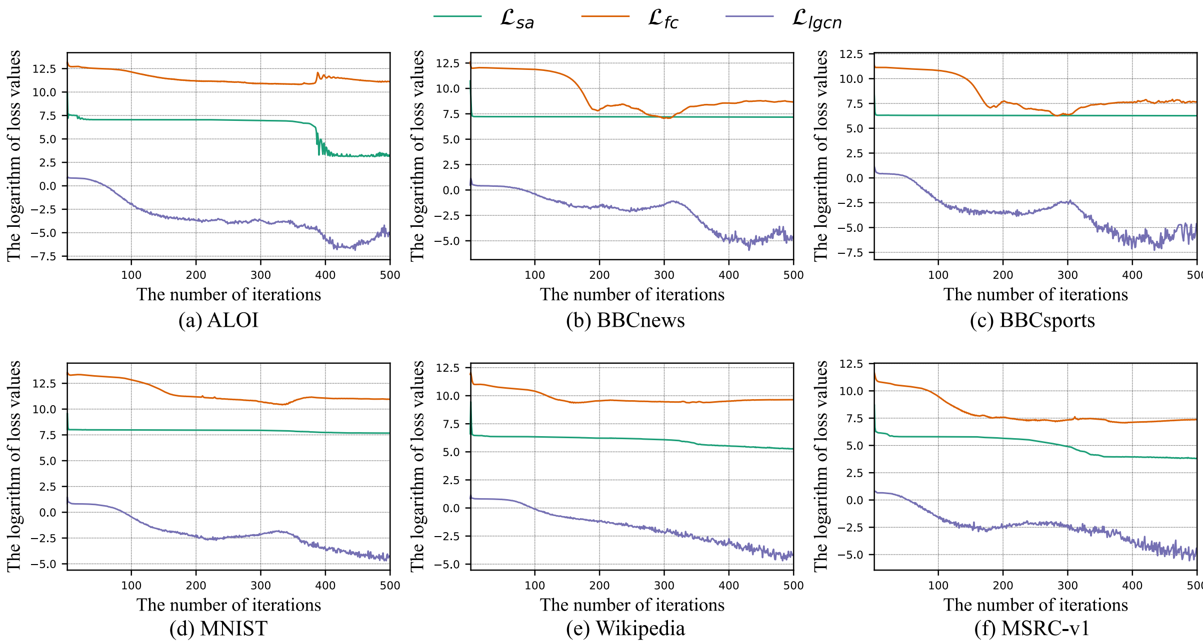


Fig. 7. Convergence curves of logarithm loss values measured by $\mathcal{L}_{sa}$, $\mathcal{L}_{fc}$ and $\mathcal{L}_{lgcn}$.

Table 3
Ablation study of the proposed LGCN-FF on all test datasets, where average accuracy (%) and standard deviation (%) are recorded.

Datasets\Methods	WGCN-FF	AWGCN-FF	LGCN-FF
ALOI	91.8 (1.5)	94.1 (2.3)	**97.1 (0.6)**
BBCnews	87.8 (1.1)	89.2 (1.3)	**91.2 (0.9)**
BBCsports	96.5 (0.3)	96.9 (0.3)	**98.2 (0.6)**
MNIST	88.0 (0.6)	88.8 (0.6)	**90.1 (0.7)**
Wikipedia	64.1 (1.5)	65.0 (1.6)	**67.4 (1.5)**
MSRC-v1	83.1 (2.4)	86.1 (2.3)	**90.4 (2.0)**
Reuters	61.4 (0.6)	65.1 (0.4)	**67.3 (0.5)**

can be regarded as the baseline accuracy. It can be observed that LGCN-FF succeeds in promoting the performance of the framework, which suggests the feasibility of the learnable GCN. This may be attributed to the reason that the learned adjacency matrix explores more discriminative relationships of nodes, and reduces the impact of noises generated by different views. Performance comparison also verifies that $\rho(\cdot)$ in LGCN-FF further promotes the accuracy of learning tasks.

Impact of β: Fig. 5 analyzes the impact of β with varying values, which is a coefficient controlling the sparsity penalty degree of sparse autoencoders in Eq. (4). Experimental results reveal that the accuracy of LGCN-FF fluctuates slightly as β changes on all datasets. Nonetheless, it is observed that the accuracy declines marginally when β decreases to 0. Namely, a vanilla autoencoder leads to undesired performance. This indicates that suitable sparseness is beneficial for exploring hidden representations via sparse autoencoders.

Runtime Comparison: We compare the training time of the proposed LGCN-FF and other GNN-based methods in Fig. 6. Because LGCN-FF has more modules like feature fusion networks, it requires more time for training compared with GCN fusion and SSGCN fusion that only consider a simple weighted sum of different graphs. Nonetheless, the running time of LGCN-FF is still fast and acceptable. Compared with Co-GCN which also focuses on multi-view learning, it gains leading performance in terms of training time. Therefore, the proposed LGCN-FF is an efficient framework addressing multi-view data.

Convergence Analyses: Fig. 7 shows the convergence of the proposed LGCN-FF. Because there are three loss functions in the framework, we plot logarithm loss values in a single figure for a better presentation. For simplicity, we draw curves of $\mathcal{L}_{sa} = \sum_{v=1}^{V} \mathcal{L}_{sa}^{(v)}$. From all subfigures in Fig. 7, we have the following beneficial observations. First of all, the loss of $\mathcal{L}_{sa}$ generally declines dramatically within 10 iterations, because of which the curve of $\mathcal{L}_{sa}$ tends to be a vertical line in the beginning of training. Then it converges slightly on most datasets except ALOI, where the loss value drops considerably around 380 iterations. Second, the loss value of $\mathcal{L}_{fc}$ starts to decrease significantly after a period of training (after 100 iterations in most cases), attributed to the fact that the feature fusion network learns discriminative fusion features when the latent embeddings learned by sparse autoencoders are relatively fixed. It is notable that the feature fusion network aims to seek a trade-off among multiple views, because of which the loss value may be relatively higher. Third, there is an interesting phenomenon that the value of $\mathcal{L}_{fc}$ may bounce marginally. On the contrary, the loss of $\mathcal{L}_{lgcn}$ may plunge almost in the same time on some datasets. This observation is more significant on ALOI (around 380 iterations), BBCnews (around 320 iterations), BBCsports (around 300 iterations) and MNIST (around 330 iterations). One reasonable explanation is that the proposed collaborative training procedure allows the feature fusion network to refine shared embeddings with promising generalization capacity via the flexible optimization, thereby leading to further improvement on the accuracy of downstream tasks. Although the value of $\mathcal{L}_{fc}$ may not reach the lowest point, the learned feature fusion is a better trade-off among multifarious views. Last but not the least, the value of $\mathcal{L}_{lgcn}$, which is directly related to the performance of downstream classification tasks, reaches the lowest point within 500 iterations on all datasets. The values of $\mathcal{L}_{lgcn}$ may fluctuate in the late period of training, owing to the varying features generated by the previous feature fusion network. Because the classification accuracy of GCN is tightly related to the input features, the cross entropy loss is sensitive to the quality of feature fusion. However, $\mathcal{L}_{lgcn}$ generally converges and it fluctuates in a small range. It is noted that the fluctuation of $\mathcal{L}_{lgcn}$ may be amplified when it is within $[0, 1]$, because we adopt the logarithm of loss values for better presentation. Actually, the fluctuation is marginal. We can terminate the network training early when the value of loss $\mathcal{L}_{lgcn}$ does not continue to drop for several iterations.

5. Conclusion

In this paper, we proposed an end-to-end neural network framework dubbed LGCN-FF which solved the multi-view learning problem with a learnable GCN and a feature fusion network. In the feature fusion network, multiple sparse autoencoders and a fully-connected network were utilized to fuse features from different views and study a unique underlying representation containing characteristics from all views. The graph fusion procedure was conducted by the learnable GCN that adaptively integrated multiple topological graphs from multifarious views. In addition, a learnable DSA function was proposed to learn a more robust shared adjacency matrix, which promoted the performance of LGCN-FF. Finally, the proposed framework divided the optimization target into several subproblems and jointly learned feature and graph fusion representations with a multi-step optimization strategy. Experimental results validated the superiority of the proposed framework in terms of multi-view semi-supervised classification tasks.

In the future, our work can be improved from the following directions. The proposed model concentrates on undirected graphs, while node relationships in real-world applications are more likely to be directed graphs. Thus, we can further investigate the GCN-based methods that process directed graphs. Because there is no natural topological network for most data in real-world applications, the graph information adopted in this paper is established via the KNN algorithm. It would be helpful if a new graph learning framework which automatically estimates node relations can be developed. In the future, we will devote more effort to feasible graph fusion learning with multi-view data.

CRediT authorship contribution statement

Zhaoliang Chen: Conceptualization, Formal analysis, Methodology, Writing – original draft. **Lele Fu:** Conceptualization, Formal analysis, Methodology, Writing – revision. **Jie Yao:** Validation, Visualization. **Wenzhong Guo:** Funding acquisition, Resources, Supervision. **Claudia Plant:** Formal analysis, Writing – review & editing. **Shiping Wang:** Funding acquisition, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Nos. U21A20472 and 61672159).

References

[1] M.-S. Chen, L. Huang, C.-D. Wang, D. Huang, J.-H. Lai, Relaxed multi-view clustering in latent embedding space, Inf. Fusion 68 (2021) 8–21.
[2] J. Wang, C. Tang, Z. Li, X. Liu, W. Zhang, E. Zhu, L. Wang, Hyperspectral band selection via region-aware latent features fusion based clustering, Inf. Fusion 79 (2022) 162–173.
[3] Y. Chen, S. Wang, C. Peng, Z. Hua, Y. Zhou, Generalized nonconvex low-rank tensor approximation for multi-view subspace clustering, IEEE Trans. Image Process. 30 (2021) 4022–4035.
[4] C. Tang, X. Zhu, X. Liu, M. Li, P. Wang, C. Zhang, L. Wang, Learning a joint affinity graph for multiview subspace clustering, IEEE Trans. Multimedia 21 (7) (2019) 1724–1736.
[5] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proceedings of the 5th International Conference on Learning Representations, 2017.
[6] L. Yao, C. Mao, Y. Luo, Graph convolutional networks for text classification, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, 2019, pp. 7370–7377.
[7] Y.-H. Chen, L. Huang, C.-D. Wang, J.-H. Lai, Hybrid-order gated graph neural network for session-based recommendation, IEEE Trans. Ind. Inform. 18 (3) (2022) 1458–1467.
[8] L. Bai, L. Cui, Y. Jiao, L. Rossi, E.R. Hancock, Learning backtrackless aligned-spatial graph convolutional networks for graph classification, IEEE Trans. Pattern Anal. Mach. Intell. 44 (2) (2022) 783–798.
[9] Y.S. Rawat, M. Song, M.S. Kankanhalli, A spring-electric graph model for socialized group photography, IEEE Trans. Multimedia 20 (3) (2018) 754–766.
[10] Y. Chen, X. Xiao, C. Peng, G. Lu, Y. Zhou, Low-rank tensor graph learning for multi-view subspace clustering, IEEE Trans. Circuits Syst. Video Technol. 32 (1) (2022) 92–104.
[11] S.-H. Wang, V.V. Govindaraj, J.M. Górriz, X. Zhang, Y.-D. Zhang, Covid-19 classification by fgcnet with deep feature fusion from graph convolutional network and convolutional neural network, Inf. Fusion 67 (2021) 208–229.
[12] A. Mosella-Montoro, J. Ruiz-Hidalgo, 2D–3d geometric fusion network using multi-neighbourhood graph convolution for rgb-d indoor scene classification, Inf. Fusion 76 (2021) 46–54.
[13] Z. Zhang, C. Chen, Y. Chang, W. Hu, X. Xing, Y. Zhou, Z. Zheng, Shne: Semantics and homophily preserving network embedding, IEEE Trans. Neural Netw. Learn. Syst. (2021) http://dx.doi.org/10.1109/TNNLS.2021.3116936.
[14] X. Zhou, F. Shen, L. Liu, W. Liu, L. Nie, Y. Yang, H.T. Shen, Graph convolutional network hashing, IEEE Trans. Cybern. 50 (4) (2020) 1460–1472.
[15] Y. Zhu, J. Ma, C. Yuan, X. Zhu, Interpretable learning based dynamic graph convolutional networks for alzheimer's disease analysis, Inf. Fusion 77 (2022) 53–61.
[16] S. Li, W. Li, W. Wang, Co-gcn for multi-view semi-supervised learning, in: Proceedings of the 34th AAAI Conference on Artificial Intelligence, 2020, pp. 4691–4698.
[17] J. Cheng, Q. Wang, Z. Tao, D. Xie, Q. Gao, Multi-view attribute graph convolution networks for clustering, in: Proceedings of the 29th International Joint Conference on Artificial Intelligence, 2020, pp. 2973–2979.
[18] Z. Li, C. Tang, X. Liu, X. Zheng, G. Yue, W. Zhang, E. Zhu, Consensus graph learning for multi-view clustering, IEEE Trans. Multimedia (2021) http://dx.doi.org/10.1109/TMM.2021.3081930.
[19] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, J. Pei, AM-GCN: adaptive multi-channel graph convolutional networks, in: Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2020, pp. 1243–1253.
[20] Z. Huang, J.T. Zhou, H. Zhu, C. Zhang, J. Lv, X. Peng, Deep spectral representation learning from multi-view data, IEEE Trans. Image Process. 30 (2021) 5352–5362.
[21] K. Gregor, Y. LeCun, Learning fast approximations of sparse coding, in: Proceedings of the 27th International Conference on Machine Learning, 2010, pp. 399–406.
[22] J.-F. Cai, E.J. Candès, Z. Shen, A singular value thresholding algorithm for matrix completion, SIAM J. Optim. 20 (4) (2010) 1956–1982.
[23] Q. Li, Z. Han, X. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, 2018, pp. 3538–3545.
[24] X. Xu, T. Wang, Y. Yang, A. Hanjalic, H.T. Shen, Radial graph convolutional network for visual question generation, IEEE Trans. Neural Netw. Learn. Syst. 32 (4) (2021) 1654–1667.
[25] K. Liu, L. Gao, N.M. Khan, L. Qi, L. Guan, A multi-stream graph convolutional networks-hidden conditional random field model for skeleton-based action recognition, IEEE Trans. Multimedia 23 (2021) 64–76.
[26] D. Bo, X. Wang, C. Shi, H. Shen, Beyond low-frequency information in graph convolutional networks, in: Proceedings of the 35th AAAI Conference on Artificial Intelligence, 2021, pp. 3950–3957.
[27] H. Zhu, P. Koniusz, Simple spectral graph convolution, in: Proceedings of the 9th International Conference on Learning Representations, 2021, pp. 1–11.
[28] Y. Guo, L. Gao, J. Song, P. Wang, N. Sebe, H.T. Shen, X. Li, Relation regularized scene graph generation, IEEE Trans. Cybern. (2021) 1–12.
[29] Z. Huang, X. Li, Y. Ye, M.K. Ng, Mr-gcn: Multi-relational graph convolutional networks based on generalized tensor product, in: Proceedings of the 39th International Joint Conference on Artificial Intelligence, 2020, pp. 1258–1264.
[30] M. Lei, P. Quan, R. Ma, Y. Shi, L. Niu, Diggcn: Learning compact graph convolutional networks via diffusion aggregation, IEEE Trans. Cybern. (2020) 1–13.
[31] K. Sun, Z. Lin, 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, 2020, pp. 5892–5899.
[32] B.-Y. Liu, L. Huang, C.-D. Wang, J.-H. Lai, P. Yu, Multi-view consensus proximity learning for clustering, IEEE Trans. Knowl. Data Eng. (2020) http://dx.doi.org/10.1109/TKDE.2020.3025759.
[33] J. Zhao, X. Xie, X. Xu, S. Sun, Multi-view learning overview: Recent progress and new challenges, Inf. Fusion 38 (2017) 43–54.
[34] S. Sun, A survey of multi-view machine learning, Neural Comput. Appl. 23 (7) (2013) 2031–2038.
[35] S. Sun, Multi-view laplacian support vector machines, in: International Conference on Advanced Data Mining and Applications, 2011, pp. 209–222.
[36] Z. Xu, S. Sun, An algorithm on multi-view adaboost, in: International Conference on Neural Information Processing, 2010, pp. 355–362.
[37] J. Li, B. Zhang, G. Lu, D. Zhang, Generative multi-view and multi-feature learning for classification, Inf. Fusion 45 (2019) 215–226.

[38] Y. Chen, X. Xiao, Z. Hua, Y. Zhou, Adaptive transition probability matrix learning for multiview spectral clustering, IEEE Trans. Neural Netw. Learn. Syst. (2021) 1–15, http://dx.doi.org/10.1109/TNNLS.2021.3059874.
[39] A. Huang, T. Zhao, C. Lin, Multi-view data fusion oriented clustering via nuclear norm minimization, IEEE Trans. Image Process. 29 (2020) 9600–9613.
[40] X. Liu, X. Zhu, M. Li, L. Wang, C. Tang, J. Yin, D. Shen, H. Wang, W. Gao, Late fusion incomplete multi-view clustering, IEEE Trans. Pattern Anal. Mach. Intell. 41 (10) (2019) 2410–2423.
[41] C. Zhang, Y. Liu, H. Fu, Ae2-nets: Autoencoder in autoencoder networks, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 2577–2585.
[42] J. Wen, Z. Zhang, Z. Zhang, L. Fei, M. Wang, Generalized incomplete multiview clustering with flexible locality structure diffusion, IEEE Trans. Cybern. 51 (1) (2021) 101–114.
[43] Y. Li, M. Yang, Z. Zhang, A survey of multi-view representation learning, IEEE Trans. Knowl. Data Eng. 31 (10) (2018) 1863–1883.
[44] S. Wang, Z. Chen, S. Du, Z. Lin, Learning deep sparse regularizers with applications to multi-view clustering and semi-supervised classification, IEEE Trans. Pattern Anal. Mach. Intell. 44 (9) (2022) 5042–5055.
[45] S. Sun, J. Shawe-Taylor, Sparse semi-supervised learning using conjugate functions, J. Mach. Learn. Res. 11 (2010) 2423–2455.
[46] S. Du, Z. Liu, Z. Chen, W. Yang, S. Wang, Differentiable bi-sparse multi-view co-clustering, IEEE Trans. Signal Process. 69 (2021) 4623–4636.
[47] S. Zhou, E. Zhu, X. Liu, T. Zheng, Q. Liu, J. Xia, J. Yin, Subspace segmentation-based robust multiple kernel clustering, Inf. Fusion 53 (2020) 145–154.
[48] C. Tang, X. Liu, X. Zheng, W. Li, J. Xiong, L. Wang, A.Y. Zomaya, A. Longo, Defusionnet: Defocus blur detection via recurrently fusing and refining discriminative multi-scale deep features, IEEE Trans. Pattern Anal. Mach. Intell. 44 (2) (2022) 955–968.
[49] F. Nie, J. Li, X. Li, Self-weighted multiview clustering with multiple graphs, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 2564–2570.
[50] A. Huang, T. Zhao, C. Lin, Multi-view data fusion oriented clustering via nuclear norm minimization, IEEE Trans. Image Process. 29 (2020) 9600–9613.
[51] F. Nie, G. Cai, X. Li, Multi-view clustering and semi-supervised classification with adaptive neighbours, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, 2017, pp. 2408–2414.
[52] X. He, Q. Liu, Y. Yang, Mv-gnn: Multi-view graph neural network for compression artifacts reduction, IEEE Trans. Image Process. 29 (2020) 6829–6840.
[53] H. Wang, Y. Yang, B. Liu, Gmc: Graph-based multi-view clustering, IEEE Trans. Knowl. Data Eng. 32 (6) (2020) 1116–1129.
[54] Z. Lin, R. Liu, Z. Su, Linearized alternating direction method with adaptive penalty for low-rank representation, in: Advances in Neural Information Processing Systems, 2011, pp. 612–620.
[55] F. Nie, J. Li, 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, 2016, pp. 1881–1887.
[56] H. Tao, C. Hou, F. Nie, J. Zhu, D. Yi, Scalable multi-view semi-supervised classification via adaptive regression, IEEE Trans. Image Process. 26 (9) (2017) 4283–4296.
[57] M. Yang, C. Deng, F. Nie, Adaptive-weighting discriminative regression for multi-view classification, Pattern Recognit. 88 (2019) 236–245.
[58] Y. Xie, W. Zhang, Y. Qu, L. Dai, D. Tao, Hyper-laplacian regularized multilinear multiview self-representations for clustering and semisupervised learning, IEEE Trans. Cybern. 50 (2) (2020) 572–586.
[59] A. Huang, Z. Wang, Y. Zheng, T. Zhao, C. Lin, Embedding regularizer learning for multi-view semi-supervised classification, IEEE Trans. Image Process. 30 (2021) 6997–7011.