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

Zhaoliang Chen\textsuperscript{a,b}, Lele Fu\textsuperscript{c}, Jie Yao\textsuperscript{a,b}, Wenzhong Guo\textsuperscript{a,b}, Claudia Plant\textsuperscript{d,e}, Shiping Wang\textsuperscript{a,b,*}

\textsuperscript{a} College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China
\textsuperscript{b} Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China
\textsuperscript{c} School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou 510260, China
\textsuperscript{d} Faculty of Computer Science, University of Vienna, Vienna 1090, Austria
\textsuperscript{e} dcUniVie, Vienna 1090, Austria

\begin{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.
\end{abstract}

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...
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 $x \in \mathbb{R}^n$ and a filter $g_\theta = \text{diag}(\theta)$, formulated as

$$g_\theta \ast x = U g_\theta U^T x,$$

(1)

where $U$ 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 $l$th layer of a spectral GCN is formally defined as

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

(2)

where $\tilde{A} = A + I$ denotes the adjacency matrix considering the self-connections, and $[D]_{ii} = \sum_j [A]_{ij}$. Layer-specific weight matrix is denoted by $W^{(l)}$. 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]. AE2-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 $X = [X^{(1)}, \ldots, X^{(V)}]$ and the scarce supervision signal $Y \in \mathbb{R}^{[m \times n]}$, where $X^{(i)} \in \mathbb{R}^{v_i \times n}$ denotes the features of the $i$th view with totally $V$ views, $Y$ is the incomplete binary label matrix generated from the labeled sample set $\Omega$ 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 space.
optimization strategy is employed for updating \( \hat{H} \) as the shared node representation in the learnable GCN. A two-step trainable fully-connected network. In fact, the trainable \( G \) a total of \( L \) layers in the fully-connected neural network, the forward propagation in the \( l \)th layer is computed by
\[
G^{(l)} = \sigma \left( G^{(l-1)} W^{(l)}_{f_c} + b^{(l)}_{f_c} \right),
\]
where \( G^{(0)} = H \). The matrix \( H \) is also a learnable input updated by gradient descent and back propagation techniques. We project the learned \( H \) onto various view-specific latent features \( \{O^{(v)} \}_{v=1}^V \) via a trainable fully-connected network. In fact, the trainable \( H \) also serves as the shared node representation in the learnable GCN. A two-step optimization strategy is employed for updating \( \{W^{(l)}_{f_c}, b^{(l)}_{f_c}\}_{l=1}^L \) and \( H \).

Both two steps share the same reconstruction loss function as defined below:
\[
\mathcal{L}_{fc} = \frac{1}{2} \sum_{v=1}^V \left\| G^{(v)} - O^{(v)} \right\|_2^2 + \beta D_{KL}(\phi(\tilde{d})||\phi(d)).
\]

With the assumption that features of each single view can be rebuilt from the intact common representation \( H \) 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
\[
A_v = \sum_{i=1}^V \pi(v) A^{(v)}_i,
\]
where \( A^{(v)}_i = (\tilde{D}^{(v)})^{-\frac{1}{2}} A^{(v)} (\tilde{D}^{(v)})^{-\frac{1}{2}} \) is the initial renormalization adjacency matrix of the \( v \)th 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 \( \sum_{i=1}^V \pi(i) = 1 \), we employ the softmax renormalization at each epoch as
\[
\pi^{(v)} \leftarrow \frac{\exp \left( \pi^{(v)} \right)}{\sum_{i=1}^V \exp \left( \pi^{(v)} \right)}
\]
for \( v = 1, \ldots, V \).
Moreover, 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 ρ(⋅) 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 ρ(A) should also be symmetric. To this end, the learnable DSA function ρ(⋅) is defined as

\[ \rho(A) = A \odot \mathrm{ReLU}(S - \Theta), \]  

(10)

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

\[ S = \text{Sigmoid}\left( \frac{1}{2}(S + S^T) \right), \]  

(11)

which is parameterized by a learnable matrix \( \hat{S} \in \mathbb{R}^{m \times m} \). 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 \( \Theta \) 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

\[ \{\Theta\}_{i,j} = (\Theta)_{i,j} = \text{Sigmoid}(\theta_i), \quad \forall i \leq j \leq m \]  

(12)

with \( \theta = [\theta_1, \ldots, \theta_m] \), where \( \theta \) is a learnable vector and \text{Sigmoid}(\cdot) admits the non-negativity of thresholds. Consequently, \( \Theta \) 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 operation 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 \( \hat{S} \) randomly to compute \( S \), and initialize \( \Theta \) as a vector to generate \( \Theta \) in the beginning of training. With these previous analyses, the \( l \)th layer of the learnable GCN is formulated by

\[ H^{(l)} = \sigma \left( \rho(A)H^{(l-1)}W^{(l)}_{\theta \Theta} \right), \]  

(13)

where \( H^{(0)} = H \). Namely, the trainable \( H \) 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 \( Z \) with

\[ Z = \text{softmax} \left( \rho(A) \sigma \left( \rho(A)H^{(1)}W^{(1)}_{\theta \Theta} \right)W^{(2)}_{\theta \Theta} \right). \]  

(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 \( \Omega \), as shown below:

\[ L_{\text{gcn}} = - \sum_{i,j=1}^{c} \sum_{l \in L} Y_{ij} \log Z_{ij}, \]  

(15)

where \( Y \in \mathbb{R}^{|\mathcal{X}| \times c} \) is the incomplete label matrix generated from \( \Omega \) satisfying \( |\Omega| \ll m \).

### 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 \( H \), 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

---

**Algorithm 1 Training Framework of LGCN-FF**

**Input:** Multi-view data \( \mathcal{X} = \{X^{(1)}, \ldots, X^{(p)}\} \) and semi-supervised information \( Y \in \mathbb{R}^{|\mathcal{X}| \times c} \).

**Output:** Node embedding \( Z \).

1. Initialize weights and biases of sparse autoencoders;
2. Initialize weights, biases and learnable input \( H \) of fully-connected networks;
3. Initialize learnable weights \( \{x^{(s)} = \frac{1}{p}V_{s}\}_{s=1}^{p} \); \( S \in \mathbb{R}^{m \times m} \) and \( \theta = [\theta_1, \ldots, \theta_m] \) of learnable GCN;
4. Initialize adjacency matrices \( A^{(1)}, \ldots, A^{(p)} \) via KNN;
5. while not convergent do
6. for \( v = 1 \rightarrow V \) do
7. Compute \( O^{(s)}_{v} \) and \( O^{(L,v)}_{v} \) of the \( v \)th sparse autoencoder with Equation (3);
8. Update \( W^{(L,v)}_{ij}, b^{(L,v)}_{i} \); \( l = 1 \) with back propagation;
9. end for
10. for \( v = 1 \rightarrow V \) do
11. Compute \( G^{(L)}_{v} \) of the fully-connected network with Equation (6);
12. end for
13. Update \( H \) with back propagation;
14. Compute \( Z = H^{(L)} \) of the learnable GCN with Equation (13);
15. Update \( W^{(l)}_{ij}, b^{(l)}_{i} \); \( l = 1 \) with back propagation;
20. end while
21. return Node embedding \( Z \).
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 are listed as follows.

At each iteration, given multi-view data \( \{X^{(v)}_m\}_{m=1}^V \) with \( V \) views, the computational complexity for sparse autoencoders and the feature fusion network is \( \mathcal{O}(2V mn + md^2) \) as 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 performance evaluation, as listed below:

- **ALOI**: This is an image dataset that contains objects that are taken under varied light conditions or rotation angles. Multi-view features including 64-D HSV color histograms, 64-D color similarities and 13-D Haralick features are extracted from each image in sum: 24-D color moment, 576-D Histogram Neighborhood Preserving Embedding (NPE) features.

- **BBCnews**: It is a collection of news reports which covers political topics, entertainment, business, sport and technology fields. There are totally four different textual features extracted from various segments to describe the news.

- **BBseconds**: 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**: 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**: 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**: 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), S12-D GIST, 256-D local binary pattern and 256-D CENTRIST features.

- **Reuters**: This 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.

#### 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-MFVs [58], ERL-MVSC [59], GCN fusion [5], SSAGCN fusion [27], GCN fusion [5], HLR-MFVs [58], AWDR [57], MLAN [51], HLR-MVSC [59], AMGL [55], AdaBoost 69.9 (9.5) 49.6 (6.6) 55.1 (9.8) 63.3 (6.0) 53.1 (0.8) 33.5 (5.2) 54.0 (0.2) 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) AWDR 52.3 (5.5) 55.6 (1.4) 88.5 (0.2) 10.1 (0.8) 85.9 (1.8) –

### Table 1

<table>
<thead>
<tr>
<th>Datasets</th>
<th># Samples</th>
<th># Views</th>
<th># Features</th>
<th># Classes</th>
<th>Data types</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALOI</td>
<td>1,079</td>
<td>4</td>
<td>64/64/77/13</td>
<td>10</td>
<td>Object images</td>
</tr>
<tr>
<td>BBCnews</td>
<td>685</td>
<td>4</td>
<td>6,659/4,633/4,665/4,684</td>
<td>5</td>
<td>Textual documents</td>
</tr>
<tr>
<td>BBsports</td>
<td>544</td>
<td>2</td>
<td>3,183/3,203</td>
<td>5</td>
<td>Textual documents</td>
</tr>
<tr>
<td>MNIST</td>
<td>10,000</td>
<td>3</td>
<td>30/9/9</td>
<td>10</td>
<td>Digit images</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>693</td>
<td>2</td>
<td>128/10</td>
<td>10</td>
<td>Textual documents</td>
</tr>
<tr>
<td>MSRC-v1</td>
<td>210</td>
<td>5</td>
<td>24/576/512/256/254</td>
<td>7</td>
<td>Object images</td>
</tr>
<tr>
<td>Reuters</td>
<td>18,758</td>
<td>5</td>
<td>21,331/24,892/34,251/15,506/11,547</td>
<td>6</td>
<td>Textual documents</td>
</tr>
</tbody>
</table>

### 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 “*”.

<table>
<thead>
<tr>
<th>Datasets \ Methods</th>
<th>ALOI</th>
<th>BBCnews</th>
<th>BBCsports</th>
<th>MNIST</th>
<th>Wikipedia</th>
<th>MSRC-v1</th>
<th>Reuters</th>
</tr>
</thead>
<tbody>
<tr>
<td>KNN</td>
<td>45.7 (3.1)</td>
<td>38.3 (9.6)</td>
<td>38.3 (9.5)</td>
<td>88.8 (0.4)</td>
<td>58.7 (3.0)</td>
<td>52.9 (8.8)</td>
<td>34.4 (0.5)</td>
</tr>
<tr>
<td>SVM</td>
<td>41.5 (6.6)</td>
<td>76.7 (7.6)</td>
<td>73.4 (4.9)</td>
<td>87.2 (1.3)</td>
<td>62.4 (4.2)</td>
<td>58.9 (5.9)</td>
<td>48.7 (0.3)</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>69.9 (9.5)</td>
<td>49.6 (6.6)</td>
<td>55.1 (9.8)</td>
<td>63.3 (6.0)</td>
<td>53.1 (3.4)</td>
<td>33.5 (5.2)</td>
<td>54.0 (0.2)</td>
</tr>
<tr>
<td>AMGL [55]</td>
<td>82.4 (3.3)</td>
<td>52.3 (5.5)</td>
<td>55.6 (1.4)</td>
<td>88.5 (0.2)</td>
<td>10.1 (0.8)</td>
<td>85.9 (1.8)</td>
<td>–</td>
</tr>
<tr>
<td>MVAR [56]</td>
<td>72.6 (5.5)</td>
<td>75.3 (5.5)</td>
<td>83.7 (3.8)</td>
<td>85.3 (0.8)</td>
<td>61.2 (3.4)</td>
<td>54.8 (7.5)</td>
<td>64.6 (0.3)</td>
</tr>
<tr>
<td>MLAN [51]</td>
<td>87.6 (1.6)</td>
<td>74.1 (0.9)</td>
<td>62.6 (2.2)</td>
<td>88.6 (0.3)</td>
<td>10.2 (0.8)</td>
<td>82.2 (5.4)</td>
<td>–</td>
</tr>
<tr>
<td>AWDR [57]</td>
<td>93.6 (1.9)</td>
<td>85.7 (1.4)</td>
<td>81.3 (3.3)</td>
<td>78.1 (0.3)</td>
<td>62.5 (5.8)</td>
<td>57.7 (7.0)</td>
<td>61.3 (0.6)</td>
</tr>
<tr>
<td>HLR-MFVs [58]</td>
<td>87.7 (1.7)</td>
<td>78.1 (2.8)</td>
<td>84.6 (0.4)</td>
<td>–</td>
<td>36.5 (3.4)</td>
<td>79.6 (8.4)</td>
<td>–</td>
</tr>
<tr>
<td>ERL-MVSC [59]</td>
<td>90.5 (2.7)</td>
<td>85.9 (2.2)</td>
<td>90.3 (1.9)</td>
<td>89.5 (0.3)</td>
<td>51.6 (2.3)</td>
<td>73.3 (3.7)</td>
<td>–</td>
</tr>
<tr>
<td>GCN fusion [5]</td>
<td>92.6 (1.2)</td>
<td>89.6 (1.8)</td>
<td>87.0 (2.0)</td>
<td>89.2 (0.5)</td>
<td>60.1 (0.6)</td>
<td>68.6 (7.2)</td>
<td>55.4 (0.3)</td>
</tr>
<tr>
<td>SSAGCN fusion [27]</td>
<td>93.4 (1.0)</td>
<td>89.7 (5.5)</td>
<td>94.1 (1.2)</td>
<td>89.3 (0.1)</td>
<td>62.1 (0.4)</td>
<td>70.3 (4.9)</td>
<td>56.8 (0.2)</td>
</tr>
<tr>
<td>Co-GCN [16]</td>
<td>96.5 (0.4)</td>
<td>81.9 (1.5)</td>
<td>84.8 (1.4)</td>
<td>89.9 (0.4)</td>
<td>57.9 (0.7)</td>
<td>62.9 (4.4)</td>
<td>60.2 (0.6)</td>
</tr>
<tr>
<td>LGCN-FF</td>
<td>97.1 (0.6)</td>
<td>91.2 (0.9)</td>
<td>98.2 (0.6)</td>
<td>90.2 (0.3)</td>
<td>67.4 (1.5)</td>
<td>90.4 (2.0)</td>
<td>67.3 (0.5)</td>
</tr>
</tbody>
</table>

Z. Chen et al.

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.

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-M²VS**: 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 \( \epsilon = 0.01 \) for the feature fusion network and learnable GCN. For all sparse autoencoders the learning rate is set to \( \epsilon = 0.001 \). We utilize \( \ell_2 \)-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 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 $\beta$ 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 $\mathcal{L}_{CE}$ 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 $\rho(.)$. Results of the ablation study are presented in Table 3. It is worth mentioning that the performance of GCN fusion in Table 2

\[ \mathcal{L}_{CE} \]
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. Sheping 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


