Multiple kernel k-means clustering with late fusion

KaiKai Zhao[a,c,*], SiWei Wang[a], XinWang Liu[a], En Zhu[a], Jianping Yin[b], You He[c]

[a]*College of Computer,*
National University of Defense Technology, Changsha 410073, China
[b]*State Key Laboratory of High Performance Computing,*
National University of Defense Technology, Changsha, 410073, China
[c]*Institute of Information fusion,*
Naval Aeronautical University, Yantai, 264001, China

Abstract

Multiple kernel k-means(MKKM) has attracted a lot of research interest in kernel methods. However, most of the existing methods suffer from high computational complexity and complicated model. To address this issue, we propose a multiple kernel k-means algorithm with late fusion which aims to integrate the various clustering results into the optimal one. We extend the method of late fusion into kernel method and enhance the diversity of clustering assignment choices with small timecost. As the promising performance demonstrated in the benchmark datasets, our algorithm achieves the equally best results compared to the state-of-art ones with a simple model and fast convergence rate, which verifies the efficiency and advantages of applying the late fusion to our model.

Keywords: Multi-view clustering, Multiple kernel clustering, late fusion, kernel k-means

1. Introduction

Clustering is one of the fundamental learning tasks in machine learning and data mining. Among those existing clustering algorithms, the k-means algorithm has been widely applied to many fields for its simplicity and effect. The k-means

* Corresponding author
Email address: zhaojaiyi11@nudt.edu.cn (KaiKai Zhao)

algorithm follows a two-step iteration prototype: i) set k landmarks as cluster centers, then assign samples to k clusters based on the k landmarks; ii) update the assignment matrix by the minimizing the sum of within-cluster distances squares and compute the new landmarks. The two steps are run iteratively until convergence. As an important extension, the kernel k-means algorithm tries to transfer the original data to a high-dimensional space which are linearly separable[1]. This extension enhances the k-means to handle the linearly non-separable problem through feature mapping.

However, the samples are presented in various forms of data in many real-world application. For example, for webpage classification, the sample usually has two or more types of data, e.g., text, hyperlinks and images, each of which can be seen as one view to the sample itself. It is natural that researchers propose methods to combine the comprehensive information collected from each view, which is known as multi-view learning in literature. For kernel method, each view can be represented by a kernel matrix and the whole view are considered as a total sum of weighted kernel matrices. The weight of a single kernel matrix can be seen as its contribution of the respective view to the whole views.

In our paper, we focus on the two-step multiple kernel learning framework that optimizes the coefficients of the pre-defined kernels iteratively[2][3][4][5]. In [2], a novel optimized kernel k-means algorithm is proposed to integrate multiple data sources for clustering performance. With [3], they design a localized kernel k-means clustering algorithm to adapt to individual samples by altering the kernels' weights respectively. Following by this line, a multiple kernel k-means clustering algorithm with a matrix-induced regularization has been proposed to reduce the redundancy and enhance the diversity of the kernels[4]. 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[5]. Those mentioned multiple kernel k-means clustering algorithms have shown promising clustering performance and

are widely used to practical applications.

Our strategy of handling multiple kernel learning has been inspired by the late fusion proposed in computer vision and document classification. Late fusion has been widely applied in computer vision[6][7][8]. To our knowledge, for the field of multi-view learning, late fusion is firstly adopted to explore cluster-cluster relationships with a latent framework model[9]. And we extend the late fusion technology to the multi-view learning based on kernel method. We are supposed to integrate the various clustering assignment matrices produced by different views respectively instead of the optimal kernel for clustering in former multiple kernel framework. On the other hand, our method can be seen as an ensemble model for clustering results. Hence it can enhance the diversity of clustering assignment matrices and is desired to find the optimal clustering assignment matrix.

In order to implement the framework mentioned above, we propose a novel multiple kernel k-means algorithm with fast convergence which we call it *multiple kernel k-means with late fusion* (MKKM-LF). To solve the resultant optimization problem, we develop an efficient algorithm with proved convergence. Extensive experimental study has been conducted on five MKL benchmark data sets to evaluate clustering performance of the proposed algorithm. As indicated, our algorithm has a very simple model with small complexity and consistently demonstrates performance equally matched with the several state-of-the-art ones. Moreover, the carefully designed optimization goal has a very fast rate of convergence, usually less than 10 times in benchmark datasets. This verifies the effectiveness and superiority of late fusion in our algorithm.

The rest of this paper is organized as follows. Section 2 outlines the related work of multiple kernel clustering. Section 3 presents the proposed optimization objective and the three-step alternate algorithm. Section 4 shows the experiment results with evaluation. Section 5 concludes the paper.

2. Related work

2.1. Kernel k-means clustering(KKM)

Let $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathscr{X}$ be a collection of n samples. The objective of kernel k-means clustering is to minimize the sum of the square of the within-cluster distance. And the kernel function $\phi(x)$ maps the origin $\mathbf{x}$ onto a reproducing kernel Hilbert space $\mathscr{H}$ which is a k-means-friendly space. By taking the assignment matrix $Z \in \{0,1\}^{n\times k}$, the optimization objective of KKM could be written as follows:

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

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 $c-th(1 \le c \le k)$ cluster respectively.

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

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

Here, we apply the kernel k-means method to the Eq.(1), $\mathbf{K}$ 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 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),

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

The optimization problem in Eq.(3) could be solved by singular value decomposition(SVD) of the kernel matrix $\mathbf{K}$ [4].

2.2. Multi-kernel k-means (MKKM)

In the multiple kernel setting, Let $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathscr{X}$ be a collection of n samples, and $\phi_p(\cdot) : \mathbf{x} \in \mathscr{X} \mapsto \mathscr{H}_p$ be the p-th feature mapping that maps $\mathbf{x}$ onto a reproducing kernel Hilbert space $\mathscr{H}_p$ $(1 \le p \le m)$. And each sample is represented

as $\phi_{\boldsymbol{\beta}}(\mathbf{x}) = [\beta_1\phi_1(\mathbf{x})^\top, \cdots, \beta_m\phi_m(\mathbf{x})^\top]^\top$, 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

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

A kernel matrix $\mathbf{K}_{\boldsymbol{\beta}}$ is then calculated by applying the kernel function $\kappa_{\boldsymbol{\beta}}(\cdot,\cdot)$ into $\{\mathbf{x}_i\}_{i=1}^n$. Based on the kernel matrix $\mathbf{K}_{\boldsymbol{\beta}}$, the objective of MKKM can be written as

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

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 given $\boldsymbol{\beta}$.** 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);

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

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 $\boldsymbol{\beta}$ given H.** With $\mathbf{H}$ fixed, $\boldsymbol{\beta}$ can be optimized via solving the following quadratic programming with linear constraints,

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

As noted in [2][3], 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 viable option, because this could make only one single kernel be activated and all the others assigned with zero weights. In the following, we apply the late fusion to multiple kernel k-means to have the optimal clustering results by integrating the multi-view results. Hence it can be seen as the ensemble method in another way.

3. Multiple kernel k-means with late fusion(MKKM-LF)

As mentioned in section 2, our work is built on the fundamental base on the multiple kernel k-means and late fusion. Different from the former framework chasing for a optimal kernel to cluster, we decide to seek the best assignment matrix from a variety of assignment matrices taking by different kernels. In other words, by taking every single kernel k-means with kernel $\{\mathbf{K}_{i=1}^{\mathbf{m}}\}$, we could have an assignment matrix $\mathbf{H}_i$ at each time. 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 $\mathbf{H}_i$ could be seen as the same results once one matrix can change to another one through column transformation. Further, with a permutation matrix $\mathbf{W}_i$, we have that the new assignment matrix $\mathbf{H}_i\mathbf{W}_i$ has the same clustering result with the original matrix $\mathbf{H}_i$.

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 assignment matrix.

3.1. Proposed formulation

With discussion mentioned above, our motivation is to find the ideal assignment matrix $\mathbf{H}$ by integrating a number of the weighted original matrices $\{\mathbf{H}_{i=1}^{m}\}$. In multiple kernel setting with a given set of m kernels $\{\mathbf{K}_{i=1}^{m}\}$, we apply kernel k-means algorithm on each kernel and have a set of resultant assignment matrices $\{\mathbf{H}_{i=1}^{m}\}$. 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 set our optimization function as follows:

$$\begin{aligned} &\min_{\mathbf{H},\{\mathbf{W_p}\}_{p=1}^{m},\gamma} \left\| \mathbf{H} - \sum_{p=1}^{m} \gamma_p \mathbf{H_p} \mathbf{W_p} \right\|_{\mathbf{F}}^{2}, \\ &s.t.\ \ \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k, \gamma^{\top} 1 = 1, \gamma \geq 0. \end{aligned} \tag{8}$$

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

By relaxing the constraints imposed on the column-transformation matrix $\{\mathbf{W}_{i=1}^{m}\}$ to the orthogonality restriction, we get a relaxed version of ,

$$\begin{aligned} &\min_{\mathbf{H},\{\mathbf{W_p}\}_{p=1}^{m},\gamma} \left\| \mathbf{H} - \sum_{p=1}^{m} \gamma_p \mathbf{H_p} \mathbf{W_p} \right\|_{\mathbf{F}}^{2}, \\ &s.t.\ \ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_k, \mathbf{W}^\top \mathbf{W} = \mathbf{I}_k, \gamma^\top 1 = 1, \gamma \geq 0. \end{aligned} \tag{9}$$

Moreover, we impose the regularization on the coefficients and get the optimization function as follows,

$$\begin{aligned} &\min_{\mathbf{H},\{\mathbf{W_p}\}_{p=1}^{m},\gamma} \left\| \mathbf{H} - \sum_{p=1}^{m} \gamma_p \mathbf{H_p} \mathbf{W_p} \right\|_{\mathbf{F}}^{2} + \lambda \|\gamma\|_{\mathbf{F}}^{2}, \\ &s.t.\ \ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_k, \mathbf{W}^\top \mathbf{W} = \mathbf{I}_k, \gamma^\top 1 = 1, \gamma \geq 0. \end{aligned} \tag{10}$$

3.2. Optimization algorithm

Although the problem in Eq.10 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, which each step could be easily solved by the existing off-the-shelf packages.

*3.2.1. **Optimization** $\mathbf{H}$ **with fixed** $\{\mathbf{W_p}\}_{p=1}^{m}$ **and** γ.*

With $\{\mathbf{W_p}\}_{p=1}^{m}$ and γ being fixed, the optimization Eq.10 could be rewritten as follows,

$$\begin{aligned} &\max_{\mathbf{H}} \mathrm{Tr}(\mathbf{H}^\top \mathbf{U}) \\ &s.t.\ \ \mathbf{H}^\top \mathbf{H} = \mathbf{I}_k, \end{aligned} \tag{11}$$

where $\mathbf{U} = \sum_{p=1}^{m} \gamma_p \mathbf{H_p} \mathbf{W_p}$. And this problem in Eq.11 could be easily solved by taking the singular value decomposition(SVD) of the given matrix $\mathbf{U}$.

*3.2.2. **Optimization** $\{\mathbf{W_p}\}_{p=1}^m$ **with fixed** $\mathbf{H}$ **and** γ.*

With $\mathbf{H}$ and γ being fixed, for each single $\mathbf{W_p}$, the optimization problem in Eq.10 is equivalent to Eq.12 as follows,

$$\begin{aligned} &\max_{\mathbf{W}_p} \mathrm{Tr}(\mathbf{W_p}^\top \mathbf{V}) \\ &s.t.\ \mathbf{W_p}^\top \mathbf{W_p} = \mathbf{I}_k, \end{aligned} \tag{12}$$

where $\mathbf{V} = \mathbf{H}_p^T(\mathbf{H} - \sum_{q=1,q\neq p}^m \gamma_q \mathbf{H_q}\mathbf{W_q})$. And this problem in Eq.12 could be easily solved by taking the singular value decomposition(SVD) of the given matrix $\mathbf{V}$. 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 $\{\mathbf{W}_p\}_{p=1}^m$.

*3.2.3. **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.10 is equivalent to the optimization problem as follows,

$$\begin{aligned} &\min_{\gamma} \frac{1}{2}\gamma^\top A\gamma - \frac{1}{k}\mathbf{f}^\top \gamma, \\ &s.t.\ \ \gamma^\top \mathbf{1} = 1, \gamma \geq 0, \end{aligned} \tag{13}$$

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

With the simplified problem proposed in Eq.(13), we have observed that this problem is a convex function and could be efficiently solved via the existing convex optimization package.

Our algorithm is outlined in Algorithm 1, where $\mathrm{obj}^{(t)}$ denotes the objective value at the t-th iterations. The objective of Algorithm 1 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 five iterations in all of our experiments.

Algorithm 1 Proposed MKKM-LF

Input: $\{\mathbf{W_p}\}_{p=1}^{m}, \tau, \lambda$ and ε_0.

Output: $\mathbf{H}, \gamma$.

Initialize $\{\mathbf{W_p}\}_{p=1}^{m} = \mathbf{I_k}, \gamma = \frac{1}{m}$ and $t = 1$.

repeat

    Update $\mathbf{H}$ by solving Eq.(11) with fixed $\{\mathbf{W_p}\}_{p=1}^{m}$ and γ.

    Update $\{\mathbf{W_p}\}_{p=1}^{m}$ with fixed $\mathbf{H}$ and γ by Eq.(12).

    Update γ by solving Eq.(13) with fixed $\mathbf{H}$ and $\{\mathbf{W_p}\}_{p=1}^{m}$.

    $t = t + 1$.

until $\left(\text{obj}^{(t-1)} - \text{obj}^{(t)}\right)/\text{obj}^{(t)} \leq \varepsilon_0$

4. Experiments

4.1. Datasets

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 Caltech102[2] and Protein fold prediction[3] and UCI-Digital[4] and Caltech 101[5]. The detailed information of the several datasets are listed in Table 1.

For the ProteinFold Dataset, we use the kernel generating method proposed by [10]. For other benchmark multiple kernel datasets, we use the pre-defined kernel matrices and download them from the official website.

4.2. Compared algorithm

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

- Average multiple kernel k-means (A-MKKM): All kernels are uniformly weighted to generate a new kernel, which is taken as the input of kernel

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

[2] http://files.is.tue.mpg.de/pgehler/projects/iccv09/

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

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

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

Table 1: Datasets used in our experiments.

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

k-means.

- Single best kernel k-means (SB-KKM): Kernel k-means is performed on each single kernel and the best result is outputed.

- Multiple kernel k-means (MKKM) ([11]): The algorithm alternatively performs kernel k-means and updates the kernel coefficients, as introduced in the related work.

- Co-regularized spectral clustering (CRSC) ([12]): CRSC provides a co-regularization way to perform spectral clustering.

- Multiple kernel k-means with Matrix-induced Regularization(MKKM-MR) ([4]): The algorithm apply 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)([5]): The algorithm maximizes the local kernel alignment with multiple kernel clustering and focuses on closer sample pairs that shall stay together.

4.3. Experiment setting

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 [13]. 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 regularization parameters λ and τ are chosen from $\left[2^{-10}, 2^{-9}, \cdots, 2^{10}\right]$ and $[0.1, 0.2, \cdots, 0.9] \times n$ 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 effectness of randomness caused by k-means, and report the best result.

4.4. Experimental results

The ACC, NMI and Purity of the compared algorithms on the five benchmark datasets are displayed in Table 2. We also plot the running time of the mentioned algorithms on each datasets in Table 3. Due to the results, we have the following conclusions:

- Our proposed algorithm always achieves the second best on the five datasets while it is much closer between ours and the best one. Taking the largest dataset Flower102 as an example, our algorithm arrives 42.43% which the best is 43.23%(MKC-LKA), and significantly outperforms the other ones.

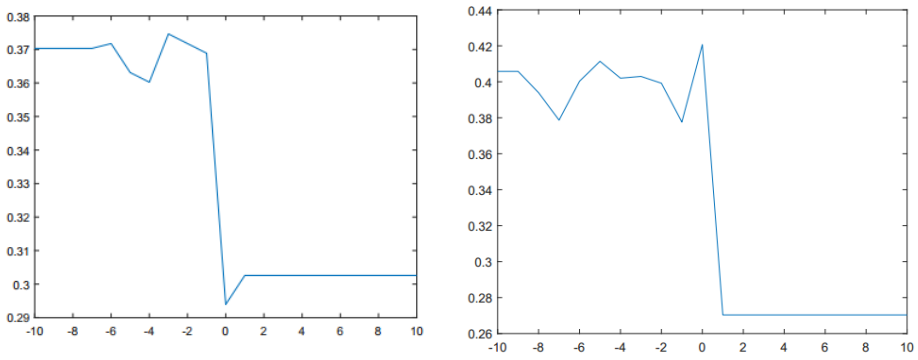
- As mentioned before, different from framework of the MKKM, MKC-LKA and MKKM-MR algorithms, the framework of ours with late fusion are more robust in the datasets, which is essential for practical use.

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

Table 2 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: i) effectively make the use of the multiple kernel k-means results and form an optimal clustering result; and ii) well jointly utilizes the contribution of each kernel in the process of clustering and reduce the high computational complexity in former methods. Our framework with late dusion is flexible and allows the prespecified kernels clustering results to be weighted for better clustering, bringing improvements on clustering performance.

4.5. Parameter selection and convergence



(a) (b)

Figure 1: The effect of the regularization parameter λ on ACC in Proteinfold(left) and Flower102(right).

Our proposed algorithm has two hyperparameter λ which represent the regularization coefficient and the neighbourhood ratio respectively. In our algorithm setting, we experimentally investigate the influence of the two hyperparameters to our clustering performance. The selection of regularization coefficient λ is from $\left[2^{-10}, 2^{-9}, \cdots, 2^{10}\right]$ and Figure 1 shows the influences by selection of λ in the two

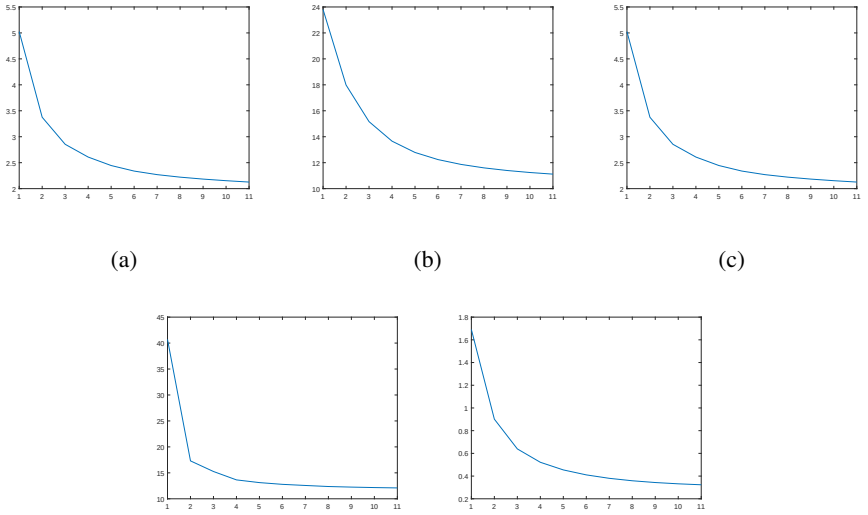
Table 2: ACC, NMI and purity comparison of different clustering algorithms on all data sets.

Datasets	A-MKKM	SB-KKM	MKKM	CSRC	MKC-LKA	MKKM-MR	Proposed
ACC							
Digital	88.75	75.40	47.00	73.20	**95.25**	90.40	92.55
Flower17	51.03	42.06	45.37	51.76	56.76	**60.00**	59.56
ProteinFold	30.69	34.58	27.23	35.59	**39.34**	36.89	39.19
Flower102	27.29	33.13	21.96	38.60	**43.23**	40.24	42.43
Caltech	35.56	33.14	34.77	34.38	**37.06**	35.82	36.21
NMI							
Digital	80.59	68.38	48.16	69.31	**89.73**	83.22	85.36
Flower17	50.19	45.14	45.35	53.19	**57.27**	57.11	54.79
ProteinFold	40.96	42.33	37.16	45.66	47.55	45.13	**49.96**
Flower102	46.32	48.99	42.30	54.95	58.05	57.27	**58.18**
Caltech	59.90	59.07	59.64	58.35	**61.58**	60.38	60.91
Purity							
Digital	88.75	76.10	0.50	76.10	**95.25**	90.40	92.55
Flower17	51.99	44.63	46.84	53.68	58.60	**61.03**	60.07
ProteinFold	37.18	41.21	33.86	42.07	45.97	43.80	**48.85**
Flower102	32.28	38.74	27.61	45.04	48.94	46.39	**48.99**
Caltech	37.12	35.10	37.25	35.95	**39.08**	37.65	38.24

Table 3: The time cost of different clustering algorithms on all data sets(sec.).

Datasets	A-MKKM	SB-KKM	MKKM	CSRC	MKC-LKA	MKKM-MR	Proposed
Digital	0.96	4.63	3.82	92.90	12.53	3.74	4.57
Flower17	0.74	4.58	2.13	46.04	6.05	4.47	4.10
ProteinFold	0.34	4.01	1.03	20.03	2.06	1.9401	2.01
Flower102	27.51	120.50	127.77	3226.60	1027.40	282.13	335.02
Caltech	1.9372	42.23	10.12	333.75	35.05	35.76	38.49

datasets. From the figure, we have the following conclusions: i)With the increasing value of λ, the ACC first increases to its highest value and then decreases; ii)



(a) (b) (c)

(d) (e)

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

The best ACC is always achieved by appropriately integrating the two terms.

We also plot the objective value of our algorithm at each iteration in Figure 2. As observed, this value is monotonically decreased and the algorithm usually converges in ten iterations.

5. 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 new algorithm, **MKKM-LF**, shows promising performance, 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.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Project NO.61672528, NO. 61671463).

References

[1] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural computation 10 (5) (1998) 1299–1319.

[2] S. Yu, L. C. Tranchevent, B. D. Moor, Y. Moreau, Optimized data fusion for kernel k-means clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (5) (2012) 1031–1039.

[3] M. Gönen, A. A. Margolin, Localized data fusion for kernel k-means clustering with application to cancer biology, in: Advances in Neural Information Processing Systems, 2014, pp. 1305–1313.

[4] X. Liu, Y. Dou, J. Yin, L. Wang, E. Zhu, Multiple kernel k -means clustering with matrix-induced regularization, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016, pp. 1888–1894.

[5] M. Li, X. Liu, L. Wang, Y. Dou, J. Yin, E. Zhu, Multiple kernel clustering with local kernel alignment maximization, in: International Joint Conference on Artificial Intelligence, 2016, pp. 1704–1710.

[6] C. G. M. Snoek, M. Worring, A. W. M. Smeulders, Early versus late fusion in semantic video analysis, in: ACM International Conference on Multimedia, 2005, pp. 399–402.

[7] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, Q. Tian, Query-adaptive late fusion for image search and person re-identification, in: Computer Vision and Pattern Recognition, 2015, pp. 1741–1750.

[8] R. T. Ionescu, S. Smeureanu, B. Alexe, M. Popescu, Unmasking the abnormal events in video.

[9] E. Bruno, S. Marchand-Maillet, Multiview clustering: a late fusion approach using latent models, in: International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, pp. 736–737.

[10] T. Damoulas, M. A. Girolami, Probabilistic multi-class multi-kernel learning: on protein fold recognition and remote homology detection, Bioinformatics 24 (10) (2008) 1264–1270.

[11] H.-C. Huang, Y.-Y. Chuang, C.-S. Chen, Multiple kernel fuzzy clustering, IEEE Transactions on Fuzzy Systems 20 (1) (2012) 120–134.

[12] A. Kumar, P. Rai, H. Daume, Co-regularized multi-view spectral clustering, in: Advances in neural information processing systems, 2011, pp. 1413–1421.

[13] C. Cortes, M. Mohri, A. Rostamizadeh, Algorithms for learning kernels based on centered alignment, Journal of Machine Learning Research 13 (Mar) (2012) 795–828.