Multi-view Clustering: A Scalable and Parameter-free Bipartite Graph Fusion Method

Feiping Nie, Han Zhang, Rong Wang, and Xuelong Li, *Fellow, IEEE*

Abstract—Multi-view clustering partitions data into different groups according to their heterogeneous features. Most existing methods degenerate the applicability of models due to their intractable hyper-parameters triggered by various regularization terms. Moreover, traditional spectral based methods always encounter the expensive time overhead and fail in exploring the explicit clusters from given graphs. In this paper, we present a scalable and parameter-free graph fusion framework for multi-view clustering, seeking for a joint graph compatible across multiple views in a self-supervised weighting manner. Our formulation coalesces multiple view-wise graphs straightforward and learns the weights as well as the joint graph interactively, which could actively release the model from any weight-related hyper-parameters. Meanwhile, we manipulate the joint graph by a connectivity constraint such that the connected components indicate clusters directly. The designed algorithm is initialization-independent and time-economical which obtains the stable performance and scales well with the data size. Substantial experiments on toy data as well as real datasets are conducted that verify the superiority of the proposed method compared to the state-of-the-arts over the clustering performance and time expenditure.

Index Terms—Multi-view clustering, scalable and parameter-free, graph fusion, connectivity constraint, initialization-independent

1 INTRODUCTION

RECENT years, an increasing number of data are generated from multiple views in the real scenarios, since information of data frequently come from diverse sources or are represented by different attributes. For instance, a video is comprised of images and sounds; a piece of content can be narrated by several languages; an object can be described from different aspects, etc. Multi-view clustering, that partitions data into different groups according to their heterogeneous features, has become a hot topic in the domain of unsupervised learning, and has been applied in many realistic tasks, such as objection recognition [1], feature selection [2], and face clustering [3], etc.

Based on the principle that information of data from diverse views is complementary yet conveys the accordant underlying clusters, extensive efforts that mine the intrinsic structure across multiple views have been made in these decades [4]–[23]. From the viewpoint of involved techniques, existing literatures are mainly categorized into three types including subspace based methods [4]–[9], matrix factorization methods [10]–[13], and graph based methods [14]–[23]. As pointed out by Kang et al. [9], subspace based multi-view clustering premises the consensus low-dimensional subspace underlying the multi-view feature domains, and matrix factorization based methods explore the common indicator matrix compatible across different views, while graph based clustering premises the consensus affinity matrix across multiple views. Meanwhile, according to the utilized data information, we could roughly summarize them into feature-driven methods and relation-driven methods, where the first three types are so-called feature-driven and the graph based type is relation-driven. To be specific, the core formulations of feature-driven methods directly utilize data features to estimate an explicit model of data distribution. For example, Cao et al. [4] learned the unified subspace representations from data features to capture both the consistency and the diversity among the multiple views. Xu et al. [12] proposed a re-weighted multiple k-means to simultaneously seek the common low-dimensional subspace and cluster indicator from data features. The performance of these feature-driven algorithms are extensively effected by the feature dimension. Different from feature-driven methods, relation-driven approaches are induced by the point-to-point relations of data which are mostly preserved in graphs, and various optimization strategies are invoked on the graphs to partition the point cloud. Owing to the powerful capacity of capturing the nonlinear structure of data, graph based methods are in hot pursuit by recent researches. The initial graph based methods are a group of spectral clustering researches, which explore the manifold representations of data embedded in given graphs by eigen-decomposition of the Laplacian matrix. Based on this, numerous multi-view spectral clustering methods are raised [15]–[21], [24]. Zhou et al. [15] stated that spectral clustering was the real-valued relaxation of combinatorial normalized cut, based on which they extended the normalized cut (Ncut) [25] from single view to multiple views, having the better interpretability of combining the graph Laplacian. Kumar et al. [16] developed the multi-view spectral clustering and explored the true underlying clustering

- *F. Nie and H. Zhang are with School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, Shaanxi, P. R. China, 710072. E-mail: {feipingnie; zhanghan9937}@gmail.com.*
- *R. Wang is with School of Cybersecurity and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, Shaanxi, P. R. China, 710072. E-mail: wangrong07@tsinghua.org.cn.*
- *X. Li is with the School of Computer Science and with the Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, Shaanxi, P. R. China, 710072. E-mail: li@nwpu.edu.cn.*

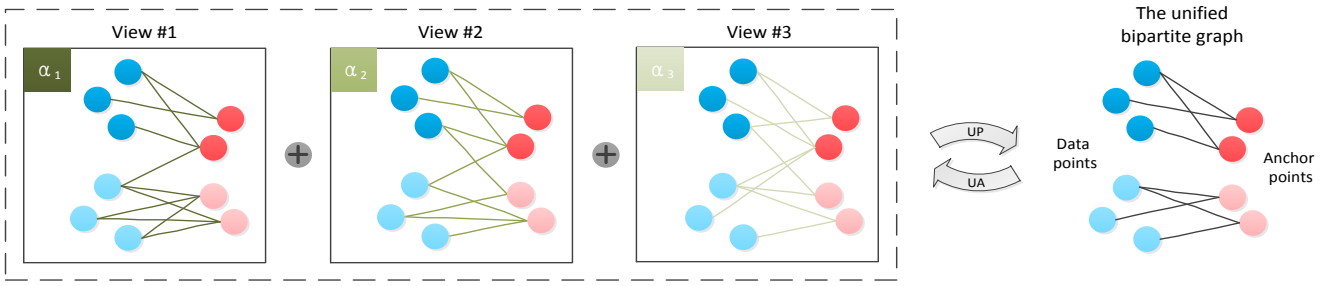


Fig. 1. Illustration of the joint structured optimal graph across three single-view bipartite graphs. Each single-view is assigned with the weight at the left-up corner, and the link parameters in each single graph are multiplied by its weight. When a rank constraint is imposed on the Laplacian matrix of the joint bipartite graph, three weighted single-view graphs are integrated to be structured optimal with exactly c-connected components ($c = 2$ here) among data points and anchor points. UP represents the process of updating the joint graph, while UA is to update weight factors.

by co-regularizing multiple spectral analysis. Hu et al. [21] presented a unified framework that integrates the nonnegative embedding and spectral embedding to enhance the quality of the traditional spectral analysis. Kang et al. [24] adaptively combined multiple self-expressiveness graphs and incorporated the fused graph into spectral clustering for structured graph learning.

Unfortunately, most of existing graph-driven methods encounter hyper-parameters, since they access to the underlying clustering across multiple view-wise graphs via the weighting or regularization manners. More specifically, these researches either employ the weight factors to discriminate different views, or utilize the regularization terms to balance these views, and thus the hyper-parameters related to weights or regularizations are inevitable. For instance, Li et al. [26] involved a hyper-parameter to control the distribution of view-wise weights. Liu et al. [19] used two parameters to balance multiple regularization terms in the model. Feng et al. [27] had two parameters as trade-off in the formulation and also had view-wise parameters to weight different views manually. Zhu et al. [20] utilized a parameter-involved regularization to prevent the weights of views from the trivial solution that only the best view obtained the valid weight. Considering these hyper-parameters do not have practical interpretations, how to tactfully determine them is a severe challenge. A common way is to search the parameters in a wide range for each dataset, intensively degrading the efficiency and the application of models however. Apart from parameter-tuning, the high time-consumption is a long-term trouble of spectral based methods. As analyzed in [28], the expensive graph computation over a regular affinity matrix makes it unbearable for large-scale data. Therefore, performing the multi-view graph clustering with a simple yet high efficient algorithm is the goal of our research.

In this paper, we simultaneously deal with the three issues of multi-view spectral clustering, i.e., **intractable parameter-tuning**, **expensive time-consuming** and **indirect cluster-obtaining**. We address the multi-view graph clustering via a scalable and parameter-free graph fusion scheme, seeking for a structured optimal joint graph that compatibly crosses multiple views. The illustration of our framework is shown in Fig. 1. Instead of regularizing or weighting the view-wise losses conventionally, our formulation coalesces the view-wise graphs straightforward to form the joint graph in a self-supervised framework. The joint graph and the view-wise weights learn from each other interactively and supervise each other adaptively, which actively releases the model from the weight-related hyper-parameters. Meanwhile, we manipulate the joint graph by a connectivity constraint, such that the connected components indicate clusters directly. In order to enhance the efficiency, we apply the bipartite graph learning in the proposed framework, in which the relations of a small size of anchor points and the data points are used to cover the affinity of the entire point cloud, significantly reducing the main computational complexity from $\mathcal{O}(n^2d)$ to $\mathcal{O}(nmd + nm^2)$ where n is the data size, m ($c \ll m \ll n$) is the anchor size, d is the dimension of data, and c is the cluster number. Accordingly, we design a simple and initialization-independent anchor-selection strategy which greatly improve the stableness of the proposed algorithm. Substantial experiments validate the superiority of the proposed method in terms of both clustering performance and running time. In general, the main contributions of this paper are threefold:

- We present a multi-view graph fusion framework that exploits the joint graph across multiple views via a self-supervised weighting manners without extra hyper-parameters.
- We design a scalable and parameter-free multi-view graph clustering algorithm based on the presented framework equipped with the structured optimal graph learning scheme.
- We provide a simple and initialization-independent anchor-selection strategy, which makes the clustering performance of the anchor-based algorithms much competitive and quite stable.

Notations. Throughout the paper, the trace of matrix $\mathbf{M}$ is written as $Tr(\mathbf{M})$; The ℓ_2-norm of vector $\mathbf{v}$ is expressed as $\|\mathbf{v}\|_2$; The Frobenius norm of matrix $\mathbf{M}$ is denoted by $\|\mathbf{M}\|_{\mathrm{F}}$; $\mathbf{M}^T$ is the transpose of matrix $\mathbf{M}$; and $\mathrm{rank}(\mathbf{M})$ denotes the rank of matrix $\mathbf{M}$. In addition, $\mathbf{M} \geq \mathbf{0}$ or $\mathbf{v} \geq \mathbf{0}$ indicates that each element in $\mathbf{M}$ or $\mathbf{v}$ is equal to or greater than zero.

2 BACKGROUND

In this section, we provide an overview to the single-view and multi-view clustering respectively, and then the previous works related to our research are introduced.

2.1 SINGLE-VIEW CLUSTERING

Graph clustering is initially put forward towards single-view data, such as [25], [29], [30]. Considering that the partial motivation of our research is inspired by a previous single-view research, we first formulate it for the better understanding of the following content. As is known, many prevalent graph clustering methods require a post-process to get the final labels since their graphs fail to achieve the cluster structure with exact cluster number. Nie et al. [31] proposed a constrained Laplacian rank method (CLR) from the perspective of direct graph clustering, formulated as follows:

$$\min_{\mathbf{S}^T\mathbf{1}=\mathbf{1},\mathbf{S}\geq\mathbf{0},\mathrm{rank}(\mathbf{L}_S)=n-c} \|\mathbf{S}-\mathbf{A}\|_{\mathrm{F}}^2, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$ is a given affinity matrix among n data points, $\mathbf{L}_S$ is the defined Laplacian matrix of $\mathbf{S}$. The goal of this model is to obtain a non-negative and normalized graph $\mathbf{S}$ that approximates to $\mathbf{A}$, and a key of this model is the rank of Laplacian matrix of $\mathbf{S}$ is constrained to be $n-c$, where c is the desired cluster number. According to Lemma 1, this constraint compels the achieved graph $\mathbf{S}$ having exact c clusters.

Lemma 1. *The multiplicity of the eigenvalue zero of the Laplacian matrix $\mathbf{L}_S$ equals to the number of connected components in the graph associated with $\mathbf{S}$.*

2.2 MULTI-VIEW CLUSTERING

We briefly revisit the popular works of multi-view spectral clustering in this part. Inheriting from the single-view spectral clustering [32]–[34], there are several forms of multi-view spectral clustering [16], [19], [20], [26], [27], whose main body could be concisely represented as

$$\begin{aligned} &\min_{\mathbf{F},\mathbf{w}} \sum_{v=1}^{n_V} w_v Tr(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}) + \gamma reg(\mathbf{w}), \\ &s.t.\ \ \mathbf{F}^T\mathbf{F}=\mathbf{I}, \mathbf{w}^T\mathbf{1}=1, \mathbf{w}\geq\mathbf{0}, \end{aligned} \tag{2}$$

where $\mathbf{F} \in \mathbb{R}^{n\times c}$ is the graph embedding. $\mathbf{L}^{(v)}$ is the graph Laplacian for v-th view. $\mathbf{w} \in \mathbb{R}^{n_V}$ is the weight factor for multiple views. $reg(\cdot)$ denotes the regularization functions, and γ is the regularization parameter. Because the weight factor is imposed on the view-wise loss objectives, i.e., $Tr(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F})$, the regularization on $\mathbf{w}$ is inevitable for smoothing the weights and avoiding the trivial solution where only the best view are assigned with 1 and others are assigned with 0. For this reason, the weighting based multi-view spectral clustering generally has one hyper-parameter at least, which needs to be manually tuned in the experiments.

2.3 RELATED WORKS

Initially, Kumar et al. proposed a co-regularized multi-view spectral clustering [16] (Co-reg) and a co-training approach for multi-view spectral clustering [35] (Co-train), which stands for the baselines in the domain of multi-view clustering. Considering the issue of parameter-tuning, there are several researches to address it. A recent work is a parameter-free auto-weighted graph learning (AMGL) method [22], which improves the weighted multi-view spectral clustering and definitely computes the weights according to the normalized cuts within each view. However, AMGL encounters the deficiencies of spectral clustering, such as requiring the two-step processing for clustering. Therefore, another research of multi-view learning with adaptive neighbors (MLAN) [23] is put forward that learns the graph with the desired cluster structure straightforward. However, this method still drops in two hyper-parameters to be determined. Nie et al. [36] (SwMC) provided an indirect strategy that employed the non-squared norm to compute the view-wise loss values instead of explicitly balancing different views. Nevertheless, this method is quite time-consuming, since an $n \times n$ (n is the data size) target graph is optimized iteratively, and the optimized graph satisfies a Laplacian rank constraint in each iteration. Towards a large-scale dataset, this rigid algorithm is expensive in time inevitably. Li et al. [26] (MVSC) proposed a large-scale multi-view spectral clustering that utilizes local manifold fusion to integrate heterogeneous features. Xu et al. [12] proposed a re-weighted multiple k-means (RDEKM) method which scales well with the data size but scales badly with the feature dimensions. Hu et al. [21] proposed an efficient framework (SMSC) that integrates the nonnegative embedding and spectral embedding on pre-defined single-view graphs to enhance the quality of the traditional spectral analysis. From the perspective of binary representation, Zheng et al. [28] (BMVC) proposed a binary multi-view clustering where each point is retold by a length-fixed binary coding previously. The time-consumption is reduced since handling binary data is much easier than tackling common data. Unfortunately, the limited representation of binary coding greatly degrades the clustering ability of models, making this method possess much lower clustering performance than others. Besides that, some existing models like [37] (MVGL) even neglect the fact that the variables of weight factor should be non-negative. Simply put, a non-negative and normalized weight vector supports the principle that the view assigned with larger weight indicates more confidence in clustering. With the ill-defined weights, how to distinguish the impact of two views with weights of 0.3 and -0.3 is a puzzle.

3 MULTI-VIEW GRAPH CLUSTERING

As analyzed above, existing multi-view spectral clustering easily drops in three deficiencies, i.e., intractable parameter-tuning, high time-consuming and poor cluster-structure. In this section, we elaborate how to effectively overcome these problems. To our best knowledge, most graph-driven methods are high time-consuming, due to the computation of the $n \times n$ graph where every pairwise relations of data points are considered. Instead of it, the bipartite graph is frequently used for scalable graph learning [38]–[40], which just depicts the relations of data w.r.t. their m anchor points via an $n \times m$ affinity matrix. Likewise, we will take advantages of the bipartite graph, and we start this section by giving the definition of multi-view bipartite graph. Then, how to build and solve the model follows.

Definition 1. *Given a dataset $\mathcal{X} = \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \cdots, \mathbf{X}^{(n_V)}\}$ consisting of n samples from n_V views in c clusters, where*

$\mathbf{X}^{(v)} \in \mathbb{R}^{d_v \times n}$ denotes the feature matrix in v-th view. The multi-view bipartite graph is defined as $\mathcal{B} = (\mathcal{X}, \mathcal{Y}, \{\mathbf{B}^{(v)}\}_{v=1}^{n_V})$, where $\mathcal{Y} = \{\mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \cdots, \mathbf{Y}^{(n_V)}\}$ is the anchor point set of $\mathcal{X}$ and $\mathbf{Y}^{(v)} \in \mathbb{R}^{d_v \times m}$ represents the features of m anchor points in the v-th view; $\mathbf{B}^{(v)} \in \mathbb{R}^{n \times m}$ is the affinity matrix in the v-th view.

3.1 Formulation of Problem

It is known that the popular spectral clustering utilizes given graphs to seek for the graph embeddings of data under the specific constraints. However, the pre-defined graphs built upon the Euclidean distance of data lacks high confidence and clear cluster-structure, leading to the concurrent shortages for the graph embedding spontaneously. From the perspectives of connectivity, the graph clustering [31] aims to acquire a c-connected approximation (where c is the cluster number) of the given graph to directly indicate the clusters of data. With multi-view graphical representations of data, we devote to Multi-view Graph Clustering (MGC), which possesses three merits: *a)* scaling well with data size for efficiently handling large-scale data; *b)* performing multi-view clustering without tuning parameters manually; *c)* obtaining cluster labels without running extra processing. To achieve them, we put forward **a graph fusion framework**, formulated as

$$\min_{\boldsymbol{\alpha}^T\mathbf{1}=1, \boldsymbol{\alpha}\geq\mathbf{0}, \mathcal{F}\in\Omega} loss(\sum_{v=1}^{n_V} \alpha^{(v)} \mathcal{S}^{(v)}, \mathcal{F}), \tag{3}$$

where $\mathcal{S}^{(v)}$ is the v-th single-view graph, and $\mathcal{F}$ denotes the fused joint graph compatible across multiple views. $\boldsymbol{\alpha} \in \mathbb{R}^{n_V}$ is the weight vector to balance the graphs of different views. Ω is the sub-domain of structured optimal $\mathcal{H}$ which satisfies the specific conditions we will clarify later. $loss(\cdot, \cdot)$ measures the discrepancies between two entries. Different from multi-view spectral clustering, the proposed framework coalesces the weighted single-view graphs, and the weights are learned along with the joint graph interactively. In this way, $\alpha^{(v)}$ depends on the discrepancies between $\mathcal{S}^{(v)}$ and $\mathcal{F}$ straightforward, by which none of single-view graph can outperform others definitely. This self-adaptive fusion framework avoids the trivial solution of $\boldsymbol{\alpha}$ that only the best view has a significant weight, and thus releases the model from hyper-parameters.

Next, we materialize the framework for clustering from three aspects:

- **Scalability.** Invoked by the bipartite graph learning, the affinity of data can be efficiently depicted by an $n \times m$ similarity matrix between n samples and their m anchor points when the anchor points well cover the entire point cloud. Considering the graph clustering needs to compute the joint graph, the time cost of our algorithm can be dramatically reduced if $m \ll n$. So, we utilize the sparse single-view bipartite graphs as defined in Definition 1. The construction of these graphs will be detailed later, and we will also provide a simple yet efficient anchor selection scheme specifically for our algorithm;
- **Connectivity.** To directly obtain c clusters from the joint graph, an intuitive way is to impose a c-connectivity constraint on $\mathcal{F}$ such that each connected component indicates a cluster precisely. Invoked by an important knowledge provided in Lemma 1, this connectivity can be achieved by a Laplacian rank constraint;
- **Robustness.** Without loss of generality, the errors are computed by subtracting the joint similarities from the accumulated similarities forthright. Then, it is common to employ the squared F-norm as the metric of loss to guarantee the robustness, which exaggerates the loss values of entries who have large errors and optimizing them more severely.

Consequently, **a scalable and parameter-free model for multi-view graph clustering** is achieved by materializing the framework (3) as below:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}^T\mathbf{1}=1, \boldsymbol{\alpha}\geq\mathbf{0}, \mathbf{P}} \left\| \sum_{v=1}^{n_V} \alpha^{(v)} \mathbf{B}^{(v)} - \mathbf{P} \right\|_{\mathrm{F}}^2, \\ &s.t.\ \ \mathbf{P}\mathbf{1} = \mathbf{1}, \mathbf{P} \geq \mathbf{0}, \mathrm{rank}(\widetilde{\mathbf{L}}_S) = n + m - c, \end{aligned} \tag{4}$$

where $\mathbf{P} \in \mathbb{R}^{n \times m}$ is the joint affinity matrix. $\widetilde{\mathbf{L}}_S = \mathbf{I} - \mathbf{D}_S^{-\frac{1}{2}} \mathbf{S} \mathbf{D}_S^{-\frac{1}{2}}$ is the normalized Laplacian matrix of $\mathbf{S}$, and $\mathbf{S}$ is the augmented graph of $\mathbf{P}$ defined as

$$\mathbf{S} = \begin{bmatrix} & \mathbf{P} \\ \mathbf{P}^T & \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}. \tag{5}$$

Noting that the principle diagonal blocks are padded with zeros. The degree matrix $\mathbf{D}_S$ is a diagonal matrix whose i-th diagonal element is computed by $\mathbf{D}_S(i,i) = \sum_{j=1}^{n+m} s_{ij}$. This Laplacian rank constraint actually ensures $\mathbf{S}$ to be c-connected according to Lemma 2. Noting that the proof of Lemma 2 is equivalent to Lemma 1 because the rank of normalized Laplacian matrix equals to its unnormalized form. Since $\mathbf{S}$ is intrinsically comprised of double $\mathbf{P}$, the c-connected $\mathbf{S}$ certainly guarantees the c-connected $\mathbf{P}$. The final cluster labels can be directly obtained from $\mathbf{P}$ without any post-processing.

Lemma 2. *The multiplicity of the eigenvalue zero of the normalized Laplacian matrix $\widetilde{\mathbf{L}}_S$ equals to the number of connected components in the graph associated with $\mathbf{S}$.*

3.2 Optimization Procedure

This section focuses on solving the optimization problem (4). We optimize two variables in model (4) by the block-coordinate descent method [41] that fixes one variable and optimizes the other alternately.

3.2.1 Fix α, and optimize $\mathbf{P}$

When the weight factor $\boldsymbol{\alpha}$ is fixed, the optimization of problem (4) is simplified into:

$$\min_{\mathbf{P}\mathbf{1}=\mathbf{1}, \mathbf{P}\geq\mathbf{0}, \mathrm{rank}(\widetilde{\mathbf{L}}_S)=n+m-c} \|\mathbf{G} - \mathbf{P}\|_{\mathrm{F}}^2, \tag{6}$$

where $\mathbf{G} = \sum_{v=1}^{n_v} \alpha^{(v)} \mathbf{B}^{(v)}$. Since the non-convex rank constraint is hard to tackle, we reach it by solving a counterpart of model (6). Let $\sigma_i(\widetilde{\mathbf{L}}_S)$ denote the i-th smallest eigenvalue of $\widetilde{\mathbf{L}}_S$. Because $\widetilde{\mathbf{L}}_S$ is semi-definite, we have $\sigma_i(\widetilde{\mathbf{L}}_S) \geq 0$ for each i. When all of the first c smallest $\sigma_i(\widetilde{\mathbf{L}}_S)$ equal to zero,

the Laplacian rank constraint is satisfied immediately. Based on this, we rewrite problem (6) into:

$$\min_{\mathbf{P1}=\mathbf{1},\mathbf{P}\geq\mathbf{0}} \|\mathbf{G}-\mathbf{P}\|_{\mathrm{F}}^2+\lambda\sum_{i=1}^{c}\sigma_i(\widetilde{\mathbf{L}}_S). \tag{7}$$

When λ is large enough [1], the first c smallest eigenvalues would infinitely approach to zero. To solve problem (7), we need an important equation known as the Ky Fan's Theoremas [42]:

$$\sum_{i=1}^{c}\sigma_i(\widetilde{\mathbf{L}}_S)=\min_{\mathbf{F}^T\mathbf{F}=\mathbf{I}} Tr(\mathbf{F}^T\widetilde{\mathbf{L}}_S\mathbf{F}), \tag{8}$$

where $\mathbf{F}=[\mathbf{f}^1;\mathbf{f}^2;\cdots;\mathbf{f}^{n+m}]\in\mathbb{R}^{(n+m)\times c}$ is the indicator matrix. Then, problem (8) could be further retold as:

$$\min_{\mathbf{P1}=\mathbf{1},\mathbf{P}\geq\mathbf{0},\mathbf{F}^T\mathbf{F}=\mathbf{I}} \|\mathbf{G}-\mathbf{P}\|_{\mathrm{F}}^2+\lambda Tr(\mathbf{F}^T\widetilde{\mathbf{L}}_S\mathbf{F}). \tag{9}$$

Currently, model (9) needs to optimize $\mathbf{P}$ along with $\mathbf{F}$. Likewise, we first update $\mathbf{F}$ with the fixed $\mathbf{P}$. Problem (9) with respect to $\mathbf{F}$ is equal to

$$\min_{\mathbf{F}^T\mathbf{F}=\mathbf{I}} Tr(\mathbf{F}^T\widetilde{\mathbf{L}}_S\mathbf{F}). \tag{10}$$

Obviously, the optimal solution to $\mathbf{F}$ should be formed by c eigenvectors of $\widetilde{\mathbf{L}}_S$ having smallest eigenvalues, which requires the computational complexity of $\mathcal{O}((n+m)^2c)$ and memory complexity of $\mathcal{O}((n+m)^2)$ at least. Instead of it, we take an efficient way to address this problem by virtue of two beneficial properties in model (9). First, $\widetilde{\mathbf{L}}_S$ is defined as $\widetilde{\mathbf{L}}_S=\mathbf{I}-\mathbf{D}_S^{-\frac{1}{2}}\mathbf{S}\mathbf{D}_S^{-\frac{1}{2}}$. Second, considering that the entire similarity matrix $\mathbf{S}$ is comprised of double bipartite matrix $\mathbf{P}$, the first n rows of $\mathbf{F}$ actually represents the indicator of data points, while the rest m rows is the indicator of anchor points. In light of the orthogonality of $\mathbf{F}$, problem (10) can be addressed by solving its equivalent maximization problem as follows:

$$\max_{\mathbf{F}_{(n)}^T\mathbf{F}_{(n)}+\mathbf{F}_{(m)}^T\mathbf{F}_{(m)}=\mathbf{I}} Tr(\mathbf{F}_{(n)}^T\mathbf{D}_{(n)}^{-\frac{1}{2}}\mathbf{P}\mathbf{D}_{(m)}^{-\frac{1}{2}}\mathbf{F}_{(m)}), \tag{11}$$

where

$$\mathbf{F}=\begin{bmatrix}\mathbf{F}_{(n)}\\ \mathbf{F}_{(m)}\end{bmatrix},\ \mathbf{D}_S=\begin{bmatrix}\mathbf{D}_{(n)} & \\ & \mathbf{D}_{(m)}\end{bmatrix}. \tag{12}$$

$\mathbf{F}_{(n)}$ reserves the first n rows of $\mathbf{F}$ and $\mathbf{F}_{(m)}$ reserves the rest m rows. $\mathbf{D}_{(n)}$ and $\mathbf{D}_{(m)}$ are respectively the left-up block and right-bottom block of $\mathbf{D}_S$ with the size of n by n and m by m. According to Lemma 3, the solution to problem (11) could be obtained by performing SVD decomposition on $\mathbf{D}_{(n)}^{-\frac{1}{2}}\mathbf{P}\mathbf{D}_{(m)}^{-\frac{1}{2}}$, which takes the computational complexity of $\mathcal{O}(m^3+m^2n)$. Due to the number of anchor points $m\ll n$ toward large-scale data clustering, it is much more efficient to solve problem (10) by tackling problem (11) instead.

1. The introduced λ needn't to be tuned manually. Our algorithm determines λ in a heuristic way automatically. To be specific, we first initialize λ with a small value, and update it according to the number of eigenvalue zero of $\mathbf{L}_S$ after each iteration. If this number is smaller than c, λ is multiplied by 2; or if it is greater than $c+1$, λ is divided by 2, otherwise we terminate the iterations.

Algorithm 1: Algorithm to solve problem (6).

Input: Given the weighted bipartite similarity $\mathbf{G}\in\mathbb{R}^{n\times m}$, the cluster number c, a small value λ.
Output: Bipartite graph $\mathbf{P}\in\mathbb{R}^{n\times m}$ with exact c-connected components.
Initialize $\mathbf{F}_{(n)}$ and $\mathbf{F}_{(m)}$ with the left and right singular vectors of $\mathbf{D}_{(m)}^{-\frac{1}{2}}\mathbf{G}\mathbf{D}_{(m)}^{-\frac{1}{2}}$ corresponding to its c largest singular values, where $d_{pr}(i,i)=\sum_{j=1}^{m}g_{ij}$ and $d_{pc}(j,j)=\sum_{i=1}^{n}g_{ij}$.
repeat
  $\forall i,j$, update $t_{ij}=\left\|\frac{\mathbf{f}_{(n)}^i}{\sqrt{D_{(n)}(i,i)}}-\frac{\mathbf{f}_{(m)}^j}{\sqrt{D_{(m)}(j,j)}}\right\|_2^2$, sorted by $t_{i1}\leq t_{i2}\leq\cdots\leq t_{im}$;
  $\forall i$, update $\mathbf{p}^i$ by solving problem (17);
  Update $\mathbf{D}_{(n)}$ and $\mathbf{D}_{(m)}$;
  Update $\mathbf{F}_{(n)}$ and $\mathbf{F}_{(m)}$ by solving problem (11);
  Update λ according to the number of the eigenvalue zero of $\widetilde{\mathbf{L}}_S$;
until *obtaining the c-connected similarity matrix* $\mathbf{P}$;

Lemma 3. *Given* $\mathbf{X}\in\mathbb{R}^{n\times m}$, *the optimal solutions of* $\mathbf{A}\in\mathbb{R}^{n\times c}$ *and* $\mathbf{B}\in\mathbb{R}^{m\times c}$ *to problem*

$$\max_{\mathbf{A}^T\mathbf{A}+\mathbf{B}^T\mathbf{B}=\mathbf{I}} Tr(\mathbf{A}^T\mathbf{X}\mathbf{B}) \tag{13}$$

are $\mathbf{A}=\frac{\sqrt{2}}{2}\mathbf{U}$ *and* $\mathbf{B}=\frac{\sqrt{2}}{2}\mathbf{V}$, *where* $\mathbf{U}$ *and* $\mathbf{V}$ *are respectively the left and right singular vectors of* $\mathbf{X}$ *corresponding to its* c *largest singular values.*

Once the optimal $\mathbf{F}$ is achieved, we optimize $\mathbf{P}$ in model (9) with this fixed $\mathbf{F}$. The following equation is needed:

$$Tr(\mathbf{F}^T\widetilde{\mathbf{L}}_S\mathbf{F})=\frac{1}{2}\sum_{i=1}^{n+m}\sum_{j=1}^{n+m}\left\|\frac{\mathbf{f}^i}{\sqrt{\mathbf{D}_S(i,i)}}-\frac{\mathbf{f}^j}{\sqrt{\mathbf{D}_S(j,j)}}\right\|_2^2 s_{ij}. \tag{14}$$

Taking in the definition of $\mathbf{F}$, $\mathbf{D}_S$ and $\mathbf{S}$, Eq.(14) is further equal to

$$Tr(\mathbf{F}^T\widetilde{\mathbf{L}}_S\mathbf{F})=\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|\frac{\mathbf{f}_{(n)}^i}{\sqrt{\mathbf{D}_{(n)}(i,i)}}-\frac{\mathbf{f}_{(m)}^j}{\sqrt{\mathbf{D}_{(m)}(j,j)}}\right\|_2^2 p_{ij}. \tag{15}$$

Then, problem (9) w.r.t. $\mathbf{P}$ is written as:

$$\min_{\mathbf{P1}=\mathbf{1},\mathbf{P}\geq\mathbf{0}}\sum_{i=1}^{n}\sum_{j=1}^{m}(g_{ij}-p_{ij})^2+\lambda t_{ij}p_{ij}. \tag{16}$$

where $t_{ij}=\left\|\frac{\mathbf{f}_{(n)}^i}{\sqrt{\mathbf{D}_{(n)}(i,i)}}-\frac{\mathbf{f}_{(m)}^j}{\sqrt{\mathbf{D}_{(m)}(j,j)}}\right\|_2^2$. Because the optimization of each element p_{ij} in problem (16) is independent, we optimize $\mathbf{P}$ by rows:

$$\min_{\mathbf{p}^i\mathbf{1}=1,\mathbf{p}^i\geq\mathbf{0}}\left\|\mathbf{p}^i-(\mathbf{g}^i-\frac{\lambda}{2}\mathbf{t}^i)\right\|_2^2. \tag{17}$$

Now, the optimal $\mathbf{p}^i$ could be efficiently solved with a closed form solution by Reference [31]. Algorithm 1 summarizes the pipeline of solving sub-problem (6).

3.2.2 Fix $\mathbf{P}$, and optimize α

When $\mathbf{P}$ is fixed by its optimal solution, the sub-problem of objective (4) in regards to $\boldsymbol{\alpha}$ arrives at

$$\min_{\boldsymbol{\alpha}^T\mathbf{1}=1,\boldsymbol{\alpha}\geq\mathbf{0}} \left\|\sum_{v=1}^{n_V}\alpha^{(v)}\mathbf{B}^{(v)}-\mathbf{P}\right\|_{\mathrm{F}}^2. \tag{18}$$

To solve it, we vectorize each matrix $\mathbf{B}^{(v)}$ into $\hat{\mathbf{b}}^{(v)}$, i.e.,

$$\hat{\mathbf{b}}^{(v)}=\left[\ \mathbf{b}_1^{(v)};\mathbf{b}_2^{(v)};\cdots;\mathbf{b}_m^{(v)}\ \right]\in\mathbb{R}^{nm\times 1}, \tag{19}$$

where $\mathbf{b}_i^{(v)}$ denotes i-th column of $\mathbf{B}^{(v)}$. So, the features of data from n_V views could be gathered into the matrix $\hat{\mathbf{B}}=[\hat{\mathbf{b}}^{(1)},\hat{\mathbf{b}}^{(2)},\cdots,\hat{\mathbf{b}}^{(n_V)}]\in\mathbb{R}^{nm\times n_V}$. Denote $\mathbf{G}=\sum_{v=1}^{n_V}\alpha^{(v)}\mathbf{B}^{(v)}$, and $\hat{\mathbf{g}}$ is the vector of $\mathbf{G}$ likewise, then we have $\hat{\mathbf{g}}=\hat{\mathbf{B}}\boldsymbol{\alpha}$. Also, $\hat{\mathbf{p}}$ denotes the vector of $\mathbf{P}$. As a result, objective (18) is equivalent to

$$\min_{\boldsymbol{\alpha}^T\mathbf{1}=1,\boldsymbol{\alpha}\geq\mathbf{0}} \left\|\hat{\mathbf{B}}\boldsymbol{\alpha}-\hat{\mathbf{p}}\right\|_2^2. \tag{20}$$

Utilizing $\|\mathbf{v}\|_2^2=\mathbf{v}^T\mathbf{v}$, we transform problem (20) into

$$\min_{\boldsymbol{\alpha}^T\mathbf{1}=1,\boldsymbol{\alpha}\geq\mathbf{0}} \boldsymbol{\alpha}^T\mathbf{A}\boldsymbol{\alpha}-\boldsymbol{\alpha}^T\mathbf{h}, \tag{21}$$

where $\mathbf{A}=\hat{\mathbf{B}}^T\hat{\mathbf{B}}$, and $\mathbf{h}=2\hat{\mathbf{B}}^T\hat{\mathbf{p}}$. Due to the semi-definite $\mathbf{A}$, problem (21) is a quadratic convex optimization, which could be efficiently solved by the classical augmented Lagrangian multiplier (ALM) [43] method referred in many literatures [44]–[46]. Via ALM, problem (21) is addressed by tackling its counterpart:

$$\min_{\boldsymbol{\alpha}^T\mathbf{1}=1,\boldsymbol{\alpha}\geq\mathbf{0},\boldsymbol{\alpha}=\boldsymbol{\beta}} \boldsymbol{\alpha}^T\mathbf{A}\boldsymbol{\beta}-\boldsymbol{\alpha}^T\mathbf{h}. \tag{22}$$

The augmented Lagrangian function of problem (22) is defined as:

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

where a penalty function term is added to ensure $\boldsymbol{\alpha}=\boldsymbol{\beta}$, and $\mu\in\mathbb{R}$ is the penalty coefficient. The optimal solution to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are iteratively optimized. In the algorithm, μ is increasingly exaggerated during each iteration, and $\boldsymbol{\eta}\in\mathbb{R}^{n_V}$ is updated by $\boldsymbol{\eta}\leftarrow\boldsymbol{\eta}+\mu(\boldsymbol{\alpha}-\boldsymbol{\beta})$.

1). Update $\boldsymbol{\beta}$ with fixed $\boldsymbol{\alpha}$. Denote the Lagrange function of objective (23) w.r.t. $\boldsymbol{\beta}$ as

$$\mathcal{L}_1(\boldsymbol{\beta})=\boldsymbol{\alpha}^{*T}\mathbf{A}\boldsymbol{\beta}+\frac{\mu}{2}\left\|\boldsymbol{\alpha}^*-\boldsymbol{\beta}+\frac{1}{\mu}\boldsymbol{\eta}\right\|_2^2, \tag{24}$$

where $\boldsymbol{\alpha}$ is fixed with $\boldsymbol{\alpha}^*$. Denote the optimal solution to problem (23) as $\boldsymbol{\beta}^*$, so $\boldsymbol{\beta}^*$ should satisfy its extreme value condition, i.e.,

$$\frac{\partial\mathcal{L}_1(\boldsymbol{\beta}^*)}{\partial\boldsymbol{\beta}}=\mathbf{0}. \tag{25}$$

Based on this, we could update $\boldsymbol{\beta}$ by

$$\boldsymbol{\beta}^*=\boldsymbol{\alpha}^*+\frac{1}{\mu}(\boldsymbol{\eta}-\mathbf{A}^T\boldsymbol{\alpha}^*). \tag{26}$$

Supposing $\boldsymbol{\beta}$ approaches to $\boldsymbol{\alpha}$ iteratively, the equality constraint $h(\boldsymbol{\beta})\rightarrow\mathbf{0}$ therewith (denote $h(\boldsymbol{\beta})=\boldsymbol{\beta}-\boldsymbol{\alpha}$). According to the updating strategy of $\boldsymbol{\eta}\leftarrow\boldsymbol{\eta}+\mu(\boldsymbol{\alpha}-\boldsymbol{\beta})$, $\boldsymbol{\eta}$ converges along with $h(\boldsymbol{\beta})\rightarrow\mathbf{0}$. Because $1\leq\rho\leq 2$, μ increases gradually and its reciprocal also tends to be zero. Accordingly, $\boldsymbol{\beta}$ updated by Eq. (26) equals $\boldsymbol{\alpha}$ after several iterations.

Algorithm 2: Augmented Lagrangian Multiplier (ALM) method to solve problem (21).

Input: a nonzero matrix $\mathbf{A}\in\mathbb{R}^{n_V\times n_V}$ and a nonzero vector $\mathbf{h}\in\mathbb{R}^{n_V}$.
Output: the weight factor $\boldsymbol{\alpha}$.
Initialize $\mu>0$, $\boldsymbol{\eta}$ and set $1\leq\rho\leq 2$. $\forall i,\alpha_v=\frac{1}{n_V}$.
repeat
  With fixed $\boldsymbol{\alpha}$, update $\boldsymbol{\beta}$ by Eq. (26);
  With fixed $\boldsymbol{\beta}$, update $\boldsymbol{\alpha}$ by solving problem (27);
  Update $\boldsymbol{\eta}\leftarrow\boldsymbol{\eta}+\mu(\boldsymbol{\alpha}-\boldsymbol{\beta})$;
  Update $\mu\leftarrow\rho\mu$;
until *convergence*;

Algorithm 3: The pipeline of solving problem (4).

Input: the single-view bipartite similarity $\{\mathbf{B}^{(v)}\}_{v=1}^{n_V}$, cluster number c, view number n_V.
Output: the joint graph $\mathbf{P}\in\mathbb{R}^{n\times m}$ with exact c-connected components.
Initialize $\alpha_v=\frac{1}{n_v}(v=1,2,\cdots,n_v)$.
repeat
  Compute $\mathbf{G}=\sum_{v=1}^{n_V}\alpha^{(v)}\mathbf{B}^{(v)}$;
  Update $\mathbf{P}$ by Algorithm 1;
  Transform $\{\mathbf{B}^{(v)}\}_{v=1}^{n_V}$ and $\mathbf{P}$ to $\hat{\mathbf{B}}$ and $\hat{\mathbf{p}}$;
  Compute $\mathbf{A}$ and $\mathbf{h}$ as objective (21);
  Update $\boldsymbol{\alpha}$ by Algorithm (2);
until $\boldsymbol{\alpha}$ *convergence*;

2). Update $\boldsymbol{\alpha}$ with fixed $\boldsymbol{\beta}$. When $\boldsymbol{\beta}$ is fixed with its optimal value $\boldsymbol{\beta}^*$, we retell the objective (23) into its equivalent counterpart as follows:

$$\min_{\boldsymbol{\alpha}^T\mathbf{1}=1,\boldsymbol{\alpha}\geq\mathbf{0}} \frac{\mu}{2}\left\|\boldsymbol{\alpha}-\boldsymbol{\beta}^*+\frac{1}{\mu}(\boldsymbol{\eta}+\mathbf{A}\boldsymbol{\beta}^*-\mathbf{h})\right\|_2^2. \tag{27}$$

The details about how to solve problem (27) refers to appendix, and the algorithm of solving problem (18) is summarized in Algorithm 2. With optimizing the sub-optimization of $\boldsymbol{\alpha}$ and $\mathbf{P}$ in objective (4) separately, we further summarize the whole pipeline of solving problem (4) in Algorithm 3. We initialize weights for each single-view evenly and then update these weights along with the uniformed bipartite graph across multiple views iteratively.

4 SINGLE-VIEW GRAPH LEARNING

The previous contents are about how to learn a structured optimal joint graph that is adaptively compatible across multiple single-view graphs. However, since the affinity graph of each view is not available and the data features are known instead in most cases, apart from graph fusion, the issue of building single-view bipartite graphs is of great significance as well. To address it, we take two steps, i.e., selecting anchors from given date points and constructing single-view bipartite graphs subsequently.

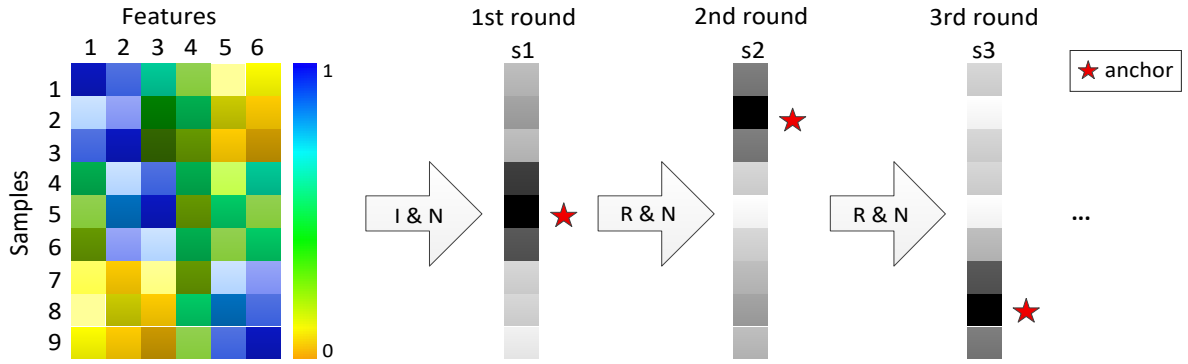


Fig. 2. Illustration of Directly Alternate Sampling (DAS). Each row is the processed feature vector of a sample. Colourful blocks represent the feature values. The gray blocks are the scores of samples, ranging from 0 to 1. Red star marks the anchor index in each iteration. I & N is the initialization and normalization of scores by Eq. (28) and Eq. (30), while R & N is the recalculation and normalization of scores by Eq. (31) and Eq. (30).

4.1 Anchor Selection of Multi-view Data

Two frequently used manners of anchor selection are random policy and k-means [47]. Random policy means randomly choosing a fraction of data as anchor points. Despite time-saving, it makes the clustering behavior neither satisfactory nor stable. By contrast, k-means manner that uses the clustering centroids of data as anchor points is more reliable in the realistic tasks. Nevertheless, for multi-view data, the anchor sets generated by different views are not consistent when k-means is performed on each view separately, which impedes to construct valid bipartite graphs across multiple views. To deal with it, a general way is to concatenate all features and then to split k-means centroids of concatenated features by views [40], [48]. However, an ineluctable issue is that k-means is sensitive to its initialization of starting centroids, for which algorithms that involve k-means as pre-processing or post-processing are also unstable and require numerous independent and repeated running to eliminate the occasionality of results like random selection. Considering that real samples belonging to the same cluster usually have similar data representations whereas samples belonging to different clusters have distinct representations, based on this, we provide a simple yet efficient anchor selection strategy, named *directly alternate sampling* method (DAS). Our strategy follows the principle that the anchor points should well cover the entire point cloud of data, and thus alternately sampling data points according to their feature similarity over all of clusters, as illustrated in Fig. 2. To be specific, denote $d_T = \sum_{v=1}^{n_V} d^{(v)}$ as the total dimension of data features, where $d^{(v)}$ is the feature dimension in the v-th view. We concatenate all features as $\hat{\mathbf{X}} = [\mathbf{X}^{(1)}; \mathbf{X}^{(2)}; \cdots; \mathbf{X}^{(n_V)}]$. Then, the intuitive score $\mathbf{s} \in \mathbb{R}^n$ of samples could be computed by

$$s_i = \sum_{j=1}^{d_T} \mathrm{Tra}(\hat{\mathbf{X}}_{ij}) \tag{28}$$

where $\mathrm{Tra}(\cdot)$ denotes the operation that transforms the raw features to be non-negative. To be specific, when feature are not non-negative, we translate the features in each dimension by subtracting the minimum value within each dimension. The motivation of operation above is that the features of samples in the same cluster generally have smaller differences than features of samples in difference clusters, and samples in the same cluster naturally obtain similar scores computed by summing the feature values. Consequently, when $\hat{\mathbf{x}}_i$ is in the same cluster as $\hat{\mathbf{x}}_j$, s_i would be close to s_j; when they are in different clusters, s_i has a big differences with s_j. In this way, we select an anchor set by using the computed score vector $\mathbf{s} = [s_1, s_2, \cdots, s_n]$. Definitely, we pick out the sample with largest score as the initial anchor point, i.e., solving

$$\arg\max_i s_i, \tag{29}$$

Noting that the intrinsic clusters of data could be reflected by their raw features to some extent, and the computed score absorbs this cluster structure straightforward. In order to make anchor points roundly cover the data points against multiple clusters, we choose the anchor points and update the scores in an alternate fashion, which aims to obtain the anchor points from different clusters alternately. To achieve it, we first normalize the score of each sample by dividing the largest score, i.e.,

$$s_i \leftarrow \frac{s_i}{\max(\mathbf{s})}, (i = 1, 2, \cdots, n). \tag{30}$$

So, the score of the last chosen sample is scaled into 1, and samples in the similar clusters with the chosen sample gains large values. Then, we update each score by

$$s_i \leftarrow s_i(1 - s_i). \tag{31}$$

This linear computing is executed m times to achieve m anchor points, where m is lager than the real class number or the desired cluster number c. The intuition behind this transformation is that the extremely high or low scores would diminish, whereas the medial scores would be exaggerated fairly in each next round, so the next anchor is of little probability being in the same clusters with the current anchor point. Besides that, since we always select the sample who has the highest score, the score of the chosen sample vanish and none of anchors can be chosen repeatedly. The necessity of our success is that we make a compromise between the random selection and the k-means selection by definitely walk between the clusters and randomly walking within each cluster. So, our strategy makes a well balance between the certainty and the randomness, providing the excellent performance especially for large-scale data.

4.2 Construction of Single-view Bipartite Graph

With the anchor points, we now narrate how to construct bipartite graphs $\{\mathbf{B}^{(v)}\}_{v=1}^{n_V}$. Since the single-view graphs are built as the same way, herein we take insight into one single-view. Considering model (4) requires a normalized and nonnegative joint graph $\mathbf{P}$, it is natural to achieve normalized and nonnegative single-view graph $\mathbf{B}^{(v)}$ for v-th view. Considering that the pairs of samples having large geometric distances generally have small similarities, we learn $\mathbf{B}^{(v)}$ by solving

$$\min_{\mathbf{b}^{i(v)}\mathbf{1}=\mathbf{1},\mathbf{b}^{i(v)}\geq\mathbf{0}} \sum_{j=1}^{m} h(\mathbf{x}_i,\mathbf{y}_j)b_{ij}^{(v)} + \gamma\sum_{j=1}^{m}(b_{ij}^{(v)})^2, \tag{32}$$

where $\mathbf{b}^{i(v)}$ denotes the i-th row of $\mathbf{B}^{(v)}$. $h(\mathbf{x}_i,\mathbf{y}_j)$ measures the differences of the i-th sample and j-th anchor, abbreviated as $h_{i,j}$. A simple metric of Euclidean distance is employed here, i.e., $h_{i,j} = \|\mathbf{x}_i - \mathbf{y}_j\|_2^2$. Therefore, the first term in objective (32) learns the local structure of data that the samples are strongly connected to their nearby anchors and weakly connected to their distant anchors. The second term in objective (32) serves as a regularization such that each anchor is assigned with the same similarity when there is no prior knowledge. Additionally, when γ is a maximal, this regularization also promotes a sparse similarity matrix. Suppose for each i, $[h_{i,1}, h_{i,2}, \cdots, h_{i,m}]$ is ordered as $h_{i,1} \leq h_{i,2} \leq \cdots \leq h_{i,m}$. Problem (32) is addressed with a closed form solution as follows:

$$b_{ij}^{(v)} = \left(\frac{h_{i,k+1} - h_{i,j}}{k h_{i,k+1} - \sum_{r=1}^{k} h_{i,r}}\right)_+, \tag{33}$$

where $(v)_+ = \max(0, v)$. k indicates the number of nonzero elements in $\mathbf{b}^{i(v)}$. When $j > k$, the solution above would be zero. By this means, each data point is only connected to its k nearest anchors, and thus $\mathbf{B}^{(v)}$ would be sparse. Therefore, we actually optimize the significant entries when learning $\mathbf{P}$ in model (6), reducing the time quite a lot.

5 DISCUSSION

In this section, we have a discussion over the computational complexity of our algorithm. When data features are known, it needs two steps to cluster data, i.e., the construction of single-view bipartite graphs and the optimization of the joint graph. In the following, we analyze them by steps.

Construction of single-view graphs. As described in the last section, we first employ DAS to generate m data points from all as anchors, which needs m iterations and each iteration performs a linear operation on a n-dimension score vector, requiring the time complexity of $\mathcal{O}(nm)$. Considering that bipartite graph $\mathbf{B}^{(v)}$ connects each data point with its k nearest anchors, we need to compute and sort the Euclidean distance between data points and anchors, which take the computational complexity of $\mathcal{O}(nmd)$ and $\mathcal{O}(nmlog(m))$ respectively. At least, constructing $\mathbf{B}^{(v)}$ by Eq. (33) takes $\mathcal{O}(nmd + nmlog(m) + nkd)$. Taking all of n_V views into consideration, the main computational complexity in this step is $\mathcal{O}(n_V nmd + n_V nmlog(m))$.

Optimization of the joint graph. With the multiple single-view graphs, Algorithm 3 outputs the joint graph along with the cluster labels by solving problem (4), which is addressed by updating $\mathbf{P}$ and $\boldsymbol{\alpha}$ iteratively. Noting that Algorithm 3 employs Algorithm 1 to update $\mathbf{P}$, which takes the computational complexity of $\mathcal{O}(nmct_1 + nm^2t_1 + m^3t_1)$ as analyzed before, where t_1 is the iteration number in Algorithm 1. Since the dimension of $\boldsymbol{\alpha}$ equaling the number of views is usually a quite small value, the updating of $\boldsymbol{\alpha}$ is pretty fast in practice. We only need to consider the computation of $\mathbf{A}$ and $\mathbf{h}$ in objective (21), which take $\mathcal{O}(n_V^2 nk)$ and $\mathcal{O}(n_V nk)$ respectively. Noting that the outer loop in Algorithm 3 depending on the convergence of $\boldsymbol{\alpha}$ is no more than five iterations. Totally, the main computational complexity in this step is $\mathcal{O}(nm^2t_1 + m^3t_1 + n_V^2 nk)$.

Due to $n \gg m$, and c, k, n_V are small constants, the main computational complexity is actually $\mathcal{O}(nmd + nm^2t_1)$. which is linear to the number of samples n. As a result, our method is certain to scale well with data size. The CPU time of executing the proposed algorithm will be tested and compared to the related approaches in the experiments.

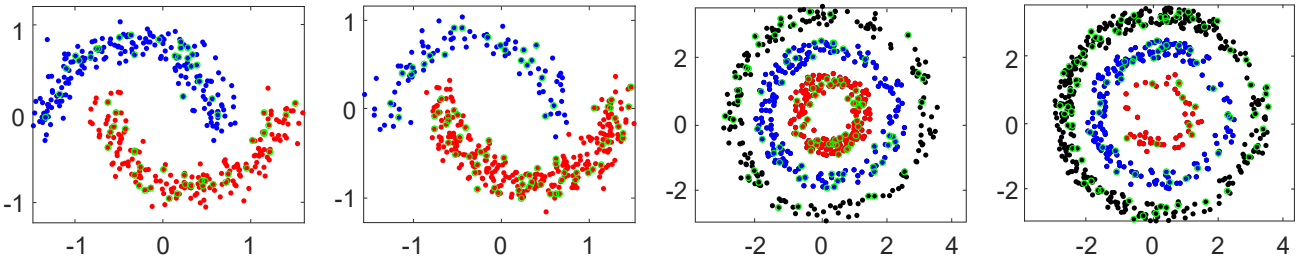
6 EXPERIMENTS

We conduct experiments to check the effectiveness and the superiority of the proposed method. All of experiments are divided into two parts. One is several validation experiments that are performed on synthetic data which helps to understand what goals our model exactly achieves, and the other is some verification experiments that are performed on real benchmark datasets which demonstrates the clustering performance of the proposed algorithm toward multi-view data. For simplicity, we abbreviate the proposed Scalable and parameter-Free Multi-view graph Clustering as SFMC.

6.1 Evaluations on Synthetic Data

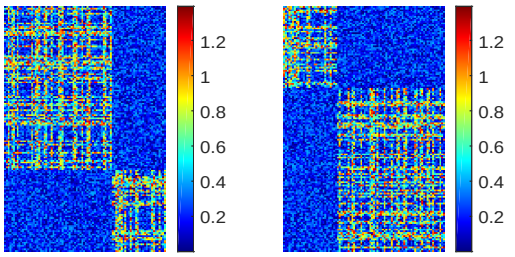
The first group of validation experiments are conducted to validate the performance of the proposed directly alternate sampling (DAS) scheme in generating anchors with the given data features. Fig. 3 shows the results of DAS towards two representative types of the point cloud, including threering data and two-moon data, where different colors of dots represent different clusters, and the green circles mark the anchor points chosen by DAS. From these results, we clearly observe that the anchor points roundly cover the point cloud evenly. To further evaluate DAS, we also discuss the cases of unbalance data where the cluster sizes are distinct as shown in Fig. 3 (b)(d). From these results, we could obtain that the selected anchor points have accordant distribution like raw points against the varying cluster size. All of these conclusions demonstrate the excellent performance of the proposed DAS in anchor selection. We would evaluate the clustering performance of the proposed SFMC equipped with DAS compared to SFMC equipped with other anchor selection schemes later.

The second group of validation experiments concerns that which view does the proposed model prefer towards multiple views and how much does the joint graph learn from each single view. Since the features from different views embed distinct characteristics of data, the graphs constructed from different views could even reflect different cluster structures. In light of this, we design three different bipartite graphs as the input of the single-view graphs, as

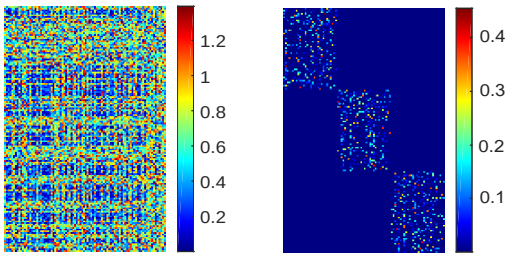


(a) Balanced two-moon data (b) Unbalanced two-moon data (c) Balanced three-ring data (d) Unbalanced three-ring data

Fig. 3. The results of anchor selection by directly alternate sampling (DAS) on two types of non-linear data. The number of data points of each cluster are balanced in (a)(c) and are unbalance in (b)(d).



(a) Input, graph in view #1 (b) Input, graph in view #2



(c) Input, graph in view #3 (d) Output, the joint graph

Fig. 4. Experiments on synthetic multi-view block diagonal graphs. (a)(b)(c) are respectively input single-view graphs. (d) is the corresponding joint graph obtained by Algorithm 3 with three clusters.

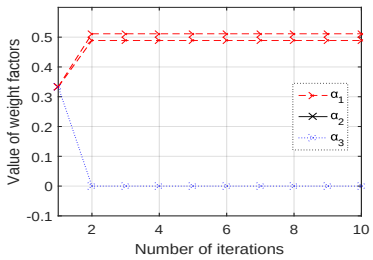


Fig. 5. The values of weights α_1, α_2 and α_3 allocated for three single-views in Fig. 4(a)(b)(c) w.r.t. the number of iterations.

shown in Fig. 4 (a)(b)(c). The graphs of the first and second views have two diagonal blocks embodying different cluster distributions, and the graph of the third view fills the Gaussian noise without any cluster structures. All of these single-view graphs are evenly added with random noises.

By using the proposed Algorithm 3 to obtain three clusters, the learned joint graph is as shown in Fig. 4 (d). We can observe that the output graph has exact 3 connected blocks, and the final clusters absorb information in both view #1 and view #2 jointly. Accordingly, Fig. 5 records the weights allocated for three views after each iteration. We conclude that i) our model explores not only informative structure within each view but also the compatible structure between multiple views; For instance, the first and third clusters are learned from view #1 and view #3 respectively, while the second cluster is learned from view #1 and view #2 jointly; ii) the appropriate weight is assigned to each view and converges fast within two iterations dramatically; iii) the severely ill-views cannot degrade the performance, since they obtain very small weights automatically like view #3.

6.2 Evaluations on Real Datasets

6.2.1 Experimental Settings

Datasets. We employ six real multi-view benchmark datasets to evaluate the proposed algorithm. *MSRC-v5* [49] is an object dataset having seven categories of tree, building, airplane, cow, face, car and bicycle and each of them contains 30 images. *Handwritten4* (HW4) is a digit dateset generated from UCI machine learning repository [50], containing 2000 instances that describe the digits from 0 to 9. *Caltech101-20* [51] (Cal-20) is the frequently used subsets of Caltech101 consisting of 20 categories of images built for object recognition tasks. *Mnist4* is a subset of MNIST [52]-a freely available and well-known handwritten database for image recognition, consisting of four categories from digit 0 to digit 3, and each category has 1000 samples evenly. Besides that, we also evaluate our algorithm on two large-scale datasets. One is *NUS-WIDE-OBJ* (NUS) dataset [53] which is consist of 30000 object images from 31 classes including bear, books, boats, computer, plane, etc, the other is a text dataset *Reuters* [54] containing 18757 documents described in five languages including English, France, German, Italian, and Spanish. Multiple kinds of features are extracted from all of these image samples to form the corresponding multi-view datasets. The detailed information of each view is introduced in TABLE 1, and the values in brackets represent the number of features in respective views.

Metrics. We employ seven criteria to comprehensively measure the clustering quality of the investigated algorithms, including ACCuracy (ACC), Normalized Mutual Information (NMI) [55], Purity, PREcision (PRE), RECall (REC), F_{score} and Adjusted Rand Index (ARI). These indicators measure the consistency between the real labels

TABLE 1
The introduction to views' information of the utilized datasets.

View	MSRC-v5	HW4	Mnist4
#1	CM(24)	FOU(76)	ISO(30)
#2	HOG(576)	FAC(216)	LDA(9)
#3	GIST(512)	ZER(47)	NPE(30)
#4	LBP(256)	MOR(6)	-
#5	CENT(254)	-	-
Total	1622	345	69
Class	7	10	4
Size	210	2000	4000
View	Cal-20	NUS	Reuters
#1	GABOR(48)	CH(64)	English(21531)
#2	WM(40)	CM(225)	France(24892)
#3	CENT(254)	CORR(144)	German(34251)
#4	HOG(1984)	EDH(73)	Italian(15506)
#5	GIST(512)	WT(129)	Spanish(11547)
#6	LBP(928)	-	-
Total	3766	635	107727
Class	20	31	6
Size	2386	26315	18758

and the predicted labels from different viewpoints. All of them indicate the better clustering performance with a larger value. ACC indicates the proportion of properly clustered samples in total samples, computed by $\mathrm{ACC} = \frac{1}{n}\sum_{i=1}^{n}\delta(r_i, map(p_i))$ where r_i denotes the real class label of i-th sample, p_i is the corresponding predicted cluster label, and $map(\cdot)$ maps the cluster labels to their best real labels according to Kuhn-Munkres algorithm, also known as Hungarian algorithm [56]. $\delta(x, y)$ is a sign function which outputs 1 when x equals y and outputs 0 otherwise. Purity is a simple and transparent evaluation measure, which computes the proportions of samples belonging to the most frequent class label within each cluster, i.e., $\mathrm{Purity}(\mathcal{S}, \mathcal{S}') = \frac{1}{n}\sum_k \max_j |\mathcal{S}_j \cap \mathcal{S}'_k|$ where $\mathcal{S}$ and $\mathcal{S}'$ denote the class label set and cluster label set respectively. $\mathcal{S}_i$ is the i-th cluster subset. High purity is easily achieved when the number of clusters is large. In particular, purity is 1 if all samples are partitioned different groups. Thus, we cannot use purity to trade off the clustering quality against the number of clusters. A measure that allows us to make this tradeoff is NMI, which improves MI against the random distribution by $\mathrm{NMI}(\mathcal{S}, \mathcal{S}') = \mathrm{MI}(\mathcal{S}, \mathcal{S}')/\sqrt{H(\mathcal{S})H(\mathcal{S}')}$, where $H(\mathcal{S})$ and $H(\mathcal{S}')$ compute the entropies of the sets $\mathcal{S}$ and $\mathcal{S}'$ respectively, and MI can be information-theoretically interpreted to measure the coherent between distributions from all overall perspective. When the observed set is a random distribution with respect to the other, NMI would be 0. F_{score} is a tradeoff of precision and recall, computed by $\frac{2*recall*precision}{recall+precision}$, where precision counts the proportion of correctly clustered pairs in all pairs having the same cluster labels, and recall counts the proportion of correctly clustered pairs in all pairs having the same real labels. ARI is a recently proposed indicator that evaluates the similarity of distributions of given sets and ranges from -1 to 1. The negative values indicate the very poor performance.

Schemes. As we desire, there is no parameters to be tuned in the proposed SFMC, and just a self-updated λ which is determined in a heuristic way automatically. In the experiments, we initialize λ as 0.1 for all of datasets referring to the literature [31]. For the fairness, we set the parameters of all algorithms as their faults or best options. We run all of algorithms 20 times independently on each datasets and report their averages as the results. The convergence conditions of all algorithms are set to 50 maximum iterations with a break condition of the objective errors of two adjacent iterations being less than 1e-4. All the algorithms are implemented in MATLAB R2016b and run on a Window 10 PC with Intel Core i7-9700F 3.00GHz CPU and 16GB RAM.

6.2.2 Experimental Results

Effect of Anchor Selection. We investigate the effect of anchor selection on the bipartite graph construction by talking about two core questions. The first one is what differences the proposed SFMC has when it employs different anchor selection strategies. The second one is how does the number of anchors affect on the performance of SFMC.

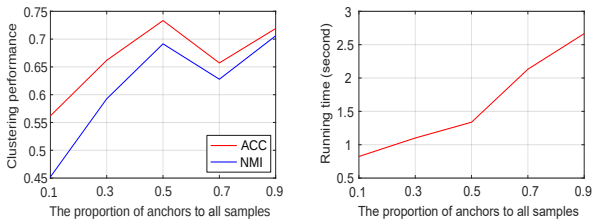
To answer the first question, we employ four anchor selection manners, including the proposed DAS, the frequently used random policy and k-means, as well as K-nearest points (KNP) which takes the nearest data points of the centroids generated by k-means as anchors. With the obtained anchor points, we construct the single-view bipartite graphs as the manner in Section 4.2, and then Algorithm 3 is invoked to obtain the joint graph and the cluster labels. To eliminate the randomness, all of the experiments are run ten times by inputting independent results of anchor selection. TABLE 2 reports the averages over ACC, NMI and CPU running time along with the standard deviations. From these results, we can see that DAS has a fairly stable performance compared to other three manners, since DAS is free of any random initialization. Besides that, DAS is time-economical and achieves a much more competitive performance than other three manners. So, the proposed DAS is the most suitable to serve as anchor selection for the bipartite graph construction.

TABLE 2
The clustering performance and CPU time (sec.) (MEAN $\pm$ STD) of SFMC equipped with different anchor selection strategies.

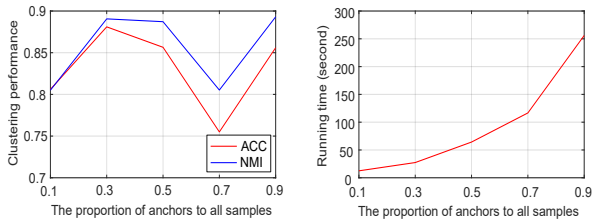
Dataset	Mnist4		
	ACC	NMI	CPU time
DAS	$\mathbf{0.916 \pm 0.000}$	$\mathbf{0.800 \pm 0.000}$	$\mathbf{127.0 \pm 1.2}$
Random	0.838 ± 0.107	0.729 ± 0.098	134.7 ± 13.2
K-mean	0.751 ± 0.165	0.637 ± 0.108	139.5 ± 5.3
KNP	0.877 ± 0.089	0.760 ± 0.086	131.3 ± 5.2
Dataset	MSRC-v5		
	ACC	NMI	CPU time
DAS	$\mathbf{0.809 \pm 0.000}$	$\mathbf{0.721 \pm 0.000}$	$\mathbf{0.781 \pm 0.012}$
Random	0.722 ± 0.083	0.680 ± 0.071	0.788 ± 0.060
K-mean	0.769 ± 0.059	0.713 ± 0.045	0.821 ± 0.033
KNP	0.763 ± 0.072	0.724 ± 0.050	0.808 ± 0.032
Dataset	Caltech101-20		
	ACC	NMI	CPU time
DAS	$\mathbf{0.642 \pm 0.000}$	$\mathbf{0.595 \pm 0.000}$	39.54 ± 0.32
Random	0.591 ± 0.080	0.569 ± 0.088	$\mathbf{39.27 \pm 1.44}$
K-means	0.554 ± 0.065	0.527 ± 0.084	42.30 ± 2.61
KNP	0.614 ± 0.028	0.588 ± 0.041	40.25 ± 2.91
Dataset	Handwritten4		
	ACC	NMI	CPU time
DAS	$\mathbf{0.853 \pm 0.000}$	$\mathbf{0.871 \pm 0.000}$	$\mathbf{24.78 \pm 0.23}$
Random	0.827 ± 0.041	0.843 ± 0.031	25.64 ± 1.82
K-means	0.837 ± 0.037	0.853 ± 0.029	26.65 ± 1.02
KNP	0.824 ± 0.049	0.844 ± 0.039	25.35 ± 0.97

TABLE 3
The clustering performance of SFMC compared to the scalable CRL (s-CLR) by using single-view features and concatenated features.

Dataset	Mnist4							Handwritten4						
Method	ACC	NMI	Purity	PRE	REC	F_{score}	ARI	ACC	NMI	Purity	PRE	REC	F_{score}	ARI
SFMC	**0.917**	**0.801**	**0.917**	**0.846**	**0.855**	**0.852**	**0.802**	**0.853**	**0.871**	**0.873**	**0.775**	**0.910**	**0.837**	**0.817**
s-CLR (#1)	0.660	0.649	0.742	0.626	0.798	0.701	0.585	0.660	0.683	0.699	0.527	0.722	0.609	0.558
s-CLR (#2)	0.843	0.762	0.744	0.640	0.824	0.721	0.655	0.698	0.731	0.731	0.592	0.803	0.681	0.640
s-CLR (#3)	0.743	0.661	0.744	0.642	0.827	0.723	0.657	0.660	0.651	0.661	0.441	0.760	0.558	0.495
s-CLR (#4)	-	-	-	-	-	-	-	0.403	0.452	0.426	0.310	0.383	0.343	0.262
s-ConcatCLR	0.897	0.747	0.897	0.813	0.822	0.817	0.756	0.759	0.751	0.760	0.610	0.865	0.716	0.678
Dataset	MSRC-v5							Caltech101-20						
Metric	ACC	NMI	Purity	PRE	REC	F_{score}	ARI	ACC	NMI	Purity	PRE	REC	F_{score}	ARI
SFMC	**0.810**	**0.721**	**0.810**	**0.657**	**0.782**	**0.714**	**0.663**	**0.642**	**0.595**	**0.748**	**0.586**	0.677	**0.628**	**0.461**
s-CLR (#1)	0.333	0.226	0.381	0.199	0.507	0.286	0.108	0.390	0.174	0.450	0.195	0.720	0.307	0.069
s-CLR (#2)	0.681	0.608	0.710	0.478	0.707	0.570	0.471	0.387	0.238	0.468	0.177	0.501	0.261	0.027
s-CLR (#3)	0.648	0.595	0.652	0.428	0.688	0.528	0.467	0.323	0.206	0.422	0.169	0.515	0.254	0.013
s-CLR (#4)	0.400	0.397	0.486	0.307	0.421	0.355	0.231	0.442	0.269	0.492	0.198	0.745	0.313	0.076
s-CLR (#5)	0.614	0.529	0.638	0.461	0.634	0.534	0.445	0.414	0.284	0.479	0.205	0.763	0.324	0.091
s-CLR (#6)	-	-	-	-	-	-	-	0.358	0.244	0.452	0.172	0.611	0.268	0.019
s-ConcatCLR	0.590	0.509	0.614	0.451	0.482	0.466	0.377	0.596	0.429	0.653	0.313	**0.817**	0.453	0.285



(a) MSRC-v5 (b) MSRC-v5



(c) Handwritten4 (d) Handwritten4

Fig. 6. The clustering ACC, NMI and CPU time of the proposed algorithm with the varying number of anchor points.

To answer the second question, we conduct a simple experiment that studies the clustering performance of the proposed algorithm with the varying number of anchor points. Specifically, we change the proportion that anchor points take in the entire data points from 0.1 to 0.9 with the interval of 0.2, then we report ACC, NMI and running time accordingly, as shown in Fig. 6. Noting that the consumed time becomes longer and longer as the number of anchors ascending, and the anchor-time curves are not straight climb. After the proportion of anchors over 0.5, the time consumption ascends more seriously. However, it is also observed that the anchor-ACC/NMI curves do not ascend monotonously, and they could reach at a peak value when a half of samples are anchor points, which indicates that it is not necessary to employ too many anchors in our algorithm. Consequently, we set the proportion of anchors to all data points as 0.5 uniformly in the comparison experiments.

Compared to Single-view Algorithms. Multi-view clustering approaches discriminately exploits features collected from different views by a straightforward way of weighting the features in each view. Apart from these multi-view approaches, a simple way of dealing with multi-view data is to concatenate all features and utilize single-view clustering methods directly, which mixes the multiple views and treats them indiscriminately. As a single-view method, CLR [31] exploits the structured optimal graph constructed among all of samples and clusters them simultaneously, inspiring the proposed model in some extent. In order to demonstrate the superiority of the multi-view graph clustering than the single-view graph clustering, we provide the experiment that CLR computes the single-view features respectively as well as the concatenated features. It is highlighted that we improve the scalability by inputting the single-view bipartite graphs constructed as SFMC, named scalable CLR (s-CLR), where anchor points are also generated by DAS. TABLE 3 shows the results on the benchmark datasets, where s-CLR (#v) denotes the clustering results of CLR by using features in v-th view and s-ConcatCLR represents the results of s-CLR by using concatenated features. From these results, we observe that

- First, the performances of using different single-views are distinguishing mostly. For mnist4, Handwritten4 and MSRC-v5, their view #2 have the best clustering results, outperforming the second best views five percents at least;
- Second, the performance of using the concatenated features defeats that using the single-view features for all of the tested datasets expect for MSRC-v5, whose view #1 is so inferior that the concatenated features are degraded a lot;
- Third, the performance of the proposed SFMC gains the superior results than the single-view clustering, since the features in different views are distinct and our model obtains more informative knowledge by jointly mining the single-view features;
- Fourth, the performance of the proposed SFMC surpasses that using concatenated features considerably,

TABLE 4
The clustering performance of the proposed SFMC compared to the state-of-the-art competitors on Handwritten4 and MSRC-v5. The algorithms gain the stable results are marked by ‡.

Dataset	Handwritten4						
Metric	ACC	NMI	Purity	PER	REC	F_{score}	ARI
SFMC‡	**0.853 (±0.000)**	**0.871 (±0.000)**	**0.873 (±0.000)**	**0.775 (±0.000)**	**0.910 (±0.000)**	**0.837 (±0.000)**	**0.817 (±0.000)**
co-train	0.381 (±0.021)	0.301 (±0.019)	0.399 (±0.019)	0.290 (±0.018)	0.295 (±0.019)	0.293 (±0.018)	0.214 (±0.021)
co-reg	0.784 (±0.010)	0.758 (±0.004)	0.795 (±0.008)	0.698 (±0.010)	0.724 (±0.005)	0.710 (±0.007)	0.667 (±0.037)
SwMC‡	0.758 (±0.000)	**0.833 (±0.000)**	0.792 (±0.000)	0.686 (±0.000)	0.867 (±0.000)	0.766 (±0.000)	0.737 (±0.000)
MVGL‡	**0.811 (±0.000)**	0.809 (±0.000)	0.831 (±0.000)	**0.721 (±0.000)**	0.826 (±0.000)	0.770 (±0.000)	0.743 (±0.000)
MVSC	0.796 (±0.059)	0.820 (±0.030)	0.808 (±0.044)	0.715 (±0.082)	0.838 (±0.035)	0.769 (±0.046)	0.741 (±0.053)
RDEKM‡	0.805 (±0.000)	0.803 (±0.000)	**0.842** (±0.000)	0.714 (±0.000)	0.806 (±0.000)	0.757 (±0.000)	0.728 (±0.000)
SMSC‡	0.742 (±0.000)	0.781 (±0.000)	0.759 (±0.000)	0.675 (±0.000)	0.767 (±0.000)	0.717 (±0.000)	0.685 (±0.000)
AMGL	0.704 (±0.045)	0.762 (±0.040)	0.732 (±0.042)	0.591 (±0.081)	0.781 (±0.022)	0.670 (±0.060)	0.628 (±0.070)
MLAN	0.785 (±0.045)	0.832 (±0.027)	0.812 (±0.045)	0.706 (±0.053)	**0.871 (±0.017)**	**0.779 (±0.039)**	**0.752 (±0.044)**
Dataset	MSRC-v5						
Metric	ACC	NMI	Purity	PRE	REC	F_{score}	ARI
SFMC‡	**0.810 (±0.000)**	**0.721 (±0.000)**	**0.810 (±0.000)**	0.657 (±0.000)	**0.782 (±0.000)**	**0.714 (±0.000)**	**0.663 (±0.000)**
co-train	0.674 (±0.070)	0.587 (±0.046)	0.690 (±0.061)	0.546 (±0.064)	0.571 (±0.063)	0.558 (±0.063)	0.485 (±0.074)
co-reg	0.635 (±0.007)	0.578 (±0.006)	0.659 (±0.006)	0.511 (±0.008)	0.535 (±0.007)	0.522 (±0.007)	0.425 (±0.030)
SwMC‡	0.776 (±0.000)	**0.774 (±0.000)**	**0.805 (±0.000)**	**0.687 (±0.000)**	**0.831 (±0.000)**	**0.752 (±0.000)**	**0.708 (±0.000)**
MVGL‡	0.690 (±0.000)	0.663 (±0.000)	0.733 (±0.000)	0.466 (±0.000)	0.715 (±0.000)	0.564 (±0.000)	0.476 (±0.000)
MVSC	**0.794 (±0.075)**	0.672 (±0.058)	0.756 (±0.071)	0.585 (±0.091)	0.779 (±0.035)	0.664 (±0.062)	0.600 (±0.079)
RDEKM‡	0.738 (±0.000)	0.650 (±0.000)	0.738 (±0.000)	0.594 (±0.000)	0.647 (±0.000)	0.619 (±0.000)	0.555 (±0.000)
SMSC‡	0.766 (±0.000)	0.717 (±0.000)	0.804 (±0.000)	**0.672** (±0.000)	0.718 (±0.000)	0.694 (±0.000)	0.643 (±0.000)
AMLG	0.751 (±0.078)	0.704 (±0.044)	0.789 (±0.056)	0.621 (±0.090)	0.744 (±0.026)	0.674 (±0.063)	0.615 (±0.079)
MLAN‡	0.681 (±0.000)	0.630 (±0.000)	0.733 (±0.000)	0.494 (±0.000)	0.718 (±0.000)	0.694 (±0.000)	0.643 (±0.000)

TABLE 5
The clustering accuracy and CPU running time (sec.) (MEAN ± STD) of the stable competitors‡ on four benchmark datasets.

Dataset	Handwritten4		MSRC-v5	
	ACC	CPU time	ACC	CPU time
SFMC‡	0.853	24.29±0.32	0.810	0.778±0.029
SwMC‡	0.758	422.17±1.29	0.776	3.593±0.027
MVGL‡	0.811	548.16±0.45	0.690	3.929±0.069
RDEKM‡	0.805	9.74±0.38	0.738	6.663±0.082
SMSC‡	0.742	12.78±0.18	0.767	0.218±0.015
Dataset	Mnist4		Caltech101-20	
	ACC	CPU time	ACC	CPU time
SFMC‡	0.917	87.07±0.27	0.642	38.01±0.51
SwMC‡	0.914	571.60±4.04	0.614	234.62±3.79
MVGL‡	0.912	6920.74±79.03	0.606	1030.17±10.56
RDEKM‡	0.885	8.75±0.77	0.424	2504.11±0.53
SMSC‡	0.913	37.84±0.41	0.588	18.78±0.14

since we assign appropriate weights to different view such that the excellent views are highly learned while the poor views are slighted learned.

Consequently, compared to individually utilizing features in some one view or indiscriminately utilizing features in multiple views, self-learning adaptive weighting these single-view features is more significant for heterogeneous data clustering.

Compared to Multi-view Algorithms. In this part, we compare the proposed SFMC with nine state-of-the-art multi-view clustering algorithms that have been mentioned at the related works of Section 2. In order to demonstrate our advantages, the clustering results of the involved algorithms are visualized including the averages as well as the standard deviations of twenty independently repeated experiments, as shown in TABLE 4 6. For clarity, the best and second best entries are in bold to highlight the prominent results. Besides that, we also mark the algorithms that gain the zero-deviation to indicate the high stableness, which is also a pivotal indicator to rank models. Since the stable algorithms we mark are independent of the initializations and are needless of repeated experiments, TABLE 5 records their CPU running time as well as the clustering ACC to further compare them. Noting that the CPU running time starts by giving the original multi-view data features and ends by outputting the cluster labels. From these results, we analyze that

- First, in terms of clustering performance, the proposed SFMC is superior to the state-of-the-art competitors in most occasions, proving its powerful capacity in multi-view clustering;
- Second, the proposed SFMC is mostly defeated by SwMC which is also the structured graph clustering like SFMC, manifesting that the structured graph learning not only immediately avoiding the post-clustering process but also effectively improving the graph quality for clustering;
- Third, despite a anchor-based algorithm, the proposed SFMC is considerably superior to most of competitors, showing that utilizing a half of data points selected by DAS instead of the entire point cloud in SFMC totally affords efficient clustering;
- Fourth, among all the competitors, the proposed SFMC as well as SwMC, MVGL, REDKM and SMSC gain the stable clustering clustering in all of datasets, and there are only SFMC and SMSC is free of parameter-tuning;
- Fifth, in terms of time consumption, SFMC, RDEKM and SMSC has a qualitative upgrade compared to SwMC and MVGL, which is the traditional graph

TABLE 6
The clustering performance of the proposed SFMC compared to the state-of-the-art competitors on Mnist4 and Calthch101-20. The algorithms gain the stable results are marked by ‡.

Dataset	Mnist4						
Metric	ACC	NMI	Purity	PRE	REC	F_{score}	ARI
SFMC‡	**0.917 (±0.000)**	**0.801 (±0.000)**	**0.917 (±0.000)**	**0.846 (±0.000)**	**0.855 (±0.000)**	**0.852 (±0.000)**	**0.802 (±0.000)**
co-train	0.758 (±0.001)	0.554 (±0.001)	0.758 (±0.001)	0.634 (±0.001)	0.644 (±0.001)	0.639 (±0.001)	0.518 (±0.002)
co-reg	0.785 (±0.003)	0.602 (±0.001)	0.786 (±0.002)	0.670 (±0.002)	0.696 (±0.002)	0.682 (±0.001)	0.575 (±0.002)
SwMC‡	**0.914 (±0.000)**	**0.799 (±0.000)**	0.912 (±0.000)	**0.844 (±0.000)**	0.852 (±0.000)	**0.848 (±0.000)**	**0.799 (±0.000)**
MVGL‡	0.912 (±0.000)	0.785 (±0.000)	0.910 (±0.000)	0.841 (±0.000)	0.844 (±0.000)	0.842 (±0.000)	0.792 (±0.000)
MVSC	0.733 (±0.115)	0.651 (±0.069)	0.780 (±0.070)	0.650 (±0.092)	0.773 (±0.041)	0.704 (±0.066)	0.592 (±0.096)
RDEKM‡	0.885 (±0.000)	0.717 (±0.000)	0.885 (±0.000)	0.795 (±0.000)	0.804 (±0.000)	0.800 (±0.000)	0.733 (±0.000)
SMSC‡	0.913 (±0.000)	0.789 (±0.000)	**0.913 (±0.000)**	0.843 (±0.000)	0.850 (±0.000)	0.846 (±0.000)	0.795 (±0.000)
AMGL‡	0.910 (±0.000)	0.785 (±0.000)	0.910 (±0.000)	0.836 (±0.000)	0.843 (±0.000)	0.840 (±0.000)	0.786 (±0.000)
MLAN	0.744 (±0.001)	0.659 (±0.001)	0.744 (±0.000)	0.643 (±0.001)	**0.921 (±0.001)**	0.757 (±0.001)	0.656 (±0.001)
Dataset	Caltech101-20						
Metric	ACC	NMI	Purity	PRE	REC	F_{score}	ARI
SFMC‡	**0.642 (±0.000)**	**0.595 (±0.000)**	0.748 (±0.000)	0.586 (±0.000)	**0.677 (±0.000)**	**0.628 (±0.000)**	**0.461 (±0.000)**
co-train	0.397 (±0.020)	0.510 (±0.016)	0.737 (±0.015)	0.690 (±0.034)	0.233 (±0.015)	0.349 (±0.020)	0.290 (±0.022)
co-reg	0.412 (±0.006)	0.587 (±0.003)	**0.754 (±0.004)**	**0.712 (±0.008)**	0.243 (±0.004)	0.363 (±0.006)	0.295 (±0.025)
SwMC‡	0.599 (±0.000)	0.493 (±0.000)	0.700 (±0.000)	0.509 (±0.000)	0.625 (±0.000)	0.431 (±0.000)	0.265 (±0.000)
MVGL‡	**0.600 (±0.000)**	0.474 (±0.000)	0.696 (±0.000)	0.325 (±0.000)	**0.653 (±0.000)**	0.440 (±0.000)	0.282 (±0.000)
MVSC	0.591 (±0.054)	**0.613 (±0.040)**	0.717 (±0.033)	0.542 (±0.091)	0.546 (±0.059)	0.541 (±0.068)	0.451 (±0.088)
RDEKM‡	0.424 (±0.000)	0.572 (±0.000)	**0.768 (±0.000)**	**0.747 (±0.000)**	0.299 (±0.000)	0.427 (±0.000)	0.368 (±0.000)
SMSC‡	0.582 (±0.000)	0.590 (±0.000)	0.748 (±0.000)	0.701 (±0.000)	0.473 (±0.000)	**0.565 (±0.000)**	**0.485 (±0.000)**
AMGL	0.557 (±0.047)	0.552 (±0.061)	0.677 (±0.058)	0.480 (±0.093)	0.539 (±0.015)	0.503 (±0.054)	0.397 (±0.080)
MLAN	0.526 (±0.007)	0.474 (±0.003)	0.666 (±0.000)	0.279 (±0.003)	0.559 (±0.020)	0.372 (±0.007)	0.198 (±0.007)

TABLE 7
The clustering performance and CPU running time (sec.) on two large-scale datasets, and "OM" represents out-of-memory.

Dataset	NUS			
	ACC	NMI	Purity	CPU time
SFMC	0.1795	0.1912	0.2556	296.12
RDEKM	0.1259	0.1247	0.2262	8417.73
LMVSC	0.1553	0.1295	0.1982	165.39
MVSC	0.1531	0.1523	0.2415	230.26
co-reg	0.1317	0.1325	0.2411	52587.16
Dataset	Reuters			
	ACC	NMI	Purity	CPU time
SFMC	0.6023	0.3541	0.6042	320.47
RDEKM	OM	OM	OM	OM
LMVSC	0.5890	0.3346	0.6145	130.73
MVSC	0.5958	0.3472	0.5741	271.87
co-reg	0.5627	0.3261	0.5523	8035.26

based algorithms that elapsed time are intolerable when the data size is large-scaled;

- Sixth, among all of the stable algorithms, the matrix factorization based RDEKM is the most scalable with the data size. Nevertheless, RDEKM is extremely expensive to deal with the data having lots of features such as Calthech101-20 and MSRC-v5, since it decomposes the features of data thoroughly.
- Seventh, SMSC has the best performance against both of data size and feature size, due to its matrix factorization on the graph Laplacian matrix. Since the graph Laplacian matrix is much smaller than feature representation matrix when the feature size is large, SMSC overcomes the deficiency of RDEKM mentioned above.

Totally speaking, the proposed SFMC is the most efficient which achieves the best clustering performance at a comparable time cost among all of the competitors. It is highlighted that except for SFMC, all other algorithms involve the parameter-tuning, such as MVGL and MLAN having two important and sensitive parameters. In this circumstance, the time shrinkage of our algorithm is of great significance than other accelerated algorithms. Moreover, we further test these algorithms on tow large-scale datasets. Due to the limited memory space of PC, many approaches that take huge storage consumption encounter the out-of-memory issue, such as co-train, SwMC, MVGL, SMSC, AMGL and MLAN. Herein, we introduce another competitor LMVSC [8] which is a subspace clustering method that incorporates subspace learning into the small graph based spectral clustering, and the algorithm is ran in linear time. The number of anchors is set as $\lceil 0.02 * n \rceil$, and the results are shown in TABLE 7. From these results, we observe that i) on two datasets the proposed SFMC obtains the competitive clustering performance with taking a little more time cost than LMVSC and MVSC; ii) RDEKM and co-reg have poor performance in dealing with these large-scale data especially that RDEKM cannot be executed toward Reuters on account of the extremely high-dimensional features. Considering that there are extra hyper-parameters in LMVSC and MVSC, our SFMC is outstanding for its clustering ability, scalability, stableness as well as the concise parameter.

7 CONCLUSION

In this paper, we propose a bipartite graph based multi-view clustering method, providing a compact yet efficient framework that not only avoids the parameter-tuning actively, but also achieves the excellent cluster structure compatible across multiple views. Different from most existing approaches which either emphasis on improving clustering

performance and then encounter the high cost of time, or focusing on accelerating algorithms but degrading the cluster performance to some extent, the advantages of our model is threefold, i.e., scalability, stableness and parameter-free. As an anchor-based graph method, we also design a simple, and initialization-independent anchor selection strategy accordingly. Taking the same order of time magnitude as the random selection, this strategy definitely determine the anchors that well cover the entire point cloud and dramatically prompt the clustering efficiency. Extensive experiments on both synthetic datasets and real-world datasets verify that both of clustering ability and time cost of the proposed method outperforms the state-of-the-art competitors.

APPENDIX

.1 The optimization of problem (27)

Denote the Lagrange function of problem (27) as

$$\mathcal{L}_2(\boldsymbol{\alpha}, \phi, \boldsymbol{\pi}) = \frac{1}{2}\|\boldsymbol{\alpha} - \mathbf{e}\|_2^2 - \phi(\boldsymbol{\alpha}^T\mathbf{1} - 1) - \boldsymbol{\pi}^T\boldsymbol{\alpha}, \quad (34)$$

where $\mathbf{e} = \boldsymbol{\beta}^* - \frac{1}{\mu}(\boldsymbol{\eta} + \mathbf{A}\boldsymbol{\beta}^* - \mathbf{h})$, $\phi \in \mathbb{R}$ and $\boldsymbol{\pi} \geq \mathbf{0} \in \mathbb{R}^{n_V}$ are Lagrange multipliers. The first-order derivative of $\mathcal{L}_2(\boldsymbol{\alpha}, \phi, \boldsymbol{\pi})$ w.r.t. $\boldsymbol{\alpha}$ could be derived by $\partial\mathcal{L}_2(\boldsymbol{\alpha}, \phi, \boldsymbol{\pi})/\partial\boldsymbol{\alpha} = \boldsymbol{\alpha} - \mathbf{e} - \phi\mathbf{1} - \boldsymbol{\pi}$. According to the KKT conditions of problem (27), the optimal solution $\boldsymbol{\alpha}^*$ of problem (27) meets that

$$\begin{cases} \boldsymbol{\alpha}^* - \mathbf{e} - \phi^*\mathbf{1} - \boldsymbol{\pi}^* = \mathbf{0}; \\ \boldsymbol{\alpha}^* \geq \mathbf{0}, \boldsymbol{\alpha}^{*T}\mathbf{1} = 1; \\ \forall v, \pi_v^* \geq 0, \pi_v^*\alpha_v^* = 0, \end{cases} \quad (35)$$

where ϕ^* and $\boldsymbol{\pi}^*$ respectively denotes the optimal Lagrange multipliers of ϕ and $\boldsymbol{\pi}$ corresponding to $\boldsymbol{\alpha}^*$. α_v is v-th element of the vector $\boldsymbol{\alpha}$, and is equivalent to the notation $\alpha^{(v)}$ in the core objective function (4). From the first condition above, we further know $(\boldsymbol{\alpha}^* - \mathbf{e} - \phi^*\mathbf{1} - \boldsymbol{\pi}^*)^T\mathbf{1} = 0$. Based on this, ϕ^* could be achieved by

$$\phi^* = \frac{1}{n_V}(1 - \mathbf{e}^T\mathbf{1} - \boldsymbol{\pi}^{*T}\mathbf{1}). \quad (36)$$

Besides that, we also have $\boldsymbol{\alpha}^* = \mathbf{e} + \phi^*\mathbf{1} + \boldsymbol{\pi}^*$. Substitute the result in Eq. (36) for ϕ^* therein and we obtain

$$\boldsymbol{\alpha}^* = \mathbf{r} - \bar{\pi}^*\mathbf{1} + \boldsymbol{\pi}^*, \quad (37)$$

where $\mathbf{r} = \mathbf{e} - \frac{1}{n_V}\mathbf{e}^T\mathbf{1}\mathbf{1} + \frac{1}{n_V}\mathbf{1}$, and $\bar{v} = \sum_i v_i$ denotes the mean of any vector $\mathbf{v}$. Take the condition of $\forall t, \pi_v \geq 0$ into consideration, we could update $\boldsymbol{\alpha}$ by

$$\alpha_v^* = (r_v - \bar{\pi}^*)_+ \quad (38)$$

where $(m)_+ = \max(m, 0)$, and $r_v = e_v - \bar{e} + \frac{1}{n_V}$. Apart from this, we still need to determine $\bar{\pi}^*$. By virtue of Eq. (37), it can be inferred that $\pi_v^* = \alpha_v^* + \bar{\pi}^* - r_v$. Since $\forall v, \alpha_v \geq 0$, we have

$$\pi_v^* = (\bar{\pi}^* - r_v)_+. \quad (39)$$

So, $\bar{\pi}^* = \frac{1}{n_V}\sum_{v=1}^{n_V}(\bar{\pi}^* - r_v)_+$. Define $\psi(\bar{\pi}^*) = \frac{1}{n_V}\sum_{v=1}^{n_V}(\bar{\pi}^* - r_v)_+ - \bar{\pi}^*$. Considering r_v involves the fixed $\boldsymbol{\beta}^*$ and given constants $\mathbf{A}$ and $\mathbf{h}$, we can update $\bar{\pi}^*$ iteratively to achieve the optimal $\bar{\pi}^*$ by virtue of Newton method [57], i.e.,

$$\bar{\pi}_{k+1}^* = \bar{\pi}_k^* - \frac{\psi(\bar{\pi}^*)}{\psi'(\bar{\pi}^*)}, \quad (40)$$

where $\bar{\pi}_k$ is the value of $\bar{\pi}$ updated in k-th iteration.

REFERENCES

[1] C. Wang, M. Pelillo, and K. Siddiqi, "Dominant set clustering and pooling for multi-view 3d object recognition," *arXiv preprint arXiv:1906.01592*, 2019.

[2] L. Zhao, Q. Hu, and W. Wang, "Heterogeneous feature selection with multi-modal deep neural networks and sparse group lass," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1936–1948, 2015.

[3] X. Cao, C. Zhang, C. Zhou, H. Fu, and H. Foroosh, "Constrained multi-view video face clustering," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4381–4393, 2015.

[4] X. Cao, C. Zhang, H. Fu, S. Liu, and H. Zhang, "Diversity-induced multi-view subspace clustering," in *Computer Vision and Pattern Recognition*, 2015, pp. 586–594.

[5] Q. Yin, S. Wu, R. He, and L. Wang, "Multi-view clustering via pairwise sparse subspace representation," *Neurocomputing*, vol. 156, pp. 12–21, 2015.

[6] C. Zhang, H. Fu, Q. Hu, X. Cao, Y. Xie, D. Tao, and D. Xu, "Generalized latent multi-view subspace clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[7] Q. Yin, S. Wu, and L. Wang, "Multiview clustering via unified and view-specific embeddings learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5541–5553, 2018.

[8] Z. Kang, W. Zhou, Z. Zhao, J. Shao, M. Han, and Z. Xu, "Large-scale multi-view subspace clustering in linear time," in *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

[9] Z. Kang, X. Zhao, C. Peng, H. Zhu, J. T. Zhou, X. Peng, W. Chen, and Z. Xu, "Partition level multiview subspace clustering," *Neural Networks*, vol. 122, pp. 279–288, 2020.

[10] J. Liu, C. Wang, J. Gao, and J. Han, "Multi-view clustering via joint nonnegative matrix factorization," in *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 2013, pp. 252–260.

[11] L. Zong, X. Zhang, L. Zhao, H. Yu, and Q. Zhao, "Multi-view clustering via multi-manifold regularized non-negative matrix factorization," *Neural Networks*, vol. 88, pp. 74–89, 2017.

[12] J. Xu, J. Han, F. Nie, and X. Li, "Re-weighted discriminatively embedded k-means for multi-view clustering," *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society*, vol. 26, no. 6, pp. 3016–3027, 2017.

[13] H. Zhao, Z. Ding, and Y. Fu, "Multi-view clustering via deep matrix factorization," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[14] S. Bickel and T. Scheffer, "Multi-view clustering," in *IEEE International Conference on Data Mining*, 2004, pp. 19–26.

[15] D. Zhou and C. J. Burges, "Spectral clustering and transductive learning with multiple views," in *Proceedings of the 24th International Conference on Machine learning*. ACM, 2007, pp. 1159–1166.

[16] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Advances in Neural Information Processing Systems*, 2011, pp. 1413–1421.

[17] Y. Wang, L. Wu, X. Lin, and J. Gao, "Multiview spectral clustering via structured low-rank matrix factorization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4833–4843, 2018.

[18] R. Xia, Y. Pan, L. Du, and J. Yin, "Robust multi-view spectral clustering via low-rank and sparse decomposition," in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[19] C. Lu, S. Yan, and Z. Lin, "Convex sparse spectral clustering: Single-view to multi-view," *IEEE Transactions on Image Processing*, vol. 25, no. 6, pp. 2833–2843, 2016.

[20] X. Zhu, S. Zhang, W. He, R. Hu, C. Lei, and P. Zhu, "One-step multi-view spectral clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 10, pp. 2022–2034, 2018.

[21] Z. Hu, F. Nie, R. Wang, and X. Li, "Multi-view spectral clustering via integrating nonnegative embedding and spectral embedding," *Information Fusion*, vol. 55, pp. 251–259, 2020.

[22] F. Nie, J. Li, and X. Li, "Parameter-free auto-weighted multiple graph learning: a framework for multiview clustering and semi-supervised classification," in *International Joint Conference on Artificial Intelligence*, 2016, pp. 1881–1887.

[23] F. Nie, G. Cai, J. Li, and X. Li, "Auto-weighted multi-view learning for image clustering and semi-supervised classification," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1501–1511, 2017.

[24] Z. Kang, G. Shi, S. Huang, W. Chen, X. Pu, J. T. Zhou, and Z. Xu, "Multi-graph fusion for multi-view spectral clustering," *Knowledge-Based Systems*, vol. 189, p. 105102, 2020.

[25] J. Shi and J. Malik, "Normalized cuts and image segmentation," *Departmental Papers (CIS)*, p. 107, 2000.
[26] Y. Li, F. Nie, H. Huang, and J. Huang, "Large-scale multi-view spectral clustering via bipartite graph," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, pp. 2750–2756.
[27] L. Feng, L. Cai, Y. Liu, and S. Liu, "Multi-view spectral clustering via robust local subspace learning," *Soft Computing*, vol. 21, no. 8, pp. 1937–1948, 2017.
[28] Z. Zhang, L. Liu, F. Shen, H. T. Shen, and L. Shao, "Binary multi-view clustering." *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 2018.
[29] Z. Kang, H. Pan, S. C. Hoi, and Z. Xu, "Robust graph learning from noisy data," *IEEE Transactions on Cybernetics*, 2019.
[30] Z. Kang, L. Wen, W. Chen, and Z. Xu, "Low-rank kernel learning for graph-based clustering," *Knowledge-Based Systems*, vol. 163, pp. 510–517, 2019.
[31] F. Nie, X. Wang, M. I. Jordan, and H. Huang, "The constrained laplacian rank algorithm for graph-based clustering," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1969–1976.
[32] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: analysis and an algorithm," *Advances in Neural Information Processing Systems*, vol. 14, pp. 849–856, 2001.
[33] S. X. Yu and J. Shi, "Multiclass spectral clustering," in *IEEE International Conference on Computer Vision*, 2003, p. 313.
[34] L. Zelnik-Manor, "Self-tuning spectral clustering," *Advances in Neural Information Processing Systems*, vol. 17, pp. 1601–1608, 2004.
[35] A. Kumar and H. D. Iii, "A co-training approach for multi-view spectral clustering," in *International Conference on International Conference on Machine Learning*, 2011, pp. 393–400.
[36] F. Nie, J. Li, and X. Li, "Self-weighted multiview clustering with multiple graphs," in *International Joint Conference on Artificial Intelligence*, 2017, pp. 2564–2570.
[37] K. Zhan, C. Zhang, J. Guan, and J. Wang, "Graph learning for multiview clustering," *IEEE Transactions on Cybernetics*, vol. 48, no. 10, pp. 2887–2895, 2017.
[38] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 679–686.
[39] D. Cai and X. Chen, "Large scale spectral clustering via landmark-based sparse representation," *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1669–1680, 2014.
[40] M. Wang, W. Fu, S. Hao, D. Tao, and X. Wu, "Scalable semi-supervised learning by efficient anchor graph regularization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1864–1877, 2016.
[41] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of Optimization Theory and Applications*, vol. 109, no. 3, pp. 475–494, 2001.
[42] K. Fan, "On a theorem of weyl concerning eigenvalues of linear transformations i," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 35, no. 11, p. 652, 1949.
[43] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*. AP, 1982.
[44] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.
[45] Z. Lin, R. Liu, and Z. Su, "Linearized alternating direction method with adaptive penalty for low-rank representation," in *Advances in Neural Information Processing Systems*, 2011, pp. 612–620.
[46] F. Du, Q.-Y. Dong, and H.-S. Li, "Truss structure optimization with subset simulation and augmented lagrangian multiplier method," *Algorithms*, vol. 10, no. 4, p. 128, 2017.
[47] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern recognition letters*, vol. 31, no. 8, pp. 651–666, 2010.
[48] D. Cai and X. Chen, "Large scale spectral clustering via landmark-based sparse representation." *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1669–1680, 2015.
[49] J. Winn and N. Jojic, "Locus: Learning object classes with unsupervised segmentation," in *Tenth IEEE International Conference on Computer Vision*, 2005, pp. 756–763.
[50] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[51] F. Li, F. Rob, and P. Peitro, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," *Computer Vision & Image Understanding*, vol. 106, no. 1, pp. 59–70, 2005.
[52] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
[53] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng, "Nus-wide: A real-world web image database from national university of singapore," in *Proc. of ACM Conf. on Image and Video Retrieval (CIVR'09)*, Santorini, Greece., July 8-10, 2009.
[54] C. Apté, F. Damerau, and S. M. Weiss, "Automated learning of decision rules for text categorization," *ACM Transactions on Information Systems (TOIS)*, vol. 12, no. 3, pp. 233–251, 1994.
[55] A. Strehl and J. Ghosh, "Cluster ensembles – a knowledge reuse framework for combining multiple partitions," *Journal on Machine Learning Research (JMLR)*, vol. 3, pp. 583–617, 2002.
[56] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics*, vol. 2, no. 1-2, pp. 83–97, 2010.
[57] A. H. Sherman, "On newton-iterative methods for the solution of systems of nonlinear equations," *Siam Journal on Numerical Analysis*, vol. 15, no. 4, pp. 755–771, 1978.

Feiping Nie received the PhD degree in computer science from Tsinghua University, China, in 2009. He is currently a full professor at Northwestern Polytechnical University, Xi'an, China. His research interests are machine learning and its application fields, such as pattern recognition, data mining, computer vision, image processing, and information retrieval. He has published more than 100 papers in the prestigious journals and conferences like TIP, TPAMI, TNNLS, TKDE, ICML, NIPS, KDD, etc. He is currently serving as an associate editor or a PC member for several prestigious journals and conferences in the related fields.

Han Zhang received the B.S. degree in Computer Science and Technology from Northwest University, Xi'an, Shaanxi, China, in 2016. She is currently working toward the Ph.D. degree under the supervision of both Prof. Xuelong Li and Prof. Feiping Nie in the School of Computer Science and the Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, Shaanxi, China. Her current research interests include machine learning.

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 Cybersecurity and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an, China. His research interests focus on machine learning and its applications.

Xuelong Li (M'02-SM'07-F'12) is a full professor with School of Computer Science and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, P.R. China.