Graph Structure Fusion for Multiview Clustering

Kun Zhan, Chaoxi Niu, Changlu Chen, Feiping Nie, Changqing Zhang, Yi Yang

Abstract—Many existing multiview clustering methods take graphs, which are usually pre-computed independently in each view, as input to uncover data distribution. These methods ignore the correlation of graph structure among multiple views, and the clustering results highly depend on the quality of predefined affinity graphs. We address the problem of multiview clustering by seamlessly integrating the graph structures of different views to fully exploit the geometric property of underlying data structure. The proposed method is based on the assumption that the intrinsic underlying graph structure would assign corresponding connected component in each graph to the same cluster. Different graphs from multiple views are integrated by using Hadamard product since different views usually together admit the same underlying structure across multiple views. Specifically, the graphs are integrated into a global one and the structure of the global graph is adaptively tuned by a well-designed objective function so that the number of components of the graph is exactly equal to the number of clusters. It is worth noting that we directly obtain cluster indicators from the graph itself without performing further graph-cut or $k$-means clustering steps. Experiments show the proposed method obtains better clustering performance than the state-of-the-art methods.

Index Terms—unsupervised learning, multiview clustering, feature extraction, multiview learning, structured graph.

1 INTRODUCTION

Since instances are generally associated with multiple types of features, multiview data are pervasive in many domains. For example, a person can be identified by face, fingerprint, iris, and signature; one news can be reported by multiple articles in different languages; and an image can be described in different features: SIFT [1], GIST [2], and HOG [3]. As the increasing popularity of multiview data, it is important to leverage multiview data to obtain more accurate clustering results than using any single view data [4]–[9].

Data clustering is a fundamental topic of unsupervised learning. Unsupervised multiview learning has emerged to be an important learning paradigm due to its ability to handle multi-feature data. Many unsupervised multiview learning methods are proposed to make the best utilization of the diversity of different views. In these methods, graphs are usually constructed to integrate heterogeneous features according to different types of descriptions. A graph is usually formed to reveal the relationship between data points, and the graph structure is also encoded by the affinity matrix [10]–[13]. Performance of existing graph-based methods is largely determined by the quality of the predefined graph. They focus on how to improve the similarity between data points and how to balance the weights of different views but ignore the geometric property of the graph structure.

In this paper, we aim to learn an intrinsic graph structure and to obtain clustering results simultaneously by exploring the geometric property of graph structure of each view. Since the different views usually admit the same underlying cluster structure, it can be seen from Fig. 1 that different graphs constructed from multiple views are integrated into an intrinsic graph $A$.

![Figure 1. Schematic diagram of graph structure fusion.](image)

The graph $W^{(v)}$ is constructed from the $v$-th view, and different graphs $W^{(v)}$, $v \in \{1, 2, 3\}$, are integrated into an intrinsic graph $A$.
in each component belong to one cluster so that we can obtain the clustering results by the graph itself directly. In practice, we use the Hadamard product, i.e., element-wise product, to preserve the common edges in multiple graphs as shown in Fig. 1. For real-world datasets, since it is difficult to obtain a fused graph with \( n_c \) connected components where \( n_c \) is the number of clusters, we design an objective function to learn an intrinsic graph so that the number of components is equal to the number of clusters exactly. Since vertices in each component belong to one cluster, the cluster indicators are obtained by the learned graph itself without performing further graph-cut or \( k \)-means clustering algorithms. We proposed an unsupervised multiview learning method based on graph structure fusion (GSF) and an efficient algorithm is presented to tackle the optimization with proved convergence. Experiments are conducted on three different datasets to show the effectiveness of the proposed GSF. GSF is compared with state-of-the-art methods on multiview clustering. The clustering results are measured in terms of seven performance metrics.

The contributions of GSF are summarized in the following three aspects:

1) The intrinsic graph structure is obtained by an efficient operator from multiple graphs. Different from most feature level fusion methods, an intrinsic graph structure is fused from multiple views directly by exploiting the geometric property rather than using a simple weighted-sum rule.

2) GSF combines similarity learning and clustering structure learning into a unified framework for the consideration of the intrinsic graph. Different from most existing two-stage methods, the graph and clustering result are learned simultaneously at one stage.

3) Since the number of connected components of the learned intrinsic graph is the same to the cluster number exactly, the clustering indicators can be directly obtained from the learned graph without any post-processing steps.

The remainder of this paper is organized as follows. In Section 2, we introduce some related works. In Section 3, an objective function is proposed for optimizing a global graph constrained by the rank of Laplacian matrix. In Section 4, we propose a novel algorithm to optimize the well-designed objective functions. In Section 5, numerical experiments are conducted. We use three datasets and compare with seven state-of-the-art methods. Section 6 concludes with some discussion.

2 Related Work

Generally, multiview learning methods are categorized into three classes: co-training, subspace learning, and multiple kernel learning. Numbers of multiview learning methods are proposed to maximize the mutual agreement across different views and to leverage the complementary principle [14], [15].

At the beginning, co-training learning model is applied to supervised multiview learning [4]. Inspired by supervised co-training learning [4], two unsupervised multiview learning methods are proposed [16], [17]. Later, to exploit the complementary information, some methods also use unsupervised co-training learning strategies [18]–[23]. These methods search for the clusters agreement across all the views and minimize the disagreement in different views.

Part of these co-training multiview learning algorithms also uses subspace learning to learn a latent subspace for data representation [20]–[23]. Similarly, unsupervised subspace learning is applied to multiview clustering to obtain a latent subspace shared by different features [5], [6], [24]–[27]. Usually, they need to perform postprocessing spectral clustering on the learned graph to obtain data labels.

Besides co-training learning and subspace learning, multi-kernel methods are usually used to process multiview features [28]–[31]. In multiple kernel learning, a set of kernels are constructed in different views to find a way to combine them [28]–[31].

In the learned data space, most of these methods need to perform postprocessing graph-cut or \( k \)-means clustering algorithms for obtaining the clustering indicators. Due to the well-defined mathematical framework of spectral clustering [32]–[34], these methods are usually designed by using spectral clustering or perform postprocessing spectral clustering in the learned subspace. These methods involve two stages in which graph or kernel is constructed beforehand, then the cluster labels are obtained by learning algorithms. A disadvantage of this two-stage process is that the final clustering structures are not represented explicitly in the graph so that they need to perform \( k \)-means clustering as a post-processing to obtain the clustering indicators.

Recently, some clustering methods are proposed to learn an adaptive graph by using a rank constraint on the Laplacian matrix [35]–[37], and many graph-based methods use a weight-sum rule to extend these single view learning to multiview learning [38]–[44]. Instead of using the weighted-sum rule, we exploit the graph structure and use Hadamard product to integrate the multiview information.

3 Adaptive Structure Clustering

3.1 Graph Structure Fusion

Assume that the data matrix is represented by \( X = [X_1, X_2, \ldots, X_n_c] = [x_{11}, x_{12}, \ldots, x_{n_c}] \in \mathbb{R}^{d \times n} \), where \( X_c \) denotes the data matrix belonging to the \( c \)-th cluster, \( x_i \) denotes a data point, \( n \) is the number of data points, \( d \) is the dimension, and \( n_c \) is the number of clusters.

In graph-based multiview learning models, data points are usually represented as vertices in an undirected graph \( W \) with weighted edges depicting the pairwise similarity of data points. For two data points \( x_i \) and \( x_j \), their similarity \( w_{ij} \) is usually assigned to a large
weight when their distance \( \| x_i - x_j \|_2^2 \) is small. If we assign \( k \) neighbors to the \( j \)-th data points, the \( j \)-th row \( w_j \) of \( W \) has \( k \) number of nonzero elements, i.e., there are \( k \) number of vertices connecting to the \( j \)-th vertex in the graph.

With multiview data, different affinity matrices \( W^{(v)} (v \in [1, n_v]) \) are constructed by a \( k \)-nearest-neighbor (\( k \)-NN) graph in this paper, where \( n_v \) is the number of views and \( v \) is the view index. In multiple graphs, different similarity matrices are well captured over the same set of vertices. Our purpose is that different graphs are integrated into a comprehensive graph over the same set of vertices. Our purpose is to learn a graph with exact number of views and \( c \) number of connected components.

3.2 Intrinsic Graph Learning

We have three observations for graph structure fusion:

1) A relatively large \( k \) has a large probability resulting in only one connected component in a graph.
2) Different \( W^{(v)} \) corresponds to a different variant of the intrinsic graph, and the connected edges between two vertices may be different from each other in multiple graphs.
3) The common edges in multiple graphs are the edges of the intrinsic structure.

To protect the intrinsic structure shared by different views, we use the Hadamard product to preserve the common edges in multiple graphs. We use the Hadamard product to fuse all the affinity matrices \( W^{(v)} \) into one affinity graph \( A \) by

\[
A = \prod_{v=1}^{n_v} W^{(v)}.
\]

Here, \( \prod \) denotes the Hadamard product of a sequence.

As shown in Fig. 1, the common edges in different graphs are preserved in the graph \( A \) and the connected components change to three. If we sum up these graphs element-wisely, the graph \( A \) has only one connected component. However, by multiplying them element-wisely, an intrinsic structure is obtained in Fig. 1. It can be seen from Fig. 1 that the graph \( A \) has exactly three connected components.

Our purpose is to learn a graph with exact \( n_c \) number of connected components for real-world datasets. It is straightforward to check that a graph fused by the Hadamard product can hardly satisfy such a constraint condition in practice. In the following sub-section, we show how to obtain such an intrinsic graph \( S \) with adaptive structure learning.

3.2 Intrinsic Graph Learning

Given a fused affinity matrix \( A \), we learn a similarity matrix \( S \) so that the corresponding graph has exactly \( n_c \) connected components. If the similarity matrix \( S \geq 0 \), whose Laplacian matrix \( L \) [45] satisfies to a theorem [46]:

**Theorem 1:** The number \( n_c \) of connected components of the graph \( S \) is equal to the multiplicity of zero as an eigenvalue of its Laplacian matrix \( L \).

For any vector \( f \in \mathbb{R}^n \), we have \( f^T L f = \frac{1}{2} \sum_{i,j=1}^{n} s_{ij} (f_i - f_j)^2 \geq 0 \), then \( L \) is a positive semi-definite matrix. Because of the positive semi-definite, \( L \) has \( n \) non-negative eigenvalues \( 0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n \). Theorem 1 indicates that if the constraint \( \sum_{c=1}^{n_c} \lambda_c = 0 \) is satisfied, the graph \( S \) has an ideal neighbors assignment and the data points are already clustered into \( n_c \) clusters. According to Fan’s theorem [47], we can obtain an objective function,

\[
\sum_{c=1}^{n_c} \lambda_c = \min \quad \langle U U^T, L \rangle
\]

\[
s.t. \quad U \in \mathbb{R}^{n \times n_c}, U^T U = I
\]

where \( \langle \cdot, \cdot \rangle \) denotes the Frobenius inner product of two matrices, \( \text{Tr}(\cdot) \) denotes the trace operator, \( U^T = [u_1, u_2, \ldots, u_{n_c}] \), \( L = D - S \) is the Laplacian matrix, \( I \in \mathbb{R}^{n_c \times n_c} \) is an identity matrix, \( D \) is a diagonal matrix and its elements are column sums of \( S^{-1} + S \).

The proof of Fan’s Theorem can be referred to [48] and [49]. Theorem 1 indicates that the component number of the graph \( S \) is exact the cluster number and each component corresponds to one cluster if the objective Eq. (2) is equal to zero. Then, we can tune adaptively the graph structure with an overall objective function to learn such a graph \( S \). \( \sum_{c=1}^{n_c} \lambda_c = 0 \) is consistent with the rank constraint \( \text{rank}(L) = n - n_c \), which means that the graph structure is tuned adaptively until the condition, \( \text{rank}(L) = n - n_c \), is satisfied [35]–[37].

Because the graph \( A \) contains edges of the intrinsic structure, we need \( S \) best approximating \( A \), then we optimize the following objective function,

\[
\max_S \quad \langle A, S \rangle
\]

\[
s.t. \quad \forall j, s_j \geq 0, 1^T s_j = 1
\]

where we constrain \( 1^T s_j = 1 \) so that \( D = I \).

By combining Eq. (2) with Eq. (3), we have,

\[
\min_{U,S} \quad \langle U U^T, L \rangle - \gamma_1 \langle A, S \rangle
\]

\[
s.t. \quad U \in \mathbb{R}^{n \times n_c}, U^T U = I,
\]

\[
\forall j, s_j \geq 0, 1^T s_j = 1
\]

where \( \gamma_1 \) is the trade-off parameter.

Since the constraint \( 1^T s_j = 1 \), \( L \) is a normalized Laplacian matrix, i.e., \( L = I - S \). Then, we have,

\[
\langle U U^T, L \rangle - \gamma_1 \langle A, S \rangle = \langle U U^T, I - S \rangle - \gamma_1 \langle A, S \rangle = \langle U U^T, I \rangle - \langle U U^T + \gamma_1 A, S \rangle.
\]

The first term of Eq. (5b), \( \langle U U^T, I \rangle \), is treated as a constant when optimizing \( S \). Considering Eq. (5), solving Eq. (4) with respect to \( S \) is the same to optimizing the following function,

\[
\max_S \quad \langle G, S \rangle
\]

\[
s.t. \quad \forall j, s_j \geq 0, 1^T s_j = 1
\]
where $G$ denotes the matrix $[UU^\top + \gamma_1 A]$. It is straightforward to check that $\langle G, S \rangle = \sum_{j=1}^n s_j g_j$ where $g_j$ denotes the $j$-th column of $G$ and constraints of different columns are independent of each other in Eq. (6). Then, solving Eq. (6) with respect to $s_j$ is equal to optimizing the following function,

$$\begin{align*}
\max_s & \quad s_j^\top g_j \\
\text{s.t.} & \quad s_j \geq 0, 1^\top s_j = 1.
\end{align*}$$

(7)

Optimizing Eq. (7) has trivial solution with respect to $s_j$, i.e., only one element is assigned to a value. The objective function value of Eq. (7) returns the maximum value $g_{ij} = \max(g_j)$, i.e., the $i$-th element of $s_j$ is assigned to one, and other elements of $s_j$ are zeroed. To avoid this case, we add $\ell_2$-norm regularization to smooth the elements of $S$. Then, we have,

$$\begin{align*}
\min_{U,S} & \quad \text{Tr}(U^\top LU) - \gamma_1 \text{Tr}(AS^\top) + \gamma_2 \|S\|_F^2 \\
\text{s.t.} & \quad U \in \mathbb{R}^{n \times n_c}, U^\top U = I, \\
& \quad \forall j, s_j \geq 0, 1^\top s_j = 1
\end{align*}$$

(8)

where $\gamma_1$ and $\gamma_2$ are two regularization parameters.

In the most existing graph-based methods, a predefined graph $S$ is generally varied by varying the graph construction methods, so it is hard to obtain the constraint $\sum_{c=1}^{n_c} \lambda_c = 0$. We tune the structure of $S$ adaptively so that we achieve the condition $\sum_{c=1}^{n_c} \lambda_c = 0$. Eq. (8) learns to obtain a graph $S$ with exactly $n_c$ number of connected components. Then, the cluster indicators are obtained by the learned $S$ without performing further graph-cut and $k$-means clustering algorithms since each component corresponds to one cluster.

As opposed to pre-computing the weights of the affinity graphs, in Eq. (8) the affinity weights of the adaptive graph $S$, i.e., $s_{ij}$, is learned by modeling fused graph $A$ from multiple views, making the learning procedures of multiple views mutually beneficial and reciprocal. By minimizing Eq. (8), the affinity graph $S$ and $U$ are learned simultaneously.

In next section, we propose a novel algorithm to optimize the objective function Eq. (8).

4 Optimization

The problem can be divided into two subproblems and alternately solved as shown in the following.

4.1 Algorithm Derivation

The first subproblem is to fix $U$, updating $S$. Then, Eq. (8) becomes

$$\begin{align*}
\min_S & \quad \text{Tr}(U^\top LU) - \gamma_1 \text{Tr}(AS^\top) + \gamma_2 \|S\|_F^2 \\
\text{s.t.} & \quad \forall j, s_j \geq 0, 1^\top s_j = 1.
\end{align*}$$

(9)

Note that the problem Eq. (9) is independent between different $j$, then we have,

$$\begin{align*}
\min_{s_j} & \quad \sum_{i=1}^n (\|u_i - u_j\|_2^2 - \gamma_1 a_{ij}) s_{ij} + \gamma_2 s_j^\top s_j \\
\text{s.t.} & \quad s_j \geq 0, 1^\top s_j = 1.
\end{align*}$$

(10)

Denoting $p_j$ as a vector with the $i$-th element equal to $p_{ij} = \|u_i - u_j\|_2^2 - \gamma_1 a_{ij}$, then optimizing Eq. (10) is equal to optimizing the following objective function,

$$\begin{align*}
\min_{s_j} & \quad \frac{1}{2} \left\| s_j + \frac{p_j}{2\gamma_2} \right\|_2^2 \\
\text{s.t.} & \quad s_j \geq 0, 1^\top s_j = 1.
\end{align*}$$

(11)

Since Eq. (11) is a Euclidean projection problem on the simplex space, there are many algorithms for solving it [50]–[52].

The Lagrangian function of Eq. (11) is

$$L(s_j, \eta, \rho) = \frac{1}{2} \left\| s_j + \frac{p_j}{2\gamma_2} \right\|_2^2 - \eta(1^\top s_j - 1) - \rho^\top s_j$$

(12)

where $\eta$ and $\rho$ are the Lagrangian multipliers.

According to the Karush-Kuhn-Tucker condition [50], it can be verified that the optimal solution $s_j^*$ is

$$s_j^* = \left( -\frac{p_j}{2\gamma_2} + \eta 1 \right)_{+}$$

(13)

The second subproblem is to fix $S$, updating $U$. Then, Eq. (8) becomes

$$\min_U \quad \text{Tr}(U^\top LU)$$

$$\begin{align*}
\text{s.t.} & \quad U \in \mathbb{R}^{n \times n_c}, U^\top U = I.
\end{align*}$$

(14)

The optimal $U$ for Eq. (14) is formed by the $n_c$ eigenvectors corresponding to the top $n_c$ smallest eigenvalues of the normalization Laplacian matrix $L$.

The detailed algorithm is summarized in Algorithm 1. According to Theorem 1 and Fan’s Theorem, the stopping condition of the algorithm is that the sum of the top $n_c$ eigenvalues of the Laplacian matrix is equal to zero [35]–[37].

4.2 Initial Graph Learning

In order to obtain a similar graph, the initial graph is learned by referring to Eq. (11). Using the raw data points represent a pairwise distance $b_{ij} = \|x_i - x_j\|_2^2$, we learn the initial graph by using the following objective function,

$$\begin{align*}
\min_{w_j} & \quad w_j^\top b_j + \gamma_j w_j^\top w_j \\
\text{s.t.} & \quad w_j \geq 0, 1^\top w_j = 1
\end{align*}$$

(15)

where $\gamma_j$ is a trade-off parameter that controls the neighbors assignment to the one data point $x_j$.

In the following, we show how to obtain a $k$-NN graph [36] with Eq. (15). Eq. (15) can be divided into two extreme graph learning cases.
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2872061, IEEE Transactions on Knowledge and Data Engineering

Algorithm 1: GSF-based multiview clustering.

input : Dataset $\mathcal{X} = \{X^{(1)}, X^{(2)}, \ldots, X^{(m)}\}$, the cluster number $n_c$, parameters $\gamma_1$ and $\gamma_2$.
output : $S$ with exactly $n_c$ connected components.
initialize: $W^{(v)}, \forall v \in \{1, 2, \ldots, m\}$, is constructed from the data matrix $X^{(v)}$ by a $k$-NN graph algorithm, $A$ is calculated by Eq. (1), and $U$ is formed by $n_c$ number of eigenvectors corresponding to the top $n_c$ smallest eigenvalues of the Laplacian matrix of $A$.

repeat
    for $j \in \{1, 2, \ldots, n\}$ do
        Update $s_j$ by using Eq. (13);
    end
    $S = S + S^\top$;
    Update $U$ by Eq. (14), i.e., $U$ is formed by $n_c$ eigenvectors with the top $n_c$ smallest eigenvalues of $L$;
until $S$ has $n_c$ connected components;

The first case is given by,
\[
\min_{s_j} b_j^\top w_j \\
\text{s.t. } w_j \geq 0, 1^\top w_j = 1. 
\tag{16}
\]

The solution of Eq. (16) is that one data point $x_j$ connects to only one other data point $x_i$ with weight $w_{ij}^* = 1$, and other weights are zeroed.

The second case is given by
\[
\min_{w_j} w_j^\top w_j \\
\text{s.t. } w_j \geq 0, 1^\top w_j = 1. 
\tag{17}
\]

The solution of Eq. (17) is that $x_j$ connects to all data points with weights $w_{ij}^* = \frac{1}{n}, \forall i$.

The initial graph is a tradeoff between these two extreme graphs. The same as Eq. (13), the solution of Eq. (15) is given by
\[
w_j^* = \left(-\frac{b_j}{2\gamma_j} + \eta 1\right)_+ . 
\tag{18}
\]

Without loss of generality, suppose that $b_{1j}, b_{2j}, \ldots, b_{nj}$ are ordered from small to large. When assigning $k$ neighbors to $x_j$, the optimal $w_j$ has $k$ nonzero elements, then according to Eq. (18), we have $w_{kj} > 0$ and $w_{k+1,j} = 0$. According to the constraint $1^\top w_j = 1$, we have
\[
\sum_{i=1}^{k} \left(-\frac{b_{ij}}{2\gamma_j} + \eta\right) = 1. 
\tag{19}
\]

Then,
\[
\eta = \frac{2\gamma_j + \sum_{i=1}^{k} b_{ij}}{2k\gamma_j}. 
\tag{20}
\]

Combining Eqs. (18) and (20) and considering $w_{kj} > 0$ and $w_{k+1,j} = 0$, we have
\[
\begin{align*}
\gamma_j &> \frac{k^2 b_{kj} - \frac{1}{2} \sum_{i=1}^{k} b_{ij}}{2k} ; \\
\gamma_j &\leq \frac{k b_{kj} - \frac{1}{2} \sum_{i=1}^{k} b_{ij}}{k}. 
\end{align*} 
\tag{21}
\]

In order to satisfy to conditions in Eq. (21), we can set
\[
\gamma_j = \frac{k}{2} b_{kj} - \frac{1}{2} \sum_{i=1}^{k} b_{ij}. 
\tag{22}
\]

Combining Eqs. (19), (20), and (22), we have
\[
\begin{align*}
w_{ij}^* &= \begin{cases} 
\frac{k b_{kj} - \sum_{i=1}^{k} b_{ij}}{kb_{kj} - \sum_{i=1}^{k} b_{ij}}, & \text{if } i \leq k; \\
0, & \text{otherwise.} 
\end{cases} 
\end{align*} 
\tag{23}
\]

Therefore, we can assign $k$ neighbors to each data point with initialing $\gamma_j$ by Eq. (22).

4.3 Convergence analysis

The original problem Eq. (8) is not a joint convex problem of $U$ and $S$. Hence, we may not obtain a global solution. We divide the original problem into two subproblems, i.e., Eq. (14) and Eq. (11). Since $L$ in Eq. (14) is positive semi-definite and Eq. (11) is a constrained quadratic minimization, each of them is the convex problem. The two subproblems are solved alternatively, so GSF will converge to a local solution. In the following section, we will show the convergence curves.

4.4 Computational analysis

The complexity of the proposed GSF mainly consists of two parts: 1) Updating $S$, and 2) Calculating the $n_c$ eigenvectors of $L$. These two parts are repeated until meeting the convergence condition, and they take $O(n^2)$ and $O(n_c n^2)$ respectively. Thus, the total complexity of Eq. (8) is
\[
O((1 + n_c)n^2 t) 
\tag{24}
\]
where $t$ is the total number of iterations.

5 Experiment

In this section, we present experiments to show the effectiveness of GSF.

5.1 Experimental Setup

We use three datasets in this paper, including:

Caltech-101: The dataset [53] consists of 101 categories with 8677 images. The widely used seven classes and 1474 images are selected. Six features are extracted from these images. They are Gabor with dimension 48, wavelet-moment with dimension 40, CENTRIST with dimension 254, HOG with dimension 1948, GIST with dimension 512, and LBP with dimension 982.

ORL: The dataset [54] contains 400 images of 40 distinct subjects and each subject has 10 different images. Three features are extracted: intensity with dimension 1474 images are selected. Six features are extracted from these images. They are Gabor with dimension 48, wavelet-moment with dimension 40, CENTRIST with dimension 254, HOG with dimension 1948, GIST with dimension 512, and LBP with dimension 982.
4096, LBP with dimension 3304, and Gabor with dimension 6750.

COIL-20: The dataset [55] is from the Columbia object image library and contains 1440 images of 20 objects. Each class contains 72 images. Three features are extracted: intensity with dimension 1024, LBP with dimension 3304, and Gabor with dimension 6750.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Caltech-101</th>
<th>ORL</th>
<th>COIL-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>View 1</td>
<td>Gabor (48)</td>
<td>Intensity (4096)</td>
<td>Intensity (1024)</td>
</tr>
<tr>
<td>View 2</td>
<td>Wavelet (40)</td>
<td>LBP (3304)</td>
<td>LBP (3304)</td>
</tr>
<tr>
<td>View 3</td>
<td>CENTRIST (254)</td>
<td>Gabor (6750)</td>
<td>Gabor (6750)</td>
</tr>
<tr>
<td>View 4</td>
<td>HOG (1984)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>View 5</td>
<td>GIST (512)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>View 6</td>
<td>LBP (928)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>n</td>
<td>1474</td>
<td>400</td>
<td>1440</td>
</tr>
<tr>
<td>n_c</td>
<td>7</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

Dataset description is summarized in Table 1. The ground-truth labels are available for these three datasets.

Seven metrics are used to evaluate the performance: clustering accuracy (ACC), normalized mutual information (NMI), purity, precision, recall, $F$-score, and adjusted rand index (ARI). These metrics are widely used, and they can be calculated by comparing the obtained label of each sample with the ground-truth provided in datasets. For these metrics, the larger value indicates the better clustering performance.

We evaluate the performance of the proposed GSF on three real-world datasets. GSF is compared with spectral clustering [33] and state-of-the-art multiview clustering methods, including co-regularized spectral clustering (CRSC) [17], robust multiview spectral clustering (RMSC) [56], robust multiview $k$-means clustering (RMKMC) [57], multiview spectral clustering (MVSC) [58], multiple kernel $k$-means clustering (MKKM) [31], and multiview clustering with graph learning (MVGL) [44].

The default parameters given by the respective authors are adopted. Specifically, the parameter setting is:

1) spectral clustering [33]: Apply spectral clustering on single view to obtain the clustering results.
2) CRSC$^1$ [17]: Co-regularize the clustering hypotheses in multiple views to enforce corresponding data in each view to have the same cluster membership based on spectral clustering. In CRSC, graphs are constructed in different views by using the Gaussian functions. In our experiment, we use the default setting.
3) RMKMC$^2$ [57]: Extend $k$-means to multiview clustering and use $\ell_{2,1}$-norm to obtain a relatively robust result. In our experiment, we search the logarithm of the parameter $\gamma$, i.e., $\log_{10} \gamma$, in the range from 0.1 to 2 with incremental 0.2 to obtain the best parameter.
4) RMSC [56]: Use low-rank and sparse decomposition to obtain a latent transition probability matrix from multiple views, and then use standard Markov chain method for spectral clustering to obtain a robust result. In our experiment, we search the parameter $\lambda$ in the range from 0.005 to 100 to obtain the best parameter.
5) MVSC [58]: By choosing salient points to speed up the graph construction and using local manifold integration to fuse heterogeneous features. In our experiment, the parameter $r$ that controls the fusion weights of all views, is searched in logarithm, i.e., $\log_{10} r$, from 0.1 to 2 with interval 0.2.
6) MKKM$^3$ [31]: Via a matrix-induced regularization, MKKM reduces the redundancy and enhances the diversity of the multiple kernels. The multiple kernels are constructed by the Gaussian function. To obtain the clustering indicators, a post-process $k$-means is needed. In our experiment, the regularization parameter $\lambda$ is searched from $\{2^{-15}, 2^{-14}, \ldots, 2^3\}$.
7) MVGL$^4$ [44]: Each graph is constructed by a learning objective function that is constrained by the rank constraint on the normalized Laplacian matrix. These learned graphs are fused and optimized to obtain a global graph with exact $n_c$ components so that the number of clusters and graph components are the same to each other. There are two parameters for MVGL, $\beta$ and $\gamma$. We use the defaulted parameters.

For these seven methods, we run each method 10 times and report the mean performance as well as the standard deviation. Because spectral clustering, CRSC, RMSC, MVSC, and MKKM need to perform $k$-means after they obtain the new representation of data points, in each experiment, we run $k$-means clustering 30 times to reduce the influence of random initialization. We report the result with the minimum value of the objective function of $k$-means among results of these 30 runs.

For GSF, both $\gamma_1$ and $\gamma_2$ are set to one. Since we learn to obtain a graph $S$ with exact $k$ components and the corresponding embedding matrix $U$ is a good data representation for clustering [32], [33], we tune $k$ to report the best results in terms of the objective value of $k$-means performed on $U$. In practice, we set $\gamma_2 = 1$ and change $\gamma_2$ in a heuristic way to accelerate the procedure [35]–[37]. Because the convergence criterion is that the sum of the top $n_c$ smallest eigenvalues of $L$ is zeroed, then we increase $\gamma_2$ if connected components of $S$ are greater than $n_c$, and decrease $\gamma_2$ if they are smaller than $n_c$ during the iteration.

2. https://sites.google.com/site/feipingnie/file/weighted_robust_multi_kmeans.m
5.2 Performance Evaluation

GSF is compared with other baseline algorithms, we show the clustering results in terms of ACC, NMI, purity, precision, recall, F-score, and ARI in Table 2, respectively. In Table 2, “SC v” means that spectral clustering is performed on the v-th view.

It can clearly be seen that GSF almost achieves the best performance. Surprisingly, GSF clusters all samples in COIL-20 correctly. The COIL-20 dataset is extracted from simple geometry. Objects belong to the same cluster are very similar to each other which implies significant low-rank property. GSF with low rank constraint achieves an integrated ideal global graph and obtains much better results than others. In Caltech-101, we obtain a large improvement in the metrics of ACC, NMI, Recall, F-score, and ARI comparing with other methods. The quantitative metrics demonstrate the superiority of it because it better utilizes the intrinsic structure of the data represented by the graph. Different from methods based on k-means clustering and spectral clustering, GSF can obtain an integrated global graph with a better structure, that is why it shows better performance than others.

5.3 Analysis of \( p \)-value

To ensure the improvement of the proposed GSF-based multiview clustering algorithm in terms of different performance metrics is statistically significant, non-parametric pair-wise Wilcoxon test is computed and conducted on the seven metrics. When the level of significance is set to 0.05, the \( p \)-value metrics of GSF achieve significant improvements in most cases. Table 3 presents the \( p \)-value of GSF against other state-of-the-art methods with respect to different clustering performance metrics. A smaller \( p \)-value means that GSF is more statistically significant, so GSF achieves statistically significant improvements.

5.4 Parameter Sensitivity

For unsupervised clustering methods, parameter insensitivity is vital for enhancing their stability. There are two parameters, \( \gamma_1 \) and \( \gamma_2 \), needed to be determined for GSF. Here, we conduct experiments on three datasets to illustrate the insensitivities of GSF to parameter variations of \( \gamma_1 \) and \( \gamma_2 \) in terms of ACC. Both of them vary in logarithm, i.e., \( \log_{10} \), in the range of \( \{ -3, -2, -1, 0, 1 \} \). It can be observed from Fig. 2 that competitive performance is obtained under a wide range of parameters.
Table 3

$p$-value of Pair-wised Wilcoxon test with respect to different metrics.

<table>
<thead>
<tr>
<th>Methods</th>
<th>ACC</th>
<th>NMI</th>
<th>Purity</th>
<th>Precision</th>
<th>Recall</th>
<th>$F$-score</th>
<th>ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caltech-101</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC 1</td>
<td>$4.14e-05$</td>
<td>$4.14e-05$</td>
<td>$5.43e-05$</td>
<td>$5.43e-05$</td>
<td>$5.43e-05$</td>
<td>$5.43e-05$</td>
<td>$5.43e-05$</td>
</tr>
<tr>
<td>SC 2</td>
<td>$2.43e-05$</td>
<td>$2.43e-05$</td>
<td>$2.43e-05$</td>
<td>$2.43e-05$</td>
<td>$2.43e-05$</td>
<td>$2.43e-05$</td>
<td>$2.43e-05$</td>
</tr>
<tr>
<td>SC 3</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
</tr>
<tr>
<td>SC 5</td>
<td>$5.35e-05$</td>
<td>$5.47e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
</tr>
<tr>
<td>SC 6</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
</tr>
<tr>
<td>CRSC</td>
<td>$5.35e-05$</td>
<td>$5.47e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
<td>$4.77e-05$</td>
</tr>
<tr>
<td>RMSC</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
</tr>
<tr>
<td>MKKM</td>
<td>$5.35e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
</tr>
<tr>
<td>MVGL</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
</tr>
<tr>
<td>ORL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC 1</td>
<td>$6.34e-05$</td>
<td>$6.39e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
</tr>
<tr>
<td>SC 2</td>
<td>$6.29e-05$</td>
<td>$6.39e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
</tr>
<tr>
<td>SC 3</td>
<td>$6.29e-05$</td>
<td>$6.39e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
</tr>
<tr>
<td>COIL-20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC 1</td>
<td>$6.39e-05$</td>
<td>$6.39e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
</tr>
<tr>
<td>SC 2</td>
<td>$6.34e-05$</td>
<td>$6.39e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
<td>$6.34e-05$</td>
</tr>
<tr>
<td>SC 3</td>
<td>$6.11e-05$</td>
<td>$6.34e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
<td>$5.47e-05$</td>
</tr>
<tr>
<td>CRSC</td>
<td>$5.94e-05$</td>
<td>$5.94e-05$</td>
<td>$5.94e-05$</td>
<td>$5.94e-05$</td>
<td>$5.94e-05$</td>
<td>$5.94e-05$</td>
<td>$5.94e-05$</td>
</tr>
<tr>
<td>MKKM</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
</tr>
<tr>
<td>MVGL</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
<td>$1.59e-05$</td>
</tr>
</tbody>
</table>

Figure 2. Sensitivity analysis on $\gamma_1$ and $\gamma_2$.

Figure 3. Convergence behaviors of GSF.
5.5 Convergence Study
To verify the convergence property of GSF, Fig. 3 shows the convergence curves of the objective value in three datasets, respectively. In Fig. 3, x-axis and y-axis denote the number of iterations and the corresponding objective values, respectively. For COIL-20 dataset, we find that the objective value decreases slightly between the first iteration and the second iteration, and then becomes stable, which implies that the structure-integrated graph A already approaches to nC connected components. The objective value is inversely proportional to the number of iterations and the objective value reaches a minimum within 10 iterations. This indicates that GSF converges sufficiently.

6 Conclusion
Combining multiview features for clustering is a fundamental research problem in data analysis. In this paper, we propose a novel multiview clustering method based on graph structure fusion, which learns a global graph with exact nC connected components reflecting cluster indicators. We use the Hadamard product on all the input graphs to preserve the intrinsic structure, and use it as a regularization when learning the global graph. Specifically, we obtain the intrinsic graph by using a theorem: The number of connected components of the graph is equal to the multiplicity of zero as an eigenvalue of the Laplacian matrix. Because the vertices in each connected component of the intrinsic graph belong to one cluster, we can obtain the clustering indicators without postprocessing steps. An efficient algorithm is provided to optimize it. Experimental results on real-world benchmark datasets demonstrate the effectiveness of GSF.

Acknowledgments
This work was supported by the National Natural Science Foundation of China under the Grant No. 61201422 and the Specialized Research Fund for the Doctoral Program of Higher Education under the Grant No. 2012021120013, and the Fundamental Research Funds for the Central Universities under the Grant No. Iuzjbky-2017-it73 and No. Iuzjbky-2017-it76.

References


Kun Zhan received the B.S. and Ph.D. degrees from the School of Information Science and Engineering, Lanzhou University, China, in 2005 and 2010, respectively. He was a visiting student with the Department of Electrical and Computer Engineering, Dalhousie University, Halifax, Canada, from 2009 to 2010. He is currently an associate professor with Lanzhou University and is a Visiting Scholar at University of Technology Sydney, Ultimo, NSW 2007, Australia. His main research interests include machine learning and data science.

Chaoxi Niu received the B.S. degree in Electronic Information Science and Technology from Lanzhou University, China, in 2017. He is currently pursuing the master’s degree at Lanzhou University.

His current research interest includes data science.

Changlu Chen received the B.S. degree in Electronic Information Science and Technology from Lanzhou University, China, in 2017. She is currently pursuing the master’s degree at Lanzhou University.

Her current research interest includes data science.

Feiping Nie received the Ph.D. degree in Computer Science from Tsinghua University, China in 2009.

He is currently a Professor with the Center for Optimal Imagery Analysis and Learning, Northwestern Polytechnical University, Shaanxi, China. His research interests are machine learning and its applications, such as pattern recognition, data mining, computer vision, image processing and information retrieval. He has published more than 100 papers in the following top journals and conferences: TPAMI, TKDE, IJCV, TIP, TNLS/TTN, TKDD, Bioinformatics, ICML, NIPS, KDD, IJCAI, AAAI, ICCV, CVPR, ACM MM, and SIGKDD Conference on Knowledge Discovery and Data Mining.

He is currently serving as Associate Editor or PC member for several prestigious journals and conferences in the related fields.
Changqing Zhang received his B.S. and M.S. degrees from the College of Computer Science, Sichuan University in 2005 and 2008, respectively, and the Ph.D. degree in Computer Science from Tianjin University, China, in 2016. He is an assistant professor at the School of Computer Science and Technology, Tianjin University. His current research interests include machine learning, data mining, and computer vision.

Yi Yang received the Ph.D. degree in computer science from Zhejiang University. He was a Post-Doctoral Research Fellow with the School of Computer Science, Carnegie Mellon University, from 2011 to 2013. He is currently a Professor with the Centre for Artificial Intelligence, School of Software, Faculty of Engineering and Information Technology, University of Technology Sydney. His research interests include multimedia, computer vision, and data science.
学霸图书馆
www.xuebalib.com

本文献由“学霸图书馆-文献云下载”收集自网络，仅供学习交流使用。

学霸图书馆（www.xuebalib.com）是一个“整合众多图书馆数据库资源，提供一站式文献检索和下载服务”的24小时在线不限IP图书馆。
图书馆致力于便利、促进学习与科研，提供最强文献下载服务。

图书馆导航：
图书馆首页 文献云下载 图书馆入口 外文数据库大全 疑难文献辅助工具