IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 22, 2019, accepted March 9, 2019, date of publication March 25, 2019, date of current version May 22, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2904887

Efficient Multiple Kernel k-Means Clustering With Late Fusion

SIWEI WANG[1], EN ZHU[1], JINGTAO HU[1], MIAOMIAO LI[1], KAIKAI ZHAO[2], NING HU[3] AND XINWANG LIU[1], (Member, IEEE)

[1]School of Computer, National University of Defense Technology, Changsha 410073, China
[2]Institute of Information Fusion, Naval Aviation University, Yantai 264001, China
[3]Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China

Corresponding authors: En Zhu (enzhu@nudt.edu.cn) and Ning Hu (huning@gzhu.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2018YFB1003203 and in part by the National Natural Science Foundation of China under Grant 61672528 and Grant 61773392.

ABSTRACT The recently proposed multiple-kernel clustering algorithms have demonstrated promising performance in various applications. However, most of the existing methods suffer from high computational complexity and intensive time cost. To address this issue, we propose to fulfill multiple kernel k-means clustering via a late fusion manner. In specific, we design two multiple kernel k-means algorithms with late fusion, whose computational complexities linearly grow with the number of samples. The proposed algorithms integrally optimize the various clustering matrices into the optimal consensus clustering results iteratively. Furthermore, we analyze the computational complexities of the proposed algorithms and theoretically prove their convergence. As demonstrated by the experiments on six benchmark datasets, our algorithms achieve comparable or better clustering performance to state-of-the-art ones with less time cost, which demonstrates the advantages of the late fusion in multiple kernel k-means.

INDEX TERMS Multiple kernel clustering, multiple view clustering, late fusion.

I. INTRODUCTION

Clustering is one of the fundamental learning tasks in machine learning and data mining communities. Among the existing clustering algorithms, the k-means algorithm has been widely applied to many academic researches and real applications. The k-means algorithm follows a two-step iteration prototype: i) setting k landmarks as cluster centers, and assigning samples to k clusters based on the k landmarks; ii) updating the assignment matrix by minimizing the sum of within-cluster distances and computing the new landmarks. The two steps are run iteratively until stopping criterion is satisfied. To improve the representation ability, the kernel k-means algorithms map the original data to a high-dimensional space which is linearly separable and more friendly to learning tasks [1]–[4]. This extension enhances the k-means algorithm to handle the linearly non-separable problem in original space through feature mapping.

Although the kernel k-means algorithms achieve great success in various applications, most of them are proposed to handle the data with single view. However, the samples are presented in various forms or views of data in many real-world applications. For example, for web-page classification, the sample usually has two or more types of data, e.g., text, hyper-links and images, each of which can be seen as one view to the data. Many researchers have proposed various methods to combine the comprehensive information collected from each view, which is known as multi-view learning in literature [5], [6]. For kernel method, each view can be represented by a kernel matrix and the weight of every single kernel matrix can be considered as its contribution to the whole view. Along this direction, many multiple-kernel clustering algorithms have been proposed to solve multi-view clustering in recent literature [7]–[15]. In [8], a three-step alternate algorithm named Nonlinear Adaptive Metric Learning (NAML) is proposed to jointly optimize clustering, the kernel coefficients and dimension reduction based on the metric of Mahalanobis distance. In [9], a novel optimization kernel k-means algorithm is applied to collect multiple data sources from various views for clustering performance. In [10] and [16] they design a localized kernel k-means clustering algorithm to adapt to locally-similar

The associate editor coordinating the review of this manuscript and approving it for publication was Hazrat Ali.

samples by altering the kernels' weights respectively. Following this strategy, a multiple kernel k-means clustering algorithm with matrix-induced regularization has been proposed to reduce the redundancy and enhance the diversity of the pre-defined kernels [11]. Furthermore, the local kernel alignment criterion has been applied to multiple kernel learning obeying the principle that the closer sample pairs shall stay together and the similarity evaluations for farther sample pairs are unreliable [12]. Those aforementioned multiple kernel k-means clustering algorithms have shown promising clustering performance and are widely used to practical applications. However, most of them suffer from high computational complexity and long training time which make them infeasible in medium or larger applications.

To address this issue, we propose a novel multiple-kernel clustering framework via a late fusion manner. Late fusion has been widely applied in computer vision and document classification [17]–[21]. We avoid the massive computation on eigenvalue decomposition of kernel matrix during the clustering process, which significantly reduces the time complexity. More specially, our algorithms' time complexities grow linearly with the sample number comparing to the former cubical growing rate. Collecting comprehensive information from multiple views, the ideal consensus clustering matrix is aligned with the different views' clustering results. Although NAML is also based on multiple kernel extension of kernel k-means clustering, the mathematical objective and the solution are different from our methods. In NAML, the metric of k-means is constructed based on the Mahalanobis distance while our approach is constructed in euclidean space. Moreover, instead of calculating the new combined kernel matrix, our algorithms only optimize the cluster assignments and kernel coefficients in an alternate procedure.

In order to implement the framework mentioned above, we propose two novel multiple kernel k-means algorithms with fast convergence, which we name them *average multiple kernel k-means with late fusion* (Average-MKKM-LF) and *Adaptive multiple kernel k-means with late fusion* (Adaptive-MKKM-LF) respectively. The *average multiple kernel k-means with late fusion* (Average-MKKM-LF) equally considers each view's contributions to the clustering performance and integrates the various clustering assignment matrices respectively instead of the optimal kernel for clustering in former multiple kernel framework. Moreover, the weight of each view should be allowed to adaptively change with various data sources. Hence we propose the *Adaptive multiple kernel k-means with late fusion* (Adaptive-MKKM-LF) to adaptively weighted with different perspectives of views. To solve the resultant optimization problem, we develop two efficient algorithms with proved convergences. Extensive experimental study has been conducted on six MKL benchmark datasets to evaluate clustering performance of the proposed algorithms. As indicated, our algorithms have small time-cost and consistently demonstrate comparable or better performance with the several state-of-the-art ones. Moreover, the carefully designed optimization goals have a very fast rate of convergence, i.e., usually less than 10 times in benchmark datasets. This verifies the effectiveness and superiority of late fusion in our algorithms.

Our contributions in this paper can be summarized as follows:

(i) We propose a multiple kernel clustering framework via a late fusion manner, which are supposed to integrate the various clustering indicator matrices produced by different views respectively instead of the optimal kernel for clustering in former multiple kernel framework. We join the clustering process and the optimization assignment into one optimization problem. To the best of our knowledge, it is the first time that late fusion is adopted into kernel method in order to enhance the diversity of the clustering results and reduce the time complexity of multiple-kernel clustering algorithms.

(ii) In order to implement the proposed framework, two novel average and adaptive approaches (*average multiple kernel k-means with late fusion* (Average-MKKM-LF)) and (*adaptive multiple kernel k-means with late fusion* (Adaptive-MKKM-LF)) with carefully designed deterministic fusion optimization goals are proposed for solving our optimization problem in multiple kernel k-means clustering with proved convergence. We theoretically demonstrate that the two algorithms' time complexities linearly grow on the size of sample number.

(iii) Extensive empirical study has been conducted on six MKL benchmark datasets. As indicated, our algorithms consistently demonstrate comparable performance with several the state-of-the-art ones or even better which validate the advantage of the proposed multiple kernel clustering framework with late fusion.

The rest of this paper is organized as follows. Section II outlines the related work of multiple kernel clustering. Section III presents the proposed optimization objective and the three-step alternate algorithms. Section IV analyses the convergence and the computational complexity of our two proposed algorithms. Section V shows the experiment results with evaluation. Section VI concludes the paper.

II. RELATED WORKS

A. KERNEL k-MEANS CLUSTERING(KKM)

Let $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X}$ be a collection of n training samples. The optimization goal of kernel k-means clustering algorithm is to minimize the square loss of the within-cluster distance in the transformed space. And the feature mapping function $\phi(x)$ transfers the origin sample $\mathbf{x}$ into a reproducing kernel Hilbert space $\mathcal{H}$ which is a k-means-friendly space and easier to cluster. By supposing the cluster indicator matrix $Z \in \{0,1\}^{n\times k}$, the optimization objective of KKM could be written as follows:

$$\min_{Z\in\{0,1\}^{n\times k}} \mathbf{Z}_{ic}\|\phi(X_i) - \mu_c\|^2 \quad s.t. \sum_{c=1}^{k} Z_{ic} = 1. \tag{1}$$

where $n_c = \sum_{i=1}^{n} Z_{ic}$ and $\mu_c = \frac{1}{n_c}\sum_{i=1}^{n} Z_{ic}\phi(X_i)$ are the number and centroid of the $c-th(1 \leq c \leq k)$ cluster respectively.

By equivalently rewritten in matrix-vector form, the function in Eq. (1) is transformed to the following problem,

$$\min_{Z\in\{0,1\}^{n\times k}} \mathrm{Tr}(\mathbf{K}) - \mathrm{Tr}(\mathbf{L}^{\frac{1}{2}}\mathbf{Z}^{\top}\mathbf{K}\mathbf{Z}\mathbf{L}^{\frac{1}{2}}) \quad s.t.\ \mathbf{Z}1_k = 1_n. \tag{2}$$

Here, we apply the kernel matrix to the Eq.(1), and $\mathbf{K}$ denotes the kernel matrix and $\mathbf{L} = diag([n_1^{-1}, n_2^{-1}, \cdot, \cdot, \cdot, n_k^{-1}])$.

Directly solving the optimization problem in Eq. (2) is difficult for the reason that the element in matrix $\mathbf{L}$ is discrete. We relax $\mathbf{L}$ to take real values, by letting the new matrix $\mathbf{H}$ follows that $\mathbf{H} = \mathbf{Z}\mathbf{L}^{\frac{1}{2}}$. Then we rewrite the problem in Eq. (2),

$$\min_{\mathbf{H}\in\mathbb{R}^{n\times k}} \mathrm{Tr}(\mathbf{K}(\mathbf{I_n} - \mathbf{H}\mathbf{H}^{\top})) \quad s.t.\ \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k, \tag{3}$$

From the formula setting proposed in Eq. (1), the kernel k-means can correctly identify and extract a far more varied collection of cluster structures than the linear k-means clustering algorithm through the non-linear feature mapping. The optimization problem in Eq. (3) could be solved by singular value decomposition(SVD) of the kernel matrix $\mathbf{K}$ [11].

However, the clustering performance of kernel k-means mostly depends on the pre-specified kernel matrix. For most of the applications in real life, it is hard for researchers to set a clustering-friendly kernel matrix for the lack of prior knowledge. Hence the multiple-kernel k-means clustering is proposed to enhance the representation ability of kernel k-means in a weighted multiple-kernel setting.

B. MULTI-KERNEL *k*-MEANS (MKKM)

In the multiple kernel setting, we suppose that $\{\mathbf{x}_i\}_{i=1}^{n} \subseteq \mathcal{X}$ is a collection of n samples, and $\phi_p(\cdot) : \mathbf{x} \in \mathcal{X} \mapsto \mathcal{H}_p$ be the p-th feature mapping which transfers $\mathbf{x}$ into a reproducing kernel Hilbert space $\mathcal{H}_p$ $(1 \leq p \leq m)$. Hence each sample is represented as $\phi_{\boldsymbol{\beta}}(\mathbf{x}) = [\beta_1\phi_1(\mathbf{x})^{\top}, \cdots, \beta_m\phi_m(\mathbf{x})^{\top}]^{\top}$ from m views, where $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_m]^{\top}$ consists of the coefficients of the m base kernels $\{\kappa_p(\cdot,\cdot)\}_{p=1}^{m}$. These coefficients will be optimized during learning. Based on the definition of $\phi_{\boldsymbol{\beta}}(\mathbf{x})$, a kernel function can be expressed as

$$\kappa_{\boldsymbol{\beta}}(\mathbf{x}_i, \mathbf{x}_j) = \phi_{\boldsymbol{\beta}}(\mathbf{x}_i)^{\top}\phi_{\boldsymbol{\beta}}(\mathbf{x}_j) = \sum_{p=1}^{m}\beta_p^2\kappa_p(\mathbf{x}_i, \mathbf{x}_j). \tag{4}$$

A kernel matrix $\mathbf{K}_{\boldsymbol{\beta}}$ is then calculated by applying the kernel function $\kappa_{\boldsymbol{\beta}}(\cdot,\cdot)$ into $\{\mathbf{x}_i\}_{i=1}^{n}$. By using the notation that kernel matrix $\mathbf{K}_{\boldsymbol{\beta}}$, the optimization goal of MKKM algorithm can be expressed as

$$\begin{aligned} &\min_{\mathbf{H},\boldsymbol{\beta}} \mathrm{Tr}(\mathbf{K}_{\boldsymbol{\beta}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^{\top})) \\ &s.t.\ \mathbf{H} \in \mathbb{R}^{n\times k},\quad \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k,\ \boldsymbol{\beta}^{\top}\mathbf{1}_m = 1,\ \beta_p \geq 0,\ \forall p. \end{aligned} \tag{5}$$

where $\mathbf{I}_k$ is an identity matrix with size $k \times k$. The optimization problem in Eq. (5) can be solved by alternately updating $\mathbf{H}$ and $\boldsymbol{\beta}$: i) **Optimizing H by fixed $\boldsymbol{\beta}$**. With the kernel coefficients $\boldsymbol{\beta}$ fixed, $\mathbf{H}$ can be obtained by solving a kernel k-means clustering optimization problem shown in Eq. (6);

$$\max_{\boldsymbol{H}} \mathrm{Tr}(\mathbf{H}^{\top}\mathbf{K}_{\boldsymbol{\beta}}\mathbf{H}) \quad s.t.\ \mathbf{H} \in \mathbb{R}^{n\times k},\ \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k, \tag{6}$$

The optimal $\mathbf{H}$ for Eq. (6) can be obtained by taking the k eigenvectors corresponding to the largest k eigenvalues of $\mathbf{K}$. ii) **Optimizing $\boldsymbol{\beta}$ by fixed H**. With $\mathbf{H}$ fixed, $\boldsymbol{\beta}$ can be optimized via solving the following quadratic programming with linear constraints,

$$\min_{\boldsymbol{\beta}} \sum_{p=1}^{m}\beta_p^2\mathrm{Tr}(\mathbf{K}_p(\mathbf{I}_n - \mathbf{H}\mathbf{H}^{\top})) \quad s.t.\ \boldsymbol{\beta}^{\top}\mathbf{1}_m = 1, \quad \beta_p \geq 0. \tag{7}$$

As noted in [9] and [10], using a convex combination of kernels $\sum_{p=1}^{m}\beta_p\mathbf{K}_p$ to replace $\mathbf{K}_{\boldsymbol{\beta}}$ in Eq. (5) is not a valid option, because this could get sparse solution and only one single kernel is performed while all the others given with zero weights. As indicated, the multiple-kernel k-means mainly promote the diversity of a set of single kernel k-means and the weight vector $\boldsymbol{\beta}$ can be regarded as the different weights contributing to the clustering from each view respectively. The weight vector $\boldsymbol{\beta}$ and the clustering indicator matrix $\mathbf{H}$ are both optimized alternately during learning process.

C. MULTIPLE-KERNEL CLUSTERING WITH LOCAL KERNEL ALIGNMENT MAXIMIZATION (MKC-LKA)

In [12], the local kernel alignment criterion has been applied to multiple kernel learning following the motivation that the similar sample pairs shall stay more closer and the similarity evaluations for farther sample pairs are unreliable because of improper metric settings. Considering locally aligning the similarity of each sample to its k-nearest neighbors with corresponding ideal kernel matrix, in specific, the local kernel alignment for the i-th can be calculated as,

$$\begin{aligned} &\max_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\beta}\in\mathbb{R}_{+}^{m}} \frac{\langle\mathbf{K}_{\boldsymbol{\beta}}^{(i)}, \mathbf{H}^{(i)}\mathbf{H}^{(i)\top}\rangle}{\sqrt{\langle\mathbf{K}_{\boldsymbol{\beta}}^{(i)}, \mathbf{K}_{\boldsymbol{\beta}}^{(i)}\rangle}} \\ &s.t.\ \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k,\ \boldsymbol{\beta}^{\top}\mathbf{1}_m = 1. \end{aligned} \tag{8}$$

where $\langle\mathbf{K}_{\boldsymbol{\beta}}^{(i)}, \mathbf{H}^{(i)}\mathbf{H}^{(i)\top}\rangle = \mathrm{Tr}(\mathbf{K}_{\boldsymbol{\beta}}^{(i)\top}\mathbf{H}^{(i)}\mathbf{H}^{(i)\top})$,$\mathbf{K}_{\boldsymbol{\beta}}^{(i)}$ and $\mathbf{H}^{(i)}$ are the sub-matrix of $\mathbf{K}_{\boldsymbol{\beta}}$ and $\mathbf{H}$ whose indices are specified by the τ-nearest neighbors of the i-th sample, and $\mathbf{M}^{(i)}$ is a matrix with $\mathbf{M}_{pq}^{(i)} = \mathrm{Tr}(\mathbf{K}_p^{(i)\top}\mathbf{K}_q^{(i)})$.

The Eq. (8) can be conceptually expressed as,

$$\begin{aligned} &\min_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\beta}\in\mathbb{R}_{+}^{m}} \mathrm{Tr}(\mathbf{K}_{\boldsymbol{\beta}}^{(i)}(\mathbf{I}_{\tau} - \mathbf{H}^{(i)}\mathbf{H}^{(i)\top})) + \frac{\lambda}{2}\boldsymbol{\beta}^{\top}\mathbf{M}^{(i)}\boldsymbol{\beta} \\ &s.t.\ \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k, \quad \boldsymbol{\beta}^{\top}\mathbf{1}_m = 1. \end{aligned} \tag{9}$$

where $\mathbf{K}_{\boldsymbol{\beta}}^{(i)} = \mathbf{S}^{(i)\top}\mathbf{K}_{\boldsymbol{\beta}}\mathbf{S}^{(i)}$, $\mathbf{H}^{(i)} = \mathbf{S}^{(i)\top}\mathbf{H}$, $\mathbf{S}^{(i)} \in \{0,1\}^{n\times\tau}$ is a matrix indicating the τ-nearest neighbors of the i-th sample and $\mathbf{I}_{\tau}$ is an identity matrix with size $\tau \times \tau$.

Moreover, an alternate optimization algorithm is proposed to solve the problem in Eq. (9) and achieves the superior performance among several multiple-kernel clustering methods. However, the computational cost for that method is high as it suffers from long training time. To address this issue, in the following section, we apply the late fusion to multiple kernel k-means to have the optimal clustering results by integrating the multi-view results. It has comparable performances with multiple-kernel clustering with local kernel alignment maximization(MKC-LKA) and smaller time complexity with less training time. Moreover it provides the multiple-kernel learning with a new framework which could be easily applied to other fields.

III. MULTIPLE-KERNEL k-MEANS WITH LATE FUSION (MKKM-LF)

As mentioned in section II, our work is built on the multiple kernel k-means and late fusion. Different from the former framework chasing for the optimal kernel to cluster, we decide to align the best assignment matrix with a variety of assignment matrices obtaining by different kernels. More specially, by taking every single kernel k-means with kernel $\left\{\mathbf{K}_p\right\}_{p=1}^m$, we could have a set of assignment matrices $\left\{\mathbf{H}_p\right\}_{p=1}^m$. For the multiple kernel settings, we have several assignment matrices, each of which can be seen as a partition to the samples. Due to clustering is unsupervised learning, different assignment matrix $\mathbf{H_p}$ could be seen as the same results if each can be aligned through column transformation. Further, with a permutation matrix $\mathbf{W_p}$, we have obtain that the new assignment matrix $\mathbf{H_pW_p}$ has the same clustering result with the original matrix $\mathbf{H_p}$.

Following the above analysis, we consider the best assignment matrix $\mathbf{H}$ as a linear combination of the permutation transformed assignment matrix. This motivates us to derive an optimization problem to best approximate the ideal consensus clustering matrix.

A. MOTIVATION ILLUSTRATION

For every multi-view clustering algorithm, the basic assumption is that all the views should share the consensus clustering results. As for kernel k-means settings, that means we should align the set of assignment matrices $\left\{\mathbf{H^m_{p=1}}\right\}$ with the consensus clustering matrix $\mathbf{H}$. However, is is noticed that clustering is actually unsupervised learning for the lack of class labels. Hence through column permutation, different assignment matrices could reflect in the same clustering results. For example, we assume that the data given has 5 samples and 3 clusters. During learning process, we have two assignment matrices $\mathbf{H_1}$ and $\mathbf{H_2}$ in the following,

$$\mathbf{H_1} = \begin{vmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}, \quad \mathbf{H_2} = \begin{vmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{vmatrix}$$

Although $\mathbf{H_1}$ and $\mathbf{H_2}$ are different in math forms, they actually performs the same clustering results. They both indicate that sample 1 and sample 2 belong to the same cluster while sample 3 and sample 4 belong to another cluster. And this could do the column exchanges to solve the problem $\mathbf{H_1} = \mathbf{H_2W_2}$, where the matrix $\mathbf{W_2}$ is permutation matrix.

$$\mathbf{W_2} = \begin{vmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{vmatrix}$$

Therefore, our motivation is to align the assignment matrices $\left\{\mathbf{H^m_{p=1}}\right\}$ with the same consensus clustering matrix $\mathbf{H}$. Moreover we set our formulas in the next section.

B. PROPOSED FORMULATIONS

With discussion mentioned above, our motivation is to find the consensus clustering matrix $\mathbf{H}$ by integrating a number of the weighted original matrices $\left\{\mathbf{H}_p\right\}_{p=1}^m$. In multiple kernel setting with a given set of m kernels $\left\{\mathbf{K}_p\right\}_{p=1}^m$, we apply kernel k-means algorithm on each kernel and have a set of resultant assignment matrices $\left\{\mathbf{H}_p\right\}_{p=1}^m$. By right-manipulating column permutation matrices, the assignment matrix could be written into its equivalent clustering result. By considering the contribution weight of different matrices to the optimal matrix, we unify our optimization function as follows:

$$\min_{\mathbf{H},\{\mathbf{W_p}\}_{p=1}^m} \left\| \mathbf{H} - \frac{1}{m}\sum_{p=1}^m \mathbf{H}_p\mathbf{W}_p \right\|_{\mathbf{F}}^2, \quad s.t.\ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k. \tag{10}$$

where $\left\{\mathbf{W_p}\right\}_{p=1}^m$ are a set of permutation matrices.

By considering the locality contribution weight of different matrices to the optimal matrix, we set our optimization function as follows:

$$\min_{\mathbf{H},\{\mathbf{W_p}\}_{p=1}^m,\boldsymbol{\gamma}} \left\| \mathbf{H} - \sum_{p=1}^m \gamma_p\mathbf{H}_p\mathbf{W}_p \right\|_{\mathbf{F}}^2, \quad s.t.\ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \quad \boldsymbol{\gamma}^\top 1 = 1,\ \boldsymbol{\gamma} \geq 0. \tag{11}$$

where $\left\{\mathbf{W_p}\right\}_{p=1}^m$ are a set of permutation matrices.

By relaxing the constraints imposed on the column-transformation matrix $\left\{\mathbf{W_p}\right\}_{p=1}^m$ to the orthogonality restriction, we get a relaxed version of Eq. 10,

$$\min_{\mathbf{H},\{\mathbf{W_p}\}_{p=1}^m} \left\| \mathbf{H} - \frac{1}{m}\sum_{p=1}^m \mathbf{H}_p\mathbf{W}_p \right\|_{\mathbf{F}}^2, \quad s.t.\ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \mathbf{W}^\top\mathbf{W} = \mathbf{I}_k. \tag{12}$$

where $\left\{\mathbf{W_p}\right\}_{p=1}^m$ are a set of permutation matrices.

We also get the relaxed version of the adaptive formula Eq.(11). By taking the contribution coefficient γ into the Eq.(12), we are supposed to adaptively consider the various cluster assignment matrices,

$$\min_{\mathbf{H},\{\mathbf{W}_p\}_{p=1}^m,\boldsymbol{\gamma}} \left\| \mathbf{H} - \sum_{p=1}^{m} \gamma_p \mathbf{H}_p \mathbf{W}_p \right\|_{\mathbf{F}}^2,$$
$$s.t.\ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_k, \quad \mathbf{W}^\top \mathbf{W} = \mathbf{I}_k, \boldsymbol{\gamma}^\top 1 = 1,\ \boldsymbol{\gamma} \geq 0. \tag{13}$$

It is worth noting that we not only set an optimization goal for the multiple-kernel k-means clustering with late fusion, but also offer a new framework to fuse various clustering methods, which implies that any kind of ensemble clustering results can be applied to our framework. Moreover, as the following optimization process shows, the proposed function could be easily solved by an alternate algorithm and every step could be easily solved with the existed optimization packages.

C. OPTIMIZATION FOR AVERAGE-WEIGHTED ALGORITHM

In order to solve the problem in Eq. (12), we design a two-step alternate optimization algorithm with a fast convergence rate, where each step could be easily solved by the existing off-the-shelf packages.

1) OPTIMIZATION **H** WITH FIXED $\{\mathbf{W}_p\}_{p=1}^m$

With $\{\mathbf{W}_p\}_{p=1}^m$ being fixed, the optimization Eq.(12) could be rewritten as follows,

$$\max_{\mathbf{H}}\ \mathrm{Tr}(\mathbf{H}^\top \mathbf{U})$$
$$s.t.\ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_k, \tag{14}$$

where $\mathbf{U} = \frac{1}{m}\sum_{p=1}^m \mathbf{H}_p \mathbf{W_p}$. And this problem in Eq.(14) could be easily solved by taking the singular value decomposition(SVD) of the given matrix $\mathbf{U}$.

2) OPTIMIZATION $\{\mathbf{W}_p\}_{p=1}^m$ WITH FIXED **H**

With **H** fixed, for each single $\mathbf{W_p}$, the optimization problem in Eq.(12) is equivalent to Eq.(15) as follows,

$$\max_{\mathbf{W}_p}\ \mathrm{Tr}(\mathbf{W}_p^\top \mathbf{V})$$
$$s.t.\ \mathbf{W}_p^\top \mathbf{W}_p = \mathbf{I}_k, \tag{15}$$

where $\mathbf{V} = \mathbf{H}_p^T(\mathbf{H} - \frac{1}{m}\sum_{q=1,q\neq p}^m \mathbf{H_q W_q})$. And this problem in Eq.(15) could be easily solved by taking the singular value decomposition(SVD) of the given matrix $\mathbf{V}$.

Our equally-weighted multiple kernel k-means algorithm with late fusion is outlined in Algorithm 1, and in the following we have proposed an adaptive algorithm in III-D to show respect to the different contribution of every single assignment matrix.

D. OPTIMIZATION FOR ADAPTIVE ALGORITHM

The average-weighted multiple-kernel k-means algorithm with late fusion proposed in section III-C naturally consider that every view shares the same coefficient contributing to the optimal clustering results. However it neglects the locality of every single view and is impractical in real applications.

Algorithm 1 Proposed Average-Weighted MKKM-LF

```
Input: $\{\mathbf{W_p}\}_{p=1}^m$ , $\tau$ and $\epsilon_0$.
Output: H.
Initialize $\{\mathbf{W_p}\}_{p=1}^m = \mathbf{I_k}$, $\boldsymbol{\gamma} = \frac{1}{m}$ and $t = 1$.
Repeat
    Update H by solving Eq.(14) with fixed $\{\mathbf{W_p}\}_{p=1}^m$ and $\boldsymbol{\gamma}$.
    Update $\{\mathbf{W_p}\}_{p=1}^m$ with fixed H and $\boldsymbol{\gamma}$ by Eq.(15).
    $t = t + 1$.
Until$\left(\mathbf{obj}^{(t-1)} - \mathbf{obj}^{(t)}\right)/\mathbf{obj}^{(t)} \leq \epsilon_0$
```

In order to allow every single view's coefficient to alter with respect to different data applications, we have proposed an adaptive algorithm and set the optimization goal as Eq.(13). Although the problem in Eq.(13) is a relaxed version, it is still troublesome to be solved with existed packages. In order to solve it, we design a three-step alternate optimization algorithm with a fast convergence rate, where each step could be easily solved by the existing off-the-shelf packages.

1) OPTIMIZATION **H** WITH FIXED $\{\mathbf{W}_p\}_{p=1}^m$ AND γ

With $\{\mathbf{W}_p\}_{p=1}^m$ and $\boldsymbol{\gamma}$ being fixed, the optimization Eq.(13) could be rewritten as follows,

$$\max_{\mathbf{H}} \mathrm{Tr}(\mathbf{H}^\top \mathbf{U})$$
$$s.t.\ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_k, \tag{16}$$

where $\mathbf{U} = \sum_{p=1}^m \boldsymbol{\gamma}_p \mathbf{H}_p \mathbf{W_p}$. And this problem in Eq.(16) could be easily solved by taking the singular value decomposition(SVD) of the given matrix $\mathbf{U}$. Here the following theorem gives a simple closed-form solution for the problem in Eq.16.

Theorem 1: Suppose that the matrix $\mathbf{U}$ *in Eq.(16) has the economic rank-k singular value decomposition form as* $\mathbf{U} = \mathbf{S}_k \boldsymbol{\Sigma}_k \mathbf{V}_k^\top$, *where* $\mathbf{S}_k \in \mathbb{R}^{n\times k}$, $\boldsymbol{\Sigma}_k \in \mathbb{R}^{k\times k}$, $\mathbf{V}_k \in \mathbb{R}^{k\times k}$. *The optimization in Eq.(16) has a closed-form solution as follows,*

$$\mathbf{H} = \mathbf{S}_k \mathbf{V}_k^\top \tag{17}$$

Proof: By taking the the normal singular value decomposition $\mathbf{U} = \mathbf{S}\boldsymbol{\Sigma}\mathbf{V}^\top$, the Eq.(16) could be changed into

$$\mathrm{Tr}(\mathbf{H}^\top \mathbf{S}\boldsymbol{\Sigma}\mathbf{V}^\top) = \mathrm{Tr}(\mathbf{V}^\top \mathbf{H}^\top \mathbf{S}\boldsymbol{\Sigma}). \tag{18}$$

Considering that $\mathbf{Q} = \mathbf{V}^\top \mathbf{H}^\top \mathbf{S}$, then we have that $\mathbf{Q}\mathbf{Q}^\top = \mathbf{V}^\top \mathbf{H}^\top \mathbf{S}\mathbf{S}^\top \mathbf{H}\mathbf{V} = \mathbf{I}_k$. Therefore we can take that $\mathrm{Tr}(\mathbf{V}^\top \mathbf{H}^\top \mathbf{S}\boldsymbol{\Sigma}) = \mathrm{Tr}(\mathbf{Q}\boldsymbol{\Sigma}) \leqslant \sum_{i=1}^k \sigma_i$. Hence to maximize the value of Eq.(16), the solution should be given as Eq.(17). □

2) OPTIMIZATION $\{\mathbf{W}_p\}_{p=1}^m$ WITH FIXED **H** AND γ

With **H** and $\boldsymbol{\gamma}$ being fixed, for each single $\mathbf{W_p}$, the optimization problem in Eq.(11) is equivalent to Eq.(19) as follows,

$$\max_{\mathbf{W}_p} \mathrm{Tr}(\mathbf{W}_p^\top \mathbf{V})$$
$$s.t.\ \mathbf{W}_p^\top \mathbf{W}_p = \mathbf{I}_k, \tag{19}$$

where $\mathbf{V} = \mathbf{H}_p^T(\mathbf{H} - \sum_{q=1,q\neq p}^{m} \boldsymbol{\gamma}_q \mathbf{H_q}\mathbf{W_q})$. And this problem in Eq.(19) could be easily solved by taking the singular value decomposition (SVD) of the given matrix $\mathbf{V}$. Like the closed-form expressed in Theorem 1, if the matrix$\mathbf{V}$ has the singular value decomposition form as $\mathbf{V} = \mathbf{S\Sigma X}^\top$, the optimization in Eq.(19) has a closed-form solution as $\mathbf{W}_p = \mathbf{SX}^\top$. Hence we optimize one $\mathbf{W}_p$ with other $\mathbf{W}_{i\neq p}$ fixed at each iteration. Finally, we can obtain a set of optimized $\{\mathbf{W}_p\}_{p=1}^m$.

3) OPTIMIZATION γ WITH FIXED $\mathbf{H}$ AND $\{\mathbf{W}_p\}_{p=1}^m$

With $\mathbf{H}$ and $\{\mathbf{W_p}\}_{p=1}^m$ being fixed, the optimization problem in Eq.(11) is equivalent to the optimization problem as follows. Suppose that $\mathbf{Q} = \sum_{p=1}^m \boldsymbol{\gamma}_p\mathbf{H}_p\mathbf{W}_p$, then we have that

$$\begin{aligned}\|\mathbf{H}-\mathbf{Q}\|_\mathbf{F}^2 &= \mathrm{Tr}((\mathbf{H}-\mathbf{Q})^\top(\mathbf{H}-\mathbf{Q}))\\ &= \mathrm{Tr}(\mathbf{H}^\top\mathbf{H}) - 2\mathrm{Tr}(\mathbf{H}^\top\mathbf{Q}) + \mathrm{Tr}(\mathbf{Q}^\top\mathbf{Q})\\ &= k - 2\mathrm{Tr}(\mathbf{H}^\top\mathbf{Q}) + \mathrm{Tr}(\mathbf{Q}^\top\mathbf{Q}) \qquad (20)\end{aligned}$$

Noting that $\mathbf{Q} = \sum_{p=1}^m \boldsymbol{\gamma}_p\mathbf{H}_p\mathbf{W}_p$, we have that $\mathbf{Q}^\top = \sum_{p=1}^m \boldsymbol{\gamma}_p\mathbf{W}_p^\top\mathbf{H}_p^\top$ and $\mathbf{Q}^\top\mathbf{Q} = (\sum_{p=1}^m \boldsymbol{\gamma}_p\mathbf{W}_p^\top\mathbf{H}_p^\top)(\sum_{p=1}^m \boldsymbol{\gamma}_p\mathbf{H}_p\mathbf{W}_p)$. And taking them into Eq.(20), the optimization can be written as follows,

$$\begin{aligned}&\min_{\boldsymbol{\gamma}} \frac{1}{2}\boldsymbol{\gamma}^\top A\boldsymbol{\gamma} - \mathbf{f}^\top\boldsymbol{\gamma},\\ &s.t.\ \boldsymbol{\gamma}^\top 1 = 1, \quad \boldsymbol{\gamma} \geq 0, \qquad (21)\end{aligned}$$

where $f = [f_1, f_2, \ldots, f_m]$ with $\mathbf{f_p} = \mathrm{Tr}(H^\top H_p W_p)$, $\mathbf{A_{pq}} = \mathrm{Tr}(W_p^\top H_p^\top H_q W_q)$.

It seems difficult to solve the Eq.(21). However the following proof illustrates the matrix $\mathbf{A}$ is a positive semidefinite (PSD) matrix. Hence, with the simplified problem proposed in Eq.(21), we have observed that this problem is a quadratic programming optimization and could be efficiently solved via the existing convex optimization package.

Lemma 1: for every $x \in \mathbb{R}^m$, we have that

$$\begin{aligned}x^\top Ax &= \sum_{p=1}^m\sum_{q=1}^m x_p x_q \mathrm{Tr}(W_p^\top H_p^\top H_q W_q),\\ &= \mathrm{Tr}(\sum_{p=1}^m\sum_{q=1}^m x_p x_q W_p^\top H_p^\top H_q W_q),\\ &= \mathrm{Tr}(\sum_{p=1}^m x_p W_p^\top H_p^\top \sum_{q=1}^m x_q H_q W_q),\\ &= \left\|\sum_{p=1}^m x_p w_p^\top H_p^\top\right\|_F^2 \geq 0. \qquad (22)\end{aligned}$$

Therefore, the matrix $\mathbf{A}$ *is a positive semidefinite matrix and the optimization in Eq.(21) could be solved by quadratic programming.*

Our adaptive algorithm is outlined in Algorithm 2, where $\mathrm{obj}^{(t)}$ denotes the objective value at the t-th iterations. The objective of Algorithm 1 and Algorithm 2 is monotonically decreased when optimizing one variable with the other fixed at each iteration. At the same time, the whole optimization problem is lower-bounded. As a result, the proposed algorithm can be verified to be convergent. We also record the objective at each iteration and the results validate the convergence. In addition, the algorithm usually converges in less than ten iterations in all of our experiments.

Algorithm 2 Proposed Adaptive MKKM-LF

```
Input: $\{\mathbf{H}_p\}_{p=1}^m$, $\tau$ and $\epsilon_0$.
Output: $\mathbf{H}$, $\boldsymbol{\gamma}$.
Initialize $\{\mathbf{W}_p\}_{p=1}^m = \mathbf{I_k}$, $\boldsymbol{\gamma} = \frac{1}{m}$ and $t = 1$.
Repeat
    Update $\mathbf{H}$ by solving Eq.(16) with fixed $\{\mathbf{W_p}\}_{p=1}^m$ and $\boldsymbol{\gamma}$.
    Update $\{\mathbf{W}_p\}_{p=1}^m$ with fixed $\mathbf{H}$ and $\boldsymbol{\gamma}$ by Eq.(19).
    Update $\boldsymbol{\gamma}$ by solving Eq.(21) with fixed $\mathbf{H}$ and $\{\mathbf{W}_p\}_{p=1}^m$.
    $t = t + 1$.
Until $\left(\mathbf{obj}^{(t-1)} - \mathbf{obj}^{(t)}\right)/\mathbf{obj}^{(t)} \leq \epsilon_0$
```

IV. ALGORITHM ANALYSIS

In this section, we present the theoretical analysis on the optimization algorithm's convergence and computational complexity to verify the efficiency of proposed algorithms.

A. CONVERGENCE ANALYSIS

As mentioned, our optimization value is monotonically decreased and the algorithm usually converges less than ten iterations. Our two algorithms both adopt the alternate optimization strategy which ensures every step of optimization goal could get decreased under conditions. The objective of Algorithm 1 and Algorithm 2 is monotonically decreased when optimizing one variable with the other fixed at each iteration. At the same time, the whole optimization problem is lower-bounded. As a result, the proposed algorithm can be verified to be convergent. In addition, our algorithm is theoretically guaranteed to converge to a local minimum according to [22].

B. COMPUTATIONAL COMPLEXITY

As shown in the former sections, our proposed algorithm achieves the comparable or even better performances than other multiple-kernel k-means clustering ones. Moreover, as our motivation mentioned in the introduction part, comparing to the best algorithm multiple-kernel clustering with local kernel alignment(MKC-LKA), our algorithm has less time complexity with learning time. And in this section, we theoretically analyze the time of the several former-mentioned algorithms.

Theoretically, we assume that the number of samples in given dataset is n, the number of clusters k and the number of kernels is m. Going back to the our optimization algorithm in 1, the total time complexity consists of three parts referring to the three alternate steps. The first step of algorithm 1,

TABLE 1. The comparison on the time complexity of comparing algorithms at each iteration.

	#Constructing the matrix	#Optimization step	#Total
MKC-LKA	$\mathcal{O}(n^3+kn^3+mn^3+m^2n^3)$	$\mathcal{O}(n^3+m^3)$	$\mathcal{O}(kn^3+mn^3+m^2n^3+n^3+m^3)$
Ours Average	$\mathcal{O}(mnk^2+m^2nk^2)$	$\mathcal{O}(nk^2+mk^3)$	$\mathcal{O}(nk^2+mnk^2+mk^3+nm^2k^2)$
Ours Adaptive	$\mathcal{O}(mnk^2+m^2nk^2)$	$\mathcal{O}(nk^2+mk^3+m^3)$	$\mathcal{O}(nk^2+mnk^2+mk^3+m^3+nm^2k^2)$

mentioned in Eq.16, actually needs an singular value decomposition(SVD) of a matrix with the size of $n \times k$ and building the matrix $\mathbf{Q} = \sum_{p=1}^{m} \boldsymbol{\gamma}_p \mathbf{H}_p \mathbf{W}_p$ needs $\mathcal{O}(mnk^2)$. Hence the time complexity of first step is $\mathcal{O}(nk^2 + mnk^2)$.

The second step follows the same optimization strategy in the first step while the matrix size reduces to $k \times k$ and building the matrix $\mathbf{V} = \mathbf{H}_p^T(\mathbf{H} - \sum_{q=1,q\neq p}^{m} \boldsymbol{\gamma}_q \mathbf{H_q W_q})$ needs $\mathcal{O}(mnk^2)$.Hence the time complexity of second step is $\mathcal{O}(m(k^3 + mnk^2))$. As for the third step, the time complexity of quadratic programming is $\mathcal{O}(m^3)$ and building the matrix $\mathbf{A}$ and f needs $\mathcal{O}(2mnk^2 + (\frac{m(m+1)}{2})nk^2)$. Hence time complexity of the third step is $\mathcal{O}(m^3 + mnk^2 + m^2nk^2)$. The whole time complexity of each iteration in our proposed algorithm is $\mathcal{O}(nk^2 + mnk^2 + mk^3 + m^3 + nm^2k^2)$.

The comparison on the time complexity of ours and the Multiple Kernel Clustering with Local Kernel Alignment Maximization (MKC-LKA) at each iteration is listed in Table 1. Comparing to MKC-LKA and MKKM-MR, our proposed algorithm has less time cost which linearly grows on the sample number n, where $n \gg k$ and $n \gg m$. And our algorithm always converge in less than 10 times on real-world datasets. The experiment results listed in Table 4 also verify our analysis.

V. EXPERIMENTS

In this section, we conduct a set of experiments to demonstrate the effectiveness of the proposed multiple-kernel k-means with late fusion (MKKM-LF). We compare our proposed average-weighted algorithm, adaptive algorithm with the several state-of-art algorithms on clustering performance and timecost listed in tables. Moreover, we also conduct several experiments to verify our analysis of time complexity.

A. EXPERIMENTAL SETTINGS

We evaluate our multiple kernel k-means with late fusion(MKKM-LF) algorithm on multiple kernel clustering benchmarks. They are Oxford Flower17 and Flower102[1] and Protein Fold prediction[2] and CCV[3] and UCI-Digital[4] and Caltech 101[5]. The detailed information of the several datasets are listed in Table 2.

TABLE 2. Datasets used in our experiments.

Dataset	#Samples	#Kernels	#Classes
Flower17	1360	7	17
Flower102	8189	4	102
Caltech	1530	25	102
ProteinFold	694	12	27
Digits	2000	3	20
CCV	6773	3	20

For the ProteinFold Dataset, we use the kernel generating method proposed by [23]. For other benchmark multiple kernel datasets, we use the pre-defined kernel matrices and download them from the official website.

In all our experiments, all base kernels are first centered and then scaled so that for all sample x_i and p, we have $K_p(x_i, x_i) = 1$ by following [24]. For all data sets, it is assumed that the true number of clusters is known and set as the true number of classes. For the proposed algorithm, its neighborhood parameter τ is chosen from $[0.1, 0.2, \cdots, 0.9, 1] \times n$ by grid search, where n is the number of samples.

The widely used clustering accuracy (ACC), normalized mutual information (NMI) and purity are applied to evaluate the clustering performance. For all algorithms, we repeat each experiment for 50 times with random initialization to reduce the effectiveness of randomness caused by k-means, and report the best result. All the experiments are performed on a desktop with Intel core i7-5820k CPU and 16G RAM.

B. COMPARED ALGORITHM

In this section, we list the compared algorithms as follows,

- Average multiple kernel k-means (A-MKKM): All kernels are averagely weighted to conduct the optimal kernel, which is used as the input of kernel k-means algorithm.
- Single best kernel k-means (SB-KKM): Kernel k-means is performed on each single kernel and the best result is outputted.
- Multiple kernel k-means (MKKM) ([25]): The algorithm alternatively performs kernel k-means and updates the kernel coefficients, as introduced in the related work.
- Optimized data fusion for kernel k-means clustering (OKKC) ([9]): The algorithm propose to jointly optimize clustering, the kernel coefficients and dimension reduction based on the metric of Mahalanobis distance, as introduced in the related work.

[1] http://www.robots.ox.ac.uk/~vgg/data/flowers/

[2] http://mkl.ucsd.edu/dataset/protein-fold-prediction

[3] http://www.ee.columbia.edu/ln/dvmm/CCV/

[4] http://archive.ics.uci.edu/ ml/datasets /optical+recognition+of+handwritten+digits

[5] http://www.vision.caltech.edu/Image_Datasets /Caltech101

TABLE 3. **ACC, NMI and purity comparison of different clustering algorithms on all data sets.**

Datasets	A-MKKM	SB-KKM	MKKM	OKKC	CSRC	MKC-LKA	MKKM-MR	Proposed	
			[25]	[9]	[26]	[12]	[11]	Average	Adaptive
ACC									
Digital	88.75	75.40	47.00	67.10	73.20	**95.25**	90.40	93.55	94.55
Flower17	51.03	42.06	45.37	44.85	51.76	61.69	60.00	61.71	**62.56**
ProteinFold	30.69	34.58	27.23	37.10	35.59	39.34	36.89	40.50	**41.49**
Flower102	27.29	33.13	21.96	22.32	38.60	43.23	40.24	**43.99**	43.78
Caltech	35.56	33.14	34.77	33.92	34.38	37.06	35.82	**37.39**	37.09
CCV	19.74	20.08	18.01	20.54	23.06	23.49	22.47	25.34	**25.67**
NMI									
Digital	80.59	68.38	48.16	64.36	69.31	**89.73**	83.22	85.05	85.36
Flower17	50.19	45.14	45.35	45.85	53.19	57.27	57.11	57.56	**57.79**
ProteinFold	40.96	42.33	37.16	40.75	45.66	47.55	45.13	48.17	**49.96**
Flower102	46.32	48.99	42.30	43.28	54.95	58.05	57.27	**59.48**	58.18
Caltech	59.90	59.07	59.64	57.22	58.35	61.58	60.38	**62.65**	61.91
CCV	17.16	17.73	15.52	16.28	18.89	17.11	18.62	19.31	**19.73**
Purity									
Digital	88.75	76.10	0.50	68.25	76.10	**95.25**	90.40	92.25	92.55
Flower17	51.99	44.63	46.84	45.00	53.68	62.79	61.03	63.32	**64.07**
ProteinFold	37.18	41.21	33.86	39.91	42.07	45.97	43.80	48.39	**48.85**
Flower102	32.28	38.74	27.61	28.12	45.04	48.94	46.39	**50.49**	48.99
Caltech	37.12	35.10	37.25	36.27	35.95	39.08	37.65	**40.28**	39.84
CCV	23.98	23.48	22.25	24.17	26.80	22.93	25.69	28.41	**28.50**

TABLE 4. **The time cost of different clustering algorithms on all data sets (sec.).**

Datasets	A-MKKM	SB-KKM	MKKM	CSRC	MKC-LKA	MKKM-MR	Proposed Average	Proposed Adaptive
Digital	0.96	4.63	3.82	92.90	12.53	4.74	6.46	4.57
Flower17	0.74	4.58	2.13	46.04	6.05	5.47	4.11	4.10
ProteinFold	0.34	4.01	1.03	20.03	2.06	1.94	1.90	1.72
Flower102	27.51	120.50	127.77	3226.60	1027.40	382.13	305.90	335.02
Caltech	1.9372	42.23	10.12	333.75	55.05	45.76	30.65	31.49
CCV	6.9760	17.84	39.39	983.96	281.92	161.20	125.43	138.06

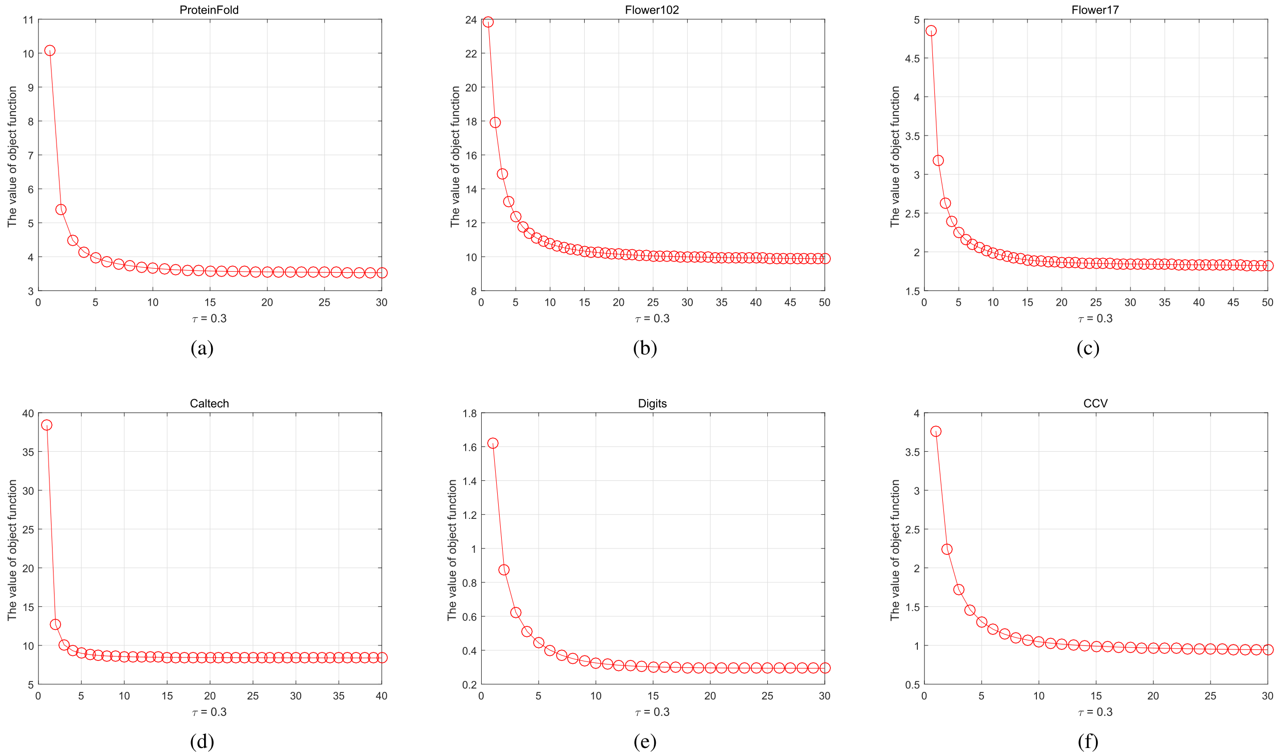
- Co-regularized spectral clustering (CRSC) ([26]): CRSC provides a co-regularization way to perform spectral clustering on multiple views.
- Multiple kernel k-means with Matrix-induced Regularization (MKKM-MR) ([11]): The algorithm applies the multiple kernel k-means clustering with a matrix-induced regularization to reduce the redundancy and enhance the diversity of the kernels.
- Multiple Kernel Clustering with Local Kernel Alignment Maximization (MKC-LKA)([12]): The algorithm maximizes the local kernel alignment with multiple kernel clustering and focuses on closer sample pairs that they shall stay together.

The Matlab codes of A-MKKM, SB-KKM and MKKM are publicly available at `http://github.com/mehmetgonen/lmkkmeans`. For the rest of algorithms, we use their matlab implementations from authors' websites in our experiments.

C. EXPERIMENTAL RESULTS

The ACC, NMI and Purity of the compared algorithms on the six benchmark datasets are displayed in Table 3. The best and second best results are presented in red and blue respectively. We also plot the running time of the mentioned algorithms on each datasets in Table 4. Due to the results, we have the following conclusions:

- Our proposed algorithm always achieves the best and second best on the four datasets while it is much closer between ours and the best one on the rest datasets. Taking the largest dataset Flower102 as an example, our average algorithm arrives 43.99% and the adaptive algorithm is 43.78% while significantly outperforms the other ones. And among the six benchmark datasets, our algorithms achieves the best results in four datasets and the second best in the rest of datasets.
- As mentioned before, different from framework of the MKKM, MKC-LKA and MKKM-MR algorithms, our framework with late fusion is more robust in the datasets, which is essential for practical use.
- OKKC has comparable performances with SB-KKM and MKKM. Because of the ℓ_1 norm constraint on the kernel coefficients, this leads a sparse solution and normally only one of the selected kernels is performed while others are given very small weight. Hence the performance of OKKC could not exceed the SB-MKKM(single-best) and MKKM too much.



(a) (b) (c)

(d) (e) (f)

FIGURE 1. The objective value of our average algorithm at each iteration in Proteinfold (a), Flower102 (b), Flower17 (c), Caltech (d), Digits (e) and CCV (f).

- As a strong baseline, MKC-LKA usually demonstrates comparable or even better performance than most of algorithms in comparison. However, the time-cost of our proposed algorithm is significantly less than MKC-LKA while ours achieves the equal performance with MKC-LKA.

Table 3 also reports the comparison of NMI and purity. Again, we observe that the proposed algorithm has promising performance among datasets. In all, these results have well verified the effectiveness of late fusion in multiple kernel k-means setting.

From the above experiments, we can conclude that our proposed algorithm has the following advantages:

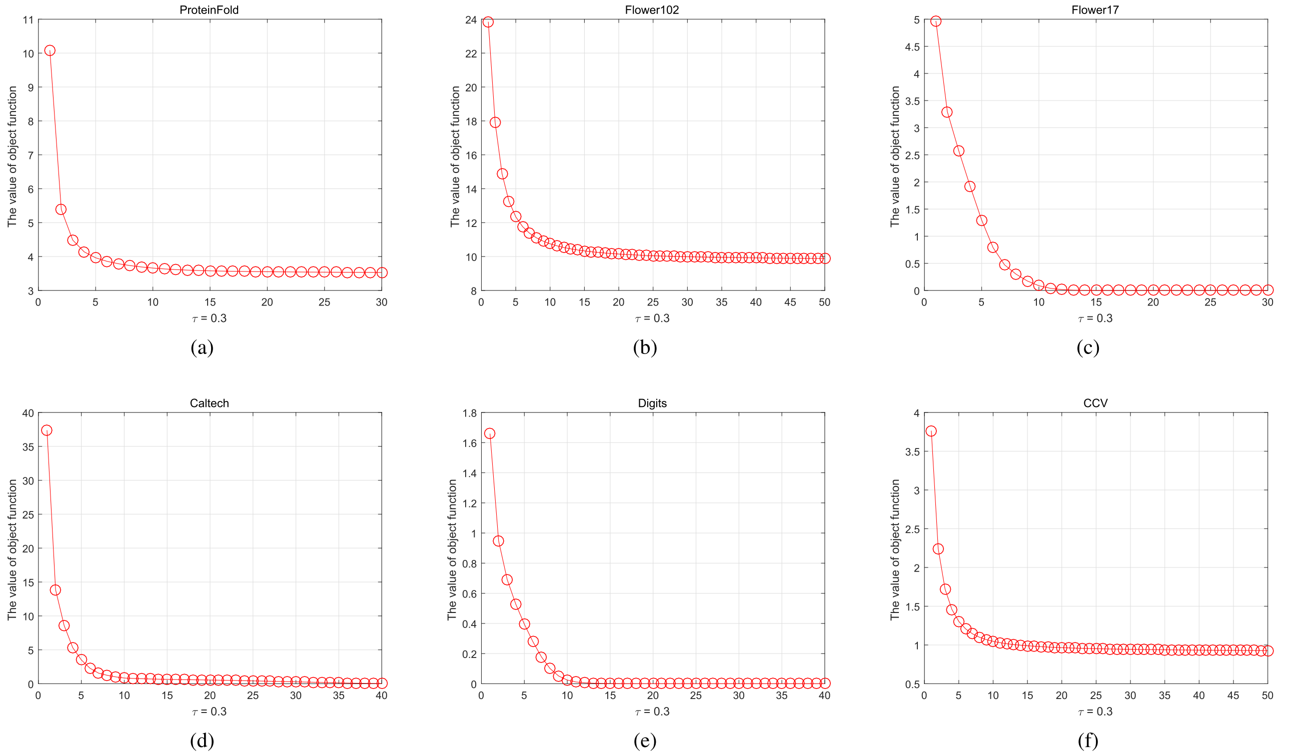
- effectively make the use of the multiple assignment matrices and alternately form an optimal clustering result;
- well jointly utilizes the contribution of each kernel in the process of clustering and reduce the high computational complexity in former methods. As the sample number increases, ours significantly outperforms MKC-LKA in time cost which we theoretically demonstrate in section IV-B.

The two proposed algorithms actually take various performance among datasets. While equally considering each view's contribution, the average-weighted algorithm aims to enhance the diversity of choices of selected assignment matrices. Moreover, the adaptive algorithm allows the weight to alternately change and join the coefficient optimization step into the whole optimization process. So it leads to sparse combination of the pre-selected assignment matrices, and hence automatically performs the matrix selection. Our framework with late fusion is flexible and allows the pre-specified kernels clustering results to be weighted for better clustering, bringing improvements on clustering performance. Hence it could be easily extended to other multiple-view clustering methods.

D. PARAMETER SELECTION

Our proposed algorithm has one hyperparameter τ which represents the neighborhood ratio respectively. The parameter τ is considered to reveal the locality of samples and the underlying inner structure of clusters. In our algorithm setting, we experimentally investigate the influence of the hyperparameter on our clustering performance. The selection neighborhood ratio of τ is from $[0.1, 0.2, \cdots, 0.9, 1]$. The Figure 3 shows the influences on the clustering accuracy by selection of τ among different datasets. From the figure, we have the following conclusions:

- The hyperparameter selection of τ should be various for different datasets. And the best performance of clustering accuracy is always achieved by appropriately selecting the range of neighbors.
- Our proposed algorithm shows comparable clustering performance across a wide range of τ values. The fluctuation of various τ on clustering accuracy is no more than 5%.
- As our figures 3 shows, our two proposed algorithms are more robust across a wide range of parameter τ.



(a) (b) (c)

(d) (e) (f)

FIGURE 2. The objective value of our adaptive algorithm at each iteration in Proteinfold (a), Flower102 (b), Flower17 (c), Caltech (d), Digits (e) and CCV (f).

Hence ours are more practical in real applications. While the two algorithms usually achieve equal performances, the adaptive algorithm is trained with less time consuming.

E. FURTHER EXPERIMENTS

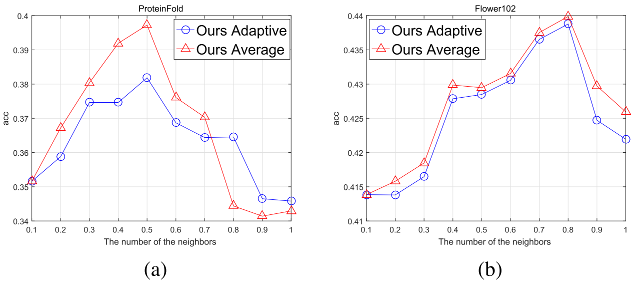
To further reveal the time complexity of our proposed algorithm comparing to the multiple kernel clustering with Local kernel alignment maximization(MKC-LKA), we are supposed to do experiments on the Caltech102[6] dataset. To investigate the clustering performance with respect to the number of samples, we select 5, 10, 15, 20, 25 and 30 samples randomly selected from each class. By this way, we generate five datasets on Caltech102, which has 102 classes and 48 base kernels. We refer to the six generated datasets as caltech102-5, caltech102-10, caltech102-15, caltech102-20, caltech102-20, caltech102-25 and caltech102-30 respectively.

The ACC of the compared algorithms on the five benchmark datasets are displayed in Table 6. The best and second best results are presented in red and blue respectively. We also plot the running time of the mentioned algorithms on each datasets in Table 7 and Figure 4a.

As the result shows, our two algorithms achieve the best and second best performances in cluster accuracy among the six datasets. Moreover as the sample of each class grows, the timecost of the our two algorithms grows linearly on sample number while the compared algorithm MKC-LKA, MKKM-MR significantly need more training time. While achieving comparable performance among the six datasets, the time cost also verifies the analysis mentioned before.

TABLE 5. Details about datasets used in further experiments.

Dataset	#Samples	#Kernels	#Classes
Caltech102-5	510	48	102
Caltech102-10	1020	48	102
Caltech102-15	1530	48	102
Caltech102-20	2040	48	102
Caltech102-25	2550	48	102
Caltech102-30	3060	48	102



(a) (b)

FIGURE 3. The effect of the neighbourhood ratio τ on ACC in adaptive algorithm among Proteinfold (a) and Flower102 (d).

[6] http://mkl.ucsd.edu/dataset/ucsd-mit-caltech-101-mkl-dataset

TABLE 6. The clustering accuracy (ACC) results in further experiments.

Dataset	#MKC-LKA	#MKKM-MR	# Our Average	#Our Adaptive
Caltech102-5	35.69	37.02	39.80	39.25
Caltech102-10	33.33	33.73	37.90	36.02
Caltech102-15	31.31	32.35	36.86	36.07
Caltech102-20	29.71	33.48	35.85	36.82
Caltech102-25	29.69	31.41	33.57	34.39
Caltech102-30	28.89	31.84	34.42	35.47

TABLE 7. The time cost of different clustering algorithms on further experiment (sec).

Dataset	#MKC-LKA	#MKKM-MR	#Ours Average	#Ours Adaptive
Caltech102-5	7.16	8.84	11.04	10.01
Caltech102-10	35.03	40.78	32.09	36.41
Caltech102-15	106.90	121.04	82.16	90.06
Caltech102-20	213.55	230.17	142.57	161.71
Caltech102-25	392.04	422.70	212.70	240.97
Caltech102-30	620.07	634.94	296.68	340.12

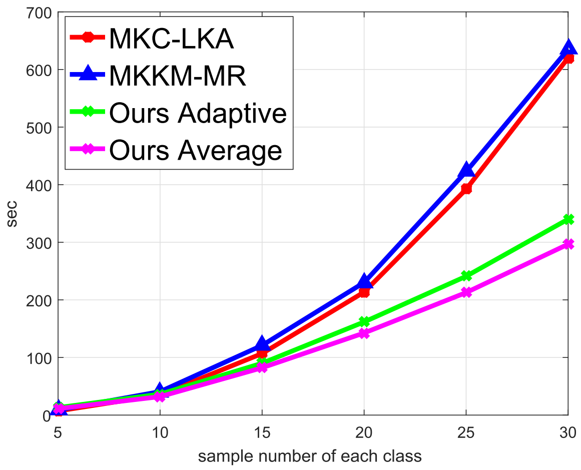


FIGURE 4. The time comparison in six Caltch102 datasets.

VI. CONCLUSION

This work has proposed a multiple kernel clustering framework with late fusion to jointly utilize the various views of clustering results. A weighted combination of the clustering matrices which reflect the different views' relevance to the clustering task is automatically updated. The two new algorithm, **MKKM-LF**, show promising performance with smaller time complexities, underlying the strength of late fusion and boosting the quality of clustering partition.

In the future, we try to apply the late fusion framework to other kernel-based learning tasks. Moreover, it is interesting to explore more possible fusion methods extended to our framework. The idea of view-weighted late fusion could be adapted to kernel-based unsupervised attribute weighting.

REFERENCES

[1] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, 1998.

[2] I. S. Dhillon, Y. Guan, and B. Kulis, *A Unified View of Kernel k-means, Spectral Clustering and Graph Cuts*. Citeseer, 2004.

[3] I. S. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors a multilevel approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 11, pp. 1944–1957, Nov. 2007.

[4] S. Wang *et al.*, "Multiple sclerosis detection based on biorthogonal wavelet transform, RBF kernel principal component analysis, and logistic regression," *IEEE Access*, vol. 4, pp. 7567–7576, 2016.

[5] J. J.-Y. Wang, J. Z. Huang, Y. Sun, and X. Gao, "Feature selection and multi-kernel learning for adaptive graph regularized nonnegative matrix factorization," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1278–1286, 2015.

[6] P. Cao *et al.*, "A multi-kernel based framework for heterogeneous feature selection and over-sampling for computer-aided detection of pulmonary nodules," *Pattern Recognit.*, vol. 64, pp. 327–346, Apr. 2017.

[7] M. Gönen and E. Alpaydin, "Localized multiple kernel learning," in *Proc. 25th Int. Conf. Mach. Learn.* New York, NY, USA: ACM, 2008, pp. 352–359.

[8] J. Chen, Z. Zhao, J. Ye, and H. Liu, "Nonlinear adaptive distance metric learning for clustering," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: ACM, 2007, pp. 123–132.

[9] S. Yu *et al.*, "Optimized data fusion for kernel k-means clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 1031–1039, May 2012.

[10] M. Gönen and A. A. Margolin, "Localized data fusion for kernel k-means clustering with application to cancer biology," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1305–1313.

[11] X. Liu, Y. Dou, J. Yin, L. Wang, and E. Zhu, "Multiple kernel k-means clustering with matrix-induced regularization," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1888–1894.

[12] M. Li, X. Liu, L. Wang, Y. Dou, J. Yin, and E. Zhu, "Multiple kernel clustering with local kernel alignment maximization," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 1704–1710.

[13] L. Chen, C. L. P. Chen, and M. Lu, "A multiple-kernel fuzzy C-means algorithm for image segmentation," *IEEE Trans. Syst., Man B, Cybern.*, vol. 41, no. 5, pp. 1263–1274, Oct. 2011.

[14] D. Guo, J. Zhang, X. Liu, Y. Cui, and C. Zhao, "Multiple kernel learning based multi-view spectral clustering," in *Proc. 22nd Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2014, pp. 3774–3779.

[15] X. Liu, L. Wang, J. Yin, E. Zhu, and J. Zhang, "An efficient approach to integrating radius information into multiple kernel learning," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 557–569, Apr. 2013.

[16] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *J. Mach. Learn. Res.*, vol. 12, pp. 2211–2268, Jul. 2011.

[17] E. Bruno and S. Marchand-Maillet, "Multiview clustering: A late fusion approach using latent models," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2009, pp. 736–737.

[18] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders, "Early versus late fusion in semantic video analysis," in *Proc. ACM Int. Conf. Multimedia*, 2005, pp. 399–402.

[19] K.-T. Lai, D. Liu, S.-F. Chang, and M.-S. Chen, "Learning sample specific weights for late fusion," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2772–2783, Sep. 2015.

[20] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, and Q. Tian, "Query-adaptive late fusion for image search and person re-identification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1741–1750.

[21] R. T. Ionescu, S. Smeureanu, B. Alexe, and M. Popescu, "Unmasking the abnormal events in video," Tech. Rep., 2017.

[22] J. C. Bezdek and R. J. Hathaway, "Convergence of alternating optimization," *Neural, Parallel Sci. Comput.*, vol. 11, no. 4, pp. 351–368, 2003.

[23] T. Damoulas and M. A. Girolami, "Probabilistic multi-class multi-kernel learning: On protein fold recognition and remote homology detection," *Bioinformatics*, vol. 24, no. 10, pp. 1264–1270, 2008.

[24] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for learning kernels based on centered alignment," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 795–828, Jan. 2012.

[25] H. C. Huang, Y. Y. Chuang, and C. S. Chen, "Multiple kernel fuzzy clustering," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 1, pp. 120–134, Feb. 2012.

[26] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1413–1421.

SIWEI WANG is currently pursuing the degree with the National University of Defense Technology, China. His current research interests include kernel learning, unsupervised multiple-view learning, scalable kernel k-means, and deep neutral networks.

EN ZHU received the Ph.D. degree from the National University of Defense Technology, China, where he is currently a Professor with the School of Computer Science. He has published 60+ peer-reviewed papers, including IEEE T-CSVT, IEEE T-NNLS, PR, AAAI, IJCAI, and so on. His main research interests include pattern recognition, image processing, machine vision, and machine learning. He was awarded the China National Excellence Doctoral Dissertation.

JINGTAO HU is currently pursuing the degree with the National University of Defense Technology, China. Her current research interests include unsupervised abnormal detection, outlier detection, and neutral networks.

MIAOMIAO LI is currently pursuing the Ph.D. degree with the National University of Defense Technology, China. She is currently a Lecturer with the Changsha College, Changsha, China. She has published several peer-reviewed papers, such as AAAI, IJCAI, *Neurocomputing*, and so on. Her current research interests include kernel learning and multi-view clustering. She served on the Technical Program Committees of IJCAI 2017 and 2018.

KAIKAI ZHAO is currently pursuing the joint Ph.D. degree with Naval Aeronautical University and Department of Computer Science, National University of Defense Technology, China. His research interests include pattern recognition, kernel learning, and large-scale machine learning.

NING HU received the Ph.D. degree from the National University of Defense Technology, China. He is currently a Professor with the Cyberspace Institute of Advanced Technology, Guangzhou University. He has published over 30 papers in journals and conferences. His current research interests include artificial intelligence safety and security. He has also achieved the Second Class Prize of Chinese State Scientific and Technological Progress Award.

XINWANG LIU (M'13) received the Ph.D. degree from the National University of Defense Technology, China, where he is currently an Assistant Researcher with the School of Computer Science. He has published 40+ peer-reviewed papers, including those in highly regarded journals and conferences, such as IEEE T-IP, IEEE T-NNLS, ICCV, AAAI, IJCAI, and so on. His current research interests include kernel learning and unsupervised feature learning. He served on the Technical Program Committees of IJCAI 2016/2017/2018 and AAAI 2016/2017/2018.