Discrete Multiple Kernel k-means

Rong Wang[1] , **Jitao Lu**[2] , **Yihang Lu**[2] , **Feiping Nie**[2*] and **Xuelong Li**[2]
[1]School of Cybersecurity and School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University, Xi'an 710072, Shaanxi, P. R. China
[2]School of Computer Science and School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University, Xi'an 710072, Shaanxi, P. R. China
wangrong07@tsinghua.org.cn, dianlujitao@gmail.com, sebastian.yihanglu@gmail.com, feipingnie@gmail.com, li@nwpu.edu.cn

Abstract

The multiple kernel k-means (MKKM) and its variants utilize complementary information from different kernels, achieving better performance than kernel k-means (KKM). However, the optimization procedures of previous works all comprise two stages, learning the continuous relaxed label matrix and obtaining the discrete one by extra discretization procedures. Such a two-stage strategy gives rise to a mismatched problem and severe information loss. To address this problem, we elaborate a novel Discrete Multiple Kernel k-means (DMKKM) model solved by an optimization algorithm that directly obtains the cluster indicator matrix without subsequent discretization procedures. Moreover, DMKKM can strictly measure the correlations among kernels, which is capable of enhancing kernel fusion by reducing redundancy and improving diversity. What's more, DMKKM is parameter-free avoiding intractable hyperparameter tuning, which makes it feasible in practical applications. Extensive experiments illustrated the effectiveness and superiority of the proposed model.

1 Introduction

Clustering is one of the most fundamental topics in machine learning and data mining. Out of various clustering algorithms, k-means [Hartigan and Wong, 1979] enjoys a huge popularity because of efficiency and simpleness, but it fails to cope with non-globular clusters which are very common in practice. Thus, researchers put forward a series of models to solve this problem, *e.g.*, kernel k-means clustering (KKM) [Schölkopf *et al.*, 1998] using a kernel function to embed the original data into a high-dimensional Reproducing Kernel Hilbert Space (RKHS) where standard k-means clustering is performed with linearly separable mapped data [Van Laarhoven and Marchiori, 2016; He and Zhang, 2018; Calandriello and Rosasco, 2018; Wang *et al.*, 2019a; Vankadara and Ghoshdastidar, 2020]. Although KKM intends to improve the clustering performance by introducing kernel functions, it is unable to identify whether a specific kernel function is suitable for a particular task in advance. To alleviate this problem, it's a good idea to allow the algorithm to adaptively choose the appropriate kernels, exploiting complementary information from different kernels to enhance learning, which is known as multiple kernel learning [Zhao *et al.*, 2009; Xu *et al.*, 2017; Kang *et al.*, 2018].

Huang *et al*. proposed multiple kernel k-means clustering (MKKM) applying multiple kernel learning settings to kernel k-means clustering [Huang *et al.*, 2011], which unifies the kernel fusion process and clustering into a single optimization framework. A concurrent work OKKC optimizes the kernel coefficients and cluster membership based on the same Rayleigh quotient objective and claims to have less complexity [Yu *et al.*, 2011]. In the past decades, many studies have been devoted to improving MKKM. Gönen *et al*. proposed localized multiple kernel k-means (LMKKM) to adaptively change the kernel coefficients with a localized data fusion approach acquiring sample-specific characteristics of the data [Gönen and Margolin, 2014]. Besides, Du *et al*. replaced the squared error term of k-means with $\ell_{2,1}$-norm based one and proposed a robust multiple kernel k-means clustering (RMKKM) algorithm to improve the robustness with respect to noises and outliers [Du *et al.*, 2015]. Liu *et al*. argued that previous works haven't significantly considered the correlation among different kernels and proposed a MKKM-MR model which conducts a matrix-induced regularization to reduce the redundancy and enhance the diversity of selected kernels [Liu *et al.*, 2016].

Although previous works made some progress in MKKM clustering performance, they suffer from various problems. Dealing with the NP-hard cluster assignment problem, they all utilize the two-stage process: learning the continuous relaxation matrix and obtaining the discrete clustering indicator matrix by extra discretization process, which result in a mismatched problem and severe information loss. Moreover, existing MKKM models overlook the correlation among different kernels, leading to fusion of mutually redundant kernels and bad effect on the diversity of information sources. What's worse, most existing models achieve desirable clustering performance by tuning hyperparameters from regularization terms, which is intractable in practice owing to tedious setting and hard searching of hyperparameters.

In this paper, we elaborate a novel Discrete Multiple Kernel

*Corresponding Author

k-means (DMKKM) clustering model, which aims at overcoming the limitations and weaknesses caused by the above problems. The major contributions of our model can be summarized as follows. Firstly, DMKKM is able to directly obtain cluster indicator matrix without subsequent steps, which works as the first model to directly solve the cluster assignment problem avoiding information loss and over reliance on extra discretization procedures. Secondly, our model is capable of measuring the correlation among kernels by penalizing the selection of highly correlated kernels, which successfully enhances kernel fusion by reducing redundancy and improving diversity. Thirdly, the proposed model is completely parameter-free avoiding intractable hyperparameter tuning, which makes it more feasible in practice. Lastly, extensive experiments conducted on several real-world benchmark datasets demonstrate the effectiveness and efficiency of our proposed model.

2 Related Work

2.1 Kernel k-means (KKM)

Given a data matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$ where n is the number of data points and d is the dimension of features. Let $\phi(\cdot) : \mathbb{R}^d \to \mathcal{H}$ be a kernel mapping that maps $\boldsymbol{X}$ onto a reproducing kernel Hilbert space $\mathcal{H}$. Let $\boldsymbol{F} \in \mathbb{B}^{n\times c}$ denote the cluster indicator matrix represented by $\boldsymbol{F} \in \mathrm{Ind}$ where c is the number of clusters. If $\phi(\boldsymbol{x}_i)$ is assigned to the j-th cluster then $F_{ij} = 1$, otherwise $F_{ij} = 0$. The objective of kernel k-means (KKM) is to minimize the sum of the squared errors defined by

$$\min_{\boldsymbol{m}_j, \boldsymbol{F}\in\mathrm{Ind}} \sum_{i=1}^{n}\sum_{j=1}^{c} \|\phi(\boldsymbol{x}_i) - \boldsymbol{m}_j\|_2^2 F_{ij}, \tag{1}$$

where $\boldsymbol{m}_j$ denotes the center of the j-th cluster. Problem (1) can be reformulated in matrix form:

$$\min_{\boldsymbol{M}, \boldsymbol{F}\in\mathrm{Ind}} \|\phi(\boldsymbol{X}) - \boldsymbol{M}\boldsymbol{F}^T\|_F^2, \tag{2}$$

where $\phi(\boldsymbol{X}) = [\phi(\boldsymbol{x}_1), \cdots, \phi(\boldsymbol{x}_n)]$ and $\boldsymbol{M} = [\boldsymbol{m}_1, \cdots, \boldsymbol{m}_c]$ denotes the clustering centroid matrix. Taking the derivative of Eq. (2) w.r.t. $\boldsymbol{M}$ and setting it to zero, we have

$$\boldsymbol{M} = \phi(\boldsymbol{X})\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}. \tag{3}$$

Substituting Eq. (3) to Eq. (2) yields

$$\min_{\boldsymbol{F}\in\mathrm{Ind}} \mathrm{Tr}(\boldsymbol{K}) - \mathrm{Tr}((\boldsymbol{F}^T\boldsymbol{F})^{-\frac{1}{2}}\boldsymbol{F}^T\boldsymbol{K}\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-\frac{1}{2}}), \tag{4}$$

where $\boldsymbol{K} = \phi(\boldsymbol{X})^T\phi(\boldsymbol{X}) \in \mathbb{R}^{n\times n}$ is a kernel matrix with the $\langle i, j\rangle$-th element $K(i, j) = \phi(\boldsymbol{x}_i)^T\phi(\boldsymbol{x}_j)$. Problem (4) is NP-hard since the elements of $\boldsymbol{F}$ are constrained to be discrete values. A widely used way is to relax the discrete constraint of $\boldsymbol{F}$ and allow $\tilde{\boldsymbol{F}} = \boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-\frac{1}{2}}$ to take arbitrary real values, so problem (4) becomes

$$\max_{\tilde{\boldsymbol{F}}^T\tilde{\boldsymbol{F}}=\boldsymbol{I}} \mathrm{Tr}(\tilde{\boldsymbol{F}}^T\boldsymbol{K}\tilde{\boldsymbol{F}}), \tag{5}$$

where $\boldsymbol{I}$ denotes the identity matrix. The optimal solution to problem (5) is formed by the eigenvectors of $\boldsymbol{K}$ corresponding to its largest c eigenvalues. Since $\tilde{\boldsymbol{F}}$ is now in relaxed continuous form and has mixed signs, we have to lean upon other discretization procedures, such as k-means, so as to obtain the discrete cluster indicator matrix $\boldsymbol{F}$.

2.2 Multiple Kernel k-means (MKKM)

In multiple kernel settings, each data point has multiple feature representations via a group of kernel mappings $\{\phi_p(\cdot)\}_{p=1}^v$ which is represented as $\phi_{\boldsymbol{\gamma}}(\boldsymbol{x}_i) = [\gamma_1\phi_1(\boldsymbol{x}_i); \cdots; \gamma_v\phi_v(\boldsymbol{x}_i)]$, where $\boldsymbol{\gamma} = [\gamma_1, \cdots, \gamma_v]^T$ denotes the coefficients of each base kernel and needs to be optimized through learning. $\boldsymbol{K}_{\boldsymbol{\gamma}} = \sum_{p=1}^v \gamma_p^2\boldsymbol{K}_p$ denotes kernel matrix with the $\langle i, j\rangle$-th element $K_{\boldsymbol{\gamma}}(i, j) = \phi_{\boldsymbol{\gamma}}(\boldsymbol{x}_i)^T\phi_{\boldsymbol{\gamma}}(\boldsymbol{x}_j) = \sum_{p=1}^v \gamma_p^2\phi_p(\boldsymbol{x}_i)^T\phi_p(\boldsymbol{x}_j)$.

By replacing the kernel matrix $\boldsymbol{K}$ in Eq. (4) with $\boldsymbol{K}_{\boldsymbol{\gamma}}$, the objective of multiple kernel k-means (MKKM) can be formulated as

$$\min_{\boldsymbol{F}\in\mathrm{Ind}, \boldsymbol{\gamma}^T\mathbf{1}=1, \gamma_p\geqslant 0, \forall p} \mathrm{Tr}(\boldsymbol{K}_{\boldsymbol{\gamma}}(\boldsymbol{I} - \boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T)), \tag{6}$$

where $\mathbf{1}$ is a column vector with all elements being 1. This problem can be solved by alternatively updating $\boldsymbol{F}$ and $\boldsymbol{\gamma}$:

1) Update $\boldsymbol{F}$ with $\boldsymbol{\gamma}$ fixed. Taking the same two-stage strategy as KKM does, problem (6) becomes

$$\max_{\tilde{\boldsymbol{F}}^T\tilde{\boldsymbol{F}}=\boldsymbol{I}} \mathrm{Tr}(\tilde{\boldsymbol{F}}^T\boldsymbol{K}_{\boldsymbol{\gamma}}\tilde{\boldsymbol{F}}), \tag{7}$$

where $\tilde{\boldsymbol{F}}$ denotes the continuous relaxation of $\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-\frac{1}{2}}$. The optimal solution $\tilde{\boldsymbol{F}}$ is obtained as the c eigenvectors of $\boldsymbol{K}_{\boldsymbol{\gamma}}$ corresponding to the largest c eigenvalues. Then, $\boldsymbol{F}$ is obtained from $\tilde{\boldsymbol{F}}$ through other discretization procedures.

2) Update $\boldsymbol{\gamma}$ with $\boldsymbol{F}$ fixed. Problem (6) becomes

$$\min_{\boldsymbol{\gamma}^T\mathbf{1}=1, \gamma_p\geqslant 0, \forall p} \sum_{p=1}^{v} \gamma_p^2\mathrm{Tr}(\boldsymbol{K}_p(\boldsymbol{I} - \boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T)), \tag{8}$$

where $\boldsymbol{\gamma}$ can be optimized via solving the above quadratic programming (QP) problem with linear constraints.

2.3 MKKM with Matrix-induced Regularization (MKKM-MR)

By observing that MKKM does not sufficiently consider the correlation among base kernels, Liu *et al*. proposed to reduce the redundancy and enhance the diversity of selected kernels by incorporating a matrix-induced regularization [Liu *et al.*, 2016], as fulfilled in the following

$$\min_{\substack{\boldsymbol{F}\in\mathrm{Ind}, \boldsymbol{\gamma}^T\mathbf{1}=1, \\ \gamma_p\geqslant 0, \forall p}} \mathrm{Tr}(\boldsymbol{K}_{\boldsymbol{\gamma}}(\boldsymbol{I} - \boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T)) + \frac{\lambda}{2}\boldsymbol{\gamma}^T\boldsymbol{M}\boldsymbol{\gamma}, \tag{9}$$

where $\boldsymbol{M}$ denotes a matrix with $M(p, q) = \mathrm{Tr}(\boldsymbol{K}_p^T\boldsymbol{K}_q)$ and λ denotes the regularization parameter. This problem can be solved by alternatively updating $\boldsymbol{F}$ and $\boldsymbol{\gamma}$:

1) Update $\boldsymbol{F}$ with $\boldsymbol{\gamma}$ fixed. The solution of $\boldsymbol{F}$ is the same as MKKM.

2) Update $\boldsymbol{\gamma}$ with $\boldsymbol{F}$ fixed. Problem (8) becomes

$$\min_{\boldsymbol{\gamma}^T\mathbf{1}=1, \gamma_p\geqslant 0, \forall p} \boldsymbol{\gamma}^T(\boldsymbol{D} + \frac{\lambda}{2}\boldsymbol{M})\boldsymbol{\gamma}, \tag{10}$$

where $\boldsymbol{D}$ is a diagonal matrix with the i-th diagonal element $D_{ii} = \mathrm{Tr}(\boldsymbol{K}_p(\boldsymbol{I} - \boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T))$ and $\boldsymbol{\gamma}$ can be optimized via solving the above quadratic programming problem with linear constraints.

3 The Proposed Discrete Multiple Kernel k-means (DMKKM)

In this section, we elaborate the Discrete Multiple Kernel k-means (DMKKM) model. We first present the formulation and then develop an efficient algorithm for optimization.

3.1 The Proposed DMKKM Model

As mentioned above, the solutions of $\boldsymbol{F}$ in KKM, MKKM and MKKM-MR comprise two independent stages: first learning continuous relaxation $\tilde{\boldsymbol{F}}$ and then learning $\boldsymbol{F}$ from $\tilde{\boldsymbol{F}}$ by the discretization procedures. Such two-stage solutions are likely to result in severe information loss and unsatisfying clustering performance. To avoid this situation, we intend to devise a novel multiple kernel k-means (DMKKM) model which is able to directly generate the discrete clustering indicator matrix $\boldsymbol{F}$.

In the multiple kernel setting, each data point $\phi_{\boldsymbol{\alpha}}(\boldsymbol{x}_i)$ is represented as $\phi_{\boldsymbol{\alpha}}(\boldsymbol{x}_i) = [\sqrt{\alpha_1}\phi_1(\boldsymbol{x}_i); \cdots; \sqrt{\alpha_v}\phi_v(\boldsymbol{x}_i)]$, where $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_v]^T$ denotes the coefficients of each base kernel and needs to be optimized during learning. $\boldsymbol{K}_{\boldsymbol{\alpha}} = \sum_{p=1}^{v} \alpha_p \boldsymbol{K}_p$ is the kernel matrix with the $\langle i, j\rangle$-th element $K_{\boldsymbol{\alpha}}(i,j) = \sum_{p=1}^{v} \alpha_p \phi_p(\boldsymbol{x}_i)^T \phi_p(\boldsymbol{x}_j)$. Our discrete multiple kernel k-means (DMKKM) model can be formulated as

$$\begin{aligned} &\min_{\boldsymbol{F},\boldsymbol{\alpha}} \|\boldsymbol{K}_{\boldsymbol{\alpha}} - \boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T\|_F^2, \\ &\text{s.t. } \boldsymbol{F} \in \text{Ind}, \boldsymbol{\alpha}^T\mathbf{1} = 1, \alpha_p \geqslant 0, \forall p, \end{aligned} \tag{11}$$

where $\boldsymbol{K}_{\boldsymbol{\alpha}} = \boldsymbol{K}_{\boldsymbol{\alpha}}^T$ is a symmetric matrix, $\boldsymbol{F} \in \text{Ind}$ denote the cluster indicator matrix and $\|\cdot\|_F$ denotes the Frobenius norm. Problem (11) can be further reduced to

$$\begin{aligned} &\min_{\boldsymbol{F},\boldsymbol{\alpha}} \text{Tr}(\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{K}_{\boldsymbol{\alpha}}) - 2\,\text{Tr}(\boldsymbol{F}^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}), \\ &\text{s.t. } \boldsymbol{F} \in \text{Ind}, \boldsymbol{\alpha}^T\mathbf{1} = 1, \alpha_p \geqslant 0, \forall p. \end{aligned} \tag{12}$$

In problem (12), minimizing $\text{Tr}(\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{K}_{\boldsymbol{\alpha}})$ is equivalent to minimizing $\sum_{p=1}^{v}\sum_{q=1}^{v} \alpha_p\alpha_q \text{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q)$, where $\text{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q)$ measures the correlation between $\boldsymbol{K}_p$ and $\boldsymbol{K}_q$. To be specific, the larger the value of $\text{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q)$ is, the higher correlation between $\boldsymbol{K}_p$ and $\boldsymbol{K}_q$, and vice versa. On the one hand, if $\boldsymbol{K}_p$ and $\boldsymbol{K}_q$ are more correlated, minimizing the value of $\alpha_p\alpha_q \text{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q)$ is able to greatly reduce the risk of simultaneously assigning α_p and α_q with large weights. On the other hand, if $\boldsymbol{K}_p$ and $\boldsymbol{K}_q$ are less correlated, minimizing the value of $\alpha_p\alpha_q\text{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q)$ is able to greatly reduce the risk of simultaneously assigning α_p and α_q with small weights. Therefore, minimizing $\text{Tr}(\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{K}_{\boldsymbol{\alpha}})$ can significantly contribute to reducing the redundancy and enforcing the diversity of selected kernels.

3.2 Optimization Algorithm

Problem (12) can be solved with an alternative optimization approach. Concretely, the following shows the alternative optimization procedures updating $\boldsymbol{F}$ and $\boldsymbol{\alpha}$.

Step 1: Update $\boldsymbol{F}$ when $\boldsymbol{\alpha}$ is fixed. Problem (12) becomes the following objective function:

$$\max_{\boldsymbol{F}\in\text{Ind}} \text{Tr}(\boldsymbol{F}^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}), \tag{13}$$

Algorithm 1 Coordinate descent to solve problem (13)

Input: $\boldsymbol{K}_{\boldsymbol{\alpha}} \in \mathbb{R}^{n\times n}$, initial cluster label $\boldsymbol{F} \in \text{Ind}$
Output: Final cluster label $\boldsymbol{F} \in \text{Ind}$

```
Precompute f_l^T K_α f_l and f_l^T f_l, ∀l ∈ {1, ..., c}.
while not converge do
    for i = 1 ... n do
        Let m be the location of 1 in the i-th row of F.
        for s = 1 ... c do
            Calculate L(s) by Eq. (22) or Eq. (27).
        end for
        s* ← arg max_s L(s).
        F(i, s*) ← 1, F(i, m) ← 0.
    end for
end while
```

which is equivalent to the following vector form:

$$\max_{\boldsymbol{F}\in\text{Ind}} \sum_{l=1}^{c} \frac{\boldsymbol{f}_l^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_l}{\boldsymbol{f}_l^T\boldsymbol{f}_l}, \tag{14}$$

where $\boldsymbol{f}_l$ denotes the l-th column of $\boldsymbol{F}$, which denotes indicator clustering matrix obtained from the latest iteration. Now, we are going to obtain the discrete clustering indicator matrix $\boldsymbol{F}$ directly by utilizing coordinate descent technique [Wright, 2015], during which all variables are fixed except the i-th row being updated to its optimal value.

When we aim at updating the i-th row of $\boldsymbol{F}$, it's clear that there are c kinds of possible situations including $\{[1, 0, \cdots, 0], \ldots, [0, 0, \cdots, 1]\}$ with varying position of element 1. To be specific, we denote $\boldsymbol{F}^{(s)}$ with $\boldsymbol{s} \in \{1, \ldots, c\}$ as $\{\boldsymbol{F}^{(1)}, \cdots, \boldsymbol{F}^{(c)}\}$ varying from different situations of the i-th row of $\boldsymbol{F}$. For example, $\boldsymbol{F}^{(s)}$ denotes that only the s-th element in the i-th row of $\boldsymbol{F}$ is 1 and the rest ones are 0's, noting that $\boldsymbol{F}^{(s)}$ and $\boldsymbol{F}$ are identical except the i-th row. Hence, the objective function of updating a row can be expressed as

$$\max_{s\in\{1,\ldots,c\}} \sum_{l=1}^{c} \frac{\boldsymbol{f}_l^{(s)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_l^{(s)}}{\boldsymbol{f}_l^{(s)T}\boldsymbol{f}_l^{(s)}}, \tag{15}$$

where $\boldsymbol{f}_l^{(s)}$ is the l-th column of $\boldsymbol{F}^{(s)}$. It's viable to solve problem (15) by directly iterating through all s to find the optimal one, but the computational complexity of the brute-force search is expensive. Next, we introduce how the computational burden can be skillfully reduced. Let's introduce a constant $\boldsymbol{F}^{(0)}$ in which all elements in the i-th row are 0, while the rest rows are identical to $\boldsymbol{F}$. Therefore, problem (15) is equivalent to

$$\max_{s\in\{1,\ldots,c\}} \sum_{l=1}^{c} \left(\frac{\boldsymbol{f}_l^{(s)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_l^{(s)}}{\boldsymbol{f}_l^{(s)T}\boldsymbol{f}_l^{(s)}} - \frac{\boldsymbol{f}_l^{(0)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_l^{(0)}}{\boldsymbol{f}_l^{(0)T}\boldsymbol{f}_l^{(0)}}\right), \tag{16}$$

where $\boldsymbol{f}_l^{(0)}$ is the l-th column of $\boldsymbol{F}^{(0)}$. It's clear that $\boldsymbol{F}^{(s)}$ and $\boldsymbol{F}^{(0)}$ are identical except the s-th column, so problem (16) can be simplified as

$$\max_{s\in\{1,\ldots,c\}} \mathcal{L}(s) = \frac{\boldsymbol{f}_s^{(s)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s^{(s)}}{\boldsymbol{f}_s^{(s)T}\boldsymbol{f}_s^{(s)}} - \frac{\boldsymbol{f}_s^{(0)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s^{(0)}}{\boldsymbol{f}_s^{(0)T}\boldsymbol{f}_s^{(0)}}. \tag{17}$$

Algorithm 2 The procedure to solve problem (11)

Input: Kernels $\{\boldsymbol{K}_p \in \mathbb{R}^{n\times n}\}_{p=1}^{v}$, number of clusters c
Output: Final cluster label $\boldsymbol{F} \in \mathrm{Ind}$

```
Let α = 1_v/v, random initialize cluster labels F.
while not converge do
    Update F by Algorithm 1.
    Update α by Eq. (29).
end while
```

In order to find the optimal s of problem (17), we need to calculate its numerators and denominators corresponding to different $s \in \{1, 2, \cdots, c\}$, but directly calculating them is still time-consuming. Inspired by the fact that $\boldsymbol{F}^{(s)}$ is closely related to $\boldsymbol{F}$, we then demonstrate how to efficiently calculate the numerators and denominators in Eq. (17) by reusing previously calculated intermediate variables. For convenience, we denote the position of element 1 in the i-th row of $\boldsymbol{F}$ as m, *i.e.*, $\boldsymbol{F}^{(s)} = \boldsymbol{F}$ when $s = m$. To be more detailed, we separately discuss as:

When $s = m$, we gain $\boldsymbol{f}_s^{(s)} = \boldsymbol{f}_s$, and $\boldsymbol{f}_s^{(0)} = \boldsymbol{f}_s^{(s)} - \boldsymbol{\delta}$ denoting $\boldsymbol{\delta} \in \mathbb{R}^n$ as a column vector with i-th element being 1 and the rest being 0, so the numerators and denominators of Eq. (17) can be obtained by

$$\boldsymbol{f}_s^{(s)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s^{(s)} = \boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s, \tag{18}$$

$$\boldsymbol{f}_s^{(s)T}\boldsymbol{f}_s^{(s)} = \boldsymbol{f}_s^T\boldsymbol{f}_s, \tag{19}$$

$$\boldsymbol{f}_s^{(0)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s^{(0)} = \boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s - 2\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}(:,i) + K_{\boldsymbol{\alpha}}(i,i), \tag{20}$$

$$\boldsymbol{f}_s^{(0)T}\boldsymbol{f}_s^{(0)} = \boldsymbol{f}_s^T\boldsymbol{f}_s - 1, \tag{21}$$

where $\boldsymbol{K}_{\boldsymbol{\alpha}}(:,i)$ is the i-th column of $\boldsymbol{K}_{\boldsymbol{\alpha}}$, and $K_{\boldsymbol{\alpha}}(i,i)$ is the element settled in the i-th row of $\boldsymbol{K}_{\boldsymbol{\alpha}}(:,i)$, and thus we have

$$\mathcal{L}(s) = \frac{\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s}{\boldsymbol{f}_s^T\boldsymbol{f}_s} - \frac{\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s - 2\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}(:,i) + K_{\boldsymbol{\alpha}}(i,i)}{\boldsymbol{f}_s^T\boldsymbol{f}_s - 1}. \tag{22}$$

When $s \neq m$, we gain $\boldsymbol{f}_s^{(0)} = \boldsymbol{f}_s$, and $\boldsymbol{f}_s^{(s)} = \boldsymbol{f}_s + \boldsymbol{\delta}$, so the numerators and denominators can be obtained by

$$\boldsymbol{f}_s^{(s)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s^{(s)} = \boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s + 2\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}(:,i) + K_{\boldsymbol{\alpha}}(i,i), \tag{23}$$

$$\boldsymbol{f}_s^{(s)T}\boldsymbol{f}_s^{(s)} = \boldsymbol{f}_s^T\boldsymbol{f}_s + 1, \tag{24}$$

$$\boldsymbol{f}_s^{(0)T}\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s^{(0)} = \boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s, \tag{25}$$

$$\boldsymbol{f}_s^{(0)T}\boldsymbol{f}_s^{(0)} = \boldsymbol{f}_s^T\boldsymbol{f}_s, \tag{26}$$

so we have

$$\mathcal{L}(s) = \frac{\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s + 2\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}(:,i) + K_{\boldsymbol{\alpha}}(i,i)}{\boldsymbol{f}_s^T\boldsymbol{f}_s + 1} - \frac{\boldsymbol{f}_s^T\boldsymbol{K}_{\boldsymbol{\alpha}}\boldsymbol{f}_s}{\boldsymbol{f}_s^T\boldsymbol{f}_s}. \tag{27}$$

According to Eq. (22) or Eq. (27), we gain the ideal value of s reaching the optimal $\mathcal{L}(s)$ and denote it as s^*, which is the position of element 1 in the i-th row. Through the repeated iteration, we finish updating the i-th row of updated optimal $\boldsymbol{F}$ as shown in Algorithm 1. In practice, we break the iteration once the increasing rate of Eq. (13) is less than the threshold with the value of 10^{-3}.

Step 2: Update $\boldsymbol{\alpha}$ when $\boldsymbol{F}$ is fixed. Problem (12) becomes:

$$\begin{aligned}\min_{\boldsymbol{\alpha}} &\sum_{p=1}^{v}\sum_{q=1}^{v}\alpha_p\alpha_q\,\mathrm{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q) \\ &- \sum_{p=1}^{v}2\alpha_p\,\mathrm{Tr}(\boldsymbol{F}^T\boldsymbol{K}_p\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}), \\ \text{s.t. } &\boldsymbol{\alpha}^T\mathbf{1} = 1, \alpha_p \geqslant 0, \forall p.\end{aligned} \tag{28}$$

For simplicity, let us introduce a matrix $\boldsymbol{M} \in \mathbb{R}^{v\times v}$ whose $\langle p,q\rangle$-th element is defined as $M(p,q) = \mathrm{Tr}(\boldsymbol{K}_p\boldsymbol{K}_q)$ and a vector $\boldsymbol{d} \in \mathbb{R}^v = [\mathrm{Tr}(\boldsymbol{F}^T\boldsymbol{K}_1\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1}), \cdots, \mathrm{Tr}(\boldsymbol{F}^T\boldsymbol{K}_v\boldsymbol{F}(\boldsymbol{F}^T\boldsymbol{F})^{-1})]^T$. Problem (28) can be transformed into the following quadratic programming (QP) problem with linear constraints

$$\begin{aligned}\min_{\boldsymbol{\alpha}}\ &\boldsymbol{\alpha}^T\boldsymbol{M}\boldsymbol{\alpha} - 2\boldsymbol{d}^T\boldsymbol{\alpha}, \\ \text{s.t. }\ &\boldsymbol{\alpha}^T\mathbf{1} = 1, \alpha_p \geqslant 0, \forall p,\end{aligned} \tag{29}$$

which can be solved by any standard QP solver.

The overall optimization procedure for our proposed DMKKM model is summarized in Algorithm 2.

3.3 Complexity Analysis

The time complexities of calculating Eqs. (22) and (27) are both $\mathcal{O}(n)$, so finding the optimal s^* is $\mathcal{O}(nc)$. Considering $\boldsymbol{F}$ contains only n 1s while the rest are all 0s, the complexity can be regarded as $\mathcal{O}(n)$. Suppose Algorithm 1 converges after t_1 iterations, its time complexity is $\mathcal{O}(n^2t_1)$. The time complexity of the rest of Algorithm 2 is dominated by calculating the vector $\boldsymbol{d}$, which takes $\mathcal{O}(n^2v)$. Since the number of kernels v is usually very small, the QP solver converges very fast in practice, and thus can be ignored. In summary, the time complexity of Algorithm 2 is $\mathcal{O}((n^2t_1 + n^2v)t)$ denoting the number of the outer loop as t. Our algorithm is highly efficient comparing with other MKKM models utilizing the two-stage optimization algorithm, in which the time complexity of eigenvalue decomposition is cubic to n.

4 Experiments

In this section, we evaluate the clustering performance of the proposed DMKKM model on a number of real-world datasets. Besides, we meticulously analyze the properties of coefficients obtained by learning, meanwhile showing convergence curves to verify the efficiency of the algorithm.

Name	# Samples	# Kernels	# Classes
Handwritten	2000	6	10
Pima	768	8	2
ProteinFold	694	12	27
SensITVehicle	1500	2	3
UCI DIGIT	2000	3	10
Washington	230	2	5
Wisconsin	265	2	5

Table 1: Dataset descriptions

Dataset	Metric	MKKM	OKKC	LMKKM	RMKKM	MKKM-MR	ONKC	MVC-LFA	DMKKM
Handwritten	ACC	0.6825	0.6870	0.6550	0.6410	0.8910	0.9235	**0.9355**	0.9160
	NMI	0.6598	0.6703	0.6475	0.6742	0.8265	0.8454	**0.8670**	0.8472
	ARI	0.5411	0.5510	0.5033	0.5384	0.7811	0.8398	**0.8631**	0.8267
Pima	ACC	0.5716	0.5104	0.5208	0.5065	0.6536	0.6549	0.6471	**0.6563**
	NMI	0.0010	0.0005	0.0048	0.0000	0.0646	0.0753	0.0819	**0.0824**
	ARI	0.0073	-0.0008	-0.0006	-0.0012	0.0928	0.0949	0.0852	**0.0966**
ProteinFold	ACC	0.2853	0.2997	0.2262	0.2882	0.3732	0.3732	0.3646	**0.3862**
	NMI	0.3507	0.3988	0.3537	0.3940	0.4454	0.4471	0.4280	**0.4717**
	ARI	0.1456	0.1385	0.0844	0.1425	0.1911	0.1927	0.1916	**0.1932**
SensITVehicle	ACC	0.5787	0.5567	0.4240	0.4120	0.6560	0.5433	**0.6927**	0.6813
	NMI	0.1389	0.1388	0.0674	0.0537	0.2113	0.1143	**0.2681**	0.2405
	ARI	0.1454	0.1584	0.0605	0.0473	0.2400	0.1149	**0.2967**	0.2739
UCI DIGIT	ACC	0.4225	0.4725	0.4780	0.4150	0.9090	0.9125	0.9020	**0.9330**
	NMI	0.4548	0.4813	0.4846	0.4687	0.8368	0.8425	0.8295	**0.8715**
	ARI	0.3272	0.3066	0.3111	0.3173	0.8131	0.8206	0.8008	**0.8589**
Washington	ACC	0.4478	0.4783	0.4957	0.4696	0.5609	0.6087	0.5870	**0.6130**
	NMI	0.0491	0.0571	0.1433	0.0433	0.3172	0.3494	0.3335	**0.3799**
	ARI	0.0662	0.0390	0.1203	0.0594	0.3086	0.3904	0.3561	**0.4033**
Wisconsin	ACC	0.5547	0.5623	0.4566	0.5245	0.5585	**0.5962**	0.5623	0.5925
	NMI	0.2288	0.2532	0.1629	0.2474	0.3323	0.3360	0.3505	**0.3565**
	ARI	0.2188	0.2028	0.0730	0.2339	0.2660	0.3073	0.2878	**0.3166**

Table 2: Clustering result on real-world benchmark datasets, the best ones are in bold.

4.1 Dataset Descriptions

Seven real-world benchmark datasets are employed to evaluate the clustering performance, including *Handwritten*, *Pima*, *ProteinFold*, *SensITVehicle*, *UCI DIGIT*, *Washington* and *Wisconsin*. All these datasets are downloaded from Xinwang Liu's page[1] and more details can be found in their published papers. More importantly, we summarize their statistical properties in Table 1.

4.2 Comparison Models

We compare our proposed DMKKM model with several multi-kernel k-means clustering models, including:

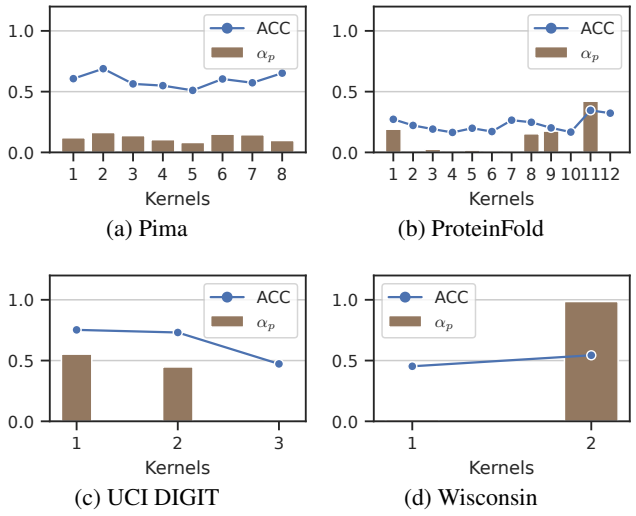
- Multiple Kernel k-means (MKKM) [Huang *et al.*, 2011]: The algorithm performs kernel k-means and updates kernel coefficients alternately, as introduced in Section 2.2.
- Optimized Kernel k-means Clustering (OKKC) [Yu *et al.*, 2011]: This is a concurrent work of MKKM, but optimizes the kernel coefficients and cluster membership based on the same Rayleigh quotient objective.
- Localized Multiple Kernel k-means (LMKKM) [Gönen and Margolin, 2014]: The model learns the kernel coefficients with a localized approach.
- Robust Multiple Kernel k-means (RMKKM) [Du *et al.*, 2015]: RMKKM replaced the squared error term of k-means with $\ell_{2,1}$-norm based one to make it more robust.
- Multiple Kernel k-means with Matrix-induced Regularization (MKKM-MR) [Liu *et al.*, 2016]: The model introduced a matrix-induced regularization to reduce the redundancy and enhance the diversity of selected kernels.
- Optimal Neighborhood Kernel Clustering (ONKC) [Liu *et al.*, 2017]: This model allows the optimal kernel to reside in the neighborhood of the base kernels to enlarge the region from which an optimal kernel can be chosen.
- Multi-view Clustering via Late Fusion Alignment Maximization (MVC-LFA) [Wang *et al.*, 2019b]: This model proposes to maximally align the consensus partition with the weighted base partitions, and they theoretically prove that it is equivalent to minimize the loss function of k-means clustering.

Among these models, MKKM, OKKC, LMKKM and DMKKM are parameter-free, while RMKKM, MKKM-MR have one hyperparameter, and ONKC, MVC-LFA have two hyperparameters.

4.3 Experimental Settings

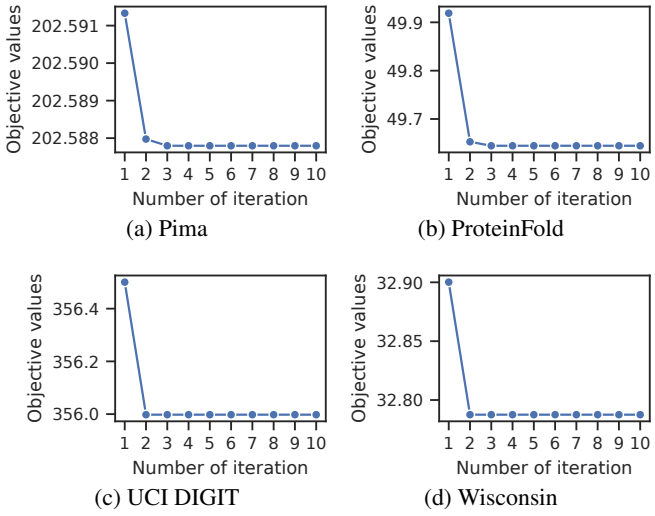
For the sake of fairness, we perform gird search to determine the hyperparameters for all comparison models with parameters choosing their best results, under the guidance of relative papers, though our DMKKM is completely parameter-free model. The source codes are downloaded from the authors' pages or requested from the authors.

[1] https://xinwangliu.github.io



(a) Pima (b) ProteinFold (c) UCI DIGIT (d) Wisconsin

Figure 1: Learned coefficients α_p and KKM accuracy of each kernel



(a) Pima (b) ProteinFold (c) UCI DIGIT (d) Wisconsin

Figure 2: Objective value of Eq. (11) after each iteration

4.4 Result Analyses

Clustering performance. Table 2 presents the results of three widely-adopted clustering performance metrics, including clustering accuracy (ACC), normalized mutual information (NMI) and adjusted rand index (ARI). According to the presented results, we can conclude through observation as follows: *a*) Our proposed DMKKM model achieved the best clustering performance on 5 out of 7 datasets, indicating its superb performance. *b*) Comparing to other parameter-free models, our model not only consistently outperforms them on all datasets, but also achievs a huge advantage over them, showing excellent superiority. *c*) Comparing to other parameterized models, our model surpasses them in most of cases. Despite MVC-LFA achieving better performance on *Handwritten* and *SensITVehicle* datasets with a very small lead. However, MVC-LFA has 2 hyperparameters and badly relies on rigorous grid search, which makes it impractical for production use. In reverse, our proposed model is totally parameter-free which is much more feasible for production use.

Properties of learned coefficients. In this part, we analyse the properties of the learned coefficients α_p of each kernel. Kernels usually work as various feature representations of data samples, but some kinds of kernels may be more discriminative than others for particular tasks, which may contribute to better clustering performance. Thus, such kinds of kernels deserve more significance, in other word, more weight, so as to achieve better performance through multi-kernel clustering models. Arguably, the single-kernel clustering models like KKM will obtain better performance with the help of more distinctive kernel, so we utilize the clustering accuracy (ACC) of KKM as a metric to identify the significance of kernels. To be specific, the higher clustering accuracy KKM can achieve, the more discriminative a kernel is. Ideally, our proposed DMKKM model is supposed to assign larger α_p to the kernels where KKM achieved higher ACC. We plot the learned α_p and corresponding KKM accuracy of each kernel on 4 datasets in Figure 1. It can be shown that our model succeeds in assigning larger coefficients to kernels which are more discriminative, so our proposed model is able to identify more discriminative kernels among multiple kernels, which may help to improve the clustering performance.

Convergence analysis. To demonstrate the effectiveness of our optimization algorithm, we plot the convergence curves of Algorithm 2 on 4 datasets in Figure 2, where the x-axes denotes the number of iterations and the y-axes denotes the objective value of Eq. (11). It can be shown that the objective values decrease monotonically until Algorithm 2 converges. All the curves plateau within 10 iterations, which means that our optimization algorithm is of high efficiency.

5 Conclusion

In this paper, we propose a novel multiple kernel k-means clustering model, namely DMKKM, utilizing an efficient iterative algorithm to solve it. Our model can directly obtain the cluster indicator matrix without subsequent discretization steps, avoiding severe information loss. Moreover, our model is capable of measuring the correlation among kernels, which greatly enhances kernel fusion through reducing redundancy and improving diversity. What's more, our model is totally parameter-free, which is more feasible in practice. Extensive experiments on several real-world datasets demonstrate its superb performance and potentiality in practical applications.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0101902, in part by the Natural Science Basic Research Program of Shaanxi (Program No. 2021JM-071), in part by the National Natural Science Foundation of China under Grant 61936014 and Grant 61772427, and in part by the Fundamental Research Funds for the Central Universities under Grant G2019KY0501.

References

[Calandriello and Rosasco, 2018] Daniele Calandriello and Lorenzo Rosasco. Statistical and computational trade-offs in kernel k-means. In *Proc. NeurIPS*, pages 9357–9367, 2018.

[Du *et al.*, 2015] Liang Du, Peng Zhou, Lei Shi, Hanmo Wang, Mingyu Fan, Wenjian Wang, and Yi-Dong Shen. Robust multiple kernel k-means using l21-norm. In *Proc. IJCAI*, pages 3476–3482, 2015.

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

[Hartigan and Wong, 1979] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society*, 28(1):100–108, 1979.

[He and Zhang, 2018] Li He and Hong Zhang. Kernel k-means sampling for nyström approximation. *IEEE Trans. Image Process.*, 27(5):2108–2120, 2018.

[Huang *et al.*, 2011] Hsin-Chien Huang, Yung-Yu Chuang, and Chu-Song Chen. Multiple kernel fuzzy clustering. *IEEE Trans. Fuzzy Syst.*, 20(1):120–134, 2011.

[Kang *et al.*, 2018] Zhao Kang, Xiao Lu, Jinfeng Yi, and Zenglin Xu. Self-weighted multiple kernel learning for graph-based clustering and semi-supervised classification. In *Proc. IJCAI*, pages 2312–2318, 2018.

[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 *Proc. AAAI*, pages 1888–1894, 2016.

[Liu *et al.*, 2017] Xinwang Liu, Sihang Zhou, Yueqing Wang, Miaomiao Li, Yong Dou, En Zhu, and Jianping Yin. Optimal neighborhood kernel clustering with multiple kernels. In *Proc. AAAI*, pages 2266–2272, 2017.

[Schölkopf *et al.*, 1998] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[Van Laarhoven and Marchiori, 2016] Twan Van Laarhoven and Elena Marchiori. Local network community detection with continuous optimization of conductance and weighted kernel k-means. *The Journal of Machine Learning Research*, 17(1):5148–5175, 2016.

[Vankadara and Ghoshdastidar, 2020] Leena C Vankadara and Debarghya Ghoshdastidar. On the optimality of kernels for high-dimensional clustering. In *Proc. AISTATS*, pages 2185–2195, 2020.

[Wang *et al.*, 2019a] Shusen Wang, Alex Gittens, and Michael W Mahoney. Scalable kernel k-means clustering with nyström approximation: relative-error bounds. *The Journal of Machine Learning Research*, 20(1):431–479, 2019.

[Wang *et al.*, 2019b] Siwei Wang, Xinwang Liu, En Zhu, Chang Tang, Jiyuan Liu, Jingtao Hu, Jingyuan Xia, and Jianping Yin. Multi-view clustering via late fusion alignment maximization. In *Proc. IJCAI*, pages 3778–3784, 2019.

[Wright, 2015] Stephen J Wright. Coordinate descent algorithms. *Math. Program.*, 151(1):3–34, 2015.

[Xu *et al.*, 2017] Jinglin Xu, Junwei Han, Feiping Nie, and Xuelong Li. Re-weighted discriminatively embedded k-means for multi-view clustering. *IEEE Trans. Image Process.*, 26(6):3016–3027, 2017.

[Yu *et al.*, 2011] Shi Yu, Leon Tranchevent, Xinhai Liu, Wolfgang Glanzel, Johan AK Suykens, Bart De Moor, and Yves Moreau. Optimized data fusion for kernel k-means clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(5):1031–1039, 2011.

[Zhao *et al.*, 2009] Bin Zhao, James T Kwok, and Changshui Zhang. Multiple kernel clustering. In *Proc. SDM*, pages 638–649, 2009.