Multiple Kernel k-Means Clustering with Matrix-induced Regularization

Xinwang Liu, Yong Dou, Jianping Yin
School of Computer
National University
of Defense Technology
Changsha, China, 410073

Lei Wang
School of Computer Science
and Software Engineering
University of Wollongong
NSW, Australia, 2522

En Zhu
School of Computer
National University
of Defense Technology
Changsha, China, 410073

Abstract

Multiple kernel k-means (MKKM) clustering aims to optimally combine a group of pre-specified kernels to improve clustering performance. However, we observe that existing MKKM algorithms do not sufficiently consider the correlation among these kernels. This could result in selecting mutually redundant kernels and affect the diversity of information sources utilized for clustering, which finally hurts the clustering performance. To address this issue, this paper proposes an MKKM clustering with a novel, effective matrix-induced regularization to reduce such redundancy and enhance the diversity of the selected kernels. We theoretically justify this matrix-induced regularization by revealing its connection with the commonly used kernel alignment criterion. Furthermore, this justification shows that maximizing the kernel alignment for clustering can be viewed as a special case of our approach and indicates the extendability of the proposed matrix-induced regularization for designing better clustering algorithms. As experimentally demonstrated on five challenging MKL benchmark data sets, our algorithm significantly improves existing MKKM and consistently outperforms the state-of-the-art ones in the literature, verifying the effectiveness and advantages of incorporating the proposed matrix-induced regularization.

Introduction

Clustering algorithms aim to partition a group of samples into k clusters, where the similarity of samples from intra-clusters shall be greater than that from inter-clusters (Hartigan 1975). As one of the classical clustering algorithms, k-means provides an intuitive and effective way to perform clustering. In specific, the k-means clustering is composed of (i) calculating k prototypes (i.e., centres of k clusters) given an assignment of samples to clusters and (ii) updating the assignment matrix by minimizing the sum-of-squares cost given the prototypes. These two steps are alternately performed until convergence. Due to its conceptual simplicity, easy-implementation and high efficiency, k-means clustering has been intensively studied and extended (Yu et al. 2012; Gönen and Margolin 2014; Cai, Nie, and Huang 2013; Du et al. 2015). As an important extension, kernel k-means first maps data onto a high-dimensional space through a feature mapping and then conducts a standard k-means clustering in that space (Schölkopf, Smola, and Müller 1998). This enables kernel k-means to handle the linearly non-separable problem in an input space that k-means suffers from.

In many practical applications of clustering, samples are represented by multiple groups of features extracted from different information sources. For example, three kinds of feature representations[1], colour, shape and texture, are extracted to distinguish one flower from another (Nilsback and Zisserman 2006). These different sources usually provide complementary information, and it is better to let learning algorithms optimally combine them in order to obtain excellent clustering. This line of research is known as multiple kernel (view) clustering in the literature.

Many efforts have been devoted to improving multiple kernel clustering from all kinds of aspects (Zhao, Kwok, and Zhang 2009; Lu et al. 2014; Xia et al. 2014; Zhou et al. 2015; Kumar and Daumé 2011). In this paper, we explore a better way to combine a set of pre-specified kernels for clustering. The existing research on this aspect can roughly be grouped into two categories. The first category learns a consensus matrix via low-rank optimization (Xia et al. 2014; Zhou et al. 2015; Kumar and Daumé 2011). In (Xia et al. 2014), a transition probability matrix is constructed from each single view, and they are used to recover a shared low-rank transition probability matrix as a crucial input to the standard Markov chain method for clustering. The work in (Zhou et al. 2015) proposes to capture the structures of noises in each kernel and integrate them into a robust and consensus framework to learn a low-rank matrix. The algorithm (Kumar and Daumé 2011) learns the clustering in one view and uses it to "label" the data in other views to modify a similarity matrix. By following multiple kernel learning (MKL) framework, the other category optimizes a group of kernel coefficients, and uses the combined kernel for clustering (Yu et al. 2012; Gönen and Margolin 2014; Du et al. 2015; Lu et al. 2014). The work in (Yu et al. 2012) proposes a multiple kernel k-means clustering algorithm. Similar work has also been done in (Gönen and Margolin 2014), with the difference that the kernels are combined in a localized way to better capture the sample-adaptive characteristics of data. Differently, by replacing the squared error in k-means with

[1] In literature, each representation is also termed as a view.

an $\ell_{2,1}$-norm based one, (Du et al. 2015) presents a robust multiple kernel k-means algorithm that simultaneously finds the best clustering labels and the optimal combination of multiple kernels. In (Lu et al. 2014), kernel alignment maximization is employed to jointly perform the k-means clustering and MKL. Our work in this paper falls into the second category.

Though the above clustering algorithms in the second category have demonstrated excellent performance in various scenarios, we find that none of them has sufficiently considered the correlation among the pre-specified kernels. Specifically, in these algorithms, the update of one kernel combination coefficient is (generally) independent of the others. Nevertheless, this would cause the following problems: i) Kernels with high correlation are selected together. This unnecessarily increases the number of kernels and information sources used for clustering; and ii) Kernels with low correlation could undesirably be suppressed due to the sparsity constraint imposed on the combination coefficients. This decreases the diversity among the selected kernels and prevents the complementary kernels from being utilized. Both problems make the pre-specified kernels less effectively utilized, and in turn adversely affect the clustering performance.

To reduce the redundancy and enhance the diversity of selected kernels, we propose a multiple kernel k-means clustering algorithm with a matrix-induced regularization, where the correlation of each pair of kernels is measured and used to constrain the cost function of kernel k-means. After that, we show that maximizing the well-known kernel alignment criterion for clustering (Lu et al. 2014) can be viewed as a special case of our approach, and this provides the theoretical justification for the incorporation of the proposed matrix-induced regularization. Also, we can see that the proposed algorithm is readily extendable to develop better clustering algorithms by designing the matrix-induced regularization appropriate for a given clustering task. 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 significantly improves the performance of MKKM and consistently demonstrates superior performance when compared with the state-of-the-art ones. This validates the effectiveness and advantage of incorporating the proposed matrix-induced regularization.

Related Work

Kernel k-means clustering (KKM)

Let $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X}$ be a collection of n samples, and $\phi(\cdot) : \mathbf{x} \in \mathcal{X} \mapsto \mathcal{H}$ be a feature mapping which maps $\mathbf{x}$ onto a reproducing kernel Hilbert space $\mathcal{H}$. The objective of kernel k-means clustering is to minimize the sum-of-squares loss over the cluster assignment matrix $\mathbf{Z} \in \{0,1\}^{n\times k}$, which can be formulated as the following optimization problem,

$$\min_{\mathbf{Z}\in\{0,1\}^{n\times k}} \sum_{i=1,c=1}^{n,k} Z_{ic}\|\phi(\mathbf{x}_i) - \boldsymbol{\mu}_c\|_2^2 \; s.t. \sum_{c=1}^{k} Z_{ic} = 1, \quad (1)$$

where $n_c = \sum_{i=1}^n Z_{ic}$ and $\boldsymbol{\mu}_c = \frac{1}{n_c}\sum_{i=1}^n Z_{ic}\phi(\mathbf{x}_i)$ are the number and centroid of the $c-\text{th}\,(1 \le c \le k)$ cluster.

The optimization problem in Eq.(1) can be equivalently rewritten as the following matrix-vector form,

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

where $\mathbf{K}$ is a kernel matrix with $K_{ij} = \phi(\mathbf{x}_i)^\top\phi(\mathbf{x}_j)$, $\mathbf{L} = \mathrm{diag}([n_1^{-1}, n_2^{-1}, \cdots, n_k^{-1}])$ and $\mathbf{1}_\ell \in \mathbb{R}^\ell$ is a column vector with all elements 1.

The variables $\mathbf{Z}$ in Eq.(2) is discrete, which makes the optimization problem very difficult to solve. However, we can approximate this problem through relaxing $\mathbf{Z}$ to take arbitrary real values. Specifically, by defining $\mathbf{H} = \mathbf{Z}\mathbf{L}^{\frac{1}{2}}$ and letting $\mathbf{H}$ take real values, we obtain a relaxed version of the above problem.

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

where $\mathbf{I}_k$ is an identity matrix with size $k \times k$. Noting that $\mathbf{Z}^\top\mathbf{Z} = \mathbf{L}^{-1}$, we have $\mathbf{L}^{\frac{1}{2}}\mathbf{Z}^\top\mathbf{Z}\mathbf{L}^{\frac{1}{2}} = \mathbf{I}_k$, and this leads to the orthogonality constraint on $\mathbf{H}$. Finally, one can obtain the optimal $\mathbf{H}$ for Eq.(3) by taking the k eigenvectors that correspond to the k largest eigenvalues of $\mathbf{K}$.

Multiple kernel k-means clustering (MKKM)

In a multiple kernel setting, each sample has multiple feature representations via a group of feature mappings $\{\phi_p(\cdot)\}_{p=1}^m$. Specifically, each sample is represented as $\phi_{\boldsymbol{\mu}}(\mathbf{x}) = [\mu_1\phi_1(\mathbf{x})^\top, \mu_2\phi_2(\mathbf{x})^\top, \cdots, \mu_m\phi_m(\mathbf{x})^\top]^\top$, where $\boldsymbol{\mu} = [\mu_1, \mu_2, \cdots, \mu_m]^\top$ denotes the coefficients of each base kernel that we need to optimize during learning. Correspondingly, the kernel function over the above mapping function can be calculated as

$$\kappa_{\boldsymbol{\mu}}(\mathbf{x}_i, \mathbf{x}_j) = \phi_{\boldsymbol{\mu}}(\mathbf{x}_i)^\top\phi_{\boldsymbol{\mu}}(\mathbf{x}_j) = \sum\nolimits_{p=1}^m \mu_p^2\kappa_p(\mathbf{x}_i, \mathbf{x}_j). \quad (4)$$

By replacing the kernel matrix $\mathbf{K}$ in Eq.(3) with $\mathbf{K}_{\boldsymbol{\mu}}$ computed via Eq.(4), we obtain the optimization objective of MKKM as follows,

$$\begin{aligned}\min_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\mu}\in\mathbb{R}_+^m} \; & \mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right) \\ s.t. \quad & \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \; \boldsymbol{\mu}^\top\mathbf{1}_m = 1.\end{aligned} \quad (5)$$

This problem can be solved by alternatively updating $\mathbf{H}$ and $\boldsymbol{\mu}$: i) **Optimizing $\mathbf{H}$ given $\boldsymbol{\mu}$.** With the kernel coefficients $\boldsymbol{\mu}$ fixed, the $\mathbf{H}$ can be obtained by solving a kernel k-means clustering optimization problem in Eq.(3); ii) **Optimizing $\boldsymbol{\mu}$ given $\mathbf{H}$.** With $\mathbf{H}$ fixed, $\boldsymbol{\mu}$ can be optimized via solving the following quadratic programming with linear constraints,

$$\min_{\boldsymbol{\mu}\in\mathbb{R}_+^m} \; \sum\nolimits_{p=1}^m \mu_p^2\mathrm{Tr}\left(\mathbf{K}_p(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right) \; \boldsymbol{\mu}^\top\mathbf{1}_m = 1. \quad (6)$$

As noted in (Yu et al. 2012; Gönen and Margolin 2014), using a convex combination of kernels $\sum_{p=1}^m \mu_p\mathbf{K}_p$ to replace $\mathbf{K}_{\boldsymbol{\mu}}$ 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.

The Proposed MKKM Clustering with Matrix-induced Regularization

As can be seen from Eq.(6), the relative value of μ_p is only dependent on $\mathbf{K}_p$ and the given $\mathbf{H}$, while independent of the other kernels. This clearly indicates that existing MKKM algorithms (Yu et al. 2012; Gönen and Margolin 2014; Du et al. 2015) do not adequately consider the mutual influence of these kernels when updating kernel coefficients. To see this point in depth, we assume that $\mathbf{K}_p$ is selected and assigned to a large weight. According to Eq.(6), the kernels with high correlation with $\mathbf{K}_p$ would be also selected together and assigned to similar important weights. This, clearly, would result in the high redundancy among the selected kernels. On the other hand, the selection of highly correlated kernels could suppress the weights of kernels that are less correlated with $\mathbf{K}_p$ due to the sparsity constraint (an ℓ_1-norm) imposed on the kernel coefficients. This would cause the low diversity among the selected kernels or even prevent complementary kernels from being utilized.

Following the above analysis, we can see that existing MKKM algorithms do not take a sufficient consideration of the characteristics of these pre-specified kernels, which could lead to unsatisfying clustering performance. This motivates us to derive a matrix-induced regularization on the kernel coefficients to improve this situation.

The proposed formulation

To reduce the redundancy and enforce the diversity of the selected kernels, we need a regularization term that is able to characterise the correlation of each pair of kernels.

To this end, we first define a criterion $\mathcal{M}(\mathbf{K}_p, \mathbf{K}_q)$ to measure the correlation between $\mathbf{K}_p$ and $\mathbf{K}_q$. A larger $\mathcal{M}(\mathbf{K}_p, \mathbf{K}_q)$ means high correlation between $\mathbf{K}_p$ and $\mathbf{K}_q$, and a smaller one implies that their correlation is low. Therefore, a natural optimization criterion to prevent two highly correlated kernels from being selected can be defined as $\mu_p\mu_q\mathcal{M}(\mathbf{K}_p, \mathbf{K}_q)$. As observed, by minimizing this term, the risk of simultaneously assigning μ_p and μ_q with large weights can be greatly reduced. Also, this regularization increases the probability of jointly assigning μ_p and μ_q with larger weights as long as $\mathbf{K}_p$ and $\mathbf{K}_q$ are less correlated. As a consequence, this criterion is beneficial to promote the diversity of selected kernels. Based on these observations, we propose the following regularization term

$$\min_{\boldsymbol{\mu}\in\mathbb{R}_+^m} \sum_{p,q=1}^{m} \mu_p\mu_q M_{pq} = \boldsymbol{\mu}^\top \mathbf{M}\boldsymbol{\mu} \;\; s.t. \;\; \boldsymbol{\mu}^\top \mathbf{1}_m = 1, \quad (7)$$

where $\mathbf{M}$ is a matrix with $M_{pq} = \mathcal{M}(\mathbf{K}_p, \mathbf{K}_q)$. We call the objective in Eq.(7) as matrix-induced regularization.

By integrating the matrix-induced regularization into the objective function of existing MKKM, we obtain the optimization problem of the proposed algorithm as follows,

$$\begin{aligned} \min_{\mathbf{H}\in\mathbb{R}^{n\times k}, \boldsymbol{\mu}\in\mathbb{R}_+^m} \; & \mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right) + \frac{\lambda}{2}\boldsymbol{\mu}^\top\mathbf{M}\boldsymbol{\mu} \\ s.t. \; & \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \; \boldsymbol{\mu}^\top\mathbf{1}_m = 1 \end{aligned} \quad (8)$$

where λ is a parameter to trade off the clustering cost function and the regularization term.

Algorithm 1 The Proposed MKKM Clustering with Matrix-induced Regularization

Input: $\{\mathbf{K}_p\}_{p=1}^m$, k, λ and ϵ_0.
Output: $\mathbf{H}$ and $\boldsymbol{\mu}$.
Initialize $\boldsymbol{\mu}^{(0)} = \mathbf{1}_m/m$ and $t = 1$.
repeat
  $\mathbf{K}_{\boldsymbol{\mu}}^{(t)} = \sum_{p=1}^m \left(\mu_p^{(t-1)}\right)^2 \mathbf{K}_p$.
  Update $\mathbf{H}^{(t)}$ by solving Eq.(3) with given $\mathbf{K}_{\boldsymbol{\mu}}^{(t)}$.
  Update $\boldsymbol{\mu}^{(t)}$ by solving Eq.(9) with given $\mathbf{H}^{(t)}$.
  $t = t + 1$.
until $\left(\mathrm{obj}^{(t-1)} - \mathrm{obj}^{(t)}\right)/\mathrm{obj}^{(t)} \leq \epsilon_0$

Alternate optimization

We propose a two-step algorithm to solve the optimization problem in Eq.(8) alternatively. (i) **Optimizing $\mathbf{H}$ with fixed $\boldsymbol{\mu}$.** Given $\boldsymbol{\mu}$, the optimization in Eq.(8) w.r.t $\mathbf{H}$ is a standard kernel k-means clustering problem, and the $\mathbf{H}$ can be obtained by solving Eq.(3) with given $\mathbf{K}_{\boldsymbol{\mu}}$; (ii) **Optimizing $\boldsymbol{\mu}$ with fixed $\mathbf{H}$.** Given $\mathbf{H}$, the optimization in Eq.(8) w.r.t $\boldsymbol{\mu}$ is a quadratic programming with linear constraints. In specific, we can obtain $\boldsymbol{\mu}$ by solving the following problem,

$$\min_{\boldsymbol{\mu}\in\mathbb{R}_+^m} \; \frac{1}{2}\boldsymbol{\mu}^\top \left(2\mathbf{Z} + \lambda\mathbf{M}\right)\boldsymbol{\mu} \quad s.t. \; \boldsymbol{\mu}^\top\mathbf{1}_m = 1, \quad (9)$$

where $\mathbf{Z} = \mathrm{diag}\Big(\left[\mathrm{Tr}\big(\mathbf{K}_1(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\big), \mathrm{Tr}\big(\mathbf{K}_2(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\big) \right.$, $\left. \cdots, \mathrm{Tr}\big(\mathbf{K}_m(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\big)\right] \Big)$.

Our algorithm for solving Eq.(8) 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 ten iterations in all of our experiments.

Discussion and extension

We revisit the objective of our algorithm from the perspective of kernel alignment maximization, with the aim to better justify the incorporation of matrix-induced regularization.

As a well-established criterion, kernel alignment maximization has been widely used to perform kernel tuning in supervised learning (Cortes, Mohri, and Rostamizadeh 2012). Nevertheless, this criterion is not directly applicable due to the absence of true labels in unsupervised learning. A promising remedy is to update kernel coefficients by maximizing the alignment between the combined kernel $\mathbf{K}_{\boldsymbol{\mu}}$ and $\mathbf{H}\mathbf{H}^\top$, where $\mathbf{H}$ is composed of the discriminative eigenvectors generated by kernel k-means (Lu et al. 2014). In specific, the kernel alignment maximization for clustering can be

fulfilled as,

$$\max_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\mu}\in\mathbb{R}_+^m} \frac{\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{H}\mathbf{H}^\top\rangle_{\mathrm{F}}}{\sqrt{\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{K}_{\boldsymbol{\mu}}\rangle_{\mathrm{F}}}} \ s.t.\ \mathbf{H}^\top\mathbf{H}=\mathbf{I}_k,\ \boldsymbol{\mu}^\top\mathbf{1}_m=1, \tag{10}$$

where $\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{H}\mathbf{H}^\top\rangle_{\mathrm{F}} = \mathrm{Tr}(\mathbf{K}_{\boldsymbol{\mu}}\mathbf{H}\mathbf{H}^\top)$, $\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{K}_{\boldsymbol{\mu}}\rangle_{\mathrm{F}} = \hat{\boldsymbol{\mu}}^\top\hat{\mathbf{M}}\hat{\boldsymbol{\mu}}$ with $\hat{\boldsymbol{\mu}} = [\mu_1^2, \mu_2^2, \cdots, \mu_m^2]^\top$ and $\hat{\mathbf{M}}$ is a matrix with $\hat{M}_{pq} = \mathrm{Tr}(\mathbf{K}_p^\top\mathbf{K}_q)$.

The optimization in Eq.(10) is readily understood. Directly optimizing Eq.(10), however, is difficult since it is a four-order fractional optimization problem. By looking into the numerator and denominator of Eq.(10) in depth, we observe that: i) The negative of the numerator of kernel alignment, i.e., $-\mathrm{Tr}(\mathbf{K}_{\boldsymbol{\mu}}\mathbf{H}\mathbf{H}^\top)$, is conceptually equivalent to the objective of MKKM, i.e., $\mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right)$; and ii) The denominator, i.e., $\hat{\boldsymbol{\mu}}^\top\hat{\mathbf{M}}\hat{\boldsymbol{\mu}}$, is a regularization on the kernel coefficients to prevent μ_p and μ_q from being jointly assigned to a large weight if $\hat{M}_{pq}$ is relatively high. From the perspective of regularization, the effect of $\boldsymbol{\mu}^\top\hat{\mathbf{M}}\boldsymbol{\mu}$ and $\hat{\boldsymbol{\mu}}^\top\hat{\mathbf{M}}\hat{\boldsymbol{\mu}}$ could be treated as the same. Comparably, we prefer to the former one since: i) This term fully fulfills our requirement to regularize kernel coefficients; and ii) It leads to a much more tractable optimization problem, i.e., a widely used quadratic programming problem with linear constraints.

Based on the above-mentioned observations, instead of rigidly maximizing the kernel alignment by solving a fractional optimization in Eq.(10), we propose to minimize the negative of the numerator $\mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right)$ and the denominator via $\boldsymbol{\mu}^\top\hat{\mathbf{M}}\boldsymbol{\mu}$, leading to the following problem:

$$\begin{aligned}\min_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\mu}\in\mathbb{R}_+^m} & \ \mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right) + \frac{\lambda}{2}\boldsymbol{\mu}^\top\hat{\mathbf{M}}\boldsymbol{\mu} \\ s.t.\ & \ \mathbf{H}^\top\mathbf{H}=\mathbf{I}_k,\ \boldsymbol{\mu}^\top\mathbf{1}_m=1,\end{aligned} \tag{11}$$

where λ is introduced to trade off the two terms.

As seen, Eq.(11) would be exactly the same as the one in Eq.(8) by setting $\mathcal{M}(\mathbf{K}_p, \mathbf{K}_q) = \mathrm{Tr}(\mathbf{K}_p^\top\mathbf{K}_q)$. The above analysis has revealed the connection between the proposed algorithm and kernel alignment maximization, and well justified the theoretical implication of incorporating matrix-induced regularization.

In addition, the proposed algorithm is readily extendable by finely designing $\mathbf{M}$ appropriate for a given clustering task. For example, $\mathcal{M}(\mathbf{K}_p, \mathbf{K}_q)$ could be defined according to some commonly used criteria such as Kullback-Leibler (KL) divergence (Topsoe 2000), maximum mean discrepancy (Gretton et al. 2006) and Hilbert-Schmidt independence criteria (HSIC), to name just a few. Throughout this paper, we set $\mathcal{M}(\mathbf{K}_p, \mathbf{K}_q) = \mathrm{Tr}(\mathbf{K}_p^\top\mathbf{K}_q)$. Designing proper $\mathbf{M}$ to satisfy various requirements of clustering tasks is interesting and worth exploring in future.

Experimental Results

Data sets

We evaluate the clustering performance of our algorithm on five MKL benchmark data sets, including Oxford Flower17[2], Protein fold prediction[3], UCI-Digital[4], Oxford Flower102[5] and Caltech102[6]. To test the performance of all algorithms with respect to the number of classes, we generate ten data sets by randomly selecting samples the first $10, 20, \cdots, 100$ classes on Flower102 and Caltech102. Detailed information on the data sets is given in Table 1.

Table 1: Datasets used in our experiments.

Dataset	#Samples	#Kernels	#Classes
Flower17	1360	7	17
Digital	2000	3	10
ProteinFold	694	12	27
Flower102	200 : 200 : 2000	25	10 : 10 : 100
Caltech102	150 : 150 : 1500	4	10 : 10 : 100

For ProteinFold, we generate 12 base kernel matrices by following (Damoulas and Girolami 2008), where the second order polynomial kernel and inner product (cosine) kernel are applied to the first ten feature sets and the last two feature sets, respectively. For the other data sets, all kernel matrices are pre-computed and can be publicly downloaded from the above websites.

Compared algorithms

Our algorithm is compared with many recently proposed counterparts, including

- **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 k-means.
- **Single best kernel k-means (SB-KKM)**: Kernel k-means is performed on each single kernel and the best result is reported.
- **Multiple kernel k-means (MKKM)** (Huang, Chuang, and Chen 2012): The algorithm alternatively performs kernel k-means and updates the kernel coefficients, as introduced in the related work.
- **Localized multiple kernel k-means (LMKKM)** (Gönen and Margolin 2014): LMMKM improves MKKM by combining the kernels in a localized way.
- **Robust multiple kernel k-means (RMKKM)** (Du et al. 2015): RMKKM improves the robustness of MKKM by replacing the sum-of-squared loss with an $\ell_{2,1}$-norm one.
- **Co-regularized spectral clustering (CRSC)** (Kumar and Daumé 2011): CRSC provides a co-regularization way to perform spectral clustering.
- **Robust multiview spectral clustering (RMSC)** (Xia et al. 2014): RMSC constructs a transition probability matrix from each single view, and uses them to recover a shared low-rank transition probability matrix for clustering.

[2] http://www.robots.ox.ac.uk/~vgg/data/flowers/17/

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

[4] http://ss.sysu.edu.cn/~py/

[5] http://www.robots.ox.ac.uk/~vgg/data/flowers/102/

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

Table 2: ACC comparison of different clustering algorithms on five benchmark data sets.

Datasets	A-MKKM	SB-KKM	MKKM	LMKKM	RMKKM	CRSC	RMSC	RMKC	Proposed
Flower17	51.03	42.06	45.37	42.94	48.38	52.72	53.90	52.35	**60.00**
Digital	88.75	75.40	47.00	47.00	40.45	84.80	90.40	88.90	**90.95**
ProteinFold	28.10	33.86	27.23	23.49	26.95	34.87	33.00	28.82	**37.32**
Flower102	45.44	42.53	40.27	39.22	39.37	46.99	52.56	46.28	**56.72**
Caltech102	40.79	40.26	40.36	38.12	36.39	41.36	39.86	41.77	**45.39**

Table 3: NMI comparison of different clustering algorithms on five benchmark data sets.

Datasets	A-MKKM	SB-KKM	MKKM	LMKKM	RMKKM	CRSC	RMSC	RMKC	Proposed
Flower17	50.19	45.14	45.35	44.12	50.73	52.13	53.89	50.42	**57.11**
Digital	80.59	68.38	48.16	48.16	46.87	73.51	81.80	80.88	**83.87**
ProteinFold	38.53	42.03	37.16	34.92	38.08	43.34	43.92	39.46	**45.89**
Flower102	60.58	57.88	57.54	57.03	57.13	61.28	66.95	60.76	**68.77**
Caltech102	57.36	56.85	56.78	55.04	52.52	57.48	57.41	58.08	**60.65**

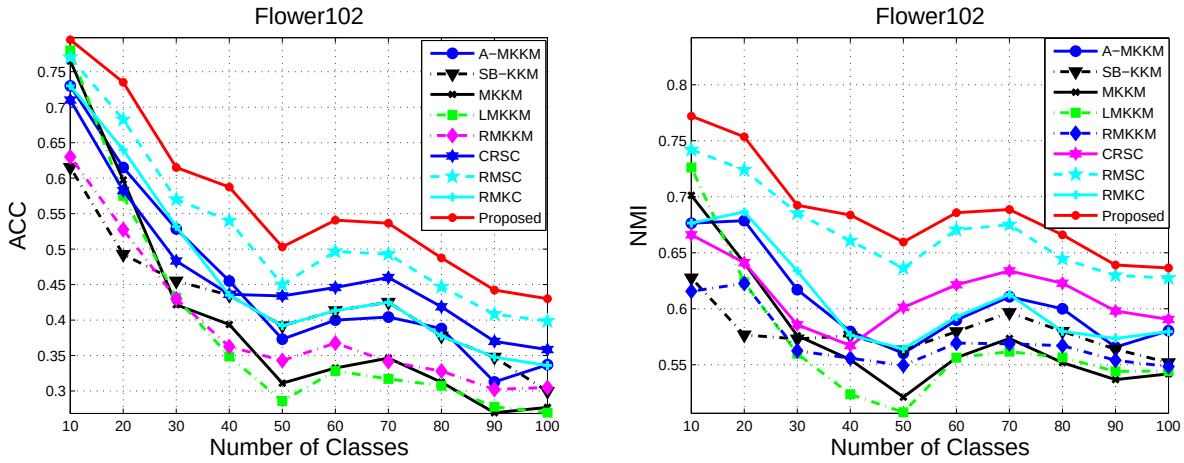

Figure 1: ACC and NMI comparison with variation of number of classes on Flower102. (left) **ACC** and (right) **NMI**

- **Robust Multiple Kernel Clustering (RMKC)** (Zhou et al. 2015): RMKC learns a robust yet low-rank kernel for clustering by capturing the structure of noises in multiple kernels.

The Matlab codes of KKM, MKKM and LMKKM are publicly available from `https://github.com/mehmetgonen/lmkkmeans`. For RMKKM, CRSC, RMSC and RCE, we download their matlab implementations from authors' websites and use them for comparison in our experiments.

Experimental settings

In all experiments, all base kernels are first centered and then scaled so that for all i and p we have $K_p(\mathbf{x}_i, \mathbf{x}_i) = 1$. For all data sets, we assume that the true number of clusters is known and we set it to be the true number of classes. In addition, the parameters of RMKKM, RMSC and RMKC are selected by grid search according to the suggestions in their papers. For our proposed algorithm, its regularization parameter is chosen from $[2^{-15}, 2^{-14}, \cdots, 2^{15}]$ by grid search.

The widely used clustering accuracy (ACC) and normalized mutual information (NMI) are applied to evaluate the clustering performance of each algorithm. For all algorithms, we repeat each experiment for 50 times with random initialization to reduce the affect of randomness caused by k-means, and report the best result. For Flower102 and Caltech102, we report the aggregated ACC (NMI) of each algorithm, which is defined as the mean of ACC (NMI) on datasets with the number of classes varied in the range of $10, 20, \cdots, 100$.

Experimental results

Table 2 reports the clustering accuracy of the above mentioned algorithms on all data sets. From these results, we have the following observations:

- The proposed algorithm consistently demonstrates the best clustering accuracy on all data sets. For example, it exceeds the second best one (RMSC) by over six percentages on Flower17. Also, its superiority is confirmed by the NMI reported in Table 3.
- Our algorithm significantly improves the performance of existing MKKM, where the kernel coefficients are updated independently. Taking the result on Digital for example, the clustering accuracy of MKKM is only 47%, which implies that it may even not work on this dataset. In contrast, our algorithm achieves 90.95%, which is the best

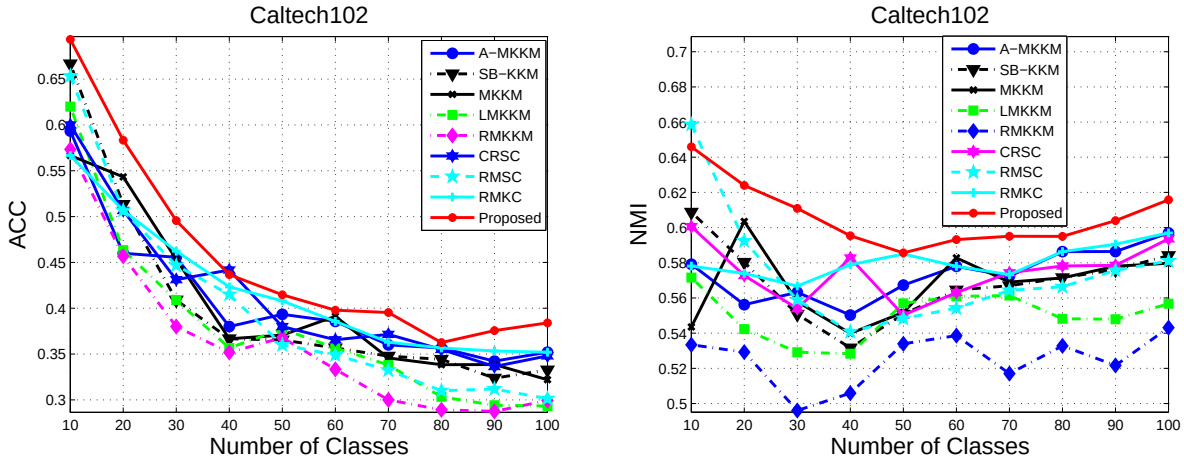

Figure 2: ACC and NMI comparison with variation of number of classes on Caltech102. (left) **ACC** and (right) **NMI**

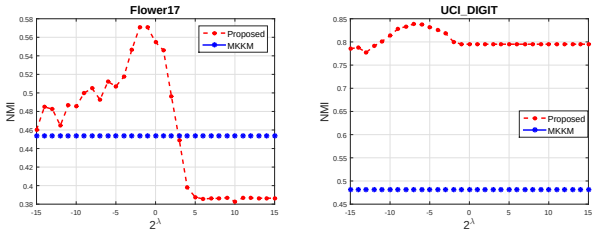

Figure 3: The effect of the regularization parameter λ on NMI. (left) **Flower17** and (right) **Digital**

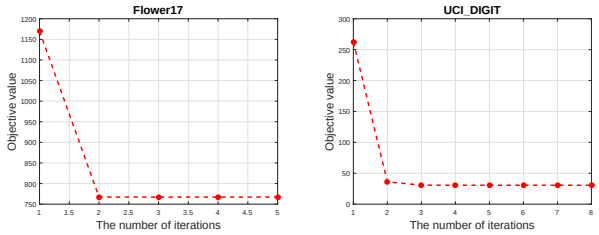

Figure 4: The objective value of our algorithm at each iteration. (left) **Flower17** and (right) **Digital**

performance among all algorithms.

- As a strong baseline, A-MKKM usually demonstrates comparable or even better clustering performance than most of algorithms in comparison. However, our algorithm outperforms this baseline consistently on all data sets, which indicates its robustness in clustering performance.

The NMI comparison of all algorithms are presented in Table 3, from which we obtain the similar observations. These results have well verified the effectiveness and advantages of incorporating matrix-induced regularization.

We also investigate the clustering performance of each algorithm with respect to the number of classes, as shown in Figure 1 and 2. Figure 1 shows the results on Flower102. As observed, our algorithm (in red) consistently keeps on the top of all sub-figures when the number of classes varies, indicating the best performance. Similar results can also be found from the Figure 2.

From the above experiments, we conclude that the proposed algorithm effectively addresses the issues of high redundancy and low diversity among the selected kernels in MKKM. This makes the pre-specified kernels be well utilized, bringing to the significant improvements on clustering performance.

Parameter selection and convergence

The proposed algorithm introduces a regularization parameter λ which balances the objective of kernel k-means and the matrix-induced regularization. We then experimentally show the effect of this parameter on the performance of our algorithm. Figure 3 plots the NMI of our algorithm by varying λ from 2^{-15} to 2^{15}, where the NMI of MKKM is also incorporated as a baseline. From these figures, we have the following observations: i) With the increasing value of λ, the NMI first increases to its highest value and then decreases; ii) The best NMI is always achieved by appropriately integrating the two terms; and iii) Our algorithm outperforms MKKM and shows stable performance across a wide range of smaller λ values.

We also plot the objective value of our algorithm at each iteration in Figure 4. As observed, this value is monotonically decreased and the algorithm usually converges in less than ten iterations.

Conclusions

This work proposes the MKKM algorithm with matrix-induced regularization—a conceptually simple but effective algorithm which well handles the high redundancy and low diversity of existing MKKM algorithms. We provide a theoretical justification to reveal the implication of incorporating matrix-induced regularization, and point out that our algorithm is readily extendable by designing various matrix-induced regularization. Extensive experimental research clearly demonstrates the superiority of our algorithm over the comparable ones in the recent literature. In the future, we plan to extend our algorithm to a general framework, and use it as a platform to revisit existing multiple kernel clustering algorithms and uncover their relationship.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (project No. 61403405, 61125201, and 60970034).

References

Cai, X.; Nie, F.; and Huang, H. 2013. Multi-view k-means clustering on big data. In *IJCAI*.

Cortes, C.; Mohri, M.; and Rostamizadeh, A. 2012. Algorithms for learning kernels based on centered alignment. *JMLR* 13:795–828.

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

Du, L.; Zhou, P.; Shi, L.; Wang, H.; Fan, M.; Wang, W.; and Shen, Y.-D. 2015. Robust multiple kernel k-means clustering using ℓ_{21}-norm. In *IJCAI*, 3476–3482.

Gönen, M., and Margolin, A. A. 2014. Localized data fusion for kernel k-means clustering with application to cancer biology. In *NIPS*, 1305–1313.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. J. 2006. A kernel method for the two-sample-problem. In *NIPS*, 513–520.

Hartigan, J. 1975. *Clustering Algorithms*. New York: John Wiley & Sons Inc.

Huang, H.; Chuang, Y.; and Chen, C. 2012. Multiple kernel fuzzy clustering. *IEEE T. Fuzzy Systems* 20(1):120–134.

Kumar, A., and Daumé, H. 2011. A co-training approach for multi-view spectral clustering. In *ICML*, 393–400.

Lu, Y.; Wang, L.; Lu, J.; Yang, J.; and Shen, C. 2014. Multiple kernel clustering based on centered kernel alignment. *Pattern Recognition* 47(11):3656 – 3664.

Nilsback, M.-E., and Zisserman, A. 2006. A visual vocabulary for flower classification. In *CVPR*, volume 2, 1447–1454.

Schölkopf, B.; Smola, A.; and Müller, K.-R. 1998. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.* 10(5):1299–1319.

Topsoe, F. 2000. Some inequalities for information divergence and related measures of discrimination. *IEEE Transactions on Information Theory* 46(4):1602–1609.

Xia, R.; Pan, Y.; Du, L.; and Yin, J. 2014. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*, 2149–2155.

Yu, S.; Tranchevent, L.-C.; Liu, X.; Glänzel, W.; Suykens, J. A. K.; Moor, B. D.; and Moreau, Y. 2012. Optimized data fusion for kernel k-means clustering. *IEEE TPAMI* 34(5):1031–1039.

Zhao, B.; Kwok, J. T.; and Zhang, C. 2009. Multiple kernel clustering. In *SDM*, 638–649.

Zhou, P.; Du, L.; Shi, L.; Wang, H.; and Shen, Y.-D. 2015. Recovery of corrupted multiple kernels for clustering. In *IJCAI*, 4105–4111.