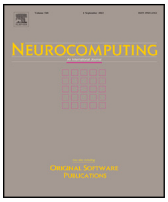
Neurocomputing 597 (2024) 128037

Contents lists available at ScienceDirect

Neurocomputing

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



Scalable and parameter-free fusion graph learning for multi-view clustering

Yu Duan [a], Danyang Wu [b,c], Rong Wang [d], Xuelong Li [d], Feiping Nie [a,*]

[a] School of Computer Science, School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), and the Key Laboratory of Intelligent Interaction and Applications (Ministry of Industry and Information Technology), Northwestern Polytechnical University, Xi'an 710072, Shaanxi, PR China
[b] College of Information Engineering, Northwest A&F University, Shaanxi, PR China
[c] Shaanxi Engineering Research Center for Intelligent Perception and Analysis of Agricultural Information, Northwest A&F University, Shaanxi, PR China
[d] School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), and the Key Laboratory of Intelligent Interaction and Applications (Ministry of Industry and Information Technology), Northwestern Polytechnical University, Xi'an 710072, Shaanxi, PR China

ARTICLE INFO

Communicated by C. Hou

Keywords:
Bipartite graph
Parameter-free fusion strategy
Multi-view clustering

ABSTRACT

Multi-view clustering aims to capture the consistency and complementary information present in view-specific data to achieve clustering alignment. However, existing multi-view clustering methods often rely on different regularization terms to quantify the importance of various views, which inevitably introduces additional hyper-parameters. It is challenging to fine-tune these additional parameters in real-world applications. Additionally, these methods suffer from high time complexity and impose substantial constraints when applied in large-scale scenarios. To address these limitations, we propose a parameter-free and time-efficient graph fusion method for multi-view clustering that can integrate view-specific graphs and directly generate clustering labels. Specifically, we introduce an anchor strategy and generate bipartite graphs on different views to enhance efficiency. Subsequently, we employ a self-weighted graph fusion strategy to merge the view-specific bipartite graphs. Finally, we propose a new solver to handle these problems, enabling the structured bipartite graphs to directly indicate clustering results. In contrast to previous clustering methods, our approach does not introduce any additional parameters and entirely relies on self-weighting for the fusion of view-specific graphs. As a result, our proposed method exhibits linear computational complexity to the data scale. Extensive experimental results on various benchmark datasets demonstrate the effectiveness and efficiency of our approach. Our code is available at https://github.com/DuannYu/MvSST.

1. Introduction

In the real-world scenarios, objects can be described from multiple perspectives. For instance, when analyzing persons, their height, weight, voice, movement, and other factors can be taken into account. This type of information is commonly referred to as multi-view data. On one hand, these view-specific features may share common information, which is known as *consistency*. On the other hand, they also contain unique information, referred to as *complementary*. The core of multi-view learning is how to effectively utilize the above properties of data to achieve the final task. In recent years, there has been significant research on multi-view data, including cross-view domain learning [1–3], multi-view classification [4–7], multi-view clustering [8–12] and so on.

This paper primarily focuses on multi-view clustering, which has witnessed remarkable contributions in recent years due to the *consistency* and *complementary* nature of multi-view data. These methods can be primarily categorized into two types: *explicit-based clustering* and *implicit-based clustering*. *Explicit-based clustering* are typically data-driven. The most representative one is subspace clustering [13–18]. Specifically, given data from the union of multiple subspaces, these methods attempt to find a joint low-dimensional embedding from multiple spaces and then perform data segmentation. For example, Cao et al. [19] learnt a common subspace that simultaneously captures the complementary and coherence of different feature spaces. Xu et al. [20] proposed an implicit multi-view k-means, which not only obtained the general subspace but also directly conducted clustering labels. Abavisani et al. [21] proposed a deep auto-encoder to affinity fusion, whose self-expression layers corresponding to different views in the network are the same and achieve good results. Moreover, Fan et al. [22] designed multi-view subspace learning via bidirectional sparsity (SLBS), which finds an effective subspace dimension and deals with outliers simultaneously.

However, in practice, the diversity in data distribution and scale hinders the performance of explicit-based clustering. In such contexts,

[*] Corresponding author.
E-mail addresses: duanyuee@gmail.com (Y. Duan), danyangwu.cs@gmail.com (D. Wu), wangrong07@tsinghua.org.cn (R. Wang), li@nwpu.edu.cn (X. Li), feipingnie@gmail.com (F. Nie).

https://doi.org/10.1016/j.neucom.2024.128037
Received 15 January 2024; Received in revised form 5 April 2024; Accepted 7 June 2024
Available online 9 June 2024

implicit-based clustering effectively addresses these limitations. These kind of methods typically rely on data similarity. For instance, Graph-based clustering, as a fundamental branch of *implicit-based clustering*, has received extensive attention [23–29]. It can effectively capture the nonlinear structural information among samples, leading to improve performance. Commonly, it first builds graphs by calculating the similarity between data from various views, then combines these graphs to obtain a unified graph through linear or nonlinear combination methods, and finally uses algorithms such as graph cuts to obtain clustering results. For example, Guo et al. [30] proposed a kernel alignment method to extend spectral clustering to a multi-view setting. Co-training and Co-reg [31] can fine-tune the coherent similarity matrix through the clustering results of each view. Zhan et al. [32] learned low-rank representations of consensus graphs from spectral embeddings. Nie et al. [33] proposed adaptive neighbor clustering, which learns the graph through adaptive local structure and imposes a rank constraint on the Laplacian matrix to obtain an ideal clustering structure. Liu et al. [34] also proposed a robust rank constrained sparse learning method, which introduces a sparsely represented $l_{2,1}$ norm objective function to learn a robust optimal graph. Moreover, Fang et al. [35] proposed UDBGL, which extract the advantage from both subspace clustering and rank constraint, achieving a discrete clustering solution without requiring additional partitioning.

Apart from the aforementioned two types of multi-view clustering methods, researchers have extended their focus to more complex scenarios. For instance, the missing of instances in certain views not only results in information loss but also poses challenges in establishing connections between different views. These challenges served as motivation for the development of incomplete multi-view clustering (IMVC) algorithms [36,37]. Moreover, in the case of two data matrices $\boldsymbol{X}^{(1)}$ and $\boldsymbol{X}^{(2)}$ corresponding to two views, only a small portion of the matrices has established prior correspondence. This partially view-aligned problem (PVP) can result in labor-intensive efforts to capture or establish aligned multi-view data [38,39]. Furthermore, some researchers have introduced the concept of ensemble learning into multi-view clustering, effectively bridging the relationship between these two fields [40]. This method is characterized by its scalability (suitable for extremely large-scale datasets), superiority (in terms of clustering performance), and simplicity (no need for dataset-specific tunning).

In sum, the methods mentioned above suffer from the following disadvantages:

- *Sensitive to parameters*. Many existing methods incorporate hyperparameters during model building or optimization. A common but crude way to find the proper parameters is grid search over a wide range. However, this manner significantly decreases efficiency. Additionally, it is impractical to adjust the parameters for every individual tasks.
- *High computational complexity*. The iterative process of existing multi-view spectral clustering methods often requires eigenvalue decomposition, which has a computational complexity of $\mathcal{O}(n^3)$. Another method based on subspace learning, which inevitably involves matrix inversion ($\mathcal{O}(n^3)$), also exhibiting high computational complexity and is unsuitable for large-scale data.
- *Require post-processing.* Graph-based clustering methods often fail to directly obtain the cluster labels after learning the optimal graph. They usually require additional steps such as k-means or spectral rotation [41] to obtain the final result. Namely, two-stage strategies may not achieve the optimal result and introduce variance due to their randomness.

To address the aforementioned issues, we propose a parameter-free and time-efficient multi-view clustering algorithm. Different with UDBGL, our proposed method is parameter-free and more focuses on connected components. Specifically, we initially construct a bipartite graph using an anchor strategy, which substantially reduces the computational complexity. Next, we employ an adaptive weighting method to seamlessly fuse bipartite graphs from different views, without any regularization or hyperparameters. Since the connected components of the graph directly represent the clustering results, we formally control the number of connected components of the graph, turning it into an optimization problem. Finally, we perform joint optimization of the adaptive fusion graph and graph structure learning in an end-to-end way. Fig. 1 shows the flow of the entire algorithm. The main contributions of this paper are summarized as follows:

Table 1
Summary of required notations.

Notations	Descriptions
$\boldsymbol{M}$	arbitrary matrix
m_{ij}	i, j-th entry of $\boldsymbol{M}$
m_i	ith column of $\boldsymbol{M}$
m^j	jth row of $\boldsymbol{M}$
$\vec{1}$	column vector with all entries 1
$\mathrm{tr}(\boldsymbol{M})$	trace of $\boldsymbol{M}$
$\|\|M\|\|_F$	$\boldsymbol{M}$'s F-norm
$diag(\cdot)$	extract a diagonal or construct a diagonal array

- *Parameter-Free*: we propose a parameter-free graph fusion strategy, which combines view-specific bipartite graphs without requiring any additional parameters or regularization. Moreover, our proposed method does not require post-processing to obtain cluster labels. It can directly obtain cluster labels in an end-to-end manner.
- *Time-Economic*: we employ an anchor strategy on multi-view data to enhance the algorithm's efficiency and enable its application to large-scale datasets. It effectively reduces the complexity from $O(n^2d)$ to linear scale with respect to the data size.
- *Theoretical Guarantee*: we find a property of the connected components in bipartite graph. Based on these observation, we propose a new solver to learn structured bipartite graph that ensures convergence theoretically.
- *Extensive Experiments*: we empirically validate the correctness of the proposed theory through a comprehensive set of experiments. The experimental results demonstrate the superiority and efficiency of our method compared to other methods.

The rest of paper are organized as follows. We introduce prior knowledge of our proposed method in Section 2. Section 3 describes the proposed approach and its optimization process. We perform a theoretical analysis in Section 4. Section 5 reports experimental results. Finally, the conclusion of the paper is provided in Section 6. For better readability, we summarize the necessary variables in Table 1.

2. Preliminaries

2.1. Graph based clustering

It is widely recognized that numerous popular graph-based methods need post-processing to obtain the final labels, which can lead to suboptimal results potentially. To handle this issue, we firstly introduce an important theorem that is shown as follow.

Theorem 1. *The multiplicity of the eigenvalue zero of the Laplacian matrix $\boldsymbol{L}_A$ equals to the number of connected components in the graph associated with $\boldsymbol{A}$.*

Inspired by Theorem 1, Nie et al. [42] introduced a method known as Constrained Laplacian Rank (CLR). It can directly obtain a structured optimal graph that indicates the exact cluster number. Its objective function is written below:

$$\min_{\boldsymbol{A}^T\vec{1}=\vec{1},\boldsymbol{A}\geq\vec{0},\mathrm{rank}(\boldsymbol{L}_A)=n-c} \|\boldsymbol{A}-\boldsymbol{W}\|_F^2, \tag{1}$$

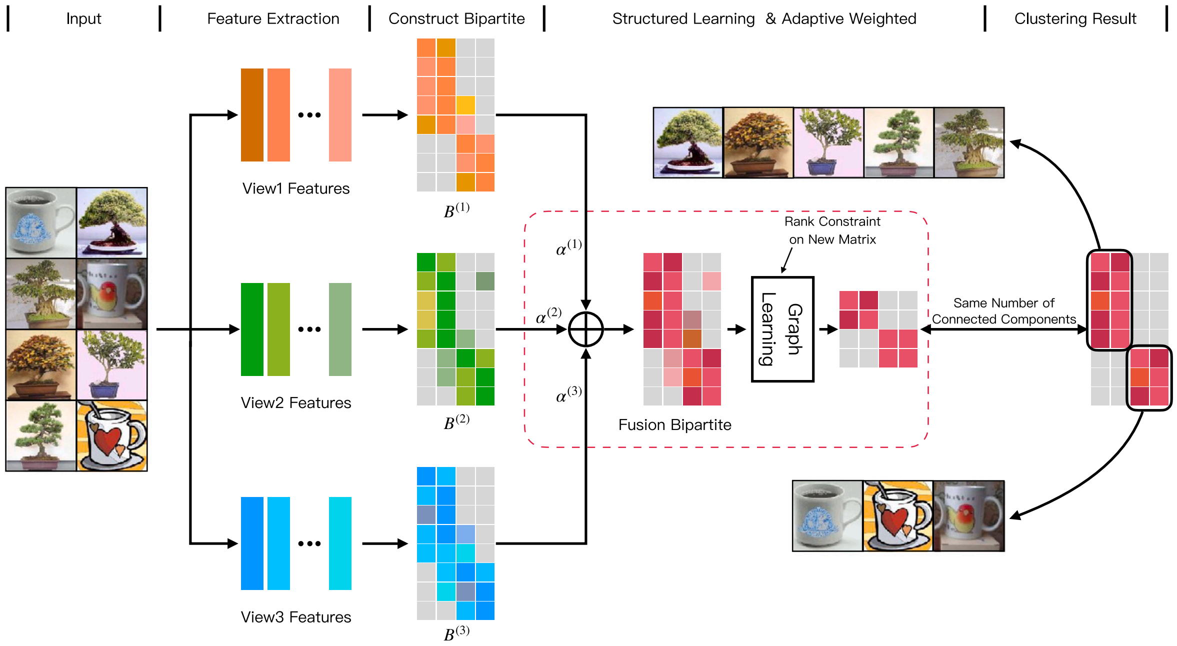


Fig. 1. Flow diagram of the proposed method. Given a set of data, we first extract features by using different way to get $V = 3$ view-specific features, namely, $\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \boldsymbol{X}^{(3)}$. We then construct view-specific bipartite $\boldsymbol{B}^{(1)}, \boldsymbol{B}^{(2)}, \boldsymbol{B}^{(3)}$. In the red dotted box, our method iteratively updates weights $\vec{\alpha}$ and learns graph until it has c connected components. Finally, we can get clustering labels according to the graph structure.

where $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ is an initial pairwise graph between n data. $\boldsymbol{L}_{\boldsymbol{A}}$ represents the corresponding Laplacian matrix of $\boldsymbol{A}$, where $\boldsymbol{A} \geq \vec{0}$ indicates that $\forall i, j = 1, 2, \ldots, n, a_{ij} \geq 0$. The core idea of this model is imposing rank constraint on the Laplacian matrix, which ensures that $\boldsymbol{A}$ has precisely c clusters, as indicated by its connected components. According to its connected components, we could easily obtain the cluster labels.

2.2. Re-weighted framework

The re-weighted method is an optimization framework widely used in many machine learning fields such as regression [43], low rank learning [44] and so on. It is also capable of optimizing robust loss functions, such as the Cauchy norm [45]. In summary, the re-weighted framework tackles the following general problems:

$$\min_{x\in\Omega} f(x) + \sum_i h(g_i(x)), \tag{2}$$

where x, $f(x)$, $g(x)$ can be scalar, vector or matrix, $h(\cdot)$ is a **concave function** and Ω is arbitrary constraints *w.r.t.* x. Let $D_i = h_i'(g_i(x))$, the Eq. (2) can be alternatively solved by the following sub-problem:

$$\min_{x\in\Omega} f(x) + \sum_i tr(D_i^T g_i(x)). \tag{3}$$

Specifically, after initializing the variable x, we alternatively solve the following two sub-problem until convergence: *(a)* calculate each D_i using the current value of x; *(b)* optimize the sub-Eq. (3) to update x based on the current D_i. Finally, we obtain the optimal solution x^*, which satisfies the KKT conditions of Eq. 1. The entire iterative procedure is summarized in Algorithm 1.

Algorithm 1 Re-Weighted Framework for Eq. (2)

```
Input: Initialized $x \in \Omega$
Output: optimal $x$
while not converge do
    Calculate $D_i$ by $h_i'(g_i(x))$;
    Update $x$ by solving Eq. (3);
end while
```

3. Proposed methods

3.1. Motivation and proposed model

Applying graph-based methods to large-scale datasets poses challenges primarily due to the optimization process and storage requirements. Moreover, these methods often require additional regularizations to effectively integrate view-specific information, introducing extra parameter tunning. Meanwhile, existing methods typically need post-processing to obtain cluster assignments. Considering the aforementioned three aspects, we propose a parameter-free and time-efficient multi-view clustering method. Firstly, we introduce the anchor strategy for constructing view-specific bipartite graphs, which reduces both time and storage costs. Additionally, we propose a parameter-free graph fusion strategy that enables adaptive weighting of different views. In sum, our model can be formulated as follow:

$$\begin{aligned} &\min_{\boldsymbol{P}, \vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0} \left\| \boldsymbol{P} - \sum_{v=1}^{V} \vec{\alpha}^{(v)} \boldsymbol{B}^{(v)} \right\|_F^2, \\ &\quad s.t. \boldsymbol{P} \geq 0, \boldsymbol{P}\vec{1} = \vec{1}, \boldsymbol{P} \in \Omega \end{aligned} \tag{4}$$

where $\boldsymbol{B}^{(v)}$ is vth view-specific bipartite graph generated between data and anchors and $\alpha^{(v)}$ measures the importance of vth view (Section 4.3 will discuss how to build $\boldsymbol{B}^{(v)}$ in detail). Assume we have known the cluster number is c, Ω in Eq. (4) denotes $\boldsymbol{P}$ has c connected components.

Like previous works [46,47], to make $\boldsymbol{P}$ has exactly c connect components, we need to impose rank constraint on Laplacian matrix corresponding to $\boldsymbol{S} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{P} \\ \boldsymbol{P}^T & \boldsymbol{0} \end{bmatrix}$, and we should leverage eigenvalue decomposition on a $(n+m)\times(n+m)$ matrix. However, they suffer from high complexity and lacked theoretical guarantees for convergence during optimization. To handle these issue, we first introduce a relationship of connected components between bipartite graph and pairwise graph, which is presented as following.

Theorem 2. *$\boldsymbol{P}$ and $\boldsymbol{P}^T\boldsymbol{P}$ have same number of connected components.*

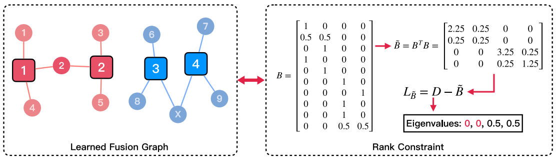


Fig. 2. An intuitive example of Theorem 2. The small circles indicate data and big squares indicate anchors.

Proof. There are two facts we have:

1. If $\boldsymbol{P}$ has c connected components, there exists permutation matrices $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{V} \in \mathbb{R}^{m\times m}$, which make $\boldsymbol{UPV}$ block diagonal with c blocks. Since $(\boldsymbol{UPV})^T\boldsymbol{UPV} = \boldsymbol{V}^T\boldsymbol{P}^T\boldsymbol{PV}$, $\boldsymbol{V}^T\boldsymbol{P}^T\boldsymbol{PV}$ is block diagonal with c blocks. **Therefore, $\boldsymbol{P}^T\boldsymbol{P}$ at least has c connected components.**
2. If there is a data x_h connected to a_i and a_j, where a_i and a_j do not belong to the same connected component, we have $p_{hi} > 0$ and $p_{hj} > 0$. More recently, $(\boldsymbol{P}^T\boldsymbol{P})_{ij} > 0$, because $(\boldsymbol{P}^T\boldsymbol{P})_{ij} = \sum_{k=1}^{n} p_{ki}\cdot p_{kj} = \sum_{k\neq h}^{n} p_{ki}\cdot p_{kj} + p_{hi}\cdot p_{hj} > 0$, contradiction. Therefore if $\boldsymbol{P}^T\boldsymbol{P}$ has c connected components, the node in the datasets cannot connect the two connected components of $\boldsymbol{P}^T\boldsymbol{P}$ at the same time. So, **$\boldsymbol{P}$ has at least c connected components.**

In sum, $\boldsymbol{P}$ and $\boldsymbol{P}^T\boldsymbol{P}$ have same connected components. □

Fig. 2 displays an intuitive example of Theorem 2. According to Theorem 2, we could impose constraint on a new Laplacian matrix corresponding to $\boldsymbol{P}^T\boldsymbol{P} \in \mathbb{R}^{m\times m}$, which greatly reduce the matrix scale, and handle the high computational complexity of eigenvalue decomposition. For simplicity, let $\tilde{\boldsymbol{P}} = \boldsymbol{P}^T\boldsymbol{P}$, Eq. (4) is further equivalent to the following:

$$\min_{\vec{\alpha}^T\vec{1}=1,\vec{\alpha}\geq 0,\boldsymbol{P}} \left\| \boldsymbol{P} - \sum_{v=1}^{V} \vec{\alpha}^{(v)}\boldsymbol{B}^{(v)} \right\|_F^2, \quad s.t. \boldsymbol{P} \geq 0, \boldsymbol{P}\vec{1} = \vec{1}, \mathrm{rank}(\boldsymbol{L}_{\tilde{\boldsymbol{P}}}) = m - c, \tag{5}$$

where $\boldsymbol{L}_{\tilde{\boldsymbol{P}}}$ stands for Laplacian matrix corresponding to $\tilde{\boldsymbol{P}}$. Once $\boldsymbol{L}_{\tilde{\boldsymbol{P}}}$ has separated connected components, we can get structured $\tilde{\boldsymbol{P}}$ and cluster labels.

3.2. Optimization

In this section, as Eq. (5) is not jointly convex respects to all variables, we divided the Eq. (5) into two sub-problems, and leverage an alternative optimization strategy to update $\vec{\alpha}$ and $\boldsymbol{P}$.

3.2.1. Fixed $\vec{\alpha}$, update $\boldsymbol{P}$

When $\vec{\alpha}$ fixed, Eq. (5) is simplified into

$$\min_{\boldsymbol{P}} \|\boldsymbol{P} - \boldsymbol{W}\|_F^2, \quad s.t. \boldsymbol{P} \geq 0, \boldsymbol{P}\vec{1} = \vec{1}, \mathrm{rank}(\boldsymbol{L}_{\tilde{\boldsymbol{P}}}) = m - c, \tag{6}$$

where $\boldsymbol{W} = \sum_{v=1}^{V} \vec{\alpha}^{(v)}\boldsymbol{B}^{(v)}$. Since rank constraint is an NP-hard problem, we should solve it in a relax manner. Let us define $\sigma_i(\boldsymbol{L}_{\tilde{\boldsymbol{P}}})$ as the ith smallest eigenvalue of $\boldsymbol{L}_{\tilde{\boldsymbol{P}}}$, it is worth noting that all eigenvalues of $\boldsymbol{L}_{\tilde{\boldsymbol{P}}}$ are greater than 0 because it is positive semi-definite. According to Theorem 1, when the first c smallest $\sigma_i(\boldsymbol{L}_{\tilde{\boldsymbol{P}}})$ equal zero while the remaining ones are greater than 0, the rank constraint is satisfied. Based on this observation, we relax Eq. (6) into

$$\min_{\boldsymbol{P}\geq 0,\boldsymbol{P}\vec{1}=\vec{1}} \|\boldsymbol{P} - \boldsymbol{W}\|_F^2 + \lambda \sum_{i=1}^{c} \sigma_i(\boldsymbol{L}_{\tilde{\boldsymbol{P}}}). \tag{7}$$

When λ is large enough, the second term will infinitely approach to zero. To solve the Eq. (7), we need to introduce a vital equation called Ky Fan's theorem [48].

Theorem 3 (*Ky Fan's theorem*). *∀ Laplacian matrix $\boldsymbol{L}$, sum of the first c minimum eigenvalues of $\boldsymbol{L}$ can be calculated by the following minimize problem:*

$$\sum_{i=1}^{c} \sigma_i(\boldsymbol{L}) = \min_{\boldsymbol{F}^T\boldsymbol{F}=\boldsymbol{I}} tr(\boldsymbol{F}^T\boldsymbol{L}\boldsymbol{F}). \tag{8}$$

According Ky Fan's theorem, Eq. (7) can be written as

$$\min_{\boldsymbol{P}\geq 0,\boldsymbol{P}\vec{1}=\vec{1},\boldsymbol{F}^T\boldsymbol{F}=\boldsymbol{I}} \|\boldsymbol{P} - \boldsymbol{W}\|_F^2 + \lambda \mathrm{tr}(\boldsymbol{F}^T\boldsymbol{L}_{\tilde{\boldsymbol{P}}}\boldsymbol{F}). \tag{9}$$

So far, we need to update two variables in problem (9) alternatively. **When $\boldsymbol{P}$ is fixed**, Eq. (9) becomes

$$\min_{\boldsymbol{F}^T\boldsymbol{F}=\boldsymbol{I}} \mathrm{tr}(\boldsymbol{F}^T\boldsymbol{L}_{\tilde{\boldsymbol{P}}}\boldsymbol{F}). \tag{10}$$

It is obvious that the optimal $\boldsymbol{F}$ is the eigenvectors of $\boldsymbol{L}_{\tilde{\boldsymbol{P}}}$ corresponding to its c smallest eigenvalues.

When $\boldsymbol{F}$ is fixed, the Eq. (9) simplifies as

$$\min_{\boldsymbol{P}\geq 0,\boldsymbol{P}\vec{1}=\vec{1}} \|\boldsymbol{P} - \boldsymbol{W}\|_F^2 + \lambda \mathrm{tr}(\boldsymbol{F}^T\boldsymbol{L}_{\tilde{\boldsymbol{P}}}\boldsymbol{F}). \tag{11}$$

To solve Eq. (11), we need introduce the following lemma.

Lemma 1. *For matrix $\boldsymbol{P}$ satisfies $\boldsymbol{P} \geq 0$ and $\boldsymbol{P}\vec{1} = \vec{1}$, we have an important equation as follow*

$$tr(\boldsymbol{F}^T\boldsymbol{L}_{\tilde{\boldsymbol{P}}}\boldsymbol{F}) = tr(\mathbf{1}diag(\boldsymbol{F}\boldsymbol{F}^T)^T\boldsymbol{P}^T) - tr(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T) \tag{12}$$

Proof. For matrix $\boldsymbol{P}$ satisfied $\boldsymbol{P} \geq 0$ and $\boldsymbol{P}\vec{1} = \vec{1}$, we have an important equations as follow

$$\begin{aligned}
&\mathrm{tr}(\boldsymbol{F}^T\boldsymbol{L}_{\tilde{\boldsymbol{P}}}\boldsymbol{F})\\
=&\mathrm{tr}(\boldsymbol{F}^T(\boldsymbol{D}_{\tilde{\boldsymbol{P}}} - \boldsymbol{P}^T\boldsymbol{P})\boldsymbol{F})\\
=&\mathrm{tr}(\boldsymbol{F}^T\boldsymbol{D}_{\tilde{\boldsymbol{P}}}\boldsymbol{F}) - \mathrm{tr}(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T)\\
=&\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T)^T\boldsymbol{P}^T\boldsymbol{P}\mathbf{1} - \mathrm{tr}(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T)\\
=&\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T)^T\boldsymbol{P}^T\mathbf{1} - \mathrm{tr}(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T)\\
=&\mathrm{tr}(\mathbf{1}\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T)^T\boldsymbol{P}^T) - \mathrm{tr}(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T).
\end{aligned} \tag{13}$$

where $\boldsymbol{D}_{\tilde{\boldsymbol{P}}}$ is degree matrix of $\boldsymbol{P}^T\boldsymbol{P}$ and $\boldsymbol{L}_{\tilde{\boldsymbol{P}}}$ is Laplacian matrix corresponding to $\boldsymbol{P}^T\boldsymbol{P}$. □

According to Lemma 1, Eq. (11) can be rewritten as

$$\begin{aligned}
\min_{\boldsymbol{P}\geq 0,\boldsymbol{P}\vec{1}=\vec{1}} &\|\boldsymbol{P} - \boldsymbol{W}\|_F^2\\
&+ \lambda\mathrm{tr}(\mathbf{1}\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T)^T\boldsymbol{P}^T) - \lambda\mathrm{tr}(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T).
\end{aligned} \tag{14}$$

Then, we rewrite the first term in a trace form, Eq. (14) is further equivalent to:

$$\begin{aligned}
\min_{\boldsymbol{P}\geq 0,\boldsymbol{P}\vec{1}=\vec{1}} &\mathrm{tr}(\boldsymbol{P}\boldsymbol{P}^T - 2\boldsymbol{P}\boldsymbol{W})\\
&+ \lambda\mathrm{tr}(\mathbf{1}\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T)^T\boldsymbol{P}^T) - \lambda\mathrm{tr}(\boldsymbol{P}\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^T).
\end{aligned} \tag{15}$$

Because each row of $\boldsymbol{P}$ is independent, we can solve above problem row by row. For each row of $\boldsymbol{P}$, we obtain the following subproblem

$$\min_{\vec{p}\geq 0,\vec{p}\vec{1}=1} \vec{p}\vec{p}^T - 2\vec{p}\vec{w}^T + \lambda\vec{p}\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T) - \lambda\vec{p}\boldsymbol{F}\boldsymbol{F}^T\vec{p}^T \tag{16}$$

where $\vec{p}$ and $\vec{w}$ are arbitrary row of $\boldsymbol{P}$ and $\boldsymbol{W}$, respectively.

To this end, we could use the re-weighted framework to solve the Eq. (16) as follow. Regarding last term as a concave function w.r.t. $\vec{p}$, Eq. (16) can be iteratively solved by the following subproblem[1]:

$$\min_{\vec{p}\geq 0,\vec{p}\vec{1}=1} \vec{p}\vec{p}^T - 2\vec{p}\vec{w}^T + \lambda\vec{p}\mathrm{diag}(\boldsymbol{F}\boldsymbol{F}^T) - 2\lambda\vec{p}\vec{d}_i, \tag{17}$$

[1] The hyper-parameter λ controls the number of connected components, and it need not be tuned manually. Actually, for accelerating convergence of the model, we use a tricky fashion to determine λ. To be specific, if connected components of $\boldsymbol{P}^T\boldsymbol{P}$ is less than cluster number, double the λ by $\lambda \leftarrow \lambda \times 2$. On the contrary, when the number of connected components is greater than the cluster number, half the λ by $\lambda \leftarrow \lambda/2$.

Algorithm 2 Algorithm to Solve Eq. (6)

```
Input: Given fusion graph $\boldsymbol{W}$, cluster number $c$
Output: structured graph $\boldsymbol{P}$ with exact $c$ connected components.
1: Initial $\vec{F}$ as eigenvectors of $\tilde{\boldsymbol{L}}_W$ corresponding to $c$-th smallest eigenvalues;
2: $\forall i = 1, 2, .., n$, initial $\vec{d}_i = \boldsymbol{F}\boldsymbol{F}^T \vec{w}^T$;
3: while not converge do
4:     while not converge for each $\vec{p}$ do
5:         Update $\vec{p}$ by solving Eq. (18);
6:         Update $\vec{d}_i = \boldsymbol{F}\boldsymbol{F}^T \vec{p}^T$;
7:     end while
8:     Update $\boldsymbol{F}$ by solving Eq. (10);
9:     Update $\lambda$ according to the number of the connected components of $\boldsymbol{P}^T\boldsymbol{P}$.
10: end while
```

where $\vec{d}_i = \boldsymbol{F}\boldsymbol{F}^T \vec{p}^T$. Furthermore, Eq. (17) can be further transformed into

$$\min_{\vec{p}\geq 0, \vec{p}\vec{1}=1} \left\| \vec{p} + (\frac{1}{2}\lambda \text{diag}(\boldsymbol{F}\boldsymbol{F}^T)^T - \vec{w} - \lambda \vec{d}_i^T) \right\|_2^2. \tag{18}$$

Finally, the optimal $\vec{p}$ could be efficiently solved with a closed form solution by [42]. Algorithm 2 summarizes the pipeline of solving Eq. (6).

3.2.2. Fixed $\boldsymbol{P}$, update $\vec{\alpha}$

When $\boldsymbol{P}$ fixed, we have

$$\min_{\vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0} \left\| \boldsymbol{P} - \sum_{v=1}^{V} \vec{\alpha}^{(v)} \boldsymbol{B}^{(v)} \right\|_F^2, \tag{19}$$

To solve above problem, we expand $\boldsymbol{B}^{(v)}$ to a column vector as $\hat{\vec{b}}^{(v)} \in \mathbb{R}^{nm\times 1}$. So we can merge all views together as a new matrix that $\hat{\boldsymbol{B}} = \left[\hat{\vec{b}}^{(1)}, \hat{\vec{b}}^{(2)}, \dots, \hat{\vec{b}}^{(V)}\right] \in \mathbb{R}^{nm\times V}$. Noting that $\boldsymbol{W} = \sum_{v=1}^{V} \vec{\alpha}^{(v)} \boldsymbol{B}^{(v)}$, we expand $\boldsymbol{W}$ into column form $\hat{\vec{w}}$, then we have $\hat{\vec{w}} = \hat{\boldsymbol{B}}\vec{\alpha}$. Also $\hat{\vec{p}}$ expanded from $\boldsymbol{P}$, Eq. (19) can be reformulated to

$$\begin{aligned} &\min_{\vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0} \left\| \hat{\vec{p}} - \hat{\boldsymbol{B}}\vec{\alpha} \right\|_2^2, \\ \Leftrightarrow &\min_{\vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0} \vec{\alpha}^T \boldsymbol{M} \vec{\alpha} - \vec{\alpha}^T \vec{h}, \end{aligned} \tag{20}$$

where $\boldsymbol{M} = \hat{\boldsymbol{B}}^T \hat{\boldsymbol{B}}$ and $\vec{h} = \hat{\boldsymbol{B}}^T \hat{\vec{p}}$. Because $\boldsymbol{M}$ is positive semi-definite, the Augmented Lagrangian Multiplier (ALM) method can solve the problem above efficiently.

Briefly speaking, Eq. (20) can be solved by its equivalent counterpart as follow:

$$\min_{\vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0, \vec{\alpha}=\vec{\beta}} \vec{\alpha}^T \boldsymbol{M} \vec{\beta} - \vec{\alpha}^T \vec{h}. \tag{21}$$

Then, the augmented Lagrangian function of above is defined as

$$\min_{\vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0, \vec{\beta}} \vec{\alpha}^T \boldsymbol{M} \vec{\beta} - \vec{\alpha}^T \vec{h} + \frac{\mu}{2} \left\| \vec{\alpha} - \vec{\beta} + \frac{\vec{\eta}}{\mu} \right\|_2^2. \tag{22}$$

So far, we could alternatively update the $\vec{\alpha}$, $\vec{\beta}$ and $\vec{\mu}$ to minimize the Eq. (22). Formally, the $\vec{\alpha}$ could be updated by solving the following problem,

$$\min_{\vec{\alpha}^T\vec{1}=1, \vec{\alpha}\geq 0} \left\| \vec{\alpha} - \vec{\beta} + \frac{1}{\mu} \left(\vec{\eta} + \boldsymbol{M}\vec{\beta} - \vec{h} \right) \right\|_2^2. \tag{23}$$

We then could leverage the close forms to update the $\vec{\beta}$ and $\vec{\mu}$ as follow,

$$\vec{\beta} = \vec{\alpha} + \frac{1}{\mu}(\vec{\eta} - \boldsymbol{M}^T \vec{\alpha}) \tag{24}$$

Algorithm 3 The Whole Algorithm to Solve Eq. (4)

```
Input: Each single-view bipartite $\boldsymbol{B}^{(v)} \in \mathbb{R}^{n\times m}$, view number $V$, cluster number $c$
Output: structured graph $\boldsymbol{P}$ with exact $c$ connected components.
Initial weight factor $\vec{\alpha} = \frac{1}{V}$, compute $\boldsymbol{W} = \sum_{v=1}^{V} \boldsymbol{B}^{(v)}$;
while not converge do
    Update $\boldsymbol{P}$ by Algorithm 2;
    Transform $\forall v$, $\boldsymbol{B}^{(v)}$ to $\hat{\boldsymbol{B}}$ and $\boldsymbol{P}$ to $\hat{\vec{p}}$ respectively;
    Compute $\boldsymbol{M} = \hat{\boldsymbol{B}}^T \hat{\boldsymbol{B}}$ and $\vec{h}$;
    Update $\vec{\alpha}$ by solving Eq. (20).
end while
```

and

$$\vec{\eta} = \vec{\eta} + \mu(\vec{\alpha} - \vec{\beta}). \tag{25}$$

To this end, We further summarize the whole flow of solving Eq. (4) in Algorithm 3.

4. Theoretical analysis

4.1. Convergence analysis

As aforementioned, solving Eq. (5) can be divided into two separate sub-problems Eq. (6) and (19). Firstly, we discuss the convergence of solving Eq. (6). Since the update of $\boldsymbol{P}$ is row independent, we will solely focus on the convergence of solving Eq. (16).

Lemma 2. *The objective value of Eq. (16) will decrease in each iteration until convergence.*

Proof. Assume that after t iterations, we have obtained $\vec{p}_t$ and $\vec{d}_t$, and in the subsequent iterations, we obtain the optimal $\vec{p}_{t+1}$. For simplicity, we omit the row and column indices without any ambiguity, and rewrite Eq. (16) below.

$$\min_{\vec{p}_t\leq 0, \vec{p}_t\vec{1}=1} \vec{p}_t\vec{p}_t^T - 2\vec{p}_t\vec{w}^T + \lambda\vec{p}_t \text{diag}(\boldsymbol{F}\boldsymbol{F}^T) - \lambda\vec{p}_t\boldsymbol{F}\boldsymbol{F}^T\vec{p}_t^T \tag{26}$$

The last term of Eq. (26) is a concave function w.r.t. $\vec{p}_t$ so that we could rewrite it into a re-weighted form:

$$\min_{\vec{p}\leq 0, \vec{p}\vec{1}=1} f(\vec{p}_t) + h(\vec{p}_t) \tag{27}$$

where $f(\vec{p}_t) = \vec{p}_t\vec{p}_t^T - 2\vec{p}_t\vec{w}^T + \lambda\vec{p}_t \text{diag}(\boldsymbol{F}\boldsymbol{F}^T)$ and $h(\vec{p}_t) = -\lambda\vec{p}_t\boldsymbol{F}\boldsymbol{F}^T\vec{p}_t^T$. For concave function $h(\vec{p})$, $\vec{p} \in \textbf{dom}h$, we have the following inequality

$$h(\vec{p}_{t+1}) - h(\vec{p}_t) \leq (\vec{p}_{t+1} - \vec{p}_t)\vec{d}_t. \tag{28}$$

Likely, $\vec{d}_t = \boldsymbol{F}\boldsymbol{F}^T\vec{p}_t^T$. Meanwhile, in the line 5 of Algorithm 2, the following inequality holds:

$$f(\vec{p}_{t+1}) + \vec{p}_{t+1}\vec{d}_t \leq f(\vec{p}_t) + \vec{p}_t\vec{d}_t. \tag{29}$$

Summing the two above we obtain

$$f(\vec{p}_{t+1}) + h(\vec{p}_{t+1}) \leq f(\vec{p}_t) + h(\vec{p}_t). \quad \square \tag{30}$$

When $\boldsymbol{P}$ fixed, we use ALM method to solve the optimal $\vec{\alpha}$. Since the ALM method converges fast [49], the objective function will converge after alternatively finding the optimal solution to each subproblem.

4.2. Complexity analysis

This section focuses on analyzing the computational complexity. Let dataset $\boldsymbol{X}^{(v)} \in \mathbb{R}^{n\times d_v}$ be obtained, where $v \in 1, 2, \dots, V$ and V is number of views, d_v is data dimension and m refers to the number of anchors. The computational complexity can be calculated as followings.

For all views of data, we need $\mathcal{O}(Vnmd_v + Vnm\log m)$ to construct all bipartite graphs. As mentioned in the previous section, we need alternatively solve two subproblems to get optimal $\boldsymbol{P}$. **When fixed** $\vec{\alpha}$, we use Algorithm 2 to obtain $\boldsymbol{P}$. Noting that using the re-weighted framework to get each row of $\boldsymbol{P}$ is fast in practice, it only takes the computational complexity of $\mathcal{O}(n)$. As for eigenvalue decomposition, it takes $\mathcal{O}(m^3)$. Finally Algorithm 2 takes $\mathcal{O}(nt + m^3t)$ computational complexity where t is iteration number.

When fixed $\boldsymbol{P}$, we ignore the complexity of obtaining $\vec{\alpha}$ because of updating it is also fast according to its relatively small dimension. Considering compute $\boldsymbol{M}$ and $\vec{h}$, which take $\mathcal{O}(V^2\ nm)$ and $\mathcal{O}(Vnm)$ respectively. Noting that the loop number of ALM is always less than five, the main computational complexity in this step is $\mathcal{O}(nt + m^3t + V^2\ nm)$.

Taking consideration that $m \ll n$, the total computational complexity of Algorithm 3 approximates as $\mathcal{O}(nt + nmV^2 + m^3)$ which has a significant computational advantage on large-scale datasets compared with $\mathcal{O}(n^2d)$ methods. The time consumption of running the proposed algorithm will be tested in the later experimental section.

4.3. Single-view initial graph construction

So far, we have extensively discussed the process of obtaining the final optimal graph $\boldsymbol{P}$ given the initial bipartite graphs. It is well known that the quality of the initial graph also impacts the final clustering result. Therefore, we mainly talk about how to initial graph for each view in this section. Motivated by CAN [50], the initial graph for each view can be obtained by solving the following problem.

$$\min_{\boldsymbol{B}\geq 0, \boldsymbol{B}\mathbf{1}=\vec{1}} \sum_{i=1}^{n}\sum_{j=1}^{m} \mathcal{D}(\vec{x}_i,\vec{z}_j)\, b_{ij}^{(v)} + \gamma \left(b_{ij}^{(v)}\right)^2, \tag{31}$$

where $\mathcal{D}(\vec{x}_i,\vec{z}_j)$ measures the difference of ith sample $\vec{x}_i$ and jth anchor $\vec{z}_j$, for short as $\mathcal{D}_{i,j} = \|\vec{x}_i - \vec{z}_j\|_2^2$. The first term of Eq. (31) makes $\boldsymbol{B}^{(v)}$ capture the local information between the samples and anchors. The closer the distance, the greater the value of $\boldsymbol{B}^{(v)}$. The second term serves as a regularization, encouraging anchors to connect with as many samples as possible. We tune the hyper-parameter γ to control the number of anchors connected to each sample. Supposed $\mathcal{D}_{i,[j]}$ is distance between ith sample and its jth nearest anchor that $\mathcal{D}_{i,[1]} \leq \mathcal{D}_{i,[2]}, \ldots, \mathcal{D}_{i,[m]}$. The closed-form solution for Eq. (31) is as follows.

$$b_{ij}^{(v)} = \max\left(\frac{\mathcal{D}_{i,[k+1]} - \mathcal{D}_{i,[j]}}{k\mathcal{D}_{i,[k+1]} - \sum_{l=1}^{k}\mathcal{D}_{i,[l]}}, 0\right), \tag{32}$$

where k is the number of each sample connects to and γ can be computed as $\gamma = \frac{k}{2}\mathcal{D}_{i,[k=1]} - \frac{1}{2}\sum_{l=1}^{k}\mathcal{D}_{i,[l]}$. The detailed discussion can be seen in [50].

5. Experiments

In this section, we firstly conduct visual experiments on toy datasets to illustrate the performance of the method. Then we present the experimental results from aspects of performance on real benchmark datasets compared with some state-of-the-art methods, computational time, convergence and parameter sensitivity.

5.1. Experiments on toy dataset

In this part, we mainly want to explain the following matters: *(a) how method combines the view-specific bipartite graph in a parameter-free manner*, and *(b) how method learns the final optimal graph*. For all experiments, we consider the algorithm to have converged if the change in the objective function value is less than $1e-4$. Moreover, the maximum number of iterations for the algorithm is set to 50.

Based on these considerations, we design a triple-view optimal graph learning task. Figs. 3(a), 3(b), and 3(c) are single-view bipartite graphs which are used as inputs. Figs. 3(a) and 3(b) exhibit two diagonal block distributions with slight noise, which are somewhat ambiguous. Fig. 3(c) is a noise view without any information. The proposed method utilizes the triple-view graph yielding optimal results, as demonstrated in Fig. 3(e). It is evident that Fig. 3(e) exhibits a distinct block-diagonal structure, derived from view-1 and view-2, while disregarding view-3. As shown in Fig. 3(f), all three view graphs have equal weights at the beginning. However, as the weights updating, the model gradually captures clustering information while disregarding irrelevant noise. Consequently, the weights of view-1 and view-2 progressively increase, whereas the weight of view-3 diminishes until it reaches zero.

If we denote Fig. 3(e) is graph $\boldsymbol{P}$, Fig. 3(d) shows pairwise graph $\boldsymbol{P}^T\boldsymbol{P}$. It directly shows that both of them have three clear connected blocks, which verifies Theorem 2.

Table 2
The detail information's of multi-view datasets.

Datasets	#Samples	#Views	#Classes
COIL20	1440	3	20
MNIST	10 000	3	10
HandWritten	2000	6	10
Caltech101-7	1474	6	7

5.2. Experimental settings

5.2.1. Datasets

We use four datasets in this paper to evaluate the proposed method. Table 2 summarizes the information of all multi-view datasets and Fig. 4 shows some examples from COIL20 and Caltech101-7. Finally, the details of datasets are discussed as follow:

COIL-20: this dataset is obtained from the Columbia object image library, which includes 1440 images with 20 objects. Furthermore, we extract three different features(Intensity, LBP, Gabor) from it.

HandWritten: this is a handwritten images dataset of '0-9' digits with 2000 images. Each digit has 1000 images. Followed by previous work [51], we use six features as different views to describe it.

MNIST: are a well-known handwritten dataset about '0-9' digits, each class has 1000 samples, respectively.

Caltech101-7: it is an object recognition dataset including 1474 images that are roughly divided into seven categories by following previous work [52]. Six types of features are extracted: Gabor, Wavelet Moments (WM), CENTRIST feature, HOG feature, GIST feature, and LBP feature.

5.2.2. Metrics

We select seven metrics to evaluate the clustering performance consisting Accuracy(ACC), Normalized Mutual Information(NMI), Purity, $\mathrm{F}_{\mathrm{score}}$, Precision(P), Recall(R) and Adjusted Rand Index(ARI). They can measure the consistency between predicted labels and ground truth from different views. For all metrics, the larger the value, the better performance.

ACC denotes the proportion of clustering samples take over the whole samples which compute as

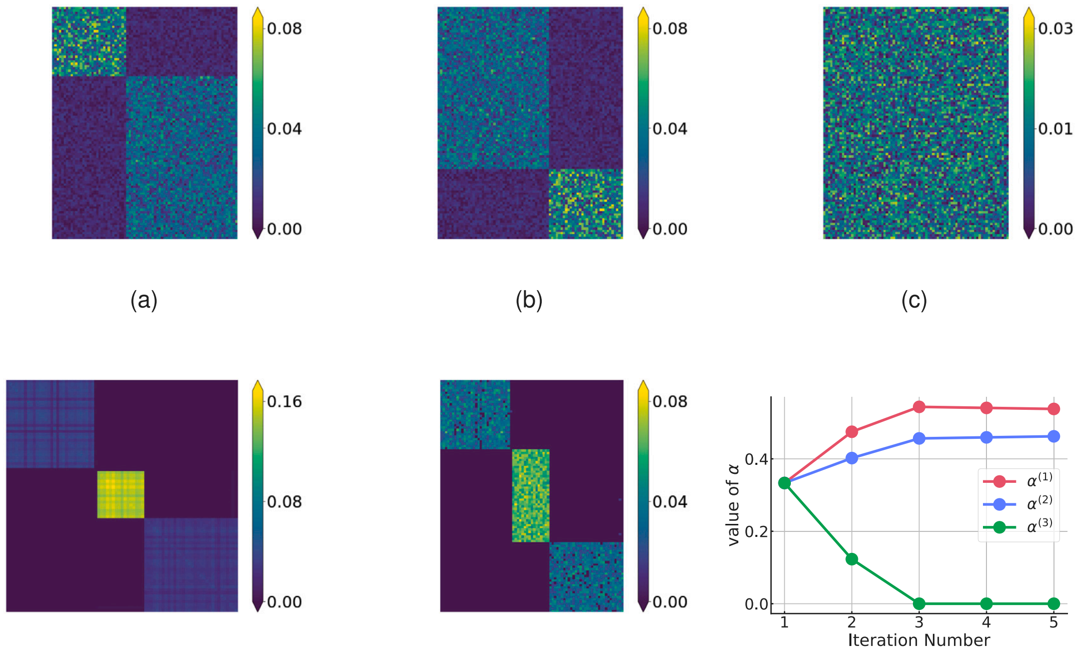
$$\mathrm{ACC} = \frac{1}{n}\sum_{i=1}^{n}\delta(y_i, \mathrm{map}(\hat{y}_i)), \tag{33}$$

where y_i and $\hat{y}_i$ are ground truth and predicted labels, map$(\cdot)$ is a function [53] that mapping the cluster labels to their best real labels.

NMI is used to measure coherence between two distributions whose definition is follow

$$\mathrm{NMI}(Y,\hat{Y}) = \mathrm{MI}(Y,\hat{Y})/\sqrt{H(Y)H(\hat{Y})}, \tag{34}$$

where $H(Y), H(Y)H(\hat{Y})$ is entropy's of distribution Y and $\hat{Y}$, MI measures the coherence between them.



(a) (b) (c)

(d) (e) (f)

Fig. 3. Experiments on toy block diagonal graph. (a), (b), (c) are respectively view-specific bipartite as input. (d) is pairwise graph whose connected components is equal to (e) and (e) is final output optimal bipartite. (f) is value of view-specific weights w.r.t. number of iterations.

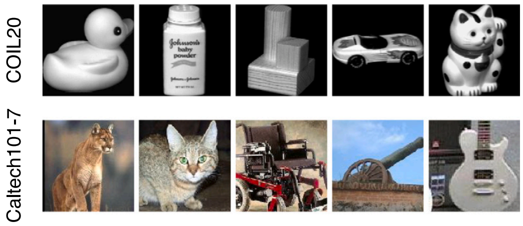


Fig. 4. Some example objects and images from COIL20 and Caltech101-7 datasets respectively.

Purity is a direct and the most intuitive way to evaluate performance. It measures the proportion that samples are assigned to the most frequent class labels within each class. Defined as

$$\text{Purity}(Y,\hat{Y}) = \frac{1}{n}\sum_{k}\max_{i}\left|Y_i \cap \hat{Y}_k\right|, \tag{35}$$

where Y and $\hat{Y}$, respectively, are ground truth and learned labels.

Precision (P) and **Recall (R)**: Suppose that true positive (TP), false positive (FP), and false negative (FN), respectively, precision and recall in above equation are defined as

$$\text{precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{36}$$

and

$$\text{recall} = \frac{\text{TP}}{\text{FP}+\text{FN}} \tag{37}$$

F_{score}**(F)** is a trade-off between precision and recall and computed as

$$\text{F}_{\text{score}} = 2\times\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}. \tag{38}$$

Adjusted Rand Index (ARI) is an improved version of the Rand Index(RI). The purpose is to remove the influence of random labels on the evaluation results of the rand coefficient, whose calculation process is similar to the calculation process of the accuracy rate. It can be computed by

$$\text{ARI} = \frac{\sum_i\sum_j C^2_{n_{ij}} - \left(\sum_i C_{n_i^\tau}\cdot\sum_i C_{n_i^r}\right)/C_n^2}{\frac{1}{2}\left(\sum_i C_{n_i^\tau}+\sum_i C_{n_i^r}\right) - \left(\sum_i C_{n_i^\tau}\cdot\sum_i C_{n_i^r}\right)/C_n^2}, \tag{39}$$

where n_i^τ denotes the number of ground truth in ith cluster and n_i^r is corresponding number of learned labels. C_n^m is combination operation.

5.2.3. Comparison methods

Here, we briefly revisit the recent popular multi-view clustering methods.

AASC [54] (Affinity Aggregation Spectral Clustering): it uses affinity aggregation on spectral clustering for multi-view context, which can ignore more ineffective affinities and irrelevant features than others.

AMGL [55] (Auto-Weighted Multiple Graph Learning): It can learn an optimal graph without additive parameters and extend to a semi-supervised version.

CDMGC [56] (Consistent and Divergent Multi-view Graph Clustering): it divides the original graph into consistency and complementary parts and learns the optimal structured graph by using rank constraint.

CGD [57] (Cross-view Graph Diffusion): it is the first attempt to employ diffusion process for multi-view clustering. Finally, it uses spectral clustering on the learned fusion graph to get results.

Co-trainSC [31] (Co-training for multi-view Spectral Clustering): it iteratively learns multiple clustering results and forces clustering results to be consistent between each other.

MCONGR [12] (Multi-view Clustering via Orthogonal and Nonnegative Graph Reconstruction): it learns a joint graph across multiple views and gets the final answer by using nonnegative constraint manipulates. In detail, there exists two different model in this paper, we denote them as **MCONGR_1** and **MCONGR_2** respectively.

SFMC [58] (Scalable and parameter-free Multiview graph Clustering): an efficient model for scalable data clustering, in which the fusion graph is learned without any hyper-parameter tunning.

SwMC [59] (Self-weighted Multiview Clustering): it is totally self-weighted clustering method for multi-view based on CLR. More importantly, once the optimal graph is obtained in our models, it can

Table 3
Clustering performance of the proposed method comparing with state-of-the-art methods on COIL20 and handwritten (%).

Methods	COIL 20							HandWritten						
	ACC	NMI	Purity	F	P	R	ARI	ACC	NMI	Purity	F	P	R	ARI
AASC	83.40	91.25	87.36	77.56	67.34	91.44	76.21	83.30	83.94	83.30	78.80	75.52	82.37	76.34
AMGL	83.40	91.25	87.36	77.56	67.34	91.44	76.21	85.10	88.79	87.65	84.61	78.50	91.75	82.76
CDMGC	85.83	**94.50**	**90.00**	84.37	75.78	95.17	83.46	84.45	89.67	88.20	84.99	79.48	91.32	83.20
CGD	79.17	87.91	82.64	75.28	76.60	70.90	83.30	85.45	88.77	87.90	83.28	**85.07**	79.01	**92.14**
CotrainSC	72.07	81.60	74.30	68.86	66.83	71.21	67.19	81.39	77.57	82.37	74.47	73.18	75.97	71.58
MCONGR_1	79.10	88.17	82.64	74.62	68.22	82.34	73.17	71.60	80.15	76.00	70.76	63.59	79.76	67.12
MCONGR_2	81.60	89.72	85.28	78.40	71.57	86.66	77.16	73.70	82.10	78.10	74.81	68.85	81.90	71.76
SFMC	80.97	91.89	84.93	81.93	**76.80**	87.79	80.92	86.35	87.47	86.35	83.94	78.13	90.69	82.02
SwMC	86.39	94.29	89.86	**84.44**	75.67	**95.51**	**83.53**	88.60	**90.18**	**88.60**	87.41	84.00	91.11	85.96
Ours	**87.22**	92.17	88.19	80.43	69.55	95.34	79.24	**90.44**	89.65	88.25	**87.44**	80.22	**96.08**	85.91
	MNIST							Caltech101-7						
	ACC	NMI	Purity	F	P	R	ARI	ACC	NMI	Purity	F	P	R	ARI
AASC	71.82	68.93	76.12	64.44	59.58	70.15	60.13	66.49	52.91	85.21	62.03	65.31	59.06	40.10
AMGL	73.91	69.72	76.56	65.65	60.86	71.27	61.51	65.81	52.52	85.41	61.41	65.63	57.71	39.65
CDMGC	–	–	–	–	–	–	–	65.20	55.88	85.14	63.63	66.41	61.07	42.33
CGD	55.61	58.85	61.53	43.75	50.38	42.70	61.44	67.64	56.11	**86.43**	54.58	68.19	87.74	55.76
CotrainSC	77.96	66.25	78.63	65.83	64.35	67.45	61.93	73.41	52.50	84.80	68.21	65.19	71.53	46.73
MCONGR_1	74.12	71.90	78.39	67.57	62.71	73.26	63.66	65.88	52.47	85.75	61.39	66.24	57.20	39.93
MCONGR_2	79.91	70.86	79.91	67.84	65.32	70.57	64.12	73.70	**82.10**	78.10	74.81	**68.85**	81.90	**71.76**
SFMC	78.03	71.90	78.05	68.21	60.25	78.59	64.16	74.15	54.20	85.21	69.36	66.19	72.85	48.60
SwMC	68.55	66.69	68.66	61.94	50.84	79.24	56.66	64.99	51.14	85.96	60.55	66.91	55.29	39.42
Ours	**81.01**	**75.28**	**81.01**	**74.09**	**68.40**	**80.80**	**70.94**	**81.68**	48.28	83.72	**77.66**	66.02	**94.28**	59.13

Table 4
Running time and complexity on different datasets.

	COIL20	HandWritten	MNIST	Caltech101-7	Complexity
AASC	68.74	39.69	1342.13	11.04	$O(n^3)$
AMGL	123.16	142.69	3874.35	7.60	$O(n^3)$
CDMGC	11.57	9.41	–	14.28	$O(n^2)$
CGD	201.13	382.82	6809.16	244.10	$O(n^3t)$
CotrainSC	211.82	432.18	9874.62	79.27	$O(n^2k)$
MCONGR_1	30.34	65.85	1629.45	37.27	$O(Vn^2d+t(n^2c+nc^2))$
MCONGR_2	21.45	23.65	597.168	17.47	$O(Vn^2d+t(n^2c+nc^2))$
SFMC	40.73	86.42	5362.79	54.67	$O(nmd+nm^2t)$
SwMC	247.83	96.76	7383.36	48.37	$O(n^3)$
Ours	5.17	55.03	2618.11	14.83	$O(nt+nmV^2+m^3)$

directly assign the cluster label to each data point and does not need any post-processing.

5.3. Experimental results

5.3.1. Comparing with multi-view clustering

In order to show the outstanding clustering performance of the proposed method, we compare it to nine state-of-the-art multi-view clustering methods.

Table 3 demonstrates the superiority of the proposed method. The top two results are indicated in bold and underlined.

Our method demonstrates superior performance compared to state-of-the-art methods in most cases. It indicates our ability to effectively capture the common information between views while preserving view-specific characteristics. Based on the above-summarized tables, we have made the following observations: particularly for Caltech101-7 with ACC, our proposed method achieves a significant improvement of 7.73% over the second-best method, thus validating its strong capability in multi-view clustering. Despite AMGL and SwMC being non-parametric multi-view clustering methods, our method outperforms them significantly. One possible reason is that it employs an implicit weighting strategy that inadequately normalizes the weights of each perspective, focusing only on perspectives with large weights while disregarding the smaller ones. Furthermore, although the SFMC shares similarities with the algorithm proposed in this paper, it lacks theoretical guarantees for convergence.

Additionally, we study the running time of the our proposed method. Table 4 presents the average CPU running time from data acquisition to final result output. As evident from the table, our method outperforms others in terms of performance and exhibits faster execution compared to related structured graph learning methods, such as SwMC, SFMC, CDMGC, etc. It highlights the effectiveness of the anchor graph approach. Theoretically, our method exhibits a linear scale with the dataset size, resulting in reduced execution time for larger datasets. At the same time, while both AMGL and SwMC involve the decomposition of $n \times n$ eigenvalues during the model optimization process, our proposed method only performs eigenvalue decomposition on $m \times m$ ($m \ll n$) graphs. It is one of the reasons for the efficiency of our method. In conclusion, our proposed method achieves a favorable balance between performance and efficiency.

5.3.2. Convergence study

Fig. 5 illustrates the convergence curve obtained by solving Eq. (5) on four datasets. All curves exhibit that algorithm will converge less than 20 iterations across all experimental datasets. It confirms the previous theoretical analysis, and also illustrates the efficiency of the algorithm from another perspective.

5.3.3. Effect of anchors

Here, we analyze the effect of the number of anchors on performance. As we all know, randomly selecting anchors is a general practice, but it does not yield favorable results. Thus, we employ the k-means to uniformly select initial anchors from the dataset. We vary the number of anchors from 0.1 to 1 in increments of 0.1, proportionate to the total number of data points, and record corresponding ACC and running time in Fig. 6. The results indicate that both the clustering

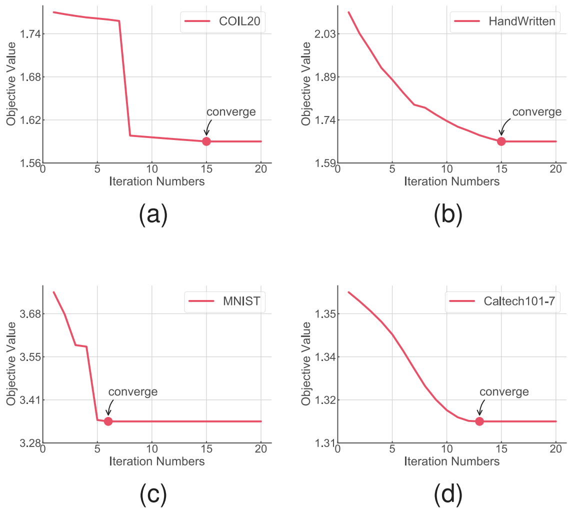


Fig. 5. The convergence curves of the objective values on four datasets. (a) COIL20. (b) HandWritten. (c) MNIST. (d) Caltech101-7.

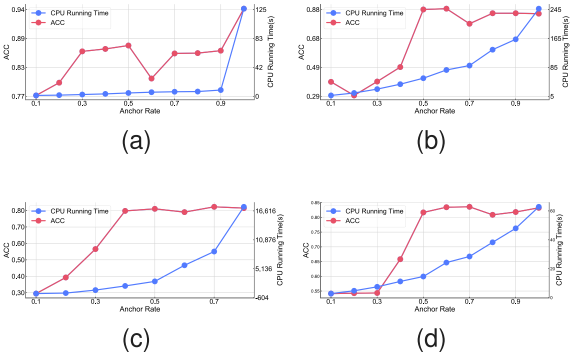


Fig. 6. The clustering performance(ACC) and CPU running time of proposed algorithm with variant number for anchors. (a) COIL20. (b) HandWritten. (c) MNIST. (d) Caltech101-7.

Table 5
Clustering performance of the proposed method comparing with single-view clustering method (%).

Methods	COIL20				MNIST			
	ACC	NMI	F	P	ACC	NMI	F	P
view1	77.76	87.98	71.46	60.58	60.13	62.75	57.49	46.22
view2	81.62	**92.51**	**80.74**	**71.21**	75.00	65.80	62.70	55.67
view3	79.85	91.68	80.09	70.84	59.17	58.98	53.90	40.57
AVG	79.65	91.30	79.55	68.88	57.44	57.24	52.04	37.55
Ours	**87.22**	92.17	80.43	69.55	**81.01**	**75.28**	**74.09**	**68.40**
Methods	HandWritten				Caltech101-7			
	ACC	NMI	F	P	ACC	NMI	F	P
view1	50.62	54.39	48.33	35.08	45.32	12.97	49.36	38.98
view2	49.24	52.96	47.73	34.60	52.31	2.80	54.68	38.59
view3	69.80	74.36	68.98	56.64	45.12	13.55	46.29	38.12
view2	76.62	80.15	75.31	64.40	69.40	**56.59**	65.74	**72.49**
view3	37.05	33.99	32.40	21.09	58.82	18.24	60.10	43.31
view4	39.41	41.94	35.66	27.28	64.04	52.57	62.50	64.17
AVG	68.80	75.79	69.11	53.76	67.64	55.91	66.91	65.70
Ours	**88.20**	**89.65**	**87.44**	**80.22**	**81.68**	48.28	**77.66**	66.02

performance and running time increase with an increasing number of anchors. It should be observed that the performance does not have any significant improvement until the anchor rate exceeds 0.5. However, the time consumption still continues to increase sharply. Thus, a good trade-off between performance and efficiency is achieved when the anchor rate is 0.5. In this case, we only need to build a bipartite graph half the size of the original image to get good performance. It shows that the anchor strategy is effective, which could reduce the complexity of the algorithm while maintaining the accuracy, so that the algorithm can adapt to a larger data set. To this end, we set the number of anchors to half the dataset size in the subsequent experiments.

5.3.4. *Effect of parameter-free graph fusion strategy*

To prove the superiority of the fusion strategy, we discuss it from two perspectives: *(a)* multi-view data obtains more information than single-view one, and the final result should be better inherently. *(b)* The parameter-free graph fusion strategy can better select and capture good features.

In order to illustrate (a), to be fair, we set $V = 1$ in Eq. (5), so that the proposed algorithm will degenerate into a single-view version. Then we perform single-view clustering on each bipartite graph separately. To demonstrate (b), we have designed the following ablation experiment. We average each single-view graph and then applying graph cuts to obtain cluster labels, which is denoted as AVG in Table 5. To mitigate the impact of randomness, we repeat each algorithm ten times and calculate the average results. It is evident that the proposed method outperforms all the single-view methods, which directly demonstrates the superiority of our multi-view clustering approach compared to single-view ones. It not only considers the complementary information among multi-view data but also captures the unique information within each view.

Meanwhile, our proposed method exhibits superior performance compared to the average fusion strategy, particularly on two handwriting datasets: MNIST and HandWritten. There are several potential reasons for this outcome. Firstly, different feature extraction algorithms yield view-specific features, which are then used to construct a various graph. Secondly, some view-specific graphs may have poor separability. The strategy of averaging fusion simply amplifies unnecessary connections, increasing in inter-cluster connections. Conversely, the algorithm employing parameter-free fusion strategy effectively addresses this issue.

6. Conclusion

In this paper, we propose a parameter-free and time-efficient method for multi-view clustering. Compared with most existing view-weighted multi-view clustering methods, our method mainly has the following three advantages: *(a)* parameter-free fusion graph learning method; *(b)* suitable for large datasets with friendly time complexity; *(c)* get the cluster labels directly from the graph structure to achieve end-to-end. Especially in the optimization process, we impose rank constraints on the equivalent pairwise graph with the same connected components of the bipartite graph, making the whole method converge to global optimal with strict theoretical proof. Extensive experiments are conducted on four datasets to show the effectiveness and efficiency of the proposed method.

In the future, we will investigate more complex multi-view clustering problems, such as incomplete multi-view clustering and partially view-aligned multi-view clustering. Due to the strong feature extraction capabilities of deep learning, we will focus on deep unsupervised and semi-supervised learning. Moreover, as an extension of multi-view analysis, we will also explore various studies based on multi-modal data.

CRediT authorship contribution statement

Yu Duan: Writing – original draft, Visualization, Methodology, Conceptualization. **Danyang Wu:** Writing – review & editing, Validation, Supervision, Data curation, Conceptualization. **Rong Wang:** Writing – review & editing, Supervision. **Xuelong Li:** Writing – review & editing, Supervision, Funding acquisition. **Feiping Nie:** Validation, Funding acquisition, Conceptualization.

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 supported in part by the National Key Research and Development Program of China under Grant 2018AAA0101902, in part by the Natural Science Basic Research Program of Shaanxi (Program No. 2021JM-071), in part by the National Natural Science Foundation of China under Grant 62176212, Grant 61936014 and Grant 61772427, and in part by the Fundamental Research Funds for the Central Universities under Grant G2019KY0501.

References

[1] Z. Xu, W. Pan, Z. Ming, A multi-view graph contrastive learning framework for cross-domain sequential recommendation, in: RecSys, ACM, 2023, pp. 491–501.

[2] J. Chen, Z. Wang, C. Zheng, K. Zeng, Q. Zou, L. Cui, GaitAMR: Cross-view gait recognition via aggregated multi-feature representation, Inform. Sci. 636 (2023) 118920.

[3] Z. Rao, M. He, Z. Zhu, Y. Dai, R. He, Bidirectional guided attention network for 3-D semantic detection of remote sensing images, IEEE Trans. Geosci. Remote Sens. 59 (7) (2020) 6138–6153.

[4] Q. Ye, P. Huang, Z. Zhang, Y. Zheng, L. Fu, W. Yang, Multiview learning with robust double-sided twin SVM, IEEE Trans. Cybern. 52 (12) (2022) 12745–12758.

[5] M. Xie, Z. Han, C. Zhang, Y. Bai, Q. Hu, Exploring and exploiting uncertainty for incomplete multi-view classification, in: CVPR, IEEE, 2023, pp. 19873–19882.

[6] W. Liu, Y. Chen, X. Yue, C. Zhang, S. Xie, Safe multi-view deep classification, in: AAAI, AAAI Press, 2023, pp. 8870–8878.

[7] R. Chen, Y. Tang, W. Zhang, W. Feng, Deep multi-view semi-supervised clustering with sample pairwise constraints, Neurocomputing 500 (2022) 832–845.

[8] Z. Li, C. Tang, X. Liu, X. Zheng, W. Zhang, E. Zhu, Consensus graph learning for multi-view clustering, IEEE Trans. Multimed. 24 (2021) 2461–2472.

[9] D. Wu, J. Xu, X. Dong, M. Liao, R. Wang, F. Nie, X. Li, GSPL: A succinct kernel model for group-sparse projections learning of multiview data, in: IJCAI, 2021, pp. 3185–3191, ijcai.org.

[10] S. Hu, Z. Lou, Y. Ye, View-wise versus cluster-wise weight: Which is better for multi-view clustering? IEEE Trans. Image Process. 31 (2021) 58–71.

[11] H. Zhang, M. Gong, Y. Gu, F. Nie, X. Li, Side-constrained graph fusion for semi-supervised multi-view clustering, Neurocomputing (2023) 127102.

[12] S. Shi, F. Nie, R. Wang, X. Li, Multi-view clustering via nonnegative and orthogonal graph reconstruction, IEEE Trans. Neural Netw. Learn. Syst. 34 (1) (2023) 201–214.

[13] X. Cai, D. Huang, G. Zhang, C. Wang, Seeking commonness and inconsistencies: A jointly smoothed approach to multi-view subspace clustering, Inf. Fusion 91 (2023) 364–375.

[14] G.-Y. Zhang, Y.-R. Zhou, C.-D. Wang, D. Huang, X.-Y. He, Joint representation learning for multi-view subspace clustering, Expert Syst. Appl. 166 (2021) 113913.

[15] G.-Y. Zhang, Y.-R. Zhou, X.-Y. He, C.-D. Wang, D. Huang, One-step kernel multi-view subspace clustering, Knowl.-Based Syst. 189 (2020) 105126.

[16] S. Yu, S. Liu, S. Wang, C. Tang, Z. Luo, X. Liu, E. Zhu, Sparse low-rank multi-view subspace clustering with consensus anchors and unified bipartite graph, IEEE Trans. Neural Netw. Learn. Syst. (2023).

[17] C. Tang, K. Sun, C. Tang, X. Zheng, X. Liu, J. Huang, W. Zhang, Multi-view subspace clustering via adaptive graph learning and late fusion alignment, Neural Netw. 165 (2023) 333–343.

[18] S. Luo, C. Zhang, W. Zhang, X. Cao, Consistent and specific multi-view subspace clustering, in: AAAI, AAAI Press, 2018, pp. 3730–3737.

[19] X. Cao, C. Zhang, H. Fu, S. Liu, H. Zhang, Diversity-induced multi-view subspace clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 586–594.

[20] J. Xu, J. Han, F. Nie, X. Li, Re-weighted discriminatively embedded k-means for multi-view clustering, IEEE Trans. Image Process. 26 (6) (2017) 3016–3027.

[21] M. Abavisani, V.M. Patel, Deep multimodal subspace clustering networks, IEEE J. Sel. Top. Sign. Proces. 12 (6) (2018) 1601–1614.

[22] R. Fan, T. Luo, W. Zhuge, S. Qiang, C. Hou, Multi-view subspace learning via bidirectional sparsity, Pattern Recognit. 108 (2020) 107524.

[23] D. Wu, F. Nie, R. Wang, X. Li, Multi-view clustering via mixed embedding approximation, in: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2020, pp. 3977–3981.

[24] J. Li, G. Zhou, Y. Qiu, Y. Wang, Y. Zhang, S. Xie, Deep graph regularized non-negative matrix factorization for multi-view clustering, Neurocomputing 390 (2020) 108–116.

[25] S. Yao, G. Yu, J. Wang, C. Domeniconi, X. Zhang, Multi-view multiple clustering, 2019, arXiv preprint arXiv:1905.05053.

[26] D. Niu, J.G. Dy, M.I. Jordan, Multiple non-redundant spectral clustering views, in: ICML, 2010.

[27] G. Andrew, R. Arora, J. Bilmes, K. Livescu, Deep canonical correlation analysis, in: International Conference on Machine Learning, PMLR, 2013, pp. 1247–1255.

[28] Z. Jiao, C. Xu, Deep multi-view robust representation learning, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2017, pp. 2851–2855.

[29] C. Lu, S. Yan, Z. Lin, Convex sparse spectral clustering: Single-view to multi-view, IEEE Trans. Image Process. 25 (6) (2016) 2833–2843.

[30] L. Guo, L. Chen, X. Lu, C.P. Chen, Membership affinity lasso for fuzzy clustering, IEEE Trans. Fuzzy Syst. 28 (2) (2019) 294–307.

[31] A. Kumar, H. Daumé, A co-training approach for multi-view spectral clustering, in: Proceedings of the 28th International Conference on Machine Learning, ICML-11, 2011, pp. 393–400, Citeseer.

[32] K. Zhan, F. Nie, J. Wang, Y. Yang, Multiview consensus graph clustering, IEEE Trans. Image Process. 28 (3) (2018) 1261–1270.

[33] F. Nie, W. Zhu, X. Li, Unsupervised feature selection with structured graph optimization, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30, (1) 2016.

[34] R. Liu, M. Chen, Q. Wang, X. Li, Robust rank constrained sparse learning: A graph-based method for clustering, in: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2020, pp. 4217–4221.

[35] S. Fang, D. Huang, X. Cai, C. Wang, C. He, Y. Tang, Efficient multi-view clustering via unified and discrete bipartite graph learning, IEEE Trans. Neural Netw. Learn. Syst. (2023).

[36] R. Fan, X. Ouyang, T. Luo, D. Hu, C. Hou, Incomplete multi-view learning under label shift, IEEE Trans. Image Process. 32 (2023) 3702–3716.

[37] Y. Lin, Y. Gou, Z. Liu, B. Li, J. Lv, X. Peng, COMPLETER: Incomplete multi-view clustering via contrastive prediction, in: CVPR, Computer Vision Foundation / IEEE, 2021, pp. 11174–11183.

[38] Z. Huang, P. Hu, J.T. Zhou, J. Lv, X. Peng, Partially view-aligned clustering, in: NeurIPS, 2020.

[39] M. Yang, Y. Li, Z. Huang, Z. Liu, P. Hu, X. Peng, Partially view-aligned representation learning with noise-robust contrastive loss, in: CVPR, Computer Vision Foundation / IEEE, 2021, pp. 1134–1143.

[40] D. Huang, C. Wang, J. Lai, Fast multi-view clustering via ensembles: Towards scalability, superiority, and simplicity, IEEE Trans. Knowl. Data Eng. 35 (11) (2023) 11388–11402.

[41] X.Y. Stella, J. Shi, Multiclass spectral clustering, in: Computer Vision, IEEE International Conference on, Vol. 2, IEEE Computer Society, 2003, p. 313.

[42] F. Nie, X. Wang, M. Jordan, H. Huang, The constrained laplacian rank algorithm for graph-based clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30, (1) 2016.

[43] F. Nie, Z. Hu, X. Li, Calibrated multi-task learning, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 2012–2021.

[44] F. Nie, H. Huang, C.H.Q. Ding, Low-rank matrix recovery via efficient schatten p-norm minimization, in: AAAI, AAAI Press, 2012, pp. 655–661.

[45] X. Li, Q. Lu, Y. Dong, D. Tao, Robust subspace clustering by cauchy loss function, IEEE Trans. Neural Netw. Learn. Syst. 30 (7) (2018) 2067–2078.

[46] F. Nie, X. Wang, C. Deng, H. Huang, Learning a structured optimal bipartite graph for co-clustering, Adv. Neural Inf. Process. Syst. 30 (2017).

[47] F. Nie, C.-L. Wang, X. Li, K-multiple-means: A multiple-means clustering method with specified k clusters, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 959–967.

[48] K. Fan, On a theorem of Weyl concerning eigenvalues of linear transformations I, Proc. Natl. Acad. Sci. USA 35 (11) (1949) 652.

[49] X. Liu, M. Li, C. Tang, J. Xia, J. Xiong, L. Liu, M. Kloft, E. Zhu, Efficient and effective regularized incomplete multi-view clustering, IEEE Trans. Pattern Anal. Mach. Intell. 43 (8) (2020) 2634–2646.

[50] F. Nie, X. Wang, H. Huang, Clustering and projected clustering with adaptive neighbors, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 977–986.

[51] C. Tang, X. Zhu, X. Liu, M. Li, P. Wang, C. Zhang, L. Wang, Learning a joint affinity graph for multiview subspace clustering, IEEE Trans. Multimed. 21 (7) (2018) 1724–1736.

[52] C. Zhang, Y. Liu, Y. Liu, Q. Hu, X. Liu, P. Zhu, FISH-MML: Fisher-HSIC multi-view metric learning, in: IJCAI, 2018, pp. 3054–3060.

[53] H.W. Kuhn, The hungarian method for the assignment problem, Nav. Res. Logist. Q. 2 (1–2) (1955) 83–97.

[54] H.-C. Huang, Y.-Y. Chuang, C.-S. Chen, Affinity aggregation for spectral clustering, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 773–780.

[55] F. Nie, J. Li, X. Li, et al., Parameter-free auto-weighted multiple graph learning: a framework for multiview clustering and semi-supervised classification, in: IJCAI, 2016, pp. 1881–1887.

[56] S. Huang, I.W. Tsang, Z. Xu, J. Lv, Measuring diversity in graph learning: A unified framework for structured multi-view clustering, IEEE Trans. Knowl. Data Eng. 34 (12) (2022) 5869–5883.

[57] C. Tang, X. Liu, X. Zhu, E. Zhu, Z. Luo, L. Wang, W. Gao, CGD: Multi-view clustering via cross-view graph diffusion, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, (04) 2020, pp. 5924–5931.

[58] X. Li, H. Zhang, R. Wang, F. Nie, Multiview clustering: A scalable and parameter-free bipartite graph fusion method, IEEE Trans. Pattern Anal. Mach. Intell. 44 (1) (2020) 330–344.

[59] F. Nie, J. Li, X. Li, et al., Self-weighted multiview clustering with multiple graphs, in: IJCAI, 2017, pp. 2564–2570.

Yu Duan received the MS degree from Northwestern Polytechnical University in 2021. He is currently pursuing a Ph.D. degree with the School of Computer Science, School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University, Xi'an, China. His research interests focus on graph learning.

Danyang Wu received the B.S. degree in Electronics Information Engineering and the Ph.D. degree in Computer Science and Technology from Northwestern Polytechnical University, Xi'an, China, in 2017 and 2022, respectively. He is currently a Full Professor with Northwest A&F University, Shaanxi, China. He has published more than 30 papers in some top-tier journals and conferences, including TPMAI, TSP, TKDE, TNNLS, TMLR, IJCAI, ACM-MM, WWW, ICASSP, etc. He is serving as a reviewer or PC member for some top-tier journals and conferences, including TPAMI, TNNLS, TCSVT, ICLR, ICML, NeurIPS, AAAI, ACM-MM, etc. His research interests include graph machine learning, graph signal processing, and the application in science and agriculture.

Rong Wang received the B.S. degree in information engineering, the M.S. degree in signal and information processing, and the Ph.D. degree in computer science from Xi'an Research Institute of Hi-Tech, Xi'an, China, in 2004, 2007 and 2013, respectively. During 2007 and 2013, he also studied in the Department of Automation, Tsinghua University, Beijing, China for his Ph.D. degree. He is currently an associate professor at the School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University, Xi'an, China. His research interests focus on machine learning and its applications.

Xuelong Li is a full professor with the School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University, Xi'an 710072, P.R. China.

Feiping Nie received the Ph.D. degree in Computer Science from Tsinghua University, China in 2009, and currently is full professor in Northwestern Polytechnical University, China. His research interests are machine learning and its applications, such as pattern recognition, data mining, computer vision, image processing and information retrieval. He has published more than 100 papers in the following journals and conferences: TPAMI, IJCV, TIP, TNNLS, TKDE, ICML, NIPS, KDD, IJCAI, AAAI, ICCV, CVPR, ACM MM. His papers have been cited more than 20000 times and the H-index is 84. He is now serving as Associate Editor or PC member for several prestigious journals and conferences in the related fields.