Neural Networks 145 (2022) 1–9

Contents lists available at ScienceDirect

Neural Networks

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

Multi-view graph embedding clustering network: Joint self-supervision and block diagonal representation

Wei Xia [a], Sen Wang [b], Ming Yang [c], Quanxue Gao [a,*], Jungong Han [d], Xinbo Gao [e]

[a] State Key Laboratory of Integrated Services Networks, Xidian University, Shaanxi 710071, China
[b] Beijing Aerospace Automatic Control Institute, Beijing 100854, China
[c] Departments of Mathematics and Computer & Information Science, Westfield State University, Westfield, MA 01086, United States of America
[d] Computer Science Department, Aberystwyth University, SY23 3FL, United Kingdom
[e] Chongqing Key Laboratory of Image Cognition, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

ARTICLE INFO

Article history:
Received 8 May 2021
Received in revised form 25 August 2021
Accepted 4 October 2021
Available online 25 October 2021

Keywords:
Multi-view clustering
Graph convolutional networks
Block diagonal representation
Self-supervision

ABSTRACT

Multi-view clustering has become an active topic in artificial intelligence. Yet, similar investigation for graph-structured data clustering has been absent so far. To fill this gap, we present a Multi-View Graph embedding Clustering network (MVGC). Specifically, unlike traditional multi-view construction methods, which are only suitable to describe Euclidean structure data, we leverage Euler transform to augment the node attribute, as a new view descriptor, for non-Euclidean structure data. Meanwhile, we impose block diagonal representation constraint, which is measured by the $\ell_{1,2}$-norm, on self-expression coefficient matrix to well explore the cluster structure. By doing so, the learned view-consensus coefficient matrix well encodes the discriminative information. Moreover, we make use of the learned clustering labels to guide the learnings of node representation and coefficient matrix, where the latter is used in turn to conduct the subsequent clustering. In this way, clustering and representation learning are seamlessly connected, with the aim to achieve better clustering performance. Extensive experimental results indicate that MVGC is superior to 11 state-of-the-art methods on four benchmark datasets. In particular, MVGC achieves an Accuracy of 96.17% (53.31%) on the ACM (IMDB) dataset, which is an up to 2.85% (1.97%) clustering performance improvement compared with the strongest baseline.

1. Introduction

As we embark on the Internet era, graph-structured data, such as data from citation networks, movie networks and social networks, are all around us. Due to the efficiency of revealing the intrinsic relationships of data, graph-structured data analysis has become prevalent in artificial intelligence community (Li, Müller, Ghanem, & Koltun, 2021; Nikolentzos, Dasoulas, & Vazirgiannis, 2020; Wu, Pan, Chen, Long, Zhang, & Yu, 2020; Xie, Zhang, Gong, Tang, & Han, 2020). To date, studies propose massive amounts of graph based learning methods, which can be divided into three main categories according to different input, *i.e.*, (1) taking the node attribute as input; (2) taking the graph structure as input; and (3) simultaneous taking the node attribute and graph structure as input. The first two categories of methods take the node attribute or graph structure as input, then learn new representation of data for downstream tasks (Kumar, Rai, & III, 2011; Luo, Huang, Ma, & Liu, 2016; Luo, Zhang, Du, & Zhang, 2020; Nie, Cai, Li, & Li, 2018; Wu, Lin, & Zha, 2019; Xie, Gao, Deng, Yang, & Gao, 2021). Although the first two categories have gained satisfactory results, they cannot simultaneously explore the information hidden in node attribute and graph structure, resulting in inferior latent representation of data. To this end, the third class represented by graph neural networks (GNNs) has become powerful tool for many machine learning tasks.

In view of the fact that manually labeling graph-structured data is time-consuming and expensive, in this paper, we study the GNNs based graph-structured data clustering problem. Clustering, which plays a crucial role in unsupervised learning, aims to divide data into several disjoint groups such that the data in the same group are similar to each other, while data in different groups are dissimilar. Numerous clustering methods have been presented, among which graph embedding clustering is one of the most representative clustering techniques due to its effectiveness in characterizing graph-structured data.

The goal of graph embedding is to learn a compact and continuous node representation. One of the most representative methods is graph auto-encoder (GAE) (Kipf & Welling, 2016). It encodes graph structure and node attribute to a node representation, on which a decoder is trained to reconstruct the graph

* Corresponding author.
E-mail address: qxgao@xidian.edu.cn (Q. Gao).

https://doi.org/10.1016/j.neunet.2021.10.006
0893-6080/

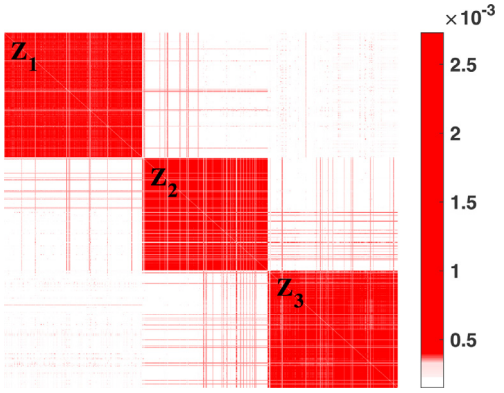


Fig. 1. Illustrations of block diagonal representation.

structure. To improve the robustness of node representation, Pan et al. (2018) incorporated adversarial learning to GAE and developed adversarial regularized graph auto-encoder (ARGE). However, both GAE and ARGE carry out node representation and clustering separately, thus resulting in suboptimal performance. To this end, Wang et al. (2019) integrated cluster centers learning and graph embedding into a unified framework, and proposed deep attentional embedded graph clustering method (DAEGC). Similarly, Bo, Wang, Shi, Zhu, Lu, and Cui (2020) proposed a structural deep embedded clustering network (SDCN), which is effective for node clustering.

Extensive studies have demonstrated that, compared with single-view data, multi-view data provide more complementary information embedded in different views (Gao, Xia, Wan, Xie, & Zhang, 2020; Xie et al., 2021; Xie, Zhang et al., 2020; Xie, Zhang, Qu, Dai, & Tao, 2020; Xu, Zhang, Xia, Gao, & Gao, 2020). Therefore, multi-view learning has attracted more and more attention. Drawing inspiration from multi-view learning, several studies bring graph neural networks (GCNs) to multi-view learning (Khan & Blumenstock, 2019; Li, Li, & Wang, 2020). Despite the promising preliminary results, they mainly study semi-supervised learning tasks. To solve this problem, inspired by deep embedding clustering model (Xie, Girshick, & Farhadi, 2016) (DEC), Cheng, Wang, Tao, Xie, and Gao (2020) employed the graph encoder to learn low-dimensional node representation for each view, then calculated the view-specific probability distribution $\mathbf{Q}^{(v)}$ and view-consensus probability distribution $\mathbf{P}^{(v)}$, where v is the vth view. By minimizing the discrepancy between $\mathbf{Q}^{(v)}$ and $\mathbf{P}^{(v)}$, they proposed a new multi-view attribute graph convolution networks for clustering (MAGCN for short in this paper). Similarly, Fan, Wang, Shi, Lu, Lin, and Wang (2020) proposed a one to multiple graph auto-encoders (O2MAC) to solve multi-view attributed graph clustering problem. It firstly learns a shared node representation by leveraging one informative graph view and node attribute to reconstruct multiple graph views, then, leverages the probability distribution constraint for clustering.

Although achieving comparable results, both MAGCN and O2MAC still suffer from two bottlenecks.

1. When extracting the view-consensus node representation, they neglect the block diagonal representation learning. Therefore, the node representation cannot characterize the cluster structure. For example, suppose there are three classes $\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$. Fig. 1 shows the block diagonal representation $\mathbf{Z}$ of nodes, where $\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3$ are node representations corresponding to $\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$. We find that the nonzero entries $\mathbf{Z}_k$ correspond to only $\mathbf{C}_k$, where $k \in \{1, 2, 3\}$ means the kth cluster. Above node representation $\mathbf{Z}$ well reveals the true membership of data and is very discriminative for clustering. Therefore, the block diagonal representation plays an important role in node clustering analysis (Lu, Feng, Lin, Mei, & Yan, 2019).
2. They failed to make full use of the information embedded in the learned clustering labels, thus leading to inferior results. In fact, although the clustering labels are inaccurate during network training, some data with accurate labels will propagate useful information, which is of benefit for achieving better clustering (Lv, Kang, Lu, & Xu, 2021).

To solve the above problems, we propose a **M**ulti-**V**iew **G**raph embedding **C**lustering network (MVGC), which is characterized by joint self-supervision and block diagonal representation (See Fig. 2). More specifically, to well exploit the cluster structure, we learn view-consensus coefficient matrix with block diagonal representation constraint (measured by the $\ell_{1,2}$-norm), such that the coefficient representation possibly leads to correct clustering. Moreover, we make good use of the clustering labels to supervise the learnings of node representation and coefficient matrix, and the latter is employed in turn to conduct the subsequent clustering. In this way, the representation learning and clustering are seamlessly connected, despite the absence of supervisory signals, such an incorporation enables the overall framework to be trained towards achieving better clustering results. In particular, we make the following contributions:

1. We discover that the $\ell_{1,2}$-norm plays a crucial role in characterizing block diagonal property, and then apply it to learn coefficient representation which well exploits the cluster structure. To the best of our knowledge, this is the first attempt to take block diagonal property into account in multi-view GCNs based clustering. Thanks to its simpleness, our method can be easily plugged into deep neural networks, thus facilitating real applications.
2. We make full use of the clustering labels to guide the optimization of the overall framework covering node representation and coefficient matrix, where the latter is employed in turn to conduct the subsequent clustering. In this way, in spite of the absence of supervisory signals, the overall framework is trained with the aim to achieve better clustering performance. Hence, we assume that this paper could provide insight toward the GCN based multi-view unsupervised learning.
3. Extensive experimental results show the promising clustering performance of the proposed method comparing with 11 state-of-the-art approaches on four challenging benchmark datasets.

2. Notations and preliminary

For convenience, we first introduce the notations and definitions that will be used in the paper. We use bold upper case letters for matrices, *e.g.,* $\mathbf{M}$, bold lower case letters for vectors, *e.g.,* $\mathbf{m}$. The Frobenius norm of $\mathbf{M} \in \mathbb{R}^{n\times d}$ is $\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{d} M_{ij}^2}$, where M_{ij} is the entries of $\mathbf{M}$. $\|\mathbf{M}\|_{1,2}^2 = \sum_{i=1}^{n}\|\mathbf{m}_i\|_1^2 = \sum_{i=1}^{n}\left(\sum_{j=1}^{d}|M_{ij}|\right)^2$ is the $\ell_{1,2}$-norm of a matrix $\mathbf{M}$.

Definition 1 (*Multi-attribute Multi-view Graph and Nodes Clustering*)**.** Multi-attribute multi-view graph is represented as $\mathbf{G} = \{\mathbf{O}, \mathbf{E}^{(1)}, \ldots, \mathbf{E}^{(V)}, \mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(V)}\}$, where $\mathbf{O} = \{o_i\}_{i=1}^{n}$ is a set of nodes in a graph and $E_{ij}^{(v)} \in \mathbf{E}$ is the edge between the node i and j in the graph structure of the vth view ($v = 1, \ldots, V$). The topological structure of $\mathbf{G}$ can be represented by adjacency matrix $\{\mathbf{A}^{(v)}\}$, and $A_{ij}^{(v)} = 1$ if $E_{ij}^{(v)} \in \mathbf{E}$, otherwise $A_{ij}^{(v)} = 0$. $\mathbf{x}_i^{(v)} \in \mathbf{X}^{(v)}$ indicates the attribute of the vth view associated with node o_i.

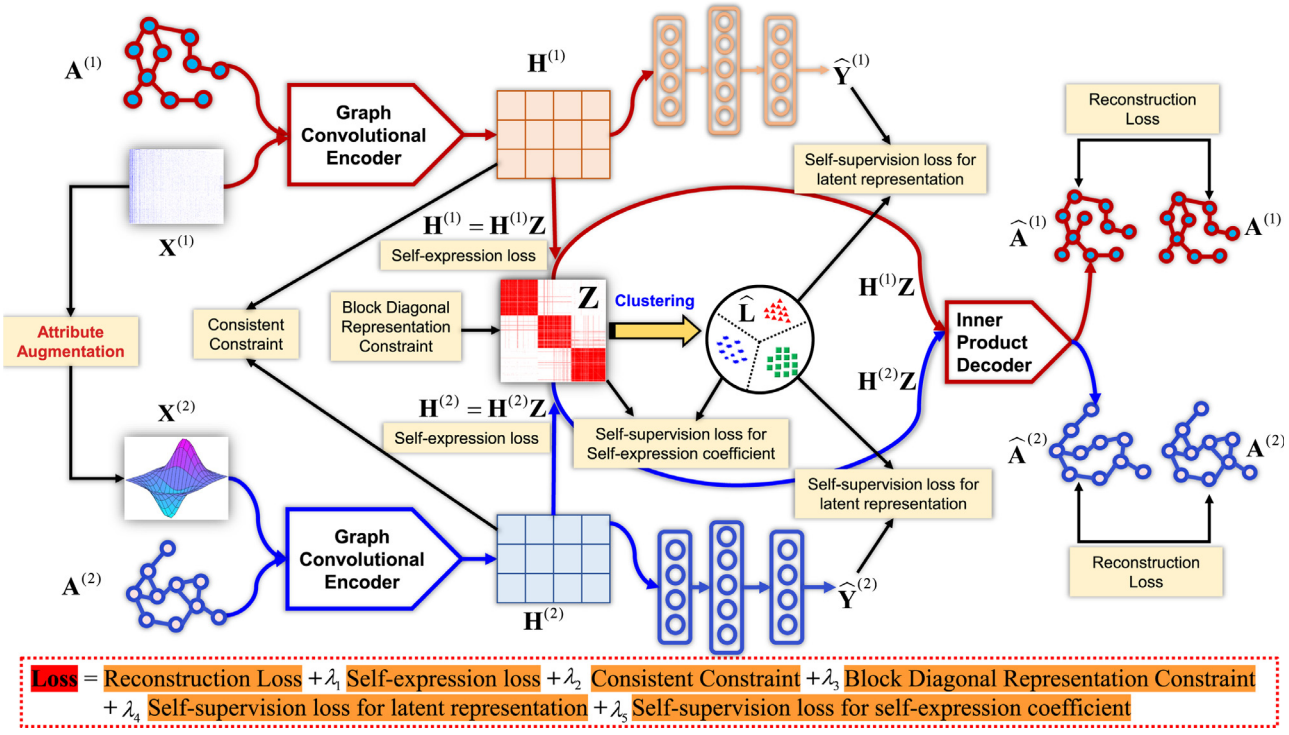


Fig. 2. The flowchart of the proposed **M**ulti-**V**iew **G**raph embedding **C**lustering network (MVGC).

Given the multi-attribute multi-view graph $\mathbf{G}$, nodes clustering aims to partition nodes in $\mathbf{G}$ into K disjoint clusters $\mathbf{C}_1, \ldots, \mathbf{C}_K$, such that the nodes in the same cluster have high correlation to each other.

3. Methodology

In this section, taking the two-view node clustering problem for example, we first introduce how to effectively augment attributes for graph-structured data, then we present the proposed MVGC for nodes clustering in an end-to-end manner.

3.1. Attribute augmentation

O2MAC is a representative of multi-view graph-structured data clustering method. Despite impressive clustering results, it is easy to see that O2MAC suffers from inadequate multi-view node representation learning due to simply encoding single-view node attribute and its corresponding graph structure. To boost the performance of multi-view learning, we propose to construct a new node attribute as a view for node clustering. With the characteristics of multi-view data, a simple and effective method is to construct multiple attributes directly by using the existing feature extraction methods, *e.g.*, Gabor, SIFT, LBP, GIST and HOG. However, these methods are suitable for processing data with a Euclidean structure, *e.g.,* face and objects. We herein study graph-structured data from non-Euclidean domains. With regard to this, inspired by the fact that kernel trick can capture the nonlinear features (Liao et al., 2018; Liwicki, Tzimiropoulos, Zafeiriou, & Pantic, 2013), we discover that Euler transform can efficiently characterize graph-structured data. Hence, we augment the raw node attribute via the Euler transform. To be specific, given a raw attribute $\mathbf{x}_i \in \mathbb{R}^{1\times d^0}$, its Euler representation can be represented by

$$\mathbf{b}_i = \frac{1}{\sqrt{2}}\begin{bmatrix} e^{j\alpha\pi X_{i1}} \\ \vdots \\ e^{j\alpha\pi X_{id^0}} \end{bmatrix} = \frac{1}{\sqrt{2}} e^{j\alpha\pi\mathbf{x}_i}, \tag{1}$$

where $\alpha \in \mathbb{R}^+$ is a parameter that is adjusted to suppress the values caused by outliers. In this article, similar to Liao et al. (2018), we take $\alpha = 1.1$ for all datasets without further tuning. $\mathbf{b}_i \in \mathbf{B} \in \mathbb{R}^{n\times d^0}$, n and d^0 are the number of nodes and attribute dimension. By doing so, we can obtain a new node attribute matrix $\mathbf{X}^{(2)} = \mathbf{B}$, raw node attribute is $\mathbf{X}^{(1)} \in \mathbb{R}^{n\times d^0}$.

3.2. Multi-view node subspace clustering module

Multi-view node subspace clustering aims to learn a coefficient representation which is shared by multiple views, then assigning each node into one of K clusters in this new subspace. With regard to this, MVGC progressively encodes both graph structure $\mathbf{A}^{(v)}$ and node attribute $\mathbf{X}^{(v)}$ of vth view into a representation $\mathbf{H}^{(v)}$ via a two-layer graph convolution encoder $\mathcal{E}^{(v)}[\cdot]$. Thus, the representation $\mathbf{H}^{(v)}$ can be expressed as

$$\begin{aligned}\mathbf{H}^{(v)} &= \mathcal{E}^{(v)}_{\text{Linear}}[\mathbf{X}^{(v)}, \mathbf{A}^{(v)}|\mathbf{W}^{(v)}_{(2)}] \\ &= f_{\text{Linear}}(\widetilde{\mathbf{D}}^{(v)^{-\frac{1}{2}}}\widetilde{\mathbf{A}}^{(v)}\widetilde{\mathbf{D}}^{(v)^{-\frac{1}{2}}}\mathbf{H}^{(v)}_{(1)}\mathbf{W}^{(v)}_{(2)})\end{aligned} \tag{2}$$

where $\widetilde{\mathbf{A}}^{(v)} = \mathbf{A}^{(v)} + \mathbf{I}$, $\widetilde{\mathbf{D}}^{(v)}_{ii} = \sum_j \widetilde{\mathbf{A}}^{(v)}_{ij}$, $\mathbf{I}$ is an identity matrix, $f_{\text{Linear}}(\cdot)$ represents the linear activation function. $\mathbf{H}^{(v)}_{(1)}$ is the output of first layer convolution operation and $\mathbf{H}^{(v)}_{(0)} = \mathbf{X}^{(v)}$. $\mathbf{W}^{(v)}_{(2)}$ is a matrix of filter parameters we need to learn in the second layer of vth view encoder.

Self-Expression Module. To make the latent representation $\mathbf{H}^{(v)}$ be more suitable for subspace clustering than raw node attribute and graph structure, we herein employ the self-expressive learning to learn a shared self-expressive coefficient representation $\mathbf{Z}$. To this end, MVGC minimizes the following self-expression loss:

$$\begin{aligned}\mathcal{L}_1 = \min_{\mathbf{Z},\mathbf{H}^{(v)},\boldsymbol{\theta}} &\sum_{v=1}^{V} \|\mathbf{H}^{(v)}\mathbf{Z} - \mathbf{H}^{(v)}\|_F^2, \\ \text{s.t.,}\quad & diag(\mathbf{Z}) = \mathbf{0},\end{aligned} \tag{3}$$

where $\mathbf{Z} \in \mathbb{R}^{n\times n}$ is the coefficient representation, $\boldsymbol{\theta}$ represents the parameter of multi-view graph auto-encoder. To avoid the trivial solution $\mathbf{Z} = \mathbf{I}$, we set the constraint term $diag(\mathbf{Z}) = \mathbf{0}$.

To make sure that the learned node representation $\mathbf{H}^{(v)}$ preserves sufficient graph structure information, the new representation $\mathbf{H}^{(v)}\mathbf{Z}$ of vth view is subsequently fed into the inner product decoder to predict whether there is a link between two nodes. We give the reconstruction loss by

$$\begin{aligned}\mathcal{L}_0 &= \min_{\mathbf{H}^{(v)},\boldsymbol{\theta}} \sum_{v=1}^{V} \mathbb{E}[-\log(\widehat{\mathbf{A}}^{(v)})] \\ &= \min_{\mathbf{H}^{(v)},\boldsymbol{\theta}} -\sum_{v=1}^{V}\sum_{i,j=1}^{N}\left[A_{i,j}^{(v)}\log(\widehat{A}_{i,j}^{(v)}) + (1-A_{i,j}^{(v)})\log(1-\widehat{A}_{i,j}^{(v)})\right],\end{aligned} \tag{4}$$

where $\widehat{\mathbf{A}}^{(v)} = \text{sigmoid}(\mathbf{S}^{(v)}\mathbf{S}^{(v)^{\mathrm{T}}})$, $\mathbf{S}^{(v)} = \mathbf{H}^{(v)}\mathbf{Z}$ is the self-expressed representation. $\widehat{A}_{i,j}^{(v)}$ predicts the links between nodes i and j in the vth view, and $\widehat{A}_{i,j}^{(v)} = 1$ if the ith node links the jth node. By minimizing above graph structure reconstruction loss, we can minimize the discrepancy between the input graph $\mathbf{A}^{(v)}$ and the reconstructed graph $\widehat{\mathbf{A}}^{(v)}$.

Consistent Representation Constraint. To make sure that the MVGC can learn a consistent subspace $\mathbf{Z}$ among different views, we herein employ a consistent representation constraint to capture the geometric relationship similarity embedded in different views. Thus, we have

$$\mathcal{L}_2 = \min_{\boldsymbol{\theta},\mathbf{H}^{(v)}} \sum_{v\neq i}^{V} \|\mathbf{H}^{(v)} - \mathbf{H}^{(i)}\|_F^2. \tag{5}$$

When we obtain the self-expressive coefficient matrix $\mathbf{Z}$, the induced affinity matrix $\boldsymbol{\Lambda}$ can be calculated by $\boldsymbol{\Lambda} = \frac{1}{2}\left(|\mathbf{Z}| + |\mathbf{Z}^{\mathrm{T}}|\right)$. Finally, we can obtain the node clustering labels by applying a spectral clustering method on $\boldsymbol{\Lambda}$. In this paper, we employ the property of normalized cut (NCut) (Shi & Malik, 2000) method to get the node clustering results.

3.3. Block diagonal representation constraint

For the learned self-expressive coefficient matrix $\mathbf{Z}$, we hope it can well characterize the cluster structure. Specifically, we hope the learned coefficient matrix $\mathbf{Z}$ complies with the block diagonal property (BDP) (Lu et al., 2019), *i.e.*, $\mathbf{Z}$ is K-block diagonal, where the nonzero entries $\{\mathbf{Z}_j\}_{j=1}^{K}$ correspond to only $\{\mathbf{X}_j\}_{j=1}^{K}$, $\mathbf{X}_j$ represents the nodes from jth cluster. Such property possibly leads to correct clustering. However, existing node clustering methods fail to take the BDP into consideration, resulting in inferior performance. To this end, we employ the $\ell_{1,2}$-norm to constrain the self-expressive coefficient representation. Thus, we have

$$\begin{aligned}\mathcal{L}_3 &= \min_{\mathbf{Z}} \|\mathbf{Z}\|_{1,2}^2 = \min_{\mathbf{Z}} \sum_{i=1}^{n} \|\mathbf{z}_i\|_1^2 \\ &= \min_{\mathbf{Z}} \sum_{i=1}^{n}\left(\sum_{j=1}^{n}|Z_{ij}|\right)^2.\end{aligned} \tag{6}$$

By minimizing Eq. (6), different elements in squared ℓ_1-norm of ith row $\mathbf{z}_i$ are competing with each other to survive, and at least one element in row $\mathbf{z}_i$ survives (remaining non-zero). By doing so, some discriminative features are survived for each cluster to provide certain flexibility in the learned coefficient representation, *i.e.*, making $\mathbf{Z}$ well preserve the block diagonal property.

3.4. Dual self-supervised mechanism

To make full use of information in the learned clustering labels $\widehat{\mathbf{L}}$, we use $\widehat{\mathbf{L}}$ to provide feedback to both self-expression module and latent representation learning module. The above idea can be fulfilled as follows.

Self-supervision for Latent Representation. We utilize the learned clustering labels $\widehat{\mathbf{L}}$ to supervise the latent representation $\mathbf{F}^{(v)}$. To this end, we feed $\mathbf{H}^{(v)}$ into a self-classification module to carry out a classification task. Let $\widehat{\mathbf{Y}}^{(v)}$ be the self-classification module of vth view, where $\widehat{\mathbf{Y}}^{(v)} \in \mathbb{R}^{n\times K}$. Here, $\widehat{\mathbf{L}}$ is treated as the target output of self-classification module. To train the graph encoder of vth view via self-supervision information, we introduce a mixture of cross-entropy (CE) loss and center loss, which is defined as

$$\mathcal{L}_4 = \min_{\boldsymbol{\theta},\boldsymbol{\phi},\mathbf{H}^{(v)}} \frac{1}{n}\sum_{v=1}^{V}(\mathbb{CE}(\widehat{\mathbf{L}}, \widehat{\mathbf{Y}}^{(v)}) + \gamma\|\widehat{\mathbf{Y}}^{(v)} - \delta_\pi(\widehat{\mathbf{L}})\|_F^2), \tag{7}$$

where $0 < \gamma \leq 1$ is a trade-off parameter, $\boldsymbol{\phi}$ represents the parameters of self-classification module. $\delta_\pi(\widehat{\mathbf{L}})$ is the corresponding cluster center of $\widehat{\mathbf{Y}}^{(v)}$. The second term in Eq. (7) is the center loss which helps compress the intra-cluster variations. In this article, the self-classification module consists of a two-layer fully-connected network.

Self-supervision for Self-expression. We utilize the clustering labels $\widehat{\mathbf{L}}$ produced by the last iteration to supervise the self-expressive coefficient matrix $\mathbf{Z}$. Specifically, for self-expression coefficient representation $\mathbf{Z}$, an entry Z_{ij} is nonzero only if the ith and jth nodes have the same cluster labels. Hence, the clustering results produced by the last iteration can provide the rich information for fine-tuning the coefficient matrix $\mathbf{Z}$, which is significant for node subspace clustering. To this end, inspired by Zhang et al. (2019), we minimize the discrepancy between $\mathbf{Z}$ and the pseudo label matrix $\widehat{\mathbf{L}}$. Thus, we have

$$\mathcal{L}_5 = \min_{\mathbf{Z}} \sum_{i,j=0}^{n} |Z_{ij}| \frac{\|\widehat{\mathbf{l}}_i - \widehat{\mathbf{l}}_j\|_2^2}{2}, \tag{8}$$

where $\widehat{\mathbf{l}}_i, \widehat{\mathbf{l}}_j \in \widehat{\mathbf{L}}$ represent the label vector corresponding to the ith and jth nodes, respectively.

3.5. Implementation details

Consequently, we integrate the above concerns into an end-to-end framework, then the objective of MVGC is induced as

$$\mathcal{L} = \min_{\boldsymbol{\theta},\boldsymbol{\phi},\mathbf{H}^{(v)},\mathbf{Z}} \mathcal{L}_0 + \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 + \lambda_3\mathcal{L}_3 + \lambda_4\mathcal{L}_4 + \lambda_5\mathcal{L}_5, \tag{9}$$

where $\lambda_i, i = 1, \ldots, 5$ are trade-off parameters.

We optimize $\mathcal{L}$ by the Adam optimizer (Kingma & Ba, 2015). The dimension of graph encoder is $d^0 \to d^1 \to d^2$, where d^0 is the dimension of the raw node attribute. The dimensions of self-classification module is $d^2 \to 1024 \to K$. The learning rate of MVGC is 3.0×10^{-5}. Due to the clustering results provided by spectral clustering are up to an unknown permutation, the clustering labels from two successive iterations might not be consistent. We adopt the Hungarian (Munkres., 1957) method to find an optimal alignment between the clustering labels of previous iterations. To improve stability, we update the other parameters in MVGC for 5 epochs, and then update $\widehat{\mathbf{L}}$ by performing spectral clustering on $\boldsymbol{\Lambda}$.

Note that, to avoid trivial solutions, we set the diagonal constraint on $\mathbf{Z}$, *i.e.*, $\text{diag}(\mathbf{Z}) = \mathbf{0}$. To this end, motivated by Elhamifar and Vidal (2013), Ji, Zhang, Li, Salzmann, and Reid (2017), taking the Python with TensorFlow 1.× platform as an example, we can do a simple trick as follows:

Algorithm 1: MVGC

Input: Node attribute: $\{\mathbf{X}^{(v)}\}_{v=1}^{V} \in \mathbb{R}^{n\times d_0}$, graph structure: $\{\mathbf{A}^{(v)}\}_{v=1}^{V} \in \mathbb{R}^{n\times n}$, cluster number K, parameters λ_1, λ_2, λ_3, λ_4, λ_5.
Output: Clustering label $\widehat{\mathbf{L}}$.

Initialize graph attention auto-encoder, self-expressive coefficient matrix $\mathbf{Z}$ and self-supervised module;
// Extract multi-view representation
Obtain the node representation $\mathbf{H}^{(v)}$ by Eq. (2);
// Calculate clustering label
Run spectral clustering on $\boldsymbol{\Lambda} = \frac{1}{2}\left(|\mathbf{Z}| + \left|\mathbf{Z}^{\mathrm{T}}\right|\right)$ to get clustering label $\widehat{\mathbf{L}}$;
// Extract the output of self-supervision for node representation module
Get $\widehat{\mathbf{Y}}^{(v)}$ by encoding node representation $\mathbf{H}^{(v)}$;
// Optimization
while *not converge* **do**
  Fix clustering label $\widehat{\mathbf{L}}$, and update other parameters of MVGC by Eq. (9);
  if *Iter % 5 ==0* **then**
    // Calculate coefficient matrix
    Calculate the coefficient matrix $\mathbf{C}$;
    // Calculate clustering label
    Run spectral clustering on $\boldsymbol{\Lambda} = \frac{1}{2}\left(|\mathbf{Z}| + \left|\mathbf{Z}^{\mathrm{T}}\right|\right)$ to update $\widehat{\mathbf{L}}$;
  end
end
return: Clustering label $\widehat{\mathbf{L}}$.

```
import tensorflow as tf
# Assuming the node representation after the graph encoder
# of v-th view is H_v and before passing to the decoder,
# Z is the coefficient matrix
# ZH_v is the self-expressed representation
ZH_v = tf.matmul((Z-tf.diag(tf.diag_part(Z))), H_v)
```

In this way, the diagonal of $\mathbf{Z}$ is set to zero. Finally, we summarize the optimization procedure of the proposed MVGC in Algorithm 1.

3.6. Discussions

⋆ The differences between MVGC and MAGCN

Although MAGCN (Cheng et al., 2020) also leveraged the Euler transform to construct multi-view descriptor, the proposed MVGC is significantly different from Cheng et al. (2020) in the following main aspects:

1. **The motivation is different.** Cheng et al. (2020) required the discrepancy between view-specific node distribution and view-consensus node distribution that could be as small as possible, and thus, a consistent embedding space can be effectively found for clustering. In contrast, our model aims to solve the GCN based multi-view subspace clustering, *i.e.*, we target at learning a good self-expressed coefficient matrix shared by different views for node clustering.
2. **The objective is different.** Cheng et al. (2020) explicitly minimized the mismatch between view-specific probability distribution and view-consensus probability distribution, In contrast, our model does not need this part. Moreover, Cheng et al. (2020) also explicitly minimized the reconstruction error between input and output node attribute, which adopted extra node attribute decoder, whereas our model is a forward neural network that only require parameter-free inner product decoder to reconstruct the input graph itself. In consequence, our method does not need to seek a good tradeoff between the attribute reconstruction and graph reconstruction errors and then enjoy a smaller parameter size.
3. In the proposed MVGC, the BDP constraint is imposed on coefficient matrix to well exploit the cluster structure. We also utilize the clustering labels to guide the learnings of node representation and coefficient matrix via a self-supervised manner. In contrast, Cheng et al. (2020) failed to consider them.

⋆ Block Diagonal Representation Regularizer

In this paper, we leverage the good property of $\ell_{1,2}$-norm to constrain the self-expressed coefficient matrix $\mathbf{Z}$, which is defined as

$$\min_{\mathbf{Z}} \|\mathbf{Z}\|_{1,2}^2 = \min_{\mathbf{Z}} \sum_{i=1}^{n} \|\mathbf{z}_i\|_1^2 = \min_{\mathbf{Z}} \sum_{i=1}^{n} \left(\sum_{j=1}^{n} |Z_{ij}| \right)^2, \tag{10}$$

by minimizing Eq. (10), different elements in squared ℓ_1-norm of ith row $\mathbf{z}_i$ are competing with each other to survive, and at least one element in row $\mathbf{z}_i$ remains non-zero (Ming & Ding, 2019). By doing so, some discriminative features are survived for each cluster to provide certain flexibility in the learned coefficient matrix, *i.e.*, $\mathbf{Z}$ obeys the block diagonal property, where the nonzero entries $\mathbf{Z}_k$ correspond to only $\mathbf{C}_k$, $k \in \{1, \ldots, K\}$, $\mathbf{C}_k$ means the nodes in kth cluster, $\mathbf{Z}_k$ is the corresponding self-expressed coefficient, K is the number of clusters.

Lu et al. proposed an alternative block diagonal regularizer, namely *k-block diagonal regularizer*, which is defined as the sum of the k smallest eigenvalues of $\mathbf{L}_{\mathbf{Z}}$ (Lu et al., 2019), *i.e.*,

$$\|\mathbf{Z}\|_{\boxed{k}} = \sum_{i=n-k+1}^{n} \rho_i\left(\mathbf{L}_{\mathbf{Z}}\right), \tag{11}$$

where $\mathbf{L}_{\mathbf{Z}}$ is the Laplacian matrix of $\mathbf{Z}$, $\rho_i\left(\mathbf{L}_{\mathbf{Z}}\right)(i \in \{1, \ldots, n\})$ is the eigenvalues of $\mathbf{L}_{\mathbf{Z}}$ in the decreasing order.

By definition, we implement block diagonal constraints differently. Our introduced manner, $\ell_{1,2}$-norm regularizer, approximates the block diagonal representation matrix by using the structure priors, *i.e.*, sparsity and smoothness, which is indirect. In contrast, Lu et al. leveraged an enforced block diagonal conditions, which is more direct. From the perspective of deep neural network optimization, our introduced block diagonal representation regularizer is easy to be implemented. In contrast, direct optimization of Eq. (11) is arduous in deep neural networks. We will continue to study this problem.

4. Experiments

We conducted extensive experiments to validate the effectiveness of the proposed MVGC and the proposed attribute augmentation and block diagonal representation strategies.

4.1. Experimental setup

We implement the proposed MVGC in TensorFlow 1.13.1 platform based on Python 3.6. All the experiments are conducted on a machine with an Intel (R) Xeon (R) Gold 6230 CPU and dual NVIDIA Tesla P100-PCIE GPUs.

Datasets. The proposed MVGC along with the compared methods are tested on four real-world datasets, *i.e.*, ACM (Tang et al.,

Table 1
The numerical introduction to real datasets, where # means "the number of".

Dataset	Size	Attribute Dimension	# Edges in each view graph	# Class	Attribute content	Clustering target
ACM (Tang, Zhang, Yao, Li, Zhang, & Su, 2008)	3,025	1,870	subject (2,210,761), paper (29,281)	3	Paper's keywords	Paper's research area
DBLP (Pan, Wu, Zhu, Zhang, & Wang, 2016)	4,057	334	conf (5,000,495), term (6,776,335), author (11,113)	4	Author's keywords	Author's research area
IMDB (Fan et al., 2020)	4,780	1,232	actor (98,010), director (21,018)	3	Movie plot's keywords	Movie's genre
HHAR (Stisen et al., 2015)	10,299	561	top-3 and top-5 nearest neighbor graph views	6	Sensor records	Human's activity

Table 2
Aggregated results of different methods on ACM, DBLP and IMDB datasets. "OM" is out-of-memory error. "N/A" means not applicable.

Datasets	ACM				DBLP				IMDB			
Metric	ACC	F_1	NMI	ARI	ACC	F_1	NMI	ARI	ACC	F_1	NMI	ARI
LINE (Tang, Qu, Wang, Zhang, Yan, & Mei, 2015)	64.79	65.95	39.41	34.33	86.89	85.46	66.76	69.88	42.68	28.70	0.31	N/A
GAE (Kipf & Welling, 2016)	84.52	84.65	55.38	59.46	61.21	61.41	30.80	22.02	42.98	40.62	4.02	4.73
ARGAE (Pan et al., 2018)	84.33	84.51	54.54	60.64	58.16	59.38	29.51	23.92	41.19	36.85	0.63	N/A
DAEGC (Wang et al., 2019)	86.94	87.07	56.18	59.35	62.05	61.75	32.49	21.03	N/A	N/A	N/A	N/A
SDCN (Bo et al., 2020)	90.45	90.42	68.31	73.91	68.05	67.71	39.50	39.15	46.97	31.83	2.85	2.84
MNE (Zhang, Qiu, Yi, & Song, 2018)	63.70	64.79	29.99	24.86	OM	OM	OM	OM	39.58	33.16	0.17	0.08
RMSC (Xia, Pan, Du, & Yin, 2014)	63.15	57.46	39.73	33.12	89.94	82.48	71.11	76.47	27.02	37.75	0.54	0.18
PWMC (Nie, Li, & Li, 2017)	41.62	37.83	3.32	3.95	32.53	28.08	1.90	1.59	24.53	31.64	0.23	0.17
SWMC (Nie et al., 2017)	38.31	47.09	8.38	1.87	65.38	56.02	37.60	38.00	26.71	37.14	0.56	0.04
O2MAC-Gabor (Fan et al., 2020)	72.53	72.97	47.88	44.75	72.93	73.02	47.21	44.42	40.55	33.89	2.56	4.82
O2MAC-FFT (Fan et al., 2020)	77.22	77.66	49.07	48.31	85.65	85.09	66.18	67.39	42.65	36.99	4.27	5.61
O2MAC-Cartesian (Fan et al., 2020)	64.40	64.50	20.53	22.97	85.21	84.67	62.85	65.99	41.72	36.63	3.31	5.43
O2MAC-Eular (Fan et al., 2020)	88.50	88.58	64.48	69.36	88.88	88.31	69.60	73.87	43.56	38.63	4.44	6.01
O2MAC (Fan et al., 2020)	90.42	90.53	69.23	73.94	90.74	90.13	72.87	77.80	45.02	**41.59**	5.24	7.53
MAGCN (Cheng et al., 2020)	93.32	93.33	75.70	80.79	88.91	88.17	70.70	74.54	51.34	35.18	4.18	6.62
MVGC-Gabor	85.32	85.05	63.27	63.08	81.53	81.72	62.49	57.95	45.67	36.98	1.63	3.68
MVGC-FFT	88.10	88.04	66.55	67.78	86.96	86.31	68.51	70.53	49.58	34.31	1.54	4.45
MVGC-Cartesian	94.08	94.07	77.68	82.95	88.07	87.23	69.11	72.94	50.19	39.59	3.13	6.91
MVGC-Eular	**96.17**	**96.08**	**84.38**	**89.17**	**92.33**	**92.25**	**74.24**	**80.41**	**53.31**	39.53	**6.23**	**8.48**

2008),[1] DBLP (Pan et al., 2016),[2] IMDB (Fan et al., 2020)[3] and heterogeneity human activity recognition (HHAR) dataset (Stisen et al., 2015) as described in Table 1. For all datasets, the 2-nd view's attribute is constructed from the corresponding raw attribute by Euler transformation, α is set to 1.1 for all dataset. Similar to Fan et al. (2020), we select the most informative graph view $\mathbf{A}^{(*)}$ as graph structure, *i.e.*, $\mathbf{A}^{(v)} = \mathbf{A}^{(*)}$.

Comparisons. We compare the proposed MVGC with 11 representative methods, *i.e.,* large-scale information network embedding (LINE) (Tang et al., 2015), GAE (Kipf & Welling, 2016), ARGAE (Pan et al., 2018), scalable multi-view network embedding (MNE) (Zhang et al., 2018), robust multi-view spectral clustering method (RMSC) (Xia et al., 2014), DAEGC (Wang et al., 2019), SDCN (Bo et al., 2020), PWMC (Nie et al., 2017), SWMC (Nie et al., 2017), O2MAC (Fan et al., 2020) and MAGCN (Cheng et al., 2020). For the methods with multi-view setting, we use the multi-view graph adjacency matrices as the input. For the methods with single-view setting, we use the most informative graph view $\mathbf{A}^{(*)}$ and raw node attribute as the input.

Evaluation Metrics. We employ four widely used measures to evaluate the clustering performances, which are accuracy (ACC), normalized mutual information (NMI), average rand index (ARI) and macro F1-score (F_1) respectively. For all metrics, a higher score indicates a better clustering quality.

Parameter Setting. To obtain a balance amongst different term in Eq. (9), the trade-off parameters are basically set roughly to be inversely proportional to the value of each objective. Thus, in the experiments, we turn λ_i in the range of $[10^{-3}, 10^{-2}, 0.1, 1, 10, 10^2, 10^2, 10^3]$ to get optimal results. Meanwhile, we tune d^1 and d^2 in the range of $[128, 256, 512, 1024, 2048, 4096]$. Specifically, d^1 and d^2 are set to 4,096 and 2,048 for all datasets, respectively. γ is set 0.8 for all datasets. λ_1, λ_2, λ_3, λ_4 and λ_5 are set to 10, 10^{-3}, 10, 10 and 10 for all datasets, respectively. For all baselines, we retain to the settings in the corresponding papers.

4.2. Comparisons with state-of-the-art methods

Table 2 presents the metrics comparison of the above methods on ACM, DBLP and IMDB datasets. For each experiment, we independently repeat the aforementioned methods 10 times and show the averages. From Table 2, we have the following interesting observations:

1. The proposed MVGC significantly and consistently outperforms classic multi-view methods, *e.g.*, RMSC, PWMC and SWMC. The reason should be that MVGC leverages the GCN to learn deep nonlinear node representation. In contrast, classic multi-view methods are linear methods.
2. Single-view GCN based methods (GAE, SDCN, ARGAE and DAEGC) are overall inferior to the proposed MVGC. These results verify the effectiveness of multiple views rather than a single view. The reason may be that multi-view methods may leverage the complementary information embedded in multi-attribute or multi-view graph, while single-view methods do not.
3. The proposed MVGC achieves comparable performance than O2MAC and MAGCN. Taking the clustering results on ACM dataset for example, it improves the O2MAC by 5.75%, 5.55%, 15.15% and 15.23% in terms of ACC, F_1, NMI and ARI,

[1] http://dl.acm.org
[2] https://dblp.uni-trier.de/
[3] https://www.imdb.com/

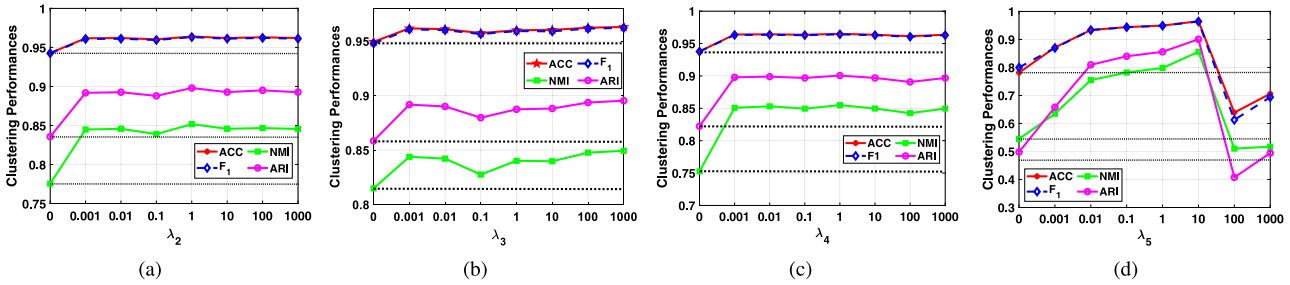


Fig. 3. The clustering results of MVGC w.r.t. λ_2, λ_3, λ_4 and λ_5 on ACM dataset.

respectively. The reason may be that MVGC explicitly takes into account the block diagonal property, and MVGC makes full use of the clustering labels to simultaneously supervise the graph encoding and coefficient matrix learning, while O2MAC and MAGCN failed to consider them.

4. We also find that the clustering performance improvement of MVGC on DBLP and IMDB datasets was not as much as on the ACM dataset. We find that the node attributes of DBLP and IMDB are too sparse. Under the circumstances, when employing the GCN to extract the node representation, node attributes can provide very limited information, the influence of graph structure on node representation may be dominant. In the future, we would like to combine MVGC with graph structure updating into a unified framework to further improve the clustering performances.
5. We also notice that the evaluation metrics of MVGC and all baselines are not very high when dealing with IMDB dataset. This phenomenon is determined by the property of the dataset. To be specific, IMDB dataset is a movie network, the node attribute consists of a bag-of-words represented of movie plots. The corresponding graph structures are represented by co-actor relationship, *i.e.*, movies are acted by the same actor, and co-director relationship, *i.e.*, movies are directed by the same director. Our goal is to divide these movies into different clusters according to movie's genre. However, there are actually many genres of movies, an actor may appear in various genres of movies, and the same director will also direct movies of different genres. Based on co-actor and co-director relationships, it is difficult to determine the genres of given movies. By contrast, ACM and DBLP are paper network and author network, respectively. When clustering ACM and DBLP datasets, we need to determine the research area of given papers and authors, respectively. According to the prosperities of these two datasets, the clustering task is relatively simple.
6. To further investigate the ability of the proposed MVGC to deal with large-scale dataset, we conduct node clustering experiment on HHAR dataset. Specifically, we compare the clustering performance of the proposed MVGC with challenging multi-view GNNs based comparing methods, *i.e.*, O2MAC and MAGCN. As reported in Fig. 4, our proposed MVGC always achieves the best performance in terms of all four metrics. In conclusion, the proposed MVGC can also be applied for large-scale data analysis.

4.3. Further evaluation

Ablation Study. We empirically analyze the effectiveness of different components in Eq. (9). To this end, Table 3 reports the clustering performance on ACM dataset. We have the following observations:

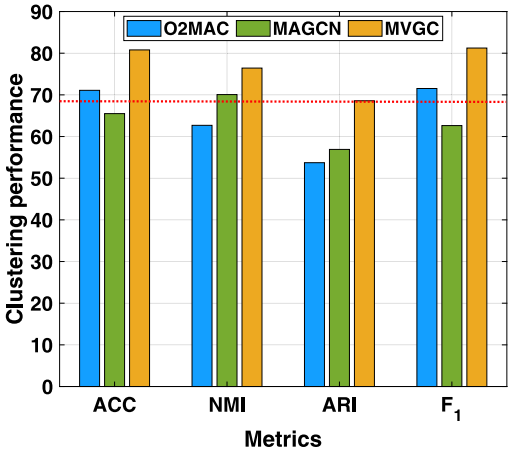


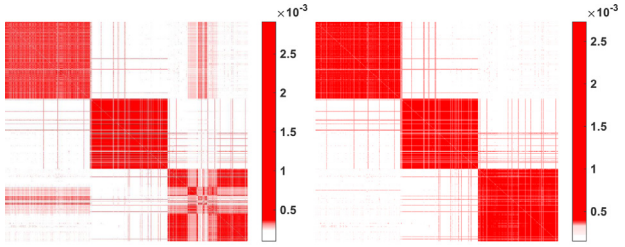
Fig. 4. The clustering results on HHAR dataset.

Table 3
Ablation study on MVGC on ACM dataset.

Strategies	ACC	F_1	NMI	ARI
$\mathcal{L}_0 + \mathcal{L}_2$	64.93	59.15	52.43	44.30
$\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2$	69.12	68.89	43.11	41.94
$\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3$	75.80	75.49	46.82	46.80
$\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_5$	88.93	88.78	67.98	70.46
$\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_4 + \mathcal{L}_5$	94.94	94.79	81.48	85.84
$\mathcal{L}$	**96.17**	**96.08**	**84.38**	**89.17**

1. Comparing the clustering results under $\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2$ and $\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3$, we find that when introducing the block diagonal representation constraint, the clustering performances are greatly improved. These results convey that both block diagonal representation are key technical choices for node subspace clustering.
2. Comparing the clustering results under $\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2$ and $\mathcal{L}_0 + \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_5$, it can be seen that the clustering results have been improved a lot. The clustering performances are further enhanced by the self-supervision loss $\mathcal{L}_4$ for latent representation. These results demonstrate that node representation extraction and coefficient matrix learning benefit from self-supervision since some data with correct clustering labels will propagate useful information.

Moreover, we vary the hyper-parameters λ_2, λ_3, λ_4 and λ_5 from 0 to 10^3 for the corresponding loss components and report the result in Fig. 3. It can be seen that every component in MVGC is helpful to improve clustering performance. Meanwhile, we can find that the proposed MVGC is in general not very sensitive to these hyper-parameters. Note that when $\lambda_5 > 10$, the clustering performances are degraded. This is because the value of $\mathcal{L}_5$ is very large in this case, making the overall objective function



(a) Without BDP constraint (b) With BDP constraint

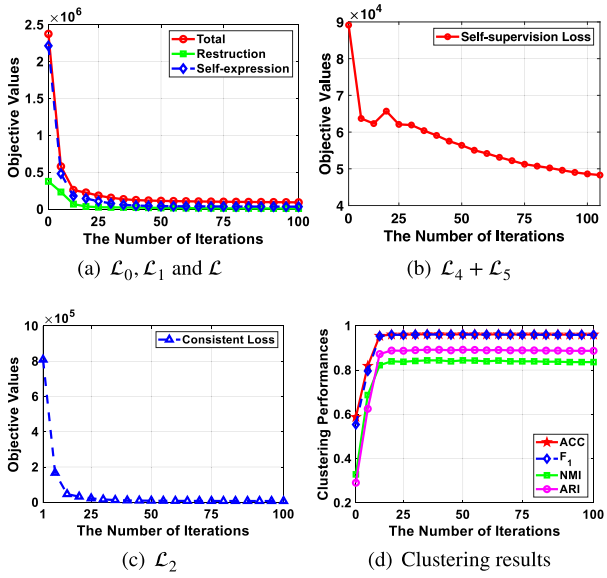
Fig. 5. The coefficient matrix visualizations on ACM dataset.

unbalanced, which in turn affects the clustering performance. Therefore, we recommend selecting λ_5 within $[0.01, 10]$. Furthermore, from the curves in Fig. 3(b), we observe that as λ_3 increases to large values, the performances ascend monotonously in general, indicating the effectiveness of incorporating $\ell_{1,2}$-norm regularization on the self-expressive coefficient matrix.

Effect of Block Diagonal Representation. To further evaluate the advantage of the proposed block diagonal representation constraint over other deep models, we provide a visualization for the coefficient matrix on ACM dataset. Two cases, *i.e.*, without $\ell_{1,2}$-norm (ℓ_1-norm constraint is adopted, which is similar to our previous work (Xia, Wang, Gao, Zhang, & Gao, 2021)) and with $\ell_{1,2}$-norm constraint, are selected for comparison. The visualization results are shown in Fig. 5. It is clear that the block diagonal learned under the block diagonal representation constraint is clearer.

Convergence Behaviors. To verify the convergence of the proposed MVGC, we record the objective values and clustering performances of MVGC with iterations. Due to space limitation, we plot partial results in Fig. 7. As observed, although the objective values do not monotonically decrease at each iteration, the overall convergence can be reached within approximately 50 steps of iterations. Moreover, we observe that clustering results gradually increase to a maximum and generally maintains it up to slight variation. These observations have clearly demonstrated that the clustering result at the last iteration guides the learning of coefficient matrix, and the latter is used in turn to carry out the subsequent clustering. In this way, the MVGC is trained by an end-to-end manner, with the target at achieving better clustering results.

Parameters Study. We conduct experiments to show the effect of graph encoder dimensions on the clustering results on ACM dataset. Fig. 6 presents four metrics of MVGC by varying d^1 and d^2 from 128 to 4,096. From these figures, we observe that four metrics obtain high value and generally maintain it up to slight variation with changing d^1 and d^2 from 512 to 4,096. This reveals that MVGC can obtain stable performance across a wide range of d^1 and d^2. While, we also notice that MVGC manifests unsatisfactory when d^1 and d^2 are in range $[128, 512]$. This is because that the self-expression module is not optimized well in such situation. To avoid this, we recommend choosing d^1 and d^2 from 512 to 4,096.



(a) $\mathcal{L}_0, \mathcal{L}_1$ and $\mathcal{L}$ (b) $\mathcal{L}_4 + \mathcal{L}_5$

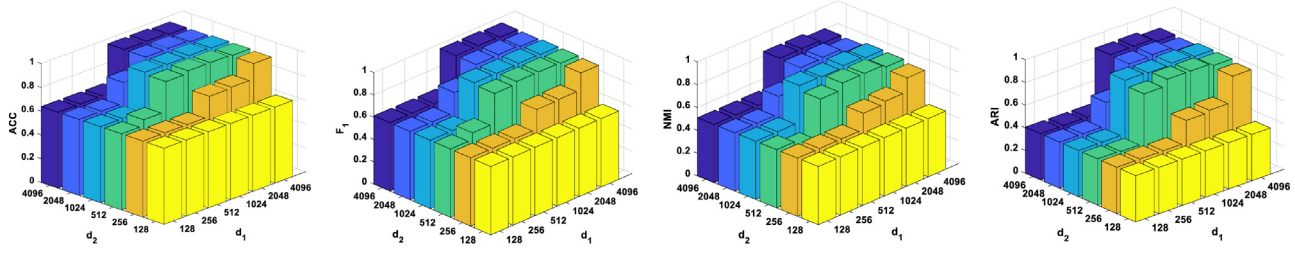
(c) $\mathcal{L}_2$ (d) Clustering results

Fig. 7. The objective values and clustering performances of MVGC with iterations on ACM dataset.

5. Conclusion and future works

We propose a Multi-View Graph embedding Clustering network (MVGC) to address multi-view graph-structured data clustering task. To augment the node attribute, MVGC uses Euler transform to effectively construct a new view descriptor for non-Euclidean structure data. By employing the block diagonal representation constraint, MVGC learns a good view-consensus coefficient matrix which has K-block diagonal. Moreover, with the aim to achieve better clustering results, MVGC seamlessly connects the clustering and representation learning via self-supervision. Extensive experiments results demonstrate the superiority of MVGC. In the future, we focus on four remained challenges:

1. Various effective and efficient node attribute augmentation methods.
2. Effective graph updating methods.
3. How to efficiently optimize an enforced block diagonal conditions (Lu et al., 2019) in deep neural networks?
4. Semi-supervised learning (Wan, Pan, Yang, & Gong, 2021) and contrastive learning (Chu, Wang, Shi, & Jiang, 2021; Zhao, Yang, Wang, Yang, & Deng, 2021) are also the hot spots in the field of GNNs. How to improve the proposed method based on this is also what we study in the future.



(a) ACC (b) F_1 (c) NMI (d) ARI

Fig. 6. Parameter sensitivity of d^1 and d^2 on ACM dataset.

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.

Acknowledgments

The authors would like to thank the anonymous reviewers and AE for their constructive comments and suggestions. This work is supported in part by National Natural Science Foundation of China under Grants 62176203, 62036007 and 62050175, in part by Equipment Pre-research Key Laboratory Funds under Grants 6142501200201, in part by Natural Science Basic Research Plan in Shaanxi Province under Grant 2020JZ-19, in part by the Fundamental Research Funds for the Central Universities, the Innovation Fund of Xidian University.

References

Bo, D., Wang, X., Shi, C., Zhu, M., Lu, E., & Cui, P. (2020). Structural deep clustering network. In *WWW* (pp. 1400–1410).

Cheng, J., Wang, Q., Tao, Z., Xie, D., & Gao, Q. (2020). Multi-view attribute graph convolution networks for clustering. In *IJCAI* (pp. 2973–2979).

Chu, G., Wang, X., Shi, C., & Jiang, X. (2021). CuCo: Graph representation with curriculum contrastive learning. In *Proc. IJCAI* (pp. 2300–2306).

Elhamifar, E., & Vidal, R. (2013). Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2765–2781.

Fan, S., Wang, X., Shi, C., Lu, E., Lin, K., & Wang, B. (2020). One2multi graph autoencoder for multi-view graph clustering. In *WWW* (pp. 3070–3076).

Gao, Q., Xia, W., Wan, Z., Xie, D., & Zhang, P. (2020). Tensor-SVD based graph learning for multi-view subspace clustering. In *Proc. AAAI* (pp. 3930–3937).

Ji, P., Zhang, T., Li, H., Salzmann, M., & Reid, I. D. (2017). Deep subspace clustering networks. In *Proc. NeurIPS* (pp. 24–33).

Khan, M. R., & Blumenstock, J. E. (2019). Multi-GCN: Graph convolutional networks for multi-view networks, with applications to global poverty. In *AAAI* (pp. 606–613).

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proc. ICLR Track Proceedings*.

Kipf, T. N., & Welling, M. (2016). Variational graph auto-encoders. CoRR abs/1611.07308.

Kumar, A., Rai, P., & III, H. D. (2011). Co-regularized multi-view spectral clustering. In *NeurIPS* (pp. 1413–1421).

Li, S., Li, W., & Wang, W. (2020). Co-GCN for multi-view semi-supervised learning. In *Proc. AAAI* (pp. 4691–4698).

Li, G., Müller, M., Ghanem, B., & Koltun, V. (2021). Training graph neural networks with 1000 layers. In *ICML*.

Liao, S., Gao, Q., Yang, Z., Chen, F., Nie, F., & Han, J. (2018). Discriminant analysis via joint Euler transform and $\ell_{2,1}$-norm. *IEEE Transactions on Image Processing*, *27*(11), 5668–5682.

Liwicki, S., Tzimiropoulos, G., Zafeiriou, S., & Pantic, M. (2013). Euler principal component analysis. *International Journal of Computer Vision*, *101*(3), 498–518.

Lu, C., Feng, J., Lin, Z., Mei, T., & Yan, S. (2019). Subspace clustering by block diagonal representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*(2), 487–501.

Luo, F., Huang, H., Ma, Z., & Liu, J. (2016). Semisupervised sparse manifold discriminative analysis for feature extraction of hyperspectral images. *IEEE Transactions on Geoscience and Remote Sensing*, *54*(10), 6197–6211.

Luo, F., Zhang, L., Du, B., & Zhang, L. (2020). Dimensionality reduction with enhanced hybrid-graph discriminant learning for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, *58*(8), 5336–5353.

Lv, J., Kang, Z., Lu, X., & Xu, Z. (2021). Pseudo-supervised deep subspace clustering. *IEEE Transactions on Image Processing*, *30*, 5252–5263.

Ming, D., & Ding, C. (2019). Robust flexible feature selection via exclusive L21 regularization. In *Proc. IJCAI* (pp. 3158–3164).

Munkres., J. (1957). Normalized cuts and image segmentation. *Journal of the Society for Industrial and Applied Mathematics*, *5*(1), 32–38.

Nie, F., Cai, G., Li, J., & Li, X. (2018). Auto-weighted multi-view learning for image clustering and semi-supervised classification. *IEEE TIP*, *27*(3), 1501–1511.

Nie, F., Li, J., & Li, X. (2017). Self-weighted multiview clustering with multiple graphs. In *IJCAI* (pp. 2564–2570).

Nikolentzos, G., Dasoulas, G., & Vazirgiannis, M. (2020). K-hop graph neural networks. *Neural Networks*, *130*, 195–205.

Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., & Zhang, C. (2018). Adversarially regularized graph autoencoder for graph embedding. In *Proc. IJCAI* (pp. 2609–2615).

Pan, S., Wu, J., Zhu, X., Zhang, C., & Wang, Y. (2016). Tri-party deep network representation. In *IJCAI* (pp. 1895–1901).

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Stisen, A., Blunck, H., Bhattacharya, S., Prentow, T. S., Kjærgaard, M. B., Dey, A. K., et al. (2015). Smart devices are different: Assessing and MitigatingMobile sensing heterogeneities for activity recognition. In *Proc. ACM SenSys* (pp. 127–140).

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). LINE: large-scale information network embedding. In *WWW* (pp. 1067–1077).

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., & Su, Z. (2008). ArnetMiner: extraction and mining of academic social networks. In *ACM SIGKDD* (pp. 990–998).

Wan, S., Pan, S., Yang, J., & Gong, C. (2021). Contrastive and generative graph convolutional networks for graph-based semi-supervised learning. In *Proc. AAAI* (pp. 10049–10057).

Wang, C., Pan, S., Hu, R., Long, G., Jiang, J., & Zhang, C. (2019). Attributed graph clustering: A deep attentional embedding approach. In *Proc. IJCAI* (pp. 3670–3676).

Wu, J., Lin, Z., & Zha, H. (2019). Essential tensor learning for multi-view spectral clustering. *IEEE TIP*, *28*(12), 5910–5922.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.

Xia, R., Pan, Y., Du, L., & Yin, J. (2014). Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI* (pp. 2149–2155).

Xia, W., Wang, Q., Gao, Q., Zhang, X., & Gao, X. (2021). Self-supervised graph convolutional network for multi-view clustering. *IEEE Transactions on Multimedia*.

Xie, D., Gao, Q., Deng, S., Yang, X., & Gao, X. (2021). Multiple graphs learning with a new weighted tensor nuclear norm. *Neural Networks*, *133*, 57–68.

Xie, J., Girshick, R. B., & Farhadi, A. (2016). Unsupervised deep embedding for clustering analysis. In *Proc. ICML (vol. 24)* (pp. 478–487).

Xie, D., Zhang, X., Gao, Q., Han, J., Xiao, S., & Gao, X. (2020). Multiview clustering by joint latent representation and similarity learning. *IEEE Transactions on Cybernetics*, *50*(11), 4848–4854.

Xie, Y., Zhang, Y., Gong, M., Tang, Z., & Han, C. (2020). MGAT: multi-view graph attention networks. *Neural Networks*, *132*, 180–189.

Xie, Y., Zhang, W., Qu, Y., Dai, L., & Tao, D. (2020). Hyper-Laplacian regularized multilinear multiview self-representations for clustering and semisupervised learning. *IEEE Transactions on Cybernetics*, *50*(2), 572–586.

Xu, H., Zhang, X., Xia, W., Gao, Q., & Gao, X. (2020). Low-rank tensor constrained co-regularized multi-view spectral clustering. *Neural Networks*, *132*, 245–252.

Zhang, J., Li, C., You, C., Qi, X., Zhang, H., Guo, J., et al. (2019). Self-supervised convolutional subspace clustering network. In *Proc. IEEE CVPR* (pp. 5473–5482).

Zhang, H., Qiu, L., Yi, L., & Song, Y. (2018). Scalable multiplex network embedding. In *IJCAI* (pp. 3082–3088).

Zhao, H., Yang, X., Wang, Z., Yang, E., & Deng, C. (2021). Graph debiased contrastive learning with joint representation clustering. In *Proc. IJCAI* (pp. 3434–3440).