Multiple Kernel Clustering Framework with Improved Kernels

Yueqing Wang, Xinwang Liu*, Yong Dou, Rongchun Li
National Laboratory for Parallel and Distributed Processing, NUDT, Changsha, China, 410073
xinwangliu@nudt.edu.cn

Abstract

Multiple kernel clustering (MKC) algorithms have been successfully applied into various applications. However, these successes are largely dependent on the quality of pre-defined base kernels, which cannot be guaranteed in practical applications. This may adversely affect the clustering performance. To address this issue, we propose a simple while effective framework to adaptively improve the quality of base kernels. Under our framework, we instantiate three MKC algorithms based on the widely used multiple kernel k-means clustering (MKKM), MKKM with matrix-induced regularization (MKKM-MR) and co-regularized multi-view spectral clustering (CRSC). After that, we design the corresponding algorithms with proved convergence to solve the resultant optimization problems. To the best of our knowledge, our framework fills the gap between kernel adaption and clustering procedure for the first time in the literature and is readily extendable. Extensive experimental research has been conducted on 7 MKC benchmarks. As is shown, our algorithms consistently and significantly improve the performance of the base MKC algorithms, indicating the effectiveness of the proposed framework. Meanwhile, our framework shows better performance than compared ones with imperfect kernels.

1 Introduction

Multiple view clustering (MVC) [Zhou and Burges, 2007; Weiran *et al.*, 2015; Jinglin *et al.*, 2016; Cao *et al.*, 2015; Zhang *et al.*, 2015; Yu *et al.*, 2016] and multiple kernel clustering (MKC) algorithms [Zhao *et al.*, 2009; Huang *et al.*, 2012; Lu *et al.*, 2014; Xia *et al.*, 2014; Zhou *et al.*, 2015; Kumar *et al.*, 2011] have been extensively studied due to their efficiency and effectiveness. Although existing MKC algorithms have demonstrated promising performance in various scenarios, we observe that the success of these algorithms is largely dependent on the quality of base kernels. In existing MKC algorithms, the base kernels are usually pre-calculated and kept unchanged during the learning procedure, which indicates that the clustering performance would be poor when the quality of pre-defined kernels is low. Unfortunately, in many practical applications, the quality of pre-specified base kernels cannot be guaranteed due to that: i) some views of a sample may be missing, leading to the incomplete base kernels, i.e. some rows or columns of the base kernels shall be absent; ii) the extracted features may not be able to produce good clustering performance even when none view of any samples is missing; and iii) inappropriate kernel types in generating base kernels, for example, some applications get better performance using linear kernels, while other applications obtain better performance using Gaussian kernels. All these factors may adversely affect the quality of base kernels, leading to unsatisfying clustering performance.

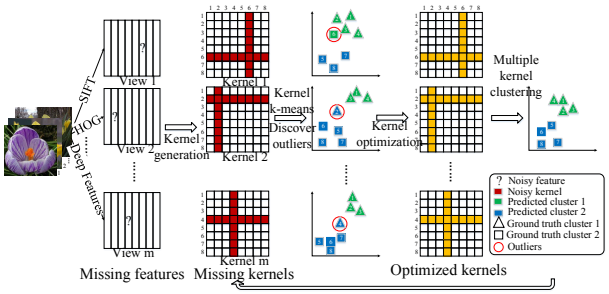

Figure 1: Proposed MKC framework with improved kernels (MKCF-IK). Given m pre-defined noisy kernels of a data set, we: i) find out outliers in each kernel, ii) design an alternative algorithm to recover those inappropriate kernel values, and iii) use MKC algorithms on refined kernels.

To reduce the influence of base kernels, we propose a simple while effective framework to adaptively improve the quality of the base kernels during the learning process of multiple kernel clustering, as illustrated in Figure 1. Our framework firstly discovers outliers in each kernel based on the previous clustering results, and then designs an alternative algorithm to optimize them iteratively. In this way, our framework enhances the negotiation between the resultant base kernels and the clustering, leading to the improvement on clustering performance. To implement this framework, two issues are naturally raised: i) how to find outliers reducing the performance; and ii) how to optimize base kernels. To address the first issue, we exploit the connections between base kernels and clustering results, and find that the samples with further

*The corresponding author.

distance to their clustering center are more unascertainable. This is rather intuitive since these unascertainable samples could be considered as outliers. To address the second issue, we propose an alternative algorithm to optimize base kernels and recover those inappropriate values in each kernel. Under our framework, we instantiate three MKC algorithms based on the widely used multiple kernel k-means clustering (MKKM) [Huang *et al.*, 2012], MKKM with matrix-induced regularization (MKKM-MR) [Liu *et al.*, 2016] and co-regularized multi-view spectral clustering (CRSC) [Kumar *et al.*, 2011], and design three efficient algorithms with proved convergence to solve the resultant optimization problems, respectively. To the best of our knowledge, our framework bridges the gap between kernel adaption and clustering procedure for the first time in the literature. More importantly, our framework is flexible and readily extendable for generating better MKC algorithms. Extensive experimental results show that our framework improves the performance of existing MKC algorithms on most datasets. Meanwhile, our framework shows better performance than compared ones with imperfect kernels.

2 Related Work

Existing MKC algorithms can roughly be grouped into two categories. The first category optimizes a group of kernel coefficients, and uses the combined kernel for clustering, called pre-fusion category, including MKKM [Huang *et al.*, 2012] and its variants such as MKKM-MR [Liu *et al.*, 2016], RMKC [Zhou *et al.*, 2015], and localized multiple kernel k-means (LMKKM) [Gönen and Margolin, 2014]. By following multiple kernel learning (MKL) framework, the other category learns the indicating matrix $\mathbf{H}$ without explicitly obtaining the combined kernel, like CRSC [Kumar *et al.*, 2011]. In this section, we give a brief introduction of three typical representatives belonging to these categories.

2.1 Multiple Kernel k-means Clustering (MKKM)

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 $\mathcal{H}$. In the 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 $\phi_{\boldsymbol{\gamma}}(\mathbf{x}) = [\gamma_1\phi_1(\mathbf{x})^\top, \gamma_2\phi_2(\mathbf{x})^\top, \cdots, \gamma_m\phi_m(\mathbf{x})^\top]^\top$, where $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \cdots, \gamma_m]^\top$ denotes the coefficients of each base kernel.

Given the combined kernel matrix $\mathbf{K}_{\boldsymbol{\gamma}}$, the optimization problem for MKKM can be written as,

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

This problem can be solved by alternatively updating $\mathbf{H}$ and $\boldsymbol{\gamma}$.

2.2 MKKM Clustering with Matrix-induced Regularization (MKKM-MR)

By integrating the matrix-induced regularization into the objective function of existing MKKM, the optimization problem of MKKM-MR can be obtained as follows,

$$\begin{aligned} \min_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\gamma}} \ & \mathrm{Tr}(\mathbf{K}_{\boldsymbol{\gamma}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)) + \frac{\lambda}{2}\boldsymbol{\gamma}^\top\mathbf{M}\boldsymbol{\gamma} \\ s.t. \ & \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \ \boldsymbol{\gamma}^\top\mathbf{1}_m = 1, \boldsymbol{\gamma} \succeq \mathbf{0}. \end{aligned} \tag{2}$$

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

2.3 Co-regularized Spectral Clustering (CRSC)

Co-regularized spectral clustering (CRSC) provides a co-regularization way to perform spectral clustering. Instead of explicitly optimizing the combined kernel, it learns the indicating matrix $\mathbf{H}$ directly. The objective function can be written as:

$$\begin{aligned} \max_{\mathbf{H}_p,\,\mathbf{H}^\star} & \sum_{p=1}^{m}\Big(\mathrm{Tr}(\mathbf{H}_p^\top\mathbf{K}_p\mathbf{H}_p) + \lambda_p\mathrm{Tr}(\mathbf{H}_p\mathbf{H}_p^\top\mathbf{H}^\star\mathbf{H}^{\star\top})\Big) \\ s.t. \ & \mathbf{H}_p^\top\mathbf{H}_p = \mathbf{I}_k, \forall 1 \le p \le m, \mathbf{H}^{\star\top}\mathbf{H}^\star = \mathbf{I}_k, \end{aligned} \tag{3}$$

where λ_p reflects the importance of $\mathbf{K}_p$. The optimal $\mathbf{H}^\star$ and $\mathbf{H}_p$ can be obtained in an alternative way. In fact, the optimal problem can be rewritten as the spectral clustering objective function when $\mathbf{H}^\star$ or $\mathbf{H}_p$ fixed.

Existing MKC algorithms have been applied to many clustering tasks successfully. However, the performance of these algorithms are largely dependent on the quality of pre-defined base kernels. Moreover, these base kernels are kept unchanged during the learning process. By this way, the performance would be poor if the base kernels have low quality. To eliminate the limitation, we design a novel clustering framework to adaptively improve the quality of the base kernels during the learning process.

3 Multiple Kernel Clustering Framework with Improved Kernels (MKCF-IK)

Although achieving promising performance, the aforementioned existing MKC algorithms largely depend on the quality of base kernels. Meanwhile, the pre-calculated kernels would not be changed during the learning procedure even if they are imperfect. To adaptively improve the quality of base kernels, we introduce our multiple kernel clustering framework with improved kernels (MKCF-IK), as shown in Algorithm 1. The proposed framework firstly adopts a simple while effective strategy to discover inappropriate values among each kernel which may be inappropriate for clustering, as described in Algorithm 2. After that, we carefully design corresponding alternative algorithms to recover these values automatically according to the previous clustering results.

In Algorithm 1, $\mathbf{O}_p^{(t)}$ means the indices of outliers, while $\mathbf{S}_p^{(t)}$ indicates the indices of stable samples, and $\rho^{(0,t)}$ represents the average kernel alignment value, which indicates the similarity between original and improved kernels. It is intuitive that the original base kernels would keep some good properties of data, so we set a threshold for $\rho^{(0,t)}$ to prevent the original kernels changing too much.

$$\rho^{(t-1,t)} = \frac{1}{m}\sum_{p=1}^{m}\frac{\mathrm{Tr}(\mathbf{K}_p^{(t-1)}\mathbf{K}_p^{(t)\top})}{\sqrt{\mathrm{Tr}(\mathbf{K}_p^{(t-1)}\mathbf{K}_p^{(t-1)\top})\mathrm{Tr}(\mathbf{K}_p^{(t)}\mathbf{K}_p^{(t)\top})}}. \tag{4}$$

Algorithm 1: Proposed MKCF-IK.

Input: $\{\mathbf{K}_p\}_{p=1}^m$, ϵ_0, η_0
Output: $\mathbf{H}^\star$, $\{\mathbf{K}_p\}_{p=1}^m$

$\mathbf{K}_p^{(0)} = \mathbf{K}_p$, $p = 1, \cdots, m$, t=1.
repeat
  Obtain indices of stable samples $\mathbf{S}_p^{(t)}$ and outliers $\mathbf{O}_p^{(t)}$ using Algorithm 2 with $\mathbf{K}_p^{(t-1)}$.
  Optimize $\mathbf{K}_p^{(t)}$ and $\mathbf{H}^\star$ using improved MKC algorithms with $\mathbf{S}_p^{(t)}$ and $\mathbf{K}_p^{(t-1)}$.
  Calculate alignment value $\rho^{(0,t)}$ using Eq.(4).
  t = t + 1.
until $\left(\rho^{(0,t)} \leq \eta_0\right) \,||\, \left((obj^{(t-1)} - obj^{(t)})/obj^{(t)} \leq \epsilon_0\right)$;

Algorithm 2: Discovering Outliers.

Input: $\{\mathbf{K}_p\}_{p=1}^m$, r
Output: $\{\mathbf{S}_p, \mathbf{O}_p\}_{p=1}^m$

for *each* $p \leq m$ **do**
  Compute labels and centers using KKM with $\mathbf{K}_p$.
  for *each clustering center* j **do**
    Collect N_j samples belongs to this center.
    Add the $N_j * r$ samples with further distance to the j-th center to $\mathbf{O}_p$.
    Add other $N_j * (1 - r)$ samples to $\mathbf{S}_p$.

3.1 Discovering Outliers

By exploiting the connections between base kernels and clustering results, we observe that samples with further distances to their clustering centers are more unascertainable. It is rather intuitive that these unascertainable samples are prone to become outliers. Moreover, different base kernels describe different relationships among samples, which implies that we can obtain different outliers based on different kernels. By this way, we are able to identify the locations of outliers for each kernel. In specific, we first perform kernel k-means (KKM) on each base kernel $\mathbf{K}_p$, which produces initial clustering labels of samples. After that, we compute the distance between each sample and its clustering center. For each center, we collect a proportion of samples with top largest distances to it, and add their indices to $\mathbf{O}_p$. We term this proportion as r. In our experiments, this r is set to be 0.05 or 0.1. In addition, we also record the relatively stable samples as $\mathbf{S}_p$.

3.2 MKC Algorithms with Enhanced Kernels

After obtaining the locations of outliers in each base kernel, we design different alternative algorithms based on different MKC algorithms to optimize the inappropriate values among these base kernels, making them better serve for clustering. Under our framework, we instantiate three classical MKC algorithms, and design three efficient algorithms with proved convergence to solve the resultant optimization problems, respectively. Here we only introduce the implementation of MKCF-IK on CRSC due to the space limitation.

3.3 An Instantiation: CRSC-IK

Let $\mathbf{S}_p (1 \leq p \leq m)$ denotes the sample indices for which the p-th view is present and $\mathbf{K}_p^{(cc)}$ be used to indicate the sub-kernel matrix computed with these samples. Then the objective function of CRSC-IK can be written as Eq.(5).

$$\begin{aligned} \max_{\mathbf{H}_p, \mathbf{H}^\star, \mathbf{K}_p} & \sum_{p=1}^m \left(\mathrm{Tr}(\mathbf{H}_p^\top \mathbf{K}_p \mathbf{H}_p) + \lambda_p \mathrm{Tr}(\mathbf{H}_p \mathbf{H}_p^\top \mathbf{H}^\star \mathbf{H}^{\star\top}) \right) \\ s.t.\ & \mathbf{H}_p^\top \mathbf{H}_p = \mathbf{I}_k, \forall 1 \leq p \leq m, \mathbf{H}^{\star\top} \mathbf{H}^\star = \mathbf{I}_k \\ & \mathbf{K}_p(\mathbf{S}_p, \mathbf{S}_p) = \mathbf{K}_p^{(cc)}, \mathbf{K}_p \succeq 0, \forall p, \end{aligned} \tag{5}$$

As can be seen, the difference between the objective function of CRSC-IK and that of CRSC in Eq.(3) is the incorporation of optimizing $\{\mathbf{K}_p\}_{p=1}^m$. Note that the constraint $\mathbf{K}_p(\mathbf{S}_p, \mathbf{S}_p) = \mathbf{K}_p^{(cc)}$ is imposed to ensure that the improved $\mathbf{K}_p$ maintains the relatively stable kernel values during the course.

CRSC-IK considers base kernels as the optimized target. It aims to enhance the base kernels for clustering by treating $\{\mathbf{K}_p\}_{p=1}^m$ as extra variables into our algorithm. To solve the problem described in Eq.(5), we design a three-step algorithm in an alternative way, as shown in Algorithm 3.

1) **Optimizing $\mathbf{H}^\star$ with $\{\mathbf{H}_p\}_{p=1}^m$ and $\{\mathbf{K}_p\}_{p=1}^m$ fixed**. Given $\mathbf{H}_p$, the $\mathbf{H}^\star$ can be obtained by solving the following objective function:

$$\begin{aligned} \max_{\mathbf{H}^\star \in \mathbb{R}^{n \times k}} & \mathrm{Tr}\left(\mathbf{H}^{\star\top} \sum_{p=1}^m \left(\lambda_p \mathbf{H}_p \mathbf{H}_p^\top \right) \mathbf{H}^\star \right), \\ & s.t.\ \mathbf{H}^{\star\top} \mathbf{H}^\star = \mathbf{I}_k, \end{aligned} \tag{6}$$

which is equivalent to solving the standard spectral clustering objective for $\mathbf{H}^\star$ with a modified Laplacian $\sum_p \lambda_p \mathbf{H}_p \mathbf{H}_p^\top$;

2) **Alternatively optimize $\mathbf{H}_p$ for $p = 1, 2, \cdots, m$ with fixed $\mathbf{H}^\star$, $\{\mathbf{K}_p\}_{p=1}^m$ and $\{\mathbf{H}_j\}_{j \neq p}$**. Given $\mathbf{H}^\star$ and all other view-specific eigenvectors, $\mathbf{H}_p$ for view p can be solved by Eq.(7).

$$\begin{aligned} \max_{\mathbf{H}_p \in \mathbb{R}^{n \times k}} & \mathrm{Tr}(\mathbf{H}_p^\top \mathbf{K}_p \mathbf{H}_p) + \lambda_p \mathrm{Tr}(\mathbf{H}_p \mathbf{H}_p^\top \mathbf{H}^\star \mathbf{H}^{\star\top}) \\ & s.t.\ \mathbf{H}_p^\top \mathbf{H}_p = \mathbf{I}_k \end{aligned} \tag{7}$$

By using the properties of matrix traces, Eq.(7) can be written as following:

$$\begin{aligned} \max_{\mathbf{H}_p \in \mathbb{R}^{n \times k}} & \mathrm{Tr}\left(\mathbf{H}_p^\top \left(\mathbf{K}_p + \lambda_p \mathbf{H}^\star \mathbf{H}^{\star\top} \right) \mathbf{H}_p \right) \\ & s.t.\ \mathbf{H}_p^\top \mathbf{H}_p = \mathbf{I}_k. \end{aligned} \tag{8}$$

which is equivalent to solving the standard spectral clustering objective for $\mathbf{H}_p$ with a modified Laplacian $\left(\mathbf{K}_p + \lambda_p \mathbf{H}^\star \mathbf{H}^{\star\top} \right)$.

3) **Optimizing $\{\mathbf{K}_p\}_{p=1}^m$ with fixed $\{\mathbf{H}_p\}_{p=1}^m$**. Given $\mathbf{H}_p$, $p \in \{1, 2, \cdots, m\}$, the optimization problem in $\mathbf{K}_p$ can be written as:

$$\begin{aligned} & \max_{\mathbf{K}_p \in \mathbb{R}^{n \times n}} \mathrm{Tr}(\mathbf{H}_p^\top \mathbf{K}_p \mathbf{H}_p), \\ & \quad s.t.\ \ \mathbf{K}_p(\mathbf{S}_p, \mathbf{S}_p) = \mathbf{K}_p^{(cc)}, \mathbf{K}_p \succeq 0, \forall p, \end{aligned} \tag{9}$$

where the constraint $\mathbf{K}_p(\mathbf{S}_p, \mathbf{S}_p) = \mathbf{K}_p^{(cc)}$ guarantees that the improved $\mathbf{K}_p$ maintains the normal values during the optimized process.

Algorithm 3: Proposed CSRC-IK.

Input: $\mathbf{K}_p^{(cc)}, \mathbf{S}_p, \forall p \in \{1, \cdots, m\}$, and ϵ_0
Output: $\mathbf{H}^\star$,$\mathbf{K}_p, \forall p \in \{1, \cdots, m\}$

For all p, initialize $\mathbf{K}_p^{(0)}$ and $t = 1$.
For all p, initialize $\mathbf{H}_p^{(0)}$ by solving Eq.(3) with given $\mathbf{K}_p^{(0)}$.
repeat
  Update $\mathbf{H}^{\star(t)}$ by solving Eq.(6) with given $\mathbf{H}_p^{(t-1)}$.
  For all p, update $\mathbf{H}_p^{(t)}$ by solving Eq.(8) with fixed $\mathbf{K}_p^{(t-1)}$, and $\mathbf{H}^{\star(t)}$.
  Update $\mathbf{K}_p^{(t)}$ by Eq.(13) with fixed $\mathbf{H}_p^{(t)}$.
  t = t + 1.
until $(obj^{(t-1)} - obj^{(t)})/obj^{(t)} \leq \epsilon_0$;

As is well known, Eq.(9) is equivalent to the Eq.(10) as follows:

$$\begin{aligned} &\min_{\mathbf{K}_p \in \mathbb{R}^{n \times n}} \mathrm{Tr}(\mathbf{K}_p(\mathbf{I} - \mathbf{H}_p\mathbf{H}_p^\top)), \\ &\qquad s.t.\, \mathbf{K}_p(\mathbf{S}_p, \mathbf{S}_p) = \mathbf{K}_p^{(cc)}, \mathbf{K}_p \succeq 0, \forall p, \end{aligned} \tag{10}$$

Note that $\mathbf{K}_p$ is positive semi-definite (PSD), so we can decompose $\mathbf{K}_p$ as $\mathbf{A}_p\mathbf{A}_p^\top$. Then we write $\mathbf{A}_p = [\mathbf{A}_p^{(c)}\mathbf{A}_p^{(m)}]$ with $\mathbf{A}_p^{(c)}{\mathbf{A}_p^{(c)}}^\top = \mathbf{K}_p^{(cc)}$, and assume that $\mathbf{Q}_p = \mathbf{I} - \mathbf{H}_p\mathbf{H}_p^\top$, then rewrite Eq.(10) as:

$$\begin{aligned} &\min_{\mathbf{A}_p^{(m)}} \mathrm{Tr}([\mathbf{A}_p^{(c)}\mathbf{A}_p^{(m)}] \begin{bmatrix} \mathbf{Q}_p^{(cc)} & \mathbf{Q}_p^{(cm)} \\ {\mathbf{Q}_p^{(cm)}}^\top & \mathbf{Q}_p^{(mm)} \end{bmatrix} [\mathbf{A}_p^{(c)}\mathbf{A}_p^{(m)}]^\top) \\ &s.t.\, \mathbf{K}_p(\mathbf{S}_p, \mathbf{S}_p) = \mathbf{K}_p^{(cc)}, \mathbf{K}_p \succeq 0, \forall p, \end{aligned} \tag{11}$$

where $\mathbf{Q}_p$ is replaced with its blocked form.

To optimize Eq.(11), we compute the derivative of it with respect to $\mathbf{A}_p^{(m)}$ and let it equals to zero. Then we can obtain an analytical solution to the optimal $\mathbf{A}_p^{(m)}$ as:

$$\mathbf{A}_p^{(m)} = \left(\mathbf{Q}_p^{(mm)}\right)^{-1} {\mathbf{Q}_p^{(cm)}}^\top \mathbf{A}_p^{(c)} \tag{12}$$

Then the optimal $\mathbf{K}_p$ in Eq.(9) can be obtained as:

$$\mathbf{K}_p = \begin{bmatrix} \mathbf{K}_p^{(cc)} & -\mathbf{K}_p^{(cc)}\mathbf{Q}_p^{(cm)}\mathbf{Q}_p^{(mm)} \\ -\mathbf{Q}_p^{(mm)}{\mathbf{Q}_p^{(cm)}}^\top\mathbf{K}_p^{(cc)} & \mathbf{Q}_p^{(mm)}{\mathbf{Q}_p^{(cm)}}^\top\mathbf{K}_p^{(cc)}\mathbf{Q}_p^{(cm)}\mathbf{Q}_p^{(mm)} \end{bmatrix} \tag{13}$$

Note that, compared with CRSC-IK, the implementations on MKKM and MKKM-MR optimize base kernels in a slightly different way. They use $\mathbf{Q}_p$ to replace $(\mathbf{I} - \mathbf{H}\mathbf{H}^\top)$ other than $(\mathbf{I} - \mathbf{H}_p\mathbf{H}_p^\top)$.

3.4 Discussion

In this section, we discuss why our framework would work. By revisiting Eq.(13), we observe that the enhanced kernels refine the inappropriate parts of each base kernel using fixed $\mathbf{K}_p^{(cc)}$ and $\mathbf{Q}$. In MKKM-IK, $\mathbf{Q}$ denotes $(\mathbf{I} - \mathbf{H}\mathbf{H}^\top)$, where $\mathbf{H}$ is calculated by combined kernel $\mathbf{K}$. Therefore, $\mathbf{Q}$ integrates the clustering information from all base kernels to optimize the p-th kernel. CRSC-IK does not use the global $\mathbf{H}$ to get information from other kernels because only $\mathbf{H}_p$ is used to optimize $\mathbf{K}_p$. However, we observe that the optimized procedure of $\mathbf{H}_p$ in CRSC-IK would introduce the $\mathbf{H}$. This indicates that the $\mathbf{H}_p$ already carries the global information from $\mathbf{H}$. As a result, CRSC-IK also exploits the information from all base kernels to improve the quality of one base kernels. In conclusion, our framework integrates the information from all kernels to fill the inappropriate parts approximately.

Table 1: Datasets used in our experiments.

Dataset	#Samples	#Kernels	#Classes
bbcsport	737	2	5
YALE	165	5	15
proteinFold	694	12	27
Caltech102	1530	25	102
Flower17	1360	7	17
Digital	2000	3	10
CCV	6773	6	20

4 Experimental Results

4.1 Datasets and Experimental Settings

To report the performance of our framework, we evaluate three instances of our framework, like MKKM-IK, CRSC-IK and MKKM-MR-IK, on seven datasets, as shown in Tabel 1. Furthermore, to test the performance of all algorithms with respect to the number of classes, we generate ten datasets by randomly selecting samples the first 10, 20, $\cdots$, 100 classes on Caltech102.

To evaluate the performance of our framework with incomplete base kernels, we randomly generate incomplete kernels with different missing ratios. Note that the missing ratio in our experiments means the percentage of samples with missing views, other than the percentage of missing rows (columns) in each kernel. Meanwhile, we randomly generate the incomplete patterns for 30 times and report the statistical results. The aggregated ACC, NMI and purity are used to evaluate the goodness of the algorithms in comparison. Following the literature [Cortes *et al.*, 2012], all base kernels are centered and scaled so that we have $\kappa_p(\mathbf{x}_i, \mathbf{x}_i) = 1$ for all i and p.

4.2 Compared Algorithms

Our algorithms are compared with several recently proposed counterparts, including

- **Average MKKM (A-MKKM):** All kernels are uniformly weighted to generate a new kernel, which is taken as the input of kernel k-means.
- **MKKM** [Huang *et al.*, 2012]: The algorithm alternatively performs kernel k-means and updates the kernel coefficients, as introduced in the related work.
- **Localized MKKM (LMKKM)** [Gönen and Margolin, 2014]: LMMKM improves MKKM by combining the kernels in a localized way.
- **Robust multi-view spectral clustering (RMSC)** [Xia *et al.*, 2014]: RMSC constructs a transition probability matrix from each single view, and uses them to recover

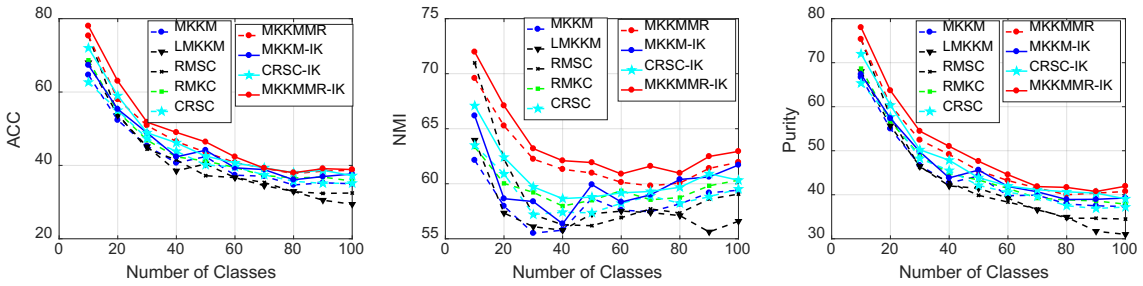

Figure 2: Clustering accuracy, NMI and purity comparison with variation of number of classes on Caltech102. (left) ACC, (middle) NMI, and (right) purity.

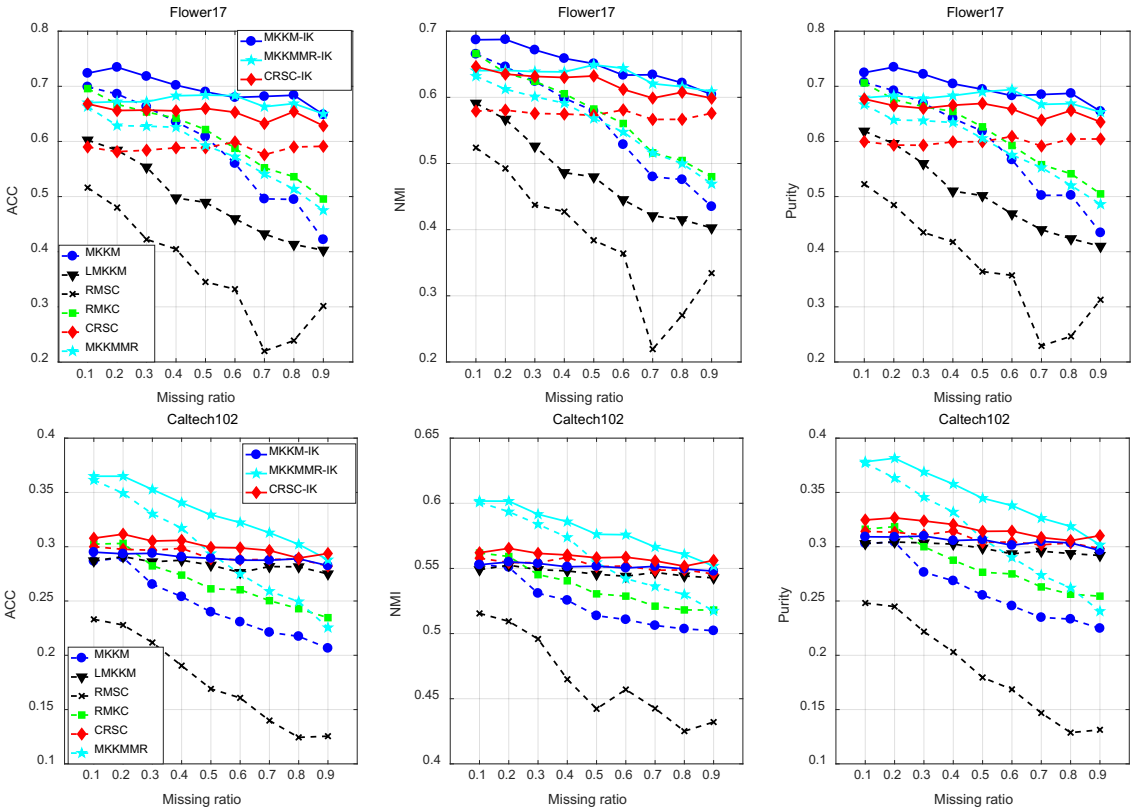

Figure 3: Clustering performance with the variation of missing ratios on Flower17 and Caltech102 data sets.

a shared low-rank transition probability matrix for clustering.

- **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.
- **CRSC:** [Kumar *et al.*, 2011]: It provides a co-regularization way to perform spectral clustering.
- **MKKM-MR** [Liu *et al.*, 2016]: MKKM-MR reduces the redundancy and enhances the diversity of base kernels by introducing a matrix-induced regularization.

4.3 Experimental Results

The clustering performance of the above mentioned algorithms on all datasets are reported in Table 2. As is shown in Table 2, three instances of our framework achieve promising clustering performance.

Taking the result on bbcsport for example, the clustering accuracy of MKKM, CRSC, and MKKM-MR are 67.91%, 80.51%, and 66.91%. In contrast, our improved algorithms, MKKM-IK, CRSC-IK, and MKKM-MR-IK, achieve 88.6%, 91.73% and 74.82%, respectively. In order to demonstrate performance more vividly, we use t-SNE [Laurens, 2013] to visualize the indicating matrices $\mathbf{H}$ of bbcsport in Figure 4. It can be observed that our algorithms would make five classes much further and more discriminative than other MKC algorithms.

We also investigate the clustering performance of each algorithm with respect to the number of classes, as shown in Figure 2. As observed, the curves of CRSC-IK and MKKM-MR-IK are above that of CRSC and MKKM-MR when the number of classes varies. Moreover, MKKM-MR-IK keep-

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

datasets	A-MKKM	MKKM	LMKKM	RMSC	RMKC	CRSC	MKKM-MR	MKKM-IK	CRSC-IK	MKKM-MR-IK
ACC										
bbcsport	66.91	67.28	66.91	86.03	66.91	80.51	66.91	88.60	**91.73**	74.82
YALE	60.00	57.58	58.18	63.03	61.21	56.36	**64.85**	61.21	57.45	**64.85**
proteinFold	32.85	25.22	24.06	36.17	33.72	35.30	39.63	26.51	36.76	**40.06**
Digital	89.00	47.00	48.50	90.60	89.05	81.25	90.15	59.41	86.14	**90.90**
Flower17	67.94	70.15	63.38	59.71	68.97	63.16	70.66	**72.50**	69.34	70.15
Caltech102	36.21	34.17	27.97	31.50	35.56	35.82	37.91	34.91	38.24	**40.33**
CCV	24.11	21.18	20.27	17.38	24.63	30.77	27.77	22.18	**31.68**	27.48
NMI										
bbcsport	55.82	55.02	55.54	75.09	55.29	63.41	55.82	70.6	**78.71**	60.99
YALE	62.85	58.35	58.92	63.98	62.85	57.25	64.73	59.67	57.38	64.31
proteinFold	43.52	37.65	36.21	46.39	42.86	43.40	**47.34**	36.05	45.73	46.37
Digital	81.18	48.16	50.06	82.21	81.15	76.09	83.81	49.95	78.14	**84.83**
Flower17	65.30	67.65	62.22	59.45	65.34	62.84	66.60	**68.75**	65.54	66.17
Caltech102	60.60	59.54	55.17	58.40	59.02	60.04	61.85	59.12	61.37	**62.88**
CCV	20.30	17.37	16.05	15.34	21.02	23.31	22.17	17.91	**25.34**	22.05
Purity										
bbcsport	54.99	77.76	54.84	73.89	77.94	80.51	77.76	88.6	**91.73**	81.43
YALE	60.61	58.18	58.18	64.24	61.21	57.58	64.31	61.21	58.06	**64.85**
proteinFold	41.07	31.99	33.14	43.37	40.92	41.50	45.39	33.29	42.71	**45.68**
Digital	89.00	49.70	51.10	90.60	89.05	81.25	90.15	59.41	86.14	**90.90**
Flower17	67.94	70.81	64.34	61.03	68.97	65.22	70.66	**72.50**	69.56	70.15
Caltech102	38.04	36.60	29.41	33.66	37.12	37.84	39.74	36.86	40.33	**42.22**
CCV	20.17	25.74	15.92	15.27	29.26	31.98	21.94	26.38	**33.65**	29.77

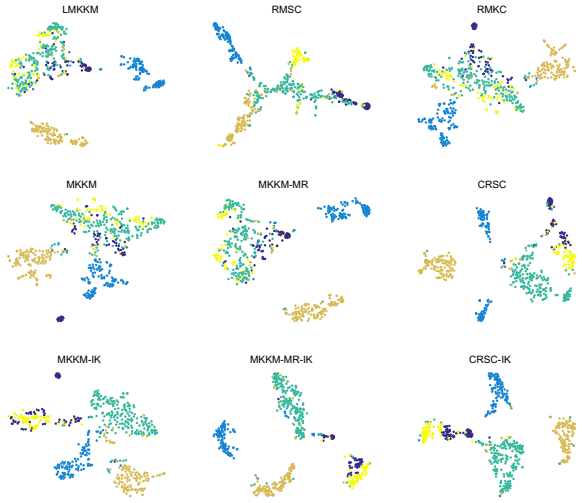

Figure 4: The effect of our framework on the clustering accuracy. The figure shows t-SNE visualization of the indicating matrices $\mathbf{H}$ using different algorithms on bbcsport.

s on the top of all sub-figures when the number of classes varies, indicating the best performance.

Figure 3 presents the clustering performance comparison of the above algorithms with absent kernels on Flower17 and Caltech102. It can be observed that: i) MKKM-IK, CRSC-IK and MKKM-MR-IK demonstrate the overall best performance in all the sub-figures; ii) the variation of MKKM-IK, CRSC-IK and MKKM-MR-IK with respect to the missing ratio is relatively smaller when compared with other algorithms, demonstrating its stability in the case of intensive absence.

Convergence. Two examples of the evolution of the objective value of MKKM-IK on Digital and YALE are demonstrated in Figure 5. As we can see, the objective value of MKKM-IK does monotonically decrease at each iteration.

From the above experiments, we conclude that the proposed framework: i) improves the performance of base MKC algorithms when the base kernels are complete; ii) achieves better clustering performance than other MKC algorithms; ii) can better recover the incomplete base kernels by taking account the goal of clustering. In short, our framework well bridges the gap between kernel adaption and clustering procedure for the first time in the literature and is readily extendable, bringing forth significant improvements on clustering performance.

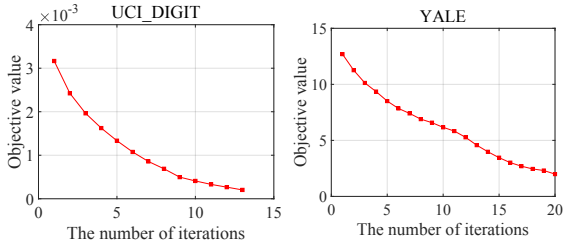

Figure 5: The objective value of our algorithms at each iteration. (left) **Digital** and (right) **YALE**

5 Conclusion

In this paper, we propose a novel framework to adaptively improve the quality of base kernels. Our framework enhances the negotiation between the base kernels and the clustering performance, leading to the improvement on clustering performance. Under our framework, we instantiate three MKC algorithms based on MKKM, MKKM-MR, and CRSC. After that, we design three alternative algorithms to solve the resultant optimization problems. To the best of our knowledge, our framework fills the gap between kernel adaption and clustering procedure for the first time in the literature. Extensive experimental research has been conducted on synthetic dataset, MKC benchmarks and some computer vision datasets. As shown, our algorithms consistently and significantly improve the performance of the base MKC algorithms, indicating the effectiveness of the proposed framework.

Acknowledgements

This work was supported in part by the Natural Science Foundation of China (61403405, U1435219, 61402499).

References

[Cao *et al.*, 2015] Xiaochun Cao, Changqing Zhang, Huazhu Fu, and Si Liu. Diversity-induced multi-view subspace clustering. In *CVPR*, pages 586–594, 2015.

[Cortes *et al.*, 2012] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1):795–828, 2012.

[Gönen and Margolin, 2014] Mehmet Gönen and Adam A Margolin. Localized data fusion for kernel k-means clustering with application to cancer biology. *NIPS*, 2:1305–1313, 2014.

[Huang *et al.*, 2012] Hsin Chien Huang, Yung Yu Chuang, and Chu Song Chen. Multiple kernel fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, 20(1):120–134, 2012.

[Jinglin *et al.*, 2016] Xu Jinglin, Han Junwei, and Nie Feiping. Discriminatively embedded k-means for multi-view clustering. In *CVPR*, pages 5356–5364, 2016.

[Kumar *et al.*, 2011] Abhishek Kumar, Piyush Rai, and Hal Daume. Co-regularized multi-view spectral clustering. In *NIPS*, pages 1413–1421, 2011.

[Laurens, 2013] Van Der Maaten Laurens. Barnes-hut-sne. *Computer Science*, 1301.3342, 2013.

[Liu *et al.*, 2016] Xinwang Liu, Yong Dou, Jianping Yin, Lei Wang, and En Zhu. Multiple kernel k-means clustering with matrix-induced regularization. In *AAAI*, 2016.

[Lu *et al.*, 2014] Yanting Lu, Liantao Wang, Jianfeng Lu, Jingyu Yang, and Chunhua Shen. Multiple kernel clustering based on centered kernel alignment. *Pattern Recognition*, 47(11):3656–3664, 2014.

[Weiran *et al.*, 2015] Wang Weiran, Arora Raman, Livescu Karen, and Bilmes Jeff. On deep multi-view representation learning. In *ICML*, pages 1083–1092, 2015.

[Xia *et al.*, 2014] Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*, pages 2149–2155, 2014.

[Yu *et al.*, 2016] Yue Yu, Huaimin Wang, Gang Yin, and Tao Wang. Reviewer recommendation for pull-requests in github: What can we learn from code review and bug assignment? *Information and Software Technology*, 74:204–218, 2016.

[Zhang *et al.*, 2015] Changqing Zhang, Huazhu Fu, Si Liu, Guangcan Liu, and Xiaochun Cao. Low-rank tensor constrained multiview subspace clustering. In *ICCV*, 2015.

[Zhao *et al.*, 2009] Bin Zhao, James T. Kwok, and Changshui Zhang. Multiple kernel clustering. In *ICDM*, 2009.

[Zhou and Burges, 2007] Dengyong Zhou and Christopher J. C. Burges. Spectral clustering and transductive learning with multiple views. In *ICML*, pages 1159–1166, 2007.

[Zhou *et al.*, 2015] Peng Zhou, Liang Du, Lei Shi, Hanmo Wang, and Yi-Dong Shen. Recovery of corrupted multiple kernels for clustering. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 4105–4111, 2015.