Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning☆

Zhenwen Ren [a,b], Haoran Li [c], Chao Yang [a,*], Quansen Sun [b,*]

[a] School of National Defence Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China
[b] Department of Computer Science, Nanjing University of Science and Technology, Nanjing 210094, China
[c] School of Information Engineering, Southwest University of Science and Technology, Mianyang 621010, China

ARTICLE INFO

Article history:
Received 21 April 2019
Received in revised form 11 August 2019
Accepted 12 September 2019
Available online xxxx

Keywords:
Multiple kernel learning
Subspace clustering
Self-expressiveness
Structure learning
Low-rank kernel

ABSTRACT

Multiple kernel learning (MKL) methods are generally believed to perform better than single kernel learning (SKL) methods in handling nonlinear subspace clustering problem, largely thanks to MKL avoids selecting and tuning a pre-defined kernel. However, previous MKL methods mainly focused on how to define a kernel weighting strategy, but ignored the structural characteristics of the input data in both the original space and the kernel space. In this paper, we first propose a novel graph-based MKL method for subspace clustering, namely, Local Structural Graph and Low-Rank Consensus Multiple Kernel Learning (LLMKL). It jointly learns an optimal affinity graph and a suitable consensus kernel for clustering purpose by elegantly integrating the MKL technology, the global structure in the kernel space, the local structure in the original space, and the Hilbert space self-expressiveness property in a unified optimization model. In particular, to capture the data global structure, we employ a substitute of the desired consensus kernel, and then introduce a low-rank constraint on the substitute to encourage that the structure of linear subspaces is present in the feature space. Moreover, the data local structure is explored by building a complete graph, where each sample is treated as a node, and an edge codes the pairwise affinity between two samples. By such, the consensus kernel learning and the affinity graph learning can promote each other such that the data in resulting Hilbert space are both self-expressive and low-rank. Experiments on both image and text clustering well demonstrate that LLMKL outperforms the state-of-the-art methods.

1. Introduction

Subspace clustering aims to cluster data points into a union of low-dimensional linear/affine subspaces [1], which has been widely used in various domains, such as data classification [2], motion segmentation [3], face clustering [4], document clustering [5], heterogeneous data analysis [6], subspace learning [7], hybrid system identification in control [8], and community clustering in social networks [9].

Existing methods can be roughly classified into four categories [10], including factorization-based methods [11], higher-order model-based methods [12,13], deep learning-based methods [14], and self-expressiveness-based methods [7,10,15–18]. Amongst them, the self-expressiveness-based methods assume that the samples can be well represented by a linear combination of other points in the same subspace. In general, such methods consist of two main steps: the first and also most crucial one aims to estimate an affinity matrix from the data; the second step then applies spectral clustering to partition the data into respective groups [16]. To learn a better affinity matrix, many scholars have proposed different regularization terms and constraints terms, such as low-rank constraint [15], sparse constraint [19], continuous label learning [20], adaptive weighted representation [21], and block diagonal representation [10]. We refer the readers to [22] for a comprehensive review.

However, most of the above methods are linear self-expressiveness ones, which cannot efficiently handle nonlinear subspaces problem. Unfortunately, in real-life applications, the data points may not fit exactly into a linear subspace model, i.e., them always lie on nonlinear subspaces. To overcome the drawback that linear subspace clustering methods cannot efficiently handle nonlinear data, the single kernel learning (SKL) methods have been widely researched [23]. Although improved performance has been reported in a wide variety of problems, the SKL methods require the user to select and tune a predefined

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105040.

* Corresponding authors.
E-mail addresses: rzw@njust.edu.cn (Z. Ren), ychao1983@126.com (C. Yang), sunquansen@njust.edu.cn (Q. Sun).

https://doi.org/10.1016/j.knosys.2019.105040
0950-7051/

kernel [24]. This is not user-friendly since the most suitable kernel for a specific task is usually challenging to decide.

In order to address the extremely difficult problem of selecting a predefined kernel in SKL, many researchers further extend their models to incorporate multiple kernel learning (MKL) [25] ability instead of using a single fixed kernel. Given a set of candidate base kernels, MKL finds an optimal combination of them to generate an optimal consensus kernel, so it has larger flexibility and has great potential to integrate complementary information. Accordingly, the performance of MKL is normally better than SKL's. For subspace clustering, in the past few years, some MKL-based methods have been proposed [24,26–29]. For example, Multiple Kernel K-means (MKKM) [26] extends K-means into a multiple kernel setting. Robust Multiple Kernel K-means (RMKKM) [27] is an extended version of MKKM, which simultaneously finds the cluster membership, the optimal combination of multiple kernels, and the best clustering labels. Due to the similarity between the affinity matrix and the kernel matrix, the Affinity Aggregation for Spectral Clustering (AASC) [28] replaces the single affinity matrix in spectral clustering with multiple matrices, which can be considered as a multiple kernel clustering version of spectral clustering. Spectral Clustering with Multiple Kernels (SCMK) [20] learns an optimal kernel using the same way adopted by RMKKM. By using the fact that the optimal kernel is a linear combination of predefined candidate base kernels, Low-rank Kernel Learning Graph-based Clustering (LKGr) [29] and Sparse Kernel Learning Graph-based Clustering (LKGs) [29] learn a low-rank kernel matrix and a sparse kernel matrix, respectively. Assume that each candidate kernel is a neighborhood of the optimal consensus kernel, a novel MKL framework, Self-weighted Multiple Kernel Learning (SMKL) [24], is proposed for subspace clustering and semisupervised classification, which introduces a more flexible kernel learning strategy to enhance the representation ability of the learned optimal consensus kernel.

Different from previous MKL subspace clustering methods that focus on the kernel weighting strategy, we aim to preserve the structure characteristics of the input data in both the original space and the kernel space. We first propose a novel MKL method for subspace clustering, called Local Structural Graph and Low-Rank Consensus Multiple Kernel Learning (LLMKL). In particular, LLMKL jointly learns an optimal affinity graph and a suitable consensus kernel for clustering purpose by elegantly integrating the MKL technology, the data global structure in the kernel space, the data local structure in the original space, and the Hilbert space self-expressiveness property in a unified optimization model. In this way, the affinity graph learning and the consensus kernel learning can be boosted mutually, which are the keys of subspace clustering and MKL respectively. In summary, the main contributions of this work lie in four-folds:

- To better handle the data with nonlinear structure, we propose a novel LLMKL method for subspace clustering. Compared with SKL methods, LLMKL can automatically learns the most suitable consensus kernel from a pool of predefined candidate base kernels. In other words, LLMKL can solve the difficult problem of how to define a suitable predefined kernel and tune its kernel parameters.
- By introducing a low-rank substitute of the consensus kernel matrix and a sparse error matrix, LLMKL can expose the latent global structure of data in Hilbert feature space, which is a better idea to deal with real data. Particularly, using low-rank constraint directly on consensus kernel cannot encourage the data in feature space to be low-rank, but the substitute can. Moreover, the error matrix can alleviate the negative effects of contaminated data points.
- By integrating the local structural graph term into our object function, LLMKL can preserve the locality relation among samples, which ensures to learn a more reasonable affinity matrix for clustering purpose. Moreover, thanks to using a unified object function, the affinity graph learning and the consensus kernel learning can be boosted mutually.
- By comparing with several state-of-the-art MKL clustering methods on nine widely used benchmark datasets, LLMKL substantially improves the clustering performance of the clustering results in terms of accuracy, NMI and purity. This demonstrates the superiority of the proposed method.

The remainder of this paper is organized as follows. Section 2 proposes our method. Section 3 shows experiments, and Section 4 presents the conclusions and further works.

2. Proposed method

2.1. Main notations

We denote matrices by boldface capital letters (e.g., $\boldsymbol{Z}$), vectors by boldface lowercase letters (e.g., $\boldsymbol{g}$), and scalars by Greek letters (e.g., λ). The (i,j)th entry of $\boldsymbol{A}$ is denoted as $\boldsymbol{A}_{ij}$. The ith row and the jth column of $\boldsymbol{A}$ are denoted as $\boldsymbol{A}^i$ and $\boldsymbol{A}_j$, respectively. The identity matrix is denoted as $\boldsymbol{I}$. The all-one vector is denoted as $\mathbf{1}$. If $\boldsymbol{A}$ is positive semi-definite, we denote as $\boldsymbol{A} \succeq 0$. Moreover, we define $[\boldsymbol{A}]_+ = \max(0, \boldsymbol{A})$ which gives the nonnegative part of $\boldsymbol{A}$, $\mathrm{diag}(\boldsymbol{A})$ as a vector with its ith entry being the ith diagonal entry of $\boldsymbol{A}$, and $\mathrm{Tr}(\boldsymbol{A})$ as trace operator of $\boldsymbol{A}$.

2.2. Problem formulation

As we know, to learn an affinity matrix, recent methods are mainly based on the self-expressiveness graph learning framework [10], which builds an affinity matrix such that data points from the same subspace have high affinity values and those from different subspaces have low affinity values , i.e.,

$$\min_{\boldsymbol{Z}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{Z}\|_F^2 + \alpha\mathcal{R}(\boldsymbol{Z}) \quad s.t.\ \boldsymbol{Z} \geqslant 0,\ \mathrm{diag}(\boldsymbol{Z}) = \mathbf{0} \tag{1}$$

where $\mathcal{R}(\boldsymbol{Z})$ is a regularization term or a discriminant term, $\alpha > 0$ defines the tradeoff between the loss function term and regularization term, and the self-expression coefficient matrix $\boldsymbol{Z}$ is often assumed to be nonnegative and $\boldsymbol{Z}_{ii} = 0$. The main difference of previous methods depend on the choice of the regularizer term $\mathcal{R}(\boldsymbol{Z})$. After obtaining $\boldsymbol{Z}$, the balanced affinity matrix is denoted as $(\boldsymbol{Z}^T + \boldsymbol{Z})/2$.

However, problem (1) cannot efficiently handle nonlinear data. To address this issue, some methods map the nonlinear data into a high-dimensional Reproducing Kernel Hilbert Space (RKHS) where a linear pattern analysis can be performed [23], i.e.,

$$\begin{aligned} &\min_{\boldsymbol{Z}} \frac{1}{2}\|\phi(\boldsymbol{X}) - \phi(\boldsymbol{X})\boldsymbol{Z}\|_F^2 + \alpha\mathcal{R}(\boldsymbol{Z}) \\ &s.t.\ \ \boldsymbol{Z} \geqslant 0,\ \mathrm{diag}(\boldsymbol{Z}) = \mathbf{0}. \\ &= \min_{\boldsymbol{Z}} \mathrm{Tr}(\phi(\boldsymbol{X})^T\phi(\boldsymbol{X}) - 2\phi(\boldsymbol{X})^T\phi(\boldsymbol{X})\boldsymbol{Z} \\ &+ \boldsymbol{Z}^T\phi(\boldsymbol{X})^T\phi(\boldsymbol{X})\boldsymbol{Z}) + \alpha\mathcal{R}(\boldsymbol{Z}) \\ &= \min_{\boldsymbol{Z}} \mathrm{Tr}(\boldsymbol{H} - 2\boldsymbol{H}\boldsymbol{Z} + \boldsymbol{Z}^T\boldsymbol{H}\boldsymbol{Z}) + \alpha\mathcal{R}(\boldsymbol{Z}) \end{aligned} \tag{2}$$

where $\boldsymbol{H}$ is a kernel Gram matrix and $\phi(\cdot)$ is a kernel mapping function.

Although the data may be nonlinear in original space, the global structure of linear subspaces should be present in RKHS, i.e., $\phi(\boldsymbol{X})$ should be low-rank [29,30]. Furthermore, we would like the data in RKHS to still lie on multiple linear subspaces with the kernel subspace self-expressiveness. Combining both of them,

Please cite this article as: Z. Ren, H. Li, C. Yang et al., Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning, Knowledge-Based Systems (2019) 105040, https://doi.org/10.1016/j.knosys.2019.105040.

we expect the feature mapping to be both self-expressive and low-rank, i.e.,

$$\min_{\boldsymbol{Z}} \frac{1}{2}\|\phi(\boldsymbol{X}) - \phi(\boldsymbol{X})\boldsymbol{Z}\|_F^2 + \alpha\mathcal{R}(\boldsymbol{Z}) + \beta\|\phi(\boldsymbol{X})\|_*. \tag{3}$$

Based on kernel trick, we do not explicitly depend on the unknown kernel mapping function $\phi(\cdot)$, but on the kernel Gram matrix $\boldsymbol{H} = \phi(\boldsymbol{X})^T\phi(\boldsymbol{X})$, where $\boldsymbol{H}_{ij} = \langle\phi(\boldsymbol{X}_i), \phi(\boldsymbol{X}_j)\rangle$. Moreover, since $\text{rank}(\boldsymbol{H}) = \text{rank}(\phi(\boldsymbol{X}))$, one may think of using $\|\boldsymbol{H}\|_*$ instead of $\|\phi(\boldsymbol{X})\|_*$. Therefore, problem (3) can be reformulated as:

$$\min_{\boldsymbol{Z},\boldsymbol{H}} \text{Tr}(\boldsymbol{H} - 2\boldsymbol{H}\boldsymbol{Z} + \boldsymbol{Z}^T\boldsymbol{H}\boldsymbol{Z}) + \alpha\mathcal{R}(\boldsymbol{Z}) + \beta\|\boldsymbol{H}\|_* \tag{4}$$

which does not explicitly depend on $\phi(\boldsymbol{X})$ anymore. However, minimizing $\|\boldsymbol{H}\|_*$ will not lead to low-rank of $\|\phi(\boldsymbol{X})\|_*$ due to $\|\boldsymbol{H}\|_* = \text{Tr}\sqrt{\boldsymbol{H}^T\boldsymbol{H}} = \text{Tr}(\phi(\boldsymbol{X})^T\phi(\boldsymbol{X})) = \|\phi(\boldsymbol{X})\|_F^2$, i.e., $\min\|\boldsymbol{H}\|_*$ is equivalent to $\min\|\phi(\boldsymbol{X})\|_F^2$, which cannot encourage the data in kernel space to be low-rank. Fortunately, since the kernel matrix $\boldsymbol{H}$ is symmetric positive semi-definite (i.e., $\boldsymbol{H} \succeq 0$), we can decompose it via an auxiliary square matrix $\boldsymbol{B}$, i.e., $\boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B}$. In other words, we have $\|\boldsymbol{B}\|_* = \|\phi(\boldsymbol{X})\|_*, \forall\boldsymbol{B} : \boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B}$. Then, we have

$$\min\|\boldsymbol{H}\|_* \Rightarrow \min\|\boldsymbol{B}\|_*, \forall\boldsymbol{B} : \boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B} \tag{5}$$

Therefore, the above discussion bear out that $\|\boldsymbol{B}\|_*$ can capture the global structure of the data in kernel space.

In addition, the local structure of the samples is useful and can also reveal the intrinsic relationships of samples [22,31]. Therefore, we further introduce a complete graph $\boldsymbol{D}$ with $\boldsymbol{D}_{ij} = \|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2^2$ named local structural graph, in which each sample is treated as a node, and the pairwise affinity between two samples is treated as edge weight. Therefore, we let

$$\mathcal{R}(\boldsymbol{Z}) = \min_{\boldsymbol{Z}} \sum_{i,j=1}^{n} \|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2^2\boldsymbol{Z}_{ij} = \min_{\boldsymbol{Z}} \text{Tr}(\boldsymbol{D}^T\boldsymbol{Z}) \tag{6}$$

where n is the number of samples and $\boldsymbol{Z}_{ij}$ represents the similarity (or probability) between $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$. The smaller $\boldsymbol{D}_{ij}$ is, the greater the probability $\boldsymbol{Z}_{ij}$ is. Analogously to Laplacian Eigenmaps (LE) [32], the local structural information among samples is preserved.

Although problem (3) can handle nonlinear data, its performance will heavily depend on the predefined kernel matrix $\boldsymbol{H}$. To overcome this drawback, we can learn a suitable consensus kernel $\boldsymbol{H}$ from r candidate base kernels $\{\boldsymbol{H}_i\}_{i=1}^r$. Inspired by [29], we use the following MKL weighting strategy:

$$\min_{\boldsymbol{H},\boldsymbol{g}} \|\boldsymbol{H} - \sum_{i=1}^{r} g_i\boldsymbol{H}_i\|_F^2 \quad s.t.\ g_i \geq 0, \sum_{i=1}^{r} g_i = 1 \tag{7}$$

where g_i is the weight of the ith base kernel $\boldsymbol{H}_i$.

If the training data is corrupted by noise, the performance is diminished [33]. To deal with the noisy data effectively, $\boldsymbol{H}$ is decomposed into $\boldsymbol{B}^T\boldsymbol{B}$ and a sparse noise component $\boldsymbol{E}$ (i.e., $\boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B} + \boldsymbol{E}$).

Hereto, by integrating the above ideas, the final objective function of LLMKL is written as follows:

$$\begin{aligned}
&\min_{\boldsymbol{Z},\boldsymbol{H},\boldsymbol{B},\boldsymbol{E},\boldsymbol{g}} \frac{1}{2}\text{Tr}\left[(\boldsymbol{I} - 2\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{Z}^T)\boldsymbol{B}^T\boldsymbol{B}\right] + \lambda_1\|\boldsymbol{B}\|_* \\
&+\frac{\lambda_2}{2}\|\boldsymbol{H} - \sum_{i=1}^{r} g_i\boldsymbol{H}_i\|_F^2 + \lambda_3\text{Tr}(\boldsymbol{D}^T\boldsymbol{Z}) + \lambda_4\|\boldsymbol{E}\|_1 \\
&s.t.\ \boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B} + \boldsymbol{E}, \text{diag}(\boldsymbol{Z}) = \boldsymbol{0}, \boldsymbol{Z}^T\boldsymbol{1} = \boldsymbol{1}, \\
&\boldsymbol{Z} \geq 0, g_i \geq 0, \sum_{i=1}^{r} g_i = 1
\end{aligned} \tag{8}$$

where $\lambda_i > 0$ ($i \in [1, 2, 3, 4]$) are tunable penalty parameters. In order to better understand the proposed method, the block diagram is illustrated in Fig. 1.

2.3. Optimization

Due to $\boldsymbol{B}^T\boldsymbol{B}$, the problem (8) is non-convex. However, ADMM has gained popularity to solve such non-convex problem effectively, and its convergence analysis for non-convex problem is provided in [34]. Therefore, in this paper, we solve (8) via the ADMM.

The augmented Lagrangian function of problem (8) is given by:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{Z},\boldsymbol{B},\boldsymbol{H},\boldsymbol{E},\boldsymbol{g}) &= \frac{1}{2}\text{Tr}\left[(\boldsymbol{I} - 2\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{Z}^T)\boldsymbol{B}^T\boldsymbol{B}\right] \\
&+\lambda_1\|\boldsymbol{B}\|_* + \frac{\lambda_2}{2}\|\boldsymbol{H} - \sum_{i=1}^{r} g_i\boldsymbol{H}_i\|_F^2 + \lambda_3\text{Tr}(\boldsymbol{D}^T\boldsymbol{Z}) \\
&+\lambda_4\|\boldsymbol{E}\|_1 + \frac{\mu}{2}\|\boldsymbol{H} - \boldsymbol{B}^T\boldsymbol{B} - \boldsymbol{E} + \frac{\boldsymbol{Y}_1}{\mu}\|_F^2
\end{aligned} \tag{9}$$

where $\boldsymbol{Y}_1$ is Lagrange multiplier, and μ is a penalty parameter.

(1) Update Z:

The optimization problem with respect to $\boldsymbol{Z}$ can be reformulated as

$$\begin{aligned}
&\min_{\boldsymbol{Z}} \frac{1}{2}\text{Tr}\left[(\boldsymbol{I} - 2\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{Z}^T)\boldsymbol{B}^T\boldsymbol{B}\right] + \lambda_3\text{Tr}(\boldsymbol{D}^T\boldsymbol{Z}) \\
&s.t.\ \text{diag}(\boldsymbol{Z}) = \boldsymbol{0}, \boldsymbol{Z}^T\boldsymbol{1} = \boldsymbol{1}, \boldsymbol{Z} \geq 0.
\end{aligned} \tag{10}$$

Analogously to [35,36], for computational simplicity and efficiency, we consider a two-step fast approximation problem to solve (10).

In step one, we first compute a latent solution $\widehat{\boldsymbol{Z}}$ by defining an auxiliary problem of (10), i.e.,

$$\min_{\boldsymbol{Z}} \frac{1}{2}\text{Tr}\left[(\boldsymbol{I} - 2\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{Z}^T)\boldsymbol{B}^T\boldsymbol{B}\right] + \lambda_3\text{Tr}(\boldsymbol{D}^T\boldsymbol{Z}). \tag{11}$$

By setting $\partial\mathcal{L}(\boldsymbol{Z})/\partial\boldsymbol{Z} = 0$, problem (11) has a closed solution as

$$\widehat{\boldsymbol{Z}} = (\boldsymbol{B}^T\boldsymbol{B})^{-1}(\boldsymbol{B}^T\boldsymbol{B} - \lambda_3\boldsymbol{D}). \tag{12}$$

In step two, to satisfy the involved constraints of problem (10), we project the latent solution $\widehat{\boldsymbol{Z}}$ to a constrained space. Thus, we can obtain the optimal solution $\boldsymbol{Z}$ via the following minimization problem:

$$\min_{\boldsymbol{Z}\geq 0, \text{diag}(\boldsymbol{Z})=0, \boldsymbol{Z}^T\boldsymbol{1}=\boldsymbol{1}} \|\boldsymbol{Z} - \widehat{\boldsymbol{Z}}\|_F^2. \tag{13}$$

Then, we can obtain

$$\boldsymbol{Z}_j^* = \max(\widehat{\boldsymbol{Z}}_j + \alpha_j\boldsymbol{1}, 0), \boldsymbol{Z}_{jj} = 0, \alpha_j = \frac{1 + \widehat{\boldsymbol{Z}}_j^T\boldsymbol{1}}{n}. \tag{14}$$

The detailed derivations of (14) is given in the Appendix.

(2) Update H:

The optimization problem with respect to $\boldsymbol{H}$ can be reformulated as

$$\min_{\boldsymbol{H}} \frac{\lambda_2}{2}\|\boldsymbol{H} - \sum_{i=1}^{r} g_i\boldsymbol{H}_i\|_F^2 + \frac{\mu}{2}\|\boldsymbol{H} - \boldsymbol{B}^T\boldsymbol{B} - \boldsymbol{E} + \frac{\boldsymbol{Y}_1}{\mu}\|_F^2. \tag{15}$$

By taking the derivative of $\mathcal{L}(\boldsymbol{H})$ w.r.t. $\boldsymbol{H}$, and setting it to zero, (15) has a closed-from solution given by

$$\boldsymbol{H}^* = \frac{\lambda_2\sum_{i=1}^{r} g_i\boldsymbol{H}_i + \mu(\boldsymbol{B}^T\boldsymbol{B} + \boldsymbol{E} - \frac{\boldsymbol{Y}_1}{\mu})}{\lambda_2 + \mu}. \tag{16}$$

Please cite this article as: Z. Ren, H. Li, C. Yang et al., Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning, Knowledge-Based Systems (2019) 105040, https://doi.org/10.1016/j.knosys.2019.105040.

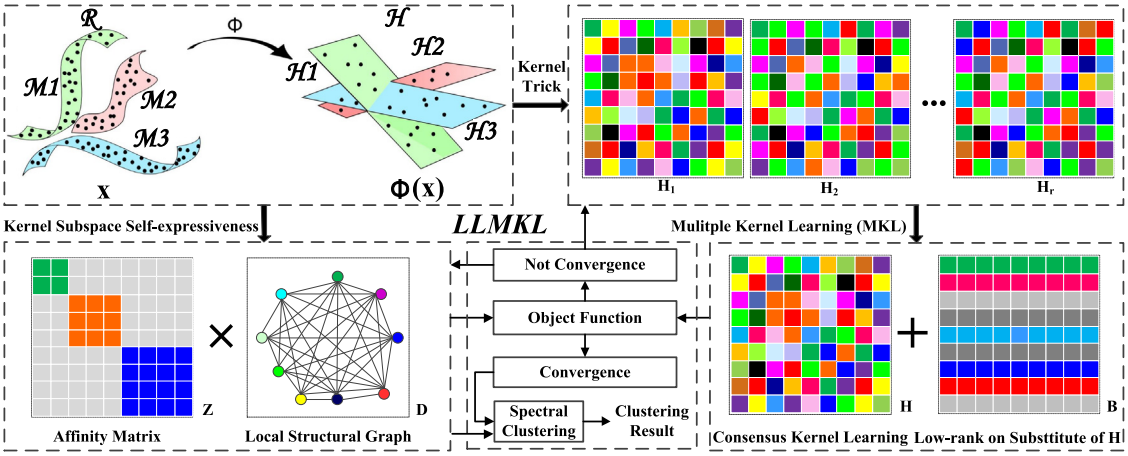


Fig. 1. The block diagram of the proposed LLMKL method.

(3) Update B:

For updating $\boldsymbol{B}$, we have the following optimization problem

$$\min_{\boldsymbol{B}} \frac{1}{2}\mathrm{Tr}\left[(\boldsymbol{I}-2\boldsymbol{Z}+\boldsymbol{Z}\boldsymbol{Z}^T)\boldsymbol{B}^T\boldsymbol{B}\right]+\lambda_1\|\boldsymbol{B}\|_* + \frac{\mu}{2}\|\boldsymbol{H}-\boldsymbol{B}^T\boldsymbol{B}-\boldsymbol{E}+\frac{\boldsymbol{Y}_1}{\mu}\|_F^2. \tag{17}$$

Let $\boldsymbol{M}=\boldsymbol{H}-\boldsymbol{E}+\frac{\boldsymbol{Y}_1}{\mu}$, the optimization problem (17) can then be reformulated as

$$\min_{\boldsymbol{B}} \lambda_1\|\boldsymbol{B}\|_* + \frac{\mu}{2}\|\boldsymbol{B}^T\boldsymbol{B}-\widetilde{\boldsymbol{M}}\|_F^2 \tag{18}$$

where $\widetilde{\boldsymbol{M}}=\boldsymbol{M}-\frac{1}{2\mu}(\boldsymbol{I}-2\boldsymbol{Z}+\boldsymbol{Z}\boldsymbol{Z}^T)$. Fortunately, such problem can be solved by Lemma 1.

Lemma 1. *With $\boldsymbol{A}\succeq 0$ let $\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$ denote its singular value decomposition. Then*

$$\min_{\boldsymbol{B}} \frac{\rho}{2}\|\boldsymbol{A}-\boldsymbol{B}^T\boldsymbol{B}\|_F^2+\tau\|\boldsymbol{B}\|_* = \sum_{i=1}^{n}\left(\frac{\rho}{2}(\sigma_i-\gamma_i^{*2})^2+\tau\gamma_i^*\right) \tag{19}$$

A minimizer $\boldsymbol{B}^*$ of (19) is given by

$$\boldsymbol{B}^*=\boldsymbol{\Gamma}^*\boldsymbol{U}^T \tag{20}$$

with $\boldsymbol{\Gamma}^*\in\boldsymbol{D}_+^n$, $\gamma_i^*\in\{\alpha\in\mathbb{R}_+|p_{\sigma_i,\tau/2\rho}(\alpha)=0\}\cup\{0\}$, where $p_{a,b}$ denotes the depressed cubic $p_{a,b}(x)=x^3-ax+b$. $\boldsymbol{D}_+^n$ is the set of $n\times n$ diagonal matrix with non-negative entries.

(4) Update E:

For updating $\boldsymbol{E}$, the optimization problem for $\boldsymbol{E}$ is

$$\min_{\boldsymbol{E}} \lambda_4\|\boldsymbol{E}\|_1+\frac{\mu}{2}\|\boldsymbol{H}-\boldsymbol{B}^T\boldsymbol{B}-\boldsymbol{E}+\frac{\boldsymbol{Y}_1}{\mu}\|_F^2. \tag{21}$$

Let $\boldsymbol{M}=\boldsymbol{H}-\boldsymbol{B}^T\boldsymbol{B}+\frac{\boldsymbol{Y}_1}{\mu}$, we then can solve problem (21) column-wise, i.e.,

$$\min_{\boldsymbol{E}_i} \frac{\mu}{2}\|\boldsymbol{E}_i-\boldsymbol{M}_i\|_2^2+\lambda_4\|\boldsymbol{E}_i\|_1 \tag{22}$$

where $\boldsymbol{E}_i$ and $\boldsymbol{M}_i$ are the ith column of $\boldsymbol{E}$ and $\boldsymbol{M}$, respectively. This problem has a closed form solution given as follows.

$$E_{ij}^* = \mathrm{sign}(\boldsymbol{M}_{ij})(\mathrm{abs}(\boldsymbol{M}_{ij})-\frac{\lambda_4}{\mu})_+ = \begin{cases} \boldsymbol{M}_{ij}-\frac{\lambda_4}{\mu}, & \text{if} \quad \boldsymbol{M}_{ij}>\frac{\lambda_4}{\mu} \\ \boldsymbol{M}_{ij}+\frac{\lambda_4}{\mu}, & \text{if} \quad \boldsymbol{M}_{ij}<-\frac{\lambda_4}{\mu} \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

(5) Update g:

For updating $\boldsymbol{g}$, the optimization problem for $\boldsymbol{g}$ is

$$\min_{\boldsymbol{g}} \lambda_2\|\boldsymbol{H}-\sum_{i=1}^{r}g_i\boldsymbol{H}_i\|_F^2 \quad s.t.\ g_i\geq 0, \sum_{i=1}^{r}g_i=1 \tag{24}$$

which can be rewritten as

$$\min_{\boldsymbol{g}} \lambda_2\boldsymbol{g}^T\boldsymbol{M}\boldsymbol{g}-\boldsymbol{a}^T\boldsymbol{g} \quad s.t.\ g_i\geq 0,\ \sum_{i=1}^{r}g_i=1 \tag{25}$$

where $a_i=\frac{\lambda_2}{2}\mathrm{Tr}(\boldsymbol{H}\boldsymbol{H}_i)$ and $\boldsymbol{M}_{ij}=\mathrm{Tr}(\boldsymbol{H}_i\boldsymbol{H}_j)$. It can be easily solved by Quadratic programming[1] [29].

(6) Update Y_1 and μ:

$\boldsymbol{Y}_1$ and μ are respectively updated by using the following formulas:

$$\begin{aligned}\boldsymbol{Y}_1 &= \boldsymbol{Y}_1+\mu(\boldsymbol{H}-\boldsymbol{B}^T\boldsymbol{B}-\boldsymbol{E}) \\ \mu &= \min(\eta\mu,\mu_{max}).\end{aligned} \tag{26}$$

Using the above steps, we alternatively update $\boldsymbol{Z}$, $\boldsymbol{B}$, $\boldsymbol{H}$, $\boldsymbol{E}$, $\boldsymbol{g}$, $\boldsymbol{Y}_1$ and μ, and repeat the process again and again until the objective function approaches to convergence or reach the maximal iterations. We summarize the above algorithm in Alg. 1

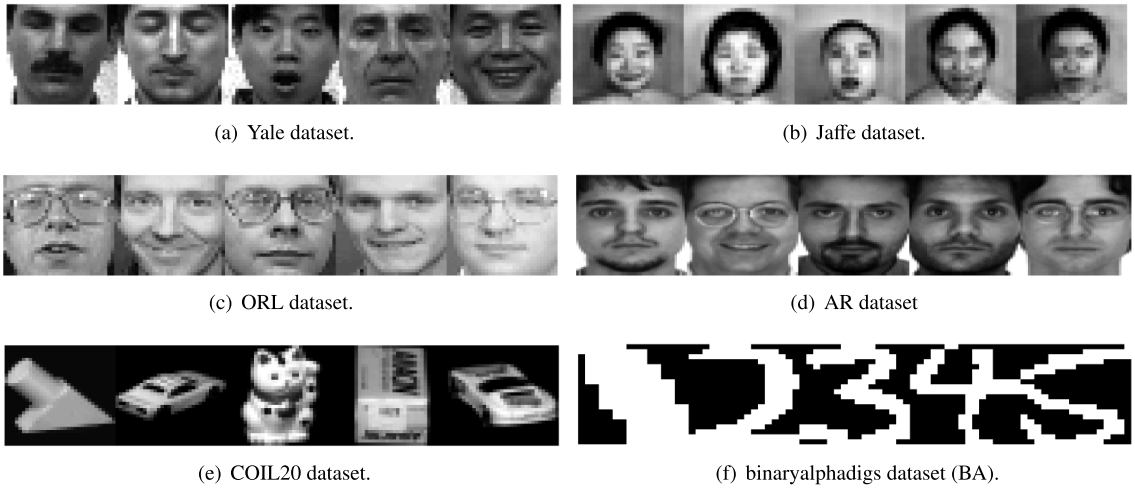
After obtaining $\boldsymbol{Z}$, the balanced affinity matrix and the corresponding Laplacian matrix are denoted as $\boldsymbol{Z}=(|\boldsymbol{Z}|+|\boldsymbol{Z}^T|)/2$ and $\boldsymbol{L}=\boldsymbol{S}-\boldsymbol{Z}$, respectively, where $\boldsymbol{S}$ is a diagonal matrix with diagonal elements $\boldsymbol{S}_{ii}=\Sigma_j\boldsymbol{Z}_{ij}$. We then perform spectral clustering via

$$\min_{\boldsymbol{F}} \mathrm{Tr}(\boldsymbol{F}^T\boldsymbol{L}\boldsymbol{F}) \quad s.t.\ \boldsymbol{F}^T\boldsymbol{F}=\boldsymbol{I} \tag{27}$$

where $\boldsymbol{F}\in\mathcal{R}^{n\times c}$ is the cluster indicator matrix.

[1] https://ww2.mathworks.cn/help/optim/ug/quadprog.html.

Please cite this article as: Z. Ren, H. Li, C. Yang et al., Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning, Knowledge-Based Systems (2019) 105040, https://doi.org/10.1016/j.knosys.2019.105040.



(a) Yale dataset. (b) Jaffe dataset.

(c) ORL dataset. (d) AR dataset

(e) COIL20 dataset. (f) binaryalphadigs dataset (BA).

Fig. 2. Cropped images of the first five subject/object form the six image datasets.

Algorithm 1 Solving LLMKL for clustering via ADMM.

Input: Data matrix $\boldsymbol{X}$, r base kernel matrices $\{\boldsymbol{H}_i\}_{i=1}^r$, and tradeoff parameters λ_1, λ_2, λ_3, λ_4.
Initialize: $\boldsymbol{Z}^1 = \boldsymbol{0}$, $\boldsymbol{E}^1 = \boldsymbol{0}$, $\boldsymbol{H}^1 = \frac{1}{r}\sum_{i=1}^r \boldsymbol{H}_i$, $\{g_i^1\}_{i=1}^r = \frac{1}{r}$, $\varepsilon = 10^{-6}$, $t = 1$, $\mu_{max} = 1e10$, $\mu = 0.1$, $\eta = 20$, and $maxIter = 1e3$.

1: **while** convergence criterion is not satisfied, and $t < maxIter$ **do**
2: Update $\boldsymbol{Z}^{t+1}$ by using (14).
3: Update $\boldsymbol{B}^{t+1}$ by using (20).
4: Update $\boldsymbol{H}^{t+1}$ by using (16).
5: Update $\boldsymbol{E}^{t+1}$ by using (23).
6: Update $\boldsymbol{g}^{t+1}$ by using (25).
7: Update $\boldsymbol{Y}_1^{t+1}$ and μ^{t+1} by using (26).
8: t = t+1;
9: **end while**
10: Define balanced affinity matrix $\boldsymbol{Z} = \frac{1}{2}(\boldsymbol{Z} + \boldsymbol{Z}^T)$.
11: Perform spectral clustering by using (27).

Output: The clustering results: ACC, NMI, Purity.

2.4. Complexity analysis

As shown in Alg. 1, for updating $\boldsymbol{Z}$, the computational complexity of matrix inverse operation (i.e., $\boldsymbol{B}^T\boldsymbol{B})^{-1}$ is $O(n^3)$. For updating $\boldsymbol{H}$, the computational complexity is $O(r)$. For updating $\boldsymbol{B}$, the computational complexity of the SVD of $\widetilde{\boldsymbol{M}}$ is $O(n^3)$. The updating of $\boldsymbol{E}$ is element-wise operation, the computational complexity is $O(n^2)$. For updating $\boldsymbol{g}$, the quadratic programming problem can be solved in polynomial time. Fortunately, the size of $\boldsymbol{g}$ is a small number r. For updating $\boldsymbol{Y}_1$, the computational complexity is $O(n)$. In sum, the computational complexity of each iteration of LLRSE is about to $O(2*n^3+n^2+n)$. Hence, the overall time complexity of LLMKL can be loosely thought of as $O(\tau * n^3)$, where τ is the number of iterations.

3. Experiments

In this section, we compare the clustering performance and computational cost of the proposed method with the several state-of-art methods on a number of benchmark datasets.

Table 1
Attributes of the 9 widely used datasets.

Dataset	# instances	# features	# classes
Yale	165	1024	15
Jaffe	213	676	10
ORL	400	1024	40
AR	840	768	120
COIL20	1440	1024	20
BA	1404	320	36
tr11	414	6429	9
tr41	878	7454	10
tr45	690	8261	10

3.1. Datasets

We collected a variety of datasets, including 6 image datasets (i.e., Yale,[2] Jaffe,[3] ORLL,[4] AR,[5] COIL20,[6] BA[7]:) and 3 text corporas[8] (i.e., tr11, tr41, tr45), most of which have been frequently used to evaluate the performance of different clustering algorithms. The statistics of these datasets are summarized in Table 1, and some samples are shown in Fig. 2. We refer the reader to [16] for more detailed descriptions.

3.2. Compared methods

We compare LLMKL with several state-of-the-art MKL clustering methods, including MKKM [26], AASC [28], RMKKM [27], SCMK [20], LKGr [29], and SMKL [24] (see Section 1). For fair comparison, the important parameters of all the compared methods are carefully tuned following the recommendations in their original works.

As in [16], to quantitatively evaluate our algorithm's performance on the clustering task, the widely used metrics, clustering accuracy (ACC), normalized mutual information (NMI) and purity, are applied.

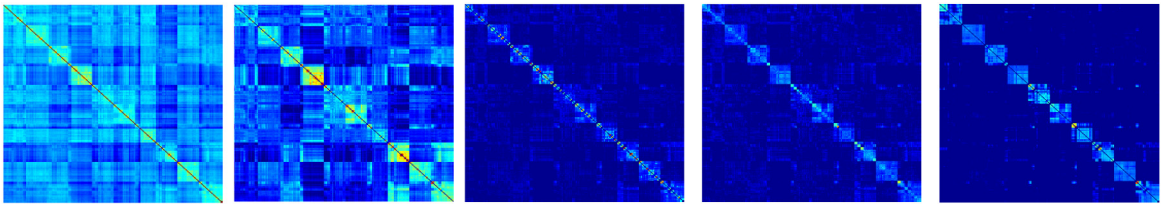
2 http://www.cvc.yale.edu/projects/yalefaces/yalefaces.html.
3 http://www.kasrl.org/jaffe.html.
4 https://www.cl.cam.ac.uk/research/dtg/attarchive/facedataset.html.
5 http://www2.ece.ohio-state.edu/~aleix/ARdataset.html.
6 http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php.
7 https://cs.nyu.edu/~roweis/data/binaryalphadigs.mat.
8 http://trec.nist.gov.

Please cite this article as: Z. Ren, H. Li, C. Yang et al., Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning, Knowledge-Based Systems (2019) 105040, https://doi.org/10.1016/j.knosys.2019.105040.

Table 2
Clustering results measured by ACC/NMI/Purity with standard deviation of the compared multiple kernel methods.

Data	Metrics	MKKM	AASC	RMKKM	SCMK	LKGr	SMKL	LLMKL
Yale	ACC	0.457(0.041)	0.406(0.027)	0.521(0.034)	0.582(0.025)	0.540(0.030)	0.582(0.017)	**0.655(0.009)**
	NMI	0.501(0.036)	0.468(0.028)	0.556(0.025)	0.576(0.012)	0.566(0.025)	0.614(0.015)	**0.646(0.007)**
	Purity	0.475(0.037)	0.423(0.026)	0.536(0.031)	0.610(0.014)	0.554(0.029)	0.667(0.014)	**0.683(0.009)**
JAFFE	ACC	0.746(0.069)	0.304(0.008)	0.871(0.053)	0.869(0.022)	0.861(0.052)	0.967(0.000)	**1.000(0.000)**
	NMI	0.798(0.058)	0.272(0.006)	0.893(0.041)	0.868(0.021)	0.869(0.031)	0.951(0.000)	**1.000(0.000)**
	Purity	0.768(0.062)	0.331(0.008)	0.889(0.045)	0.882(0.023)	0.859(0.038)	0.967(0.000)	**1.000(0.000)**
ORL	ACC	0.475(0.023)	0.272(0.009)	0.556(0.024)	0.656(0.015)	0.616(0.016)	0.573(0.032)	**0.800(0.003)**
	NMI	0.689(0.016)	0.438(0.007)	0.748(0.018)	0.808(0.008)	0.794(0.008)	0.733(0.027)	**0.891(0.003)**
	Purity	0.514(0.021)	0.316(0.007)	0.602(0.024)	0.699(0.015)	0.658(0.017)	0.648(0.017)	**0.839(0.009)**
AR	ACC	0.286(0.014)	0.332(0.006)	0.344(0.012)	0.544(0.024)	0.314(0.015)	0.263(0.009)	**0.853(0.013)**
	NMI	0.592(0.014)	0.651(0.005)	0.655(0.015)	0.775(0.009)	0.648(0.007)	0.568(0.014)	**0.935(0.003)**
	Purity	0.305(0.012)	0.350(0.006)	0.368(0.010)	0.642(0.014)	0.330(0.014)	0.530(0.014)	**0.897(0.003)**
COIL20	ACC	0.548(0.058)	0.349(0.050)	**0.667(0.028)**	0.591(0.028)	0.618(0.051)	0.487(0.031)	0.636(0.010)
	NMI	0.707(0.033)	0.419(0.027)	0.773(0.017)	0.726(0.011)	0.766(0.023)	0.628(0.018)	**0.806(0.004)**
	Purity	0.590(0.053)	0.391(0.044)	0.699(0.022)	0.635(0.013)	0.650(0.039)	0.683(0.004)	**0.714(0.010)**
BA	ACC	0.405(0.019)	0.271(0.003)	0.434(0.018)	0.384(0.014)	0.444(0.018)	0.246(0.012)	**0.482(0.010)**
	NMI	0.569(0.008)	0.423(0.004)	0.585(0.011)	0.544(0.012)	0.604(0.009)	0.486(0.011)	**0.619(0.007)**
	Purity	0.435(0.014)	0.303(0.004)	0.463(0.015)	0.606(0.009)	0.479(0.017)	**0.623(0.011)**	0.593(0.009)
TR11	ACC	0.501(0.048)	0.472(0.008)	0.577(0.094)	0.549(0.015)	0.607(0.043)	0.708(0.033)	**0.718(0.001)**
	NMI	0.446(0.046)	0.394(0.003)	0.561(0.118)	0.371(0.018)	0.597(0.031)	0.557(0.068)	**0.633(0.002)**
	Purity	0.655(0.044)	0.547(0.000)	0.729(0.096)	0.783(0.011)	0.776(0.030)	**0.835(0.048)**	0.801(0.002)
TR41	ACC	0.561(0.068)	0.459(0.001)	0.627(0.073)	0.650(0.068)	0.595(0.020)	0.671(0.002)	**0.689(0.004)**
	NMI	0.578(0.042)	0.431(0.000)	0.635(0.092)	0.492(0.017)	0.604(0.023)	0.625(0.004)	**0.666(0.003)**
	Purity	0.728(0.042)	0.621(0.001)	0.776(0.065)	0.758(0.034)	0.759(0.031)	0.761(0.003)	**0.817(0.003)**
TR45	ACC	0.585(0.066)	0.526(0.008)	0.640(0.071)	0.634(0.058)	0.663(0.042)	0.671(0.004)	**0.745(0.000)**
	NMI	0.562(0.056)	0.420(0.014)	0.627(0.092)	0.584(0.051)	0.671(0.020)	0.622(0.007)	**0.726(0.000)**
	Purity	0.691(0.058)	0.575(0.011)	0.752(0.074)	0.728(0.048)	0.800(0.026)	0.816(0.004)	**0.797(0.000)**
Avg	ACC	0.507(0.045)	0.377(0.013)	0.582(0.045)	0.607(0.030)	0.584(0.032)	0.574(0.016)	**0.731(0.006)**
	NMI	0.605(0.034)	0.435(0.010)	0.670(0.048)	0.638(0.018)	0.680(0.020)	0.643(0.018)	**0.769(0.003)**
	Purity	0.573(0.038)	0.429(0.012)	0.646(0.042)	0.705(0.020)	0.650(0.027)	0.726(0.013)	**0.793(0.005)**

The last three rows are the mean ACC, NMI, and purity, respectively, for each separate datasets.



(a) AASC (b) LKGr (c) SMKL (d) SCMK (e) LLMKL

Fig. 3. Visualization of the learned affinity matrices on the Jaffe dataset from AASC, LKGr, SMKL, SCMK, and LLMKL, respectively. The Jaffe dataset consists of 10 clusters. Our method LLMKL has a dense affinity matrix with better block diagonal structure.

3.3. Experimental settings

Following the settings in [27], we construct 12 base kernels (i.e., $r = 12$) in this work, including a cosine kernel $\boldsymbol{H}_{ij} = (\boldsymbol{X}_i^T\boldsymbol{X}_j)/(\|\boldsymbol{X}_i\| \cdot \|\boldsymbol{X}\|)$, 4 polynomial kernels $\boldsymbol{H}_{ij} = (a + \boldsymbol{X}_i^T\boldsymbol{X}_j)^b$ with $a = \{0, 1\}$ and $b = \{2, 4\}$, and 7 RBF kernels $\boldsymbol{H}_{ij} = \exp - \|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2^2/(2t\sigma^2)$, where σ is the maximum distance between samples and t varies in the range of $\{0.01, 0.05, 0.1, 1, 10, 50, 100\}$. Finally, all the kernels are rescaled to [0, 1] range.

3.4. Experimental results

The average clustering results of 20 times independent experiments with standard deviation of all the methods on 9 widely used datasets are presented in Table 2. Results that are significantly better than others are indicated in boldface. Besides, the learned affinity matrices on the Jaffe dataset are illustrated in Fig. 3. From the results, we have the following observations.

(1) According to the results in Table 2, we can see that the proposed LLMKL method gains the best performance among all of the compared methods on all datasets. More specifically, the mean ACC of the LLMKL is higher than the second best method SCMK by 7.3%, 13.1%, 14.4%, 30.9%, 4.5%, 9.8%, 16.9%, 3.9%, and 11.1% on the Yale, Jaffe, ORL, AR, COIL20, BA, TR11, TR41, and TR45 datasets, respectively. Especially on the AR database, LLMKL outperforms the latest MKL methods, SCMK, LKGr and SMKL, by over 30.9%, 53.9%, 59.0% improvements in term of ACC, respectively. Similar improvements can also be observed when using other metrics.

(2) Among all the MKL methods, the proposed LLMKL method has the smallest standard deviation, meaning that LLMKL has good stability. More specifically, the standard deviation of LLMKL is less than 1.0% on all datasets. In particular, for Jaffe and TR45 databases, the standard deviations are zero.

(3) Compared to the related method, LKGr, the proposed LLMKL method achieves significant improvements, the average improvements are 14.69%, 9.08%, 13.72% in terms of ACC, NMI, and purity, respectively. This demonstrates the necessity and advantage of the introduced substitute of consensus kernel and local structural graph. As we discussed earlier, the low-rank regularization term (i.e., $\|\boldsymbol{H}\|_*$) of LKGr cannot encourage the data in kernel feature space to be low-rank, but in our method,

the low-rank constraint on the substitute of consensus kernel (i.e., $\|\boldsymbol{B}\|_*$) really can capture the global structure of the input data.

(4) In most cases, the spectral clustering-based methods, AASC, LKGr, SMKL, SCMK, are superior to the k-means based methods in general, such as MKKM and RMKKM. It indicates that the spectral clustering based method usually achieves better performance than the standard k-means based one.

(5) Our model contains three regularization terms, i.e., $\mathrm{Tr}(\boldsymbol{D}^T\boldsymbol{Z})$, $\|\boldsymbol{B}\|_*$, and $\|\boldsymbol{E}\|_1$. It is very difficult to analyze them separately because they work together. However, we can explain their contributes through experience. For $\mathrm{Tr}(\boldsymbol{D}^T\boldsymbol{Z})$, the graph learning has been widely studied, the effect of local structural graph has been verified for clustering [22]. For $\|\boldsymbol{E}\|_1$, integrating a noisy component $\boldsymbol{E}$ is a universal technology, which can alleviate impacts by noise and outliers [37]. For $\|\boldsymbol{B}\|_*$, LRR is designed to find underlying structures of noisy data [19,38]; moreover, due to $\boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B}$, the better $\boldsymbol{B}$ is, the better the quality of consensual kernel $\boldsymbol{H}$ is. From Table 2, it can be seen that the biggest improvements are the experiments on the ORL and AR datasets, the improvements of LLMKL over the second best, SCMK, are 14.4% and 30.9%, respectively. The main reason is that low-rank constraint $\|\boldsymbol{B}\|_*$ can better expose the underlying global structure of samples in kernel space, which helps us to remove corrupted parts greatly (such as expressions and myopia glasses). Therefore, $\mathrm{Tr}(\boldsymbol{D}^T\boldsymbol{Z})$ and $\|\boldsymbol{B}\|_*$ can encourage LLMKL to learn a better graph affinity matrix and a better consensual kernel, while $\|\boldsymbol{B}\|_*$ and $\|\boldsymbol{E}\|_1$ can improve the robustness to noise and outliers.

(6) From Fig. 3, it can be seen that our proposed LLMKL obtains a very dense affinity matrix with better 10-block diagonal structure. Therefore, we believe that the local structural graph can encourage the affinity matrix to be or close to be appropriate block diagonal.

3.5. Effectiveness of low-rank substitute kernel

As seen in Eqs. (4) and (5), the low-rank kernel $\|\boldsymbol{H}\|_*$ cannot lead to low-rank of $\phi(\boldsymbol{X})$, but the low-rank substitute kernel $\|\boldsymbol{B}\|_*$ can. Therefore, we can get the conclusion: $\|\boldsymbol{H}\|_* \to \|\phi(\boldsymbol{X})\|_F^2$ and $\|\boldsymbol{B}\|_* \to \|\phi(\boldsymbol{X})\|_*$. To prove this point, we propose the following two objective functions to compare the performance of $\|\boldsymbol{H}\|_*$ and $\|\boldsymbol{B}\|_*$.

Firstly, we introduce $\|\boldsymbol{H}\|_*$ and propose the following objective function:

$$\begin{aligned}\min_{\boldsymbol{Z},\boldsymbol{H}} &\frac{1}{2}\left[\mathrm{Tr}(\boldsymbol{H} - 2\boldsymbol{H}\boldsymbol{Z} + \boldsymbol{Z}^T\boldsymbol{H}\boldsymbol{Z})\right] \\ &+ \alpha\|\boldsymbol{Z}\|_F^2 + \beta\|\boldsymbol{H}\|_* + \frac{\gamma}{2}\|\boldsymbol{H} - \boldsymbol{H}_i\|_F^2\end{aligned} \tag{28}$$

where α, β and γ are penalty parameters. This problem is named Low-Rank Kernel (LRK), which can be solved by ADMM.

Secondly, we introduce $\|\boldsymbol{B}\|_*$ and propose the following objective function:

$$\begin{aligned}\min_{\boldsymbol{Z},\boldsymbol{B}} &\frac{1}{2}\left[\mathrm{Tr}(\boldsymbol{I} - 2\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{Z}^T)\boldsymbol{B}^T\boldsymbol{B}\right] \\ &+ \alpha\|\boldsymbol{Z}\|_F^2 + \beta\|\boldsymbol{B}\|_* + \frac{\gamma}{2}\|\boldsymbol{B}^T\boldsymbol{B} - \boldsymbol{H}_i\|_F^2\end{aligned} \tag{29}$$

where α, β and γ are penalty parameters, $\boldsymbol{B}$ is the substitute of $\boldsymbol{H}$ (i.e., $\boldsymbol{H} = \boldsymbol{B}^T\boldsymbol{B}$). This problem is named Low-Rank Substitute Kernel (LRSK), which can be solved by an alternative iterative algorithm.

We evaluate the performance of LRK and LRSK on the 12 kernels from the Yale dataset, where these kernels are constructed according to Section 3.3, and the parameters α, β, and γ are set as 1, 3, and 10, respectively. The clustering performance and the rank of the learned optimal kernel $\boldsymbol{H}$ are reported in Table 3. As seen, LRSK obtains the highest performance and the lowest rank as compare to LRK in most cases, which demonstrates that $\|\boldsymbol{H}\|_*$ cannot lead to low-rank of $\phi(\boldsymbol{X})$, but $\|\boldsymbol{B}\|_*$ can.

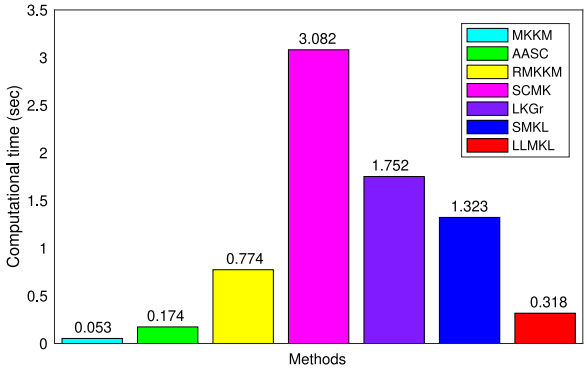


Fig. 4. Computational time (in seconds) of the compared MKL methods on the Yale dataset.

3.6. Timing experiments

Furthermore, we give the average computation time (seconds) of all compared MKL methods (i.e., MKKM, AASC, RMKKM, SCMK, LKGr, and SMKL). Our experimental platform is a Mac mini 2018 with an Intel Core i7 (3.2 GHz) CPU and 16-GB RAM, the operating system is macOS Mojave, and the simulation software is MATLAB 2016b. Due to space limit, we only report the computation time of all compared MKL methods on the Yale dataset, which is presented in Fig. 4. The result suggests that our method is much faster than other methods, except MKKM and AASC. Note that MKKM and AASC are faster due to their simple learning strategy, but their clustering performance are significantly lower than the proposed method LLMKL and the latest methods (i.e., SCMK, LKGr, and SMKL). Besides, SCMK ranks second in clustering performance, but has the highest computational cost. The main reason is that SCMK contains two sub-algorithms in each iterative. Therefore, our method is a fast and efficient MKL clustering method.

3.7. Parameter sensitivity and convergence analysis

In the proposed method, there are four essential parameters λ_i ($i = 1, 2, 3, 4$) that are used to balance the effects of loss terms and regularization terms. In order to test how the four parameters affect the performance, we consider a range of possible values for λ_i ($i = 1, 2, 3, 4$), and select the one with the best clustering performance.

Take the widely used Yale database for example, we first fix $\lambda_1 = 10^{-2}$, $\lambda_2 = 10^5$, and then tune $\lambda_3 \in [10^{-3}, 10^{-2}, 10^{-1}, 0.2, \ldots, 1.0, 10^1, 10^2]$ and $\lambda_4 \in [10^{-4}, 10^{-3}, \ldots, 10^3, 10^4]$. Fig. 5 shows the ACC of the proposed LLMKL versus the parameters λ_3 and λ_4. We can see that our algorithm is not sensitive to λ_4 (i.e., λ_4 can be selected from $[10^1, \ldots, 10^4]$). Besides, the results suggest that small λ_3 and large λ_3 both degrade the performance. In fact, we only need to tune λ_3 from $[0.1, 0.2, \ldots, 0.9, 1.0]$ for all databases basing on the experience.

Next, we fix $\lambda_3 = 0.3$, $\lambda_4 = 10^2$, and then tune $\lambda_1 \in [10^{-5}, 10^{-4}, \ldots, 1]$ and $\lambda_2 \in [10^{-3}, 10^{-2}, \ldots, 10^4, 10^5]$. Fig. 6 shows the ACC of the proposed LLMKL versus the parameters λ_1 and λ_2. From Fig. 6, we can see that LLMKL can achieve the desired effect when $\lambda_1 \in [10^{-5}, 10^1]$ and $\lambda_2 \in [10^1, 10^5]$. Besides, when the parameter $\lambda_1 > 1$, the performance of LLMKL will drop sharply.

Please cite this article as: Z. Ren, H. Li, C. Yang et al., Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning, Knowledge-Based Systems (2019) 105040, https://doi.org/10.1016/j.knosys.2019.105040.

Table 3
Clustering results of LRK and LRSK methods on the 12 kernels from the Yale dataset.

Method	Metrics	$\boldsymbol{H}_1$	$\boldsymbol{H}_2$	$\boldsymbol{H}_3$	$\boldsymbol{H}_4$	$\boldsymbol{H}_5$	$\boldsymbol{H}_6$	$\boldsymbol{H}_7$	$\boldsymbol{H}_8$	$\boldsymbol{H}_9$	$\boldsymbol{H}_{10}$	$\boldsymbol{H}_{11}$	$\boldsymbol{H}_{12}$
LRK	ACC	0.560	0.278	0.200	0.485	**0.258**	**0.207**	0.213	**0.277**	0.216	0.562	0.566	0.267
	NMI	0.571	0.324	0.233	0.510	**0.338**	**0.241**	0.259	**0.369**	0.257	0.569	0.566	0.297
	Purity	0.589	0.296	**0.727**	**0.673**	0.285	0.232	0.231	0.318	0.233	0.589	0.602	0.293
	rank($\boldsymbol{H}$)	165	165	165	165	165	165	165	165	165	165	165	165
LRSK	ACC	**0.586**	**0.412**	**0.224**	**0.501**	0.165	0.175	**0.534**	0.194	**0.531**	**0.577**	**0.575**	**0.588**
	NMI	**0.603**	**0.499**	**0.277**	**0.531**	0.214	0.220	**0.560**	0.230	**0.549**	**0.569**	**0.582**	**0.606**
	Purity	**0.639**	**0.454**	0.622	0.588	**0.724**	**0.721**	**0.571**	**0.737**	**0.581**	**0.609**	**0.618**	**0.633**
	rank($\boldsymbol{H}$)	**67**	**165**	**165**	**165**	**1**	**1**	**12**	**1**	**35**	**77**	**134**	**34**

Note that the rank of the ith input kernel equals 165 (i.e., rank($\boldsymbol{H}_i$) = 165).

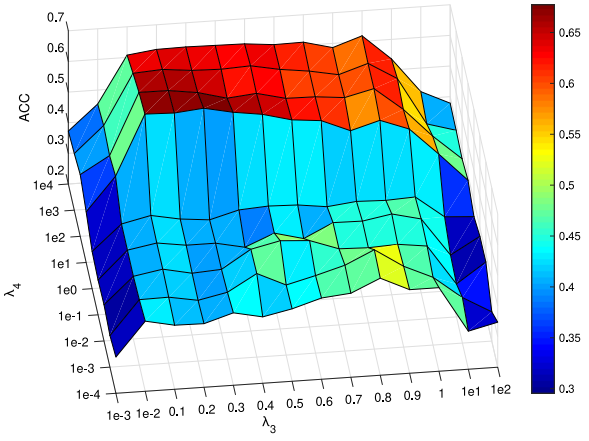


Fig. 5. The clustering accuracy of LLMKL on the Yale dataset versus λ_3 and λ_4 when fix $\lambda_1 = 10^{-2}$ and $\lambda_2 = 10^5$.

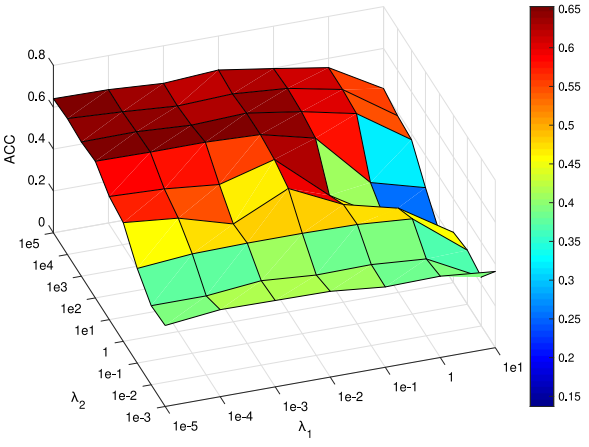


Fig. 6. The clustering accuracy of LLMKL on the Yale dataset versus λ_1 and λ_2 when fix $\lambda_3 = 0.3$ and $\lambda_4 = 10^2$.

Therefore, if the parameters are within the appropriate ranges, LLMKL can obtain better performance. These results demonstrate that LLMKL is stable and easier to tame. Further, we examine the convergence of LLMKL. We compute the values of object function and the primal residuals of (10) at each iteration, where the primal residual is defined as $\|\boldsymbol{H}^{t+1} - \boldsymbol{H}^t\|_F^2$. Fig. 7 shows the convergence curve and the primal residuals curve. It can be seen that LLMKL converges within 6 iterations.

4. Conclusions and further work

In this paper, we proposed an efficient method named LLMKL for multiple kernel subspace clustering, which integrates the MKL technology, the global and local structure learning, and the

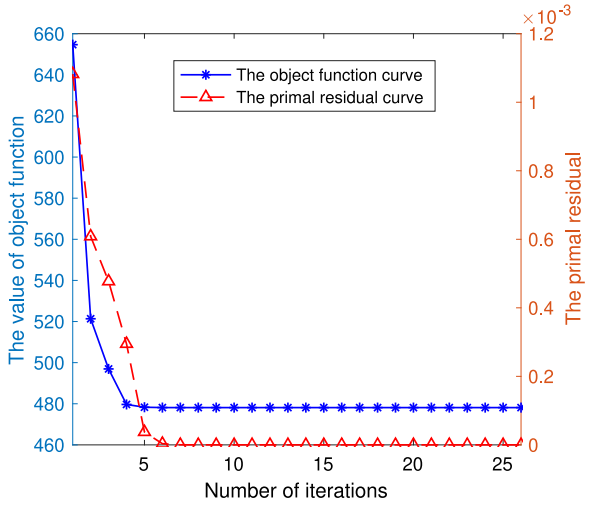


Fig. 7. The convergence curve and the primal residual curve of our algorithm on the Yale database.

Hilbert space self-expressiveness property in a unified optimization problem. A novel algorithm based ADMM was developed to solve the optimization problem. Furthermore, the sensitivity of the trade-off parameters and the convergence of LLMKL were verified. Adequate experimental results on nine datasets shown that LLMKL is superior to the state-of-the-arts.

In the future we hope to address three areas where LLMKL can be improved: (1) Applying Schatten p-Norm or Weighted Schatten p-norm to more accurately approximate the rank function. (2) Planing to further extend LLMKL to handle a large-scale multi-view [39] subspace clustering problem. (3) Exploring other robust structure learning technology to excavate the intrinsic structure of the input data for clustering purpose.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant No. 61673220), Department of Science and Technology of Sichuan Province, China (No. ZYF-2018-106), Sichuan Province Science and Technology Support Program, China (No. 2018TZDZX0002), and State Administration for Science, Technology and Industry for National Defense, China (Nos. JCKY2017209B010 and JCKY2018209B001).

The authors would like to thank Liang Du, who shared their RMKKM code on their homepage, and to thank Zhao Kang, who shared the codes of SMKL and SCMK. The authors would also like to thank the anonymous reviewers who provided substantive suggestions for improving our work.

Appendix

The minimization problem of $\boldsymbol{Z}$ is as follows:

$$\min_{\boldsymbol{Z}\geq 0,\mathrm{diag}(\boldsymbol{Z})=0,\boldsymbol{Z}^T\mathbf{1}=\mathbf{1}} \|\boldsymbol{Z}-\widehat{\boldsymbol{Z}}\|_F^2 \tag{30}$$

which is equivalent to the following expanding minimization problem:

$$\min_{\boldsymbol{Z}\geq 0,\mathrm{diag}(\boldsymbol{Z})=0,\boldsymbol{Z}^T\mathbf{1}=\mathbf{1}} \sum_{i=1}^{n}\sum_{j=1}^{n}(\boldsymbol{Z}_{ij}-\widehat{\boldsymbol{Z}}_{ij})^2. \tag{31}$$

It can be seen that (31) is independent for different j. So each column $\boldsymbol{Z}_j$ is calculated via

$$\min_{\boldsymbol{Z}\geq 0,\mathrm{diag}(\boldsymbol{Z})=0,\boldsymbol{Z}^T\mathbf{1}=\mathbf{1}} \sum_{j=1}^{n}(\boldsymbol{Z}_j-\widehat{\boldsymbol{Z}}_j)^2. \tag{32}$$

To calculate $\boldsymbol{Z}_j$, we first transform Eq. (32) into the following Lagrangian function

$$\mathcal{L}(\boldsymbol{Z}_j,\alpha_j,\beta_j)=\frac{1}{2}\|\boldsymbol{Z}_j-\widehat{\boldsymbol{Z}}_j\|_2^2-\alpha_j(\boldsymbol{Z}_j^T\mathbf{1}-1)-\beta_j^T\boldsymbol{Z}_j \tag{33}$$

where α_j and $\beta_j>0$ are the Lagrangian multipliers.

By setting the derivative of Eq. (33) with respect to $\boldsymbol{Z}_j$ to zero, we have

$$\boldsymbol{Z}_j-\widehat{\boldsymbol{Z}}_j-\alpha_j\mathbf{1}-\beta_j=0. \tag{34}$$

According to the Karush–Kuhn–Tucker (KKT) condition that $\beta_j\odot\boldsymbol{Z}_j=0$ [36], we have

$$\boldsymbol{Z}_j=[\widehat{\boldsymbol{Z}}_j+\alpha_j\mathbf{1}]_+,\boldsymbol{Z}_{jj}=0. \tag{35}$$

According to the affine constraint $\boldsymbol{Z}_j^T\mathbf{1}=\mathbf{1}$, we obtain

$$\sum_{i=1}^{n}(\alpha_j+\widehat{\boldsymbol{Z}}_{ij})=1\rightarrow\alpha_j=(1+\widehat{\boldsymbol{Z}}_j^T\mathbf{1})/n \tag{36}$$

For each column, after obtaining α_j, $\boldsymbol{Z}_j$ can be obtained by Eq. (36) so that the optimal solution $\boldsymbol{Z}$ is obtained.

The proof is completed.

References

[1] M.C. Tsakiris, R. Vidal, Algebraic clustering of affine subspaces, IEEE Trans. Pattern Anal. Mach. Intell. 40 (2) (2018) 482–489.
[2] R. Zhu, J.-H. Xue, On the orthogonal distance to class subspaces for high-dimensional data classification, Inform. Sci. 417 (2017) 262–273.
[3] W. Wang, J. Shen, R. Yang, F. Porikli, Saliency-aware video object segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 40 (1) (2017) 20–33.
[4] X. Shi, Z. Guo, F. Xing, J. Cai, L. Yang, Self-learning for face clustering, Pattern Recognit. 79 (2018) 279–289.
[5] S. Huang, Z. Xu, J. Lv, Adaptive local structure learning for document co-clustering, Knowl.-Based Syst. 148 (2018) 74–84.
[6] F. Bu, A high-order clustering algorithm based on dropout deep learning for heterogeneous data in cyber-physical-social systems, IEEE Access 6 (2018) 11687–11693.
[7] X. Peng, J. Feng, S. Xiao, W.-Y. Yau, J.T. Zhou, S. Yang, Structured autoencoders for subspace clustering, IEEE Trans. Image Process. 27 (10) (2018) 5076–5086.
[8] Y. Pan, H. Yu, Biomimetic hybrid feedback feedforward neural-network learning control, IEEE Trans. Neural Netw. Learn. Syst. 28 (6) (2016) 1481–1487.
[9] Z. Bu, J. Cao, H.-J. Li, G. Gao, H. Tao, Gleam: a graph clustering framework based on potential game optimization for large-scale social networks, Knowl. Inf. Syst. 55 (3) (2018) 741–770.
[10] C. Lu, J. Feng, Z. Lin, T. Mei, S. Yan, Subspace clustering by block diagonal representation, IEEE Trans. Pattern Anal. Mach. Intell. 41 (2) (2018) 487–501.
[11] D. Tolić, N. Antulov-Fantulin, I. Kopriva, A nonlinear orthogonal non-negative matrix factorization approach to subspace clustering, Pattern Recognit. 82 (2018) 40–55.
[12] P. Purkait, T.-J. Chin, A. Sadri, D. Suter, Clustering with hypergraphs: the case for large hyperedges, IEEE Trans. Pattern Anal. Mach. Intell. 39 (9) (2017) 1697–1711.
[13] H. Wang, G. Xiao, Y. Yan, D. Suter, Searching for representative modes on hypergraphs for robust geometric model fitting, IEEE Trans. Pattern Anal. Mach. Intell. 41 (3) (2019) 697–711.
[14] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, J. Long, A survey of clustering with deep learning: From the perspective of network architecture, IEEE Access 6 (2018) 39501–39514.
[15] M. Yin, J. Gao, Z. Lin, Q. Shi, Y. Guo, Dual graph regularized latent low-rank representation for subspace clustering, IEEE Trans. Image Process. 24 (12) (2015) 4918–4933.
[16] C. Yang, Z. Ren, Q. Sun, M. Wu, M. Yin, Y. Sun, Joint correntropy metric weighting and block diagonal regularizer for robust multiple kernel subspace clustering, Inform. Sci. 500 (2019) 48–66.
[17] C. Peng, Z. Kang, Y. Hu, J. Cheng, Q. Cheng, Robust graph regularized nonnegative matrix factorization for clustering, ACM Trans. Knowl. Discov. Data (TKDD) 11 (3) (2017) 33.
[18] Y. Chen, Z. Yi, Locality-constrained least squares regression for subspace clustering, Knowl.-Based Syst. 163 (2019) 51–56.
[19] X. Zhu, S. Zhang, Y. Li, J. Zhang, L. Yang, Y. Fang, Low-rank sparse subspace for spectral clustering, IEEE Trans. Knowl. Data Eng. 31 (8) (2019) 1532–1543.
[20] Z. Kang, C. Peng, Q. Cheng, Z. Xu, Unified spectral clustering with optimal graph, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[21] J. Wen, B. Zhang, Y. Xu, J. Yang, N. Han, Adaptive weighted nonnegative low-rank representation, Pattern Recognit. 81 (2018) 326–340.
[22] H. Wang, Y. Yang, B. Liu, H. Fujita, A study of graph-based system for multi-view clustering, Knowl.-Based Syst. 163 (2019) 1009–1019.
[23] X. Xie, X. Guo, G. Liu, J. Wang, Implicit block diagonal low-rank representation, IEEE Trans. Image Process. 27 (1) (2017) 477–489.
[24] Z. Kang, X. Lu, J. Yi, Z. Xu, Self-weighted multiple kernel learning for graph-based clustering and semi-supervised classification, arXiv preprint arXiv:1806.07697, 2018.
[25] Y. Gu, T. Liu, X. Jia, J.A. Benediktsson, J. Chanussot, Nonlinear multiple kernel learning with multiple-structure-element extended morphological profiles for hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 54 (6) (2016) 3235–3247.
[26] H.-C. Huang, Y.-Y. Chuang, C.-S. Chen, Multiple kernel fuzzy clustering, IEEE Trans. Fuzzy Syst. 20 (1) (2012) 120–134.
[27] L. Du, P. Zhou, L. Shi, H. Wang, M. Fan, W. Wang, Y.-D. Shen, Robust multiple kernel k-means using l21-norm, in: IJCAI, 2015, pp. 3476–3482.
[28] H.-C. Huang, Y.-Y. Chuang, C.-S. Chen, Affinity aggregation for spectral clustering, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 773–780.
[29] Z. Kang, L. Wen, W. Chen, Z. Xu, Low-rank kernel learning for graph-based clustering, Knowl.-Based Syst. 163 (2019) 510–517.
[30] X. Zhang, H. Sun, Z. Liu, Z. Ren, Q. Cui, Y. Li, Robust low-rank kernel multi-view subspace clustering based on the schatten p-norm and correntropy, Inform. Sci. 477 (2019) 430–447.
[31] S. Yi, Y. Liang, Z. He, Y. Li, W. Liu, Y.-m. Cheung, Dual pursuit for subspace learning, IEEE Trans. Multimed. 21 (6) (2019) 1399–1411.
[32] Y. Zhang, Y. Yang, T. Li, H. Fujita, A multitask multiview clustering algorithm in heterogeneous situations based on lle and le, Knowl.-Based Syst. 163 (2019) 776–786.
[33] B. Li, H. Lu, Y. Zhang, Z. Lin, W. Wu, Subspace clustering under complex noise, IEEE Trans. Circuits Syst. Video Technol. (2018).
[34] M. Hong, Z.-Q. Luo, M. Razaviyayn, Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems, SIAM J. Optim. 26 (1) (2016) 337–364.
[35] M. Iliadis, H. Wang, R. Molina, A.K. Katsaggelos, Robust and low-rank representation for fast face identification with occlusions, IEEE Trans. Image Process. 26 (5) (2017) 2203–2218.
[36] J. Wen, X. Fang, Y. Xu, C. Tian, L. Fei, Low-rank representation with adaptive graph regularization, Neural Netw. 108 (2018) 83–96.
[37] R. He, Y. Zhang, Z. Sun, Q. Yin, Robust subspace clustering with complex noise, IEEE Trans. Image Process. 24 (11) (2015) 4001–4013.
[38] S. Li, Y. Fu, Learning robust and discriminative subspace with low-rank constraints, IEEE Trans. Neural Netw. Learn. Syst. 27 (11) (2015) 2160–2173.
[39] Q. Xiao, J. Dai, J. Luo, H. Fujita, Multi-view manifold regularized learning-based method for prioritizing candidate disease mirnas, Knowl.-Based Syst. 175 (2019) 118–129.

Please cite this article as: Z. Ren, H. Li, C. Yang et al., Multiple kernel subspace clustering with local structural graph and low-rank consensus kernel learning, Knowledge-Based Systems (2019) 105040, https://doi.org/10.1016/j.knosys.2019.105040.