Incomplete Multiple Kernel Alignment Maximization for Clustering

Xinwang Liu, *Senior Member, IEEE*

Abstract—Multiple kernel alignment (MKA) maximization criterion has been widely applied into multiple kernel clustering (MKC) and many variants have been recently developed. Though demonstrating superior clustering performance in various applications, it is observed that none of them can effectively handle incomplete MKC, where parts or all of the pre-specified base kernel matrices are incomplete. To address this issue, we propose to integrate the imputation of incomplete kernel matrices and MKA maximization for clustering into a unified learning framework. The clustering of MKA maximization guides the imputation of incomplete kernel elements, and the completed kernel matrices are in turn combined to conduct the subsequent MKC. These two procedures are alternately performed until convergence. By this way, the imputation and MKC processes are seamlessly connected, with the aim to achieve better clustering performance. Besides theoretically analyzing the clustering generalization error bound, we empirically evaluate the clustering performance on several multiple kernel learning (MKL) benchmark datasets, and the results indicate the superiority of our algorithm over existing state-of-the-art counterparts. Our codes and data are publicly available at https://xinwangliu.github.io/.

Index Terms—multiple kernel clustering, multi-view clustering, kernel alignment maximization

1 Introduction

KERNEL alignment criterion [1] measures the agreement between a kernel and a given learning task. It is simple, effective and easy-to-implement, and therefore has been widely adopted in kernel methods such as kernel parameter tuning [2], multiple kernel learning (MKL) [3], to name just a few. The recent work in [4] extends the kernel alignment criterion from supervised learning scenarios to multiple kernel clustering (MKC) by jointly maximizing the clustering partition matrix and kernel coefficients. More importantly, it builds up the theoretical connection between multiple kernel alignment (MKA) for clustering and existing multiple kernel k-means, by which the objectives of existing MKC algorithms can be unified from the perspective of kernel alignment. MKA based clustering algorithms have been intensively studied and demonstrated promising clustering in various applications such as image fusion [5], image retrieval [6], document/video analysis [7], to name just a few. It well indicates the importance and effectiveness of this criterion [8].

As a representative of MKA based clustering algorithms, the more recent work in [9] revisits the MKA criterion and proposes a novel MKC algorithm termed SimpleMKKM. Different from existing minimization-minimization optimization framework [10], SimpleMKKM firstly reformulates the MKA criterion for clustering as a minimization-maximization optimization problem, and then develops a reduced-gradient descend algorithm to solve it. Despite its simplicity, SimpleMKKM is considered as one of the state-of-the-art MKC algorithms by showing competitive clustering performance in empirical study.

Though demonstrating promising clustering performance, SimpleMKKM assumes that all the pre-calculated kernel matrices are complete. In some real-world applications [11], [12], [13], [14], [15], [16], [17], however, it is not unusual to see that some kinds of features of a sample are unavailable, which causes the corresponding rows and columns of related kernel matrices unfilled. As a result, the violation to this assumption makes SimpleMKKM inapplicable for conducting clustering in this challenging setting, which is called incomplete MKC in this paper. There are two categories of methods to address incomplete MKC in the literature. The first category is termed "two-stage" methods, which firstly impute incomplete kernels with filling algorithms [18], [19], [20], [21] and then apply MKC with the imputed kernels. It is recognized that these "two-stage" algorithms share a drawback that they disconnect the processes of imputation and clustering, and this prevents the two learning processes from negotiating with each other to achieve the optimal clustering. Differently, the second category, termed "one-stage" methods [13], [22], [23], [24], addresses this issue by designing a clustering-oriented imputation algorithm to impute the missing parts of a kernel during the optimization for clustering. Conceptually, our work in this paper belongs to the second category, but it enjoys clear advantages over the existing ones in terms of optimization efficiency and clustering performance.

In this paper, we develop a simple while effective algorithm to enable SimpleMKKM to well handle MKC with incomplete kernels. Specifically, the proposed algorithm unifies the imputation of incomplete kernels and the clustering task into a single objective function. The clustering guides the imputation of incomplete kernels, and the completed kernels are in turn optimally combined to conduct the subsequent MKC. These two procedures are alternately performed until convergence. By this way, the imputation and MKC processes are seamlessly connected, with the

- *X. Liu is with College of Computer, National University of Defense Technology, Changsha, 410073, China. E-mail: xinwangliu@nudt.edu.cn.*

Manuscript received June 27, 2021.

aim to achieve better clustering performance. Further, we theoretically analyze the performance of our algorithm on unseen data via deriving its clustering generalization error bound. Extensive experimental study on several multiple kernel learning (MKL) benchmark datasets indicates the superiority of our algorithm over existing state-of-the-art counterparts.

The rest of this paper is organized as follows: Section 2 briefly introduces several related work. Section 3 is devoted to our proposed incomplete multiple kernel alignment maximization for clustering. Section 4 analyzes the generalization bound of the proposed algorithm. Extensive experiments are conducted in Section 5 to support our claims. Our work is concluded in Section 6.

2 Related Work

In this section, we briefly review the most related, including multiple kernel k-means (MKKM), multiple kernel k-means with incomplete kernels (MKKM-IK) [13] and the recently proposed SimpleMKKM [9].

2.1 MKKM

Given a group of pre-calculated kernel matrices $\{\mathbf{K}_p\}_{p=1}^m$, MKKM assumes that the optimal kernel matrix $\mathbf{K}_{\gamma}$ can be parameterized as $\mathbf{K}_{\gamma} = \sum_{p=1}^m \gamma_p^2 \mathbf{K}_p$, where $\gamma \in \Delta = \{\gamma \in \mathbb{R}^m | \sum_{p=1}^m \gamma_p = 1,\, \gamma_p \geq 0,\, \forall p\}$ represents the kernel weights of these base kernel matrices. It jointly learns the kernel weights γ and the clustering partition matrix $\mathbf{H}$ by optimizing Eq. (1).

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

In literature, the optimization problem in Eq. (1) is usually be solved by alternatively updating $\mathbf{H}$ and γ: (i) **Optimizing $\mathbf{H}$ given γ.** With the kernel coefficients γ fixed, $\mathbf{H}$ can be obtained by solving a kernel k-means clustering optimization problem; (ii) **Optimizing γ given $\mathbf{H}$.** With $\mathbf{H}$ fixed, γ can be optimized via solving the following quadratic programming with linear constraints,

$$\min_{\gamma \in \Delta}\ \sum_{p=1}^m \gamma_p^2 \mathrm{Tr}\left(\mathbf{K}_p(\mathbf{I}_n - \mathbf{H}\mathbf{H}^{\top})\right), \tag{2}$$

which has a closed-form solution.

Algorithm 1 presents the whole algorithm to solve the optimization problem in Eq. (1). This algorithm adopts an alternate way to optimize $\mathbf{H}$ and γ. Specifically, one variable is optimized with the other fixed. By this way, the objective in Eq. (1) can be monotonically minimized. Meanwhile, this objective is lower-bounded. As a result, the Algorithm in 1 is theoretically guaranteed to converge to a (local) optimum.

As seen from Eq. (2), using a linear combination of kernels $\sum_{p=1}^m \gamma_p \mathbf{K}_p$ to replace $\sum_{p=1}^m \gamma_p^2 \mathbf{K}_p$ is not a viable option, because this could make only one single kernel activate and all the others assigned with zero weight. Other recent work using ℓ_2-norm combinations can be found in [13], [25], [26].

Algorithm 1 MKKM

```
Input: $\{\mathbf{K}_p\}_{p=1}^m$, $k$.
Output: $\mathbf{H}$ and $\gamma$.
Initialize $\gamma^{(1)} = 1/m$, flag $= 1$ and $t = 1$.
while flag do
    compute $\mathbf{H}$ by solving a kernel k-means with $\mathbf{K}_{\gamma^{(t)}} = \sum_{p=1}^m \left(\gamma_p^{(t)}\right)^2 \mathbf{K}_p$.
    update $\gamma^{(t+1)}$ with Eq. (2).
    if max $|\gamma^{(t)} - \gamma^{(t-1)}| \leq 1e-4$ then
        flag=0.
    end if
    $t \leftarrow t+1$.
end while
```

2.2 MKKM-IK

The recently proposed MKKM-IK [10] has extended the existing MKKM in Eq. (1) to enable it to handle multiple kernel clustering with incomplete kernels. It unifies the imputation and clustering procedure into a single optimization objective and alternately optimizes each of them. That is, i) imputing the absent kernels under the guidance of clustering; and ii) updating the clustering with the imputed kernels. The above idea is mathematically fulfilled as,

$$\begin{aligned} \min_{\mathbf{H},\, \gamma,\, \{\mathbf{K}_p\}_{p=1}^m}\ & \mathrm{Tr}(\mathbf{K}_{\gamma}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^{\top})) \\ s.t.\ & \mathbf{H} \in \mathbb{R}^{n \times k}, \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k, \\ & \gamma^{\top}\mathbf{1}_m = 1, \gamma_p \geq 0, \\ & \mathbf{K}_p(\mathbf{s}_p, \mathbf{s}_p) = \mathbf{K}_p^{(oo)},\ \mathbf{K}_p \succeq 0,\ \forall p, \end{aligned} \tag{3}$$

where $\mathbf{s}_p$ $(1 \leq p \leq m)$ denote the sample indices for which the p-th view is observed and $\mathbf{K}_p^{(oo)}$ be used to denote the kernel sub-matrix computed with these samples. The constraint $\mathbf{K}_p(\mathbf{s}_p, \mathbf{s}_p) = \mathbf{K}_p^{(oo)}$ is imposed to ensure that $\mathbf{K}_p$ maintains the known entries during the course. Different from the optimization in MKKM, [10] incorporates an extra step to impute the missing entries of base kernels, leading to a three-step alternate optimization algorithm. Specifically, MKKM-IK optimizes one variable by keeping other variables fixed at each iteration, as outlined in Algorithm 2. Interested readers are referred to [10].

Algorithm 2 MKKM-IK

```
Input: $\{\mathbf{K}_p^{(oo)}\}_{p=1}^m$, $k$, and $\{\mathbf{s}_p\}_{p=1}^m$.
Output: $\mathbf{H}$, $\gamma$ and $\{\mathbf{K}_p\}_{p=1}^m$.
Initialize $\gamma^{(0)} = \mathbf{1}_m/m$, $\{\mathbf{K}_p^{(0)}\}_{p=1}^m$, and $t = 1$.
repeat
    $\mathbf{K}_{\gamma^{(t)}} = \sum_{p=1}^m \left(\gamma_p^{(t-1)}\right)^2 \mathbf{K}_p^{(t-1)}$.
    Update $\mathbf{H}^{(t)}$ by solving kernel $k$-means with given $\mathbf{K}_{\gamma^{(t)}}$.
    Update each $\mathbf{K}_p^{(t)}$ with $\mathbf{H}^{(t)}$ and $\{\mathbf{K}_q^{(t-1)}\}_{q=1,q\neq p}^m$.
    Update $\gamma^{(t)}$ with given $\mathbf{H}^{(t)}$ and $\{\mathbf{K}_p^{(t)}\}_{p=1}^m$.
    $t = t+1$.
until max $|\gamma^{(t)} - \gamma^{(t-1)}| \leq 1e-4$
```

2.3 SimpleMKKM

We briefly introduce simple multiple kernel k-means (SimpleMKKM), which is recently developed for MKC in [9] and will serve the base for the proposed algorithm to deal with incomplete MKC. SimpleMKKM revisits the criterion of MKA, and develops a simple yet effective multiple kernel clustering algorithm. Different from existing $\min_{\gamma}\min_{\mathbf{H}}$ optimization framework in Eq. (1), SimpleMKKM proposes a minimization-maximization problem with respect to the kernel coefficient and clustering partition matrix as follows.

$$\begin{aligned} \min\nolimits_{\gamma\in\Delta}\ \max\nolimits_{\mathbf{H}}\ &\langle \mathbf{K}_{\gamma}, \mathbf{H}\mathbf{H}^{\top}\rangle \\ s.t.\ &\mathbf{H}\in\mathbb{R}^{n\times k},\, \mathbf{H}^{\top}\mathbf{H}=\mathbf{I}_k, \end{aligned} \tag{4}$$

where $\{\mathbf{K}_p\}_{p=1}^{m}$ are m pre-calculated base kernel matrices, $\mathbf{K}_{\gamma}=\sum_{p=1}^{m}\gamma_p^2\mathbf{K}_p$, $\Delta=\{\gamma\in\mathbb{R}^m|\gamma^{\top}\mathbf{1}=1,\,\gamma_p\geq 0,\forall p\}$ and $\mathbf{H}$ denotes the clustering partition matrix satisfying the orthogonal constraint $\mathbf{H}^{\top}\mathbf{H}=\mathbf{I}_k$. Noting that $\langle\mathbf{K}_{\gamma},\mathbf{H}\mathbf{H}^{\top}\rangle$ is just the MKA criterion previously mentioned.

Instead of solving the formulation in Eq. (4) by the widely adopted alternate optimization, the work in [9] designs an efficient and effective reduced gradient descent algorithm. Firstly, the optimization in Eq. (4) is equivalently rewritten as,

$$\min\nolimits_{\gamma\in\Delta}\ \mathcal{J}(\gamma), \tag{5}$$

with

$$\mathcal{J}(\gamma)=\left\{\max\nolimits_{\mathbf{H}}\ \mathrm{Tr}\left(\mathbf{K}_{\gamma}\mathbf{H}\mathbf{H}^{\top}\right)\ \ s.t.\ \mathbf{H}^{\top}\mathbf{H}=\mathbf{I}_k\right\}. \tag{6}$$

By this way, the min-max optimization is transformed to a minimization one, where its objective is a kernel k-means optimal value function. After proving the differentiability of $\mathcal{J}(\gamma)$ w.r.t γ, SimpleMKKM applies the reduced gradient descent algorithm to decrease Eq. (5), where the equality and positive constraints on γ are guaranteed at each iteration. The whole algorithm procedure solving the optimization problem in Eq. (4) is outlined in Algorithm 3. Interested readers are referred to [9] for the detailed optimization.

Algorithm 3 SimpleMKKM

```
Input: $\{\mathbf{K}_p\}_{p=1}^{m}$ and $k$.
Output: $\mathbf{H}$ and $\gamma$.
Initialize $\gamma^{(1)}=1/m$, flag $=1$ and $t=1$.
while flag do
    compute $\mathbf{H}$ by solving a kernel k-means with $\mathbf{K}_{\gamma^{(t)}}=\sum_{p=1}^{m}(\gamma_p^{(t)})^2\mathbf{K}_p$.
    compute $\frac{\partial\mathcal{J}(\gamma)}{\partial\gamma_p}\,(p=1,\cdots,m)$ and the descent direction $\mathbf{d}^{(t)}$.
    update $\gamma^{(t+1)}\leftarrow\gamma^{(t)}+\alpha\mathbf{d}^{(t)}$.    ▷ $\alpha$ is the optimal step size
    if max $|\gamma^{(t)}-\gamma^{(t-1)}|\leq 1e-4$ then
        flag=0.
    end if
    $t\leftarrow t+1$.
end while
```

Though SimpleMKKM in Eq. (4) demonstrates superior clustering performance in various applications, we observe that it cannot efficiently deal with MKC with incomplete kernels. For example, in some practical applications such as Alzheimer's disease prediction [11] and cardiac disease discrimination [27], it is not uncommon to see that some kinds of features of a sample are missing, and this causes the corresponding rows and columns of related base kernels unfilled. The presence of incomplete base kernels makes it difficult to utilize the information of all kernels for clustering. In the following, we further develop a simple while effective algorithm, termed incomplete multiple kernel alignment maximization for clustering, to address this issue.

3 Incomplete Multiple Kernel Alignment Maximization for Clustering

3.1 The Proposed Formulation

Let $\mathbf{s}_p\,(1\leq p\leq m)$ denote the sample indices for which the p-th base kernel is observed and $\mathbf{K}_p^{(oo)}$ be used to denote the kernel sub-matrix computed with these samples. Our learning task is to integrate m incomplete kernel matrices, i.e., $\{\mathbf{K}_p^{(oo)}\}_{p=1}^{m}$ for clustering. This incomplete setting is not uncommon in various real-world applications such as cancer biology [28], analysis of multiple heterogeneous neuroimaging data [11], and Alzheimer's disease diagnosis [29].

Existing two-stage approaches first impute these base kernels which are then taken as the input of a conventional MKC algorithm. It is observed that the imputation by such manners may not be able to serve or even hurt the subsequent clustering tasks [22]. To maintain the advantage brought by the recently proposed one-stage methods such as learning clustering-oriented imputation [13], [22], we shall aim to directly improve the clustering by treating the absent kernel entries as auxiliary unknowns during this course. That is to jointly perform imputation and clustering: i) impute the missing parts of kernels under the guidance of clustering; and ii) update the clustering by optimally combined the imputed kernels. By this way, the above two learning processes can be seamlessly coupled and they are allowed to negotiate with each other to achieve better clustering. On top of this, we seek a more natural and reasonable manner to deal with the incompleteness in multiple kernel clustering. In specific, we propose to jointly impute the incomplete kernels and combine them for clustering by maximizing the aforementioned multiple kernel alignment in Eq. (4), leading to the following new optimization problem,

$$\begin{aligned} \max\nolimits_{\{\mathbf{K}_p\}_{p=1}^{m}}\min\nolimits_{\gamma\in\Delta}\max\nolimits_{\mathbf{H}}\ &\langle\mathbf{K}_{\gamma},\mathbf{H}\mathbf{H}^{\top}\rangle \\ s.t.\ &\mathbf{H}\in\mathbb{R}^{n\times k},\,\mathbf{H}^{\top}\mathbf{H}=\mathbf{I}_k, \\ &\mathbf{K}_p\succeq 0,\ \mathbf{K}_p(\mathbf{s}_p,\mathbf{s}_p)=\mathbf{K}_p^{(oo)}. \end{aligned} \tag{7}$$

Noting that compared with SimpleMKKM, we now have to optimize extra variables $\{\mathbf{K}_p\}_{p=1}^{m}$ due to the presence of incomplete kernels. The equality constraint, i.e., $\mathbf{K}_p(\mathbf{s}_p,\mathbf{s}_p)=\mathbf{K}_p^{(oo)}$, ensures that the observed part of the p-th base kernel matrix will be faithfully maintained during optimization. In addition, the imputed $\mathbf{K}_p$ shall retain its PSD property.

The formulation in Eq. (7) has the following advantages when compared with MKKM-IK [22]. Firstly, it considerably extends the widely adopted MKA criterion, making it capable of effectively dealing with incomplete MKC. When $\{\mathbf{K}_p\}_{p=1}^{m}$ become available, the proposed Eq. (7) is degenerated to Eq. (4). Secondly, it inherits the effective objective and novel optimization technique from SimpleMKKM, which is considered to be the most effective MKC algorithm based on the MKA criterion so far. In addition, our algorithm in Eq. (7) is free of hyper-parameters, which largely reduces the parameter-tuning burden of practitioners, making them more convenient for practical applications.

3.2 The Proposed Optimization

Although Eq. (7) enables existing MKA to effectively tackle incomplete MKC, the optimization with respect to incomplete kernels, especially the positive semidefinite (PSD) and equality constraints on each $\mathbf{K}_p$, makes the optimization more challenging. In the following, we reformulate Eq. (7) and develop an efficient algorithm to solve it.

Let $\mathbf{\Phi}_p^{(o)}$ and $\mathbf{\Phi}_p^{(u)}$ be the observed and unobserved implicit feature maps corresponding to the p-th kernel, with $\mathbf{K}_p^{(oo)} = \mathbf{\Phi}_p^{(o)}\mathbf{\Phi}_p^{(o)\top}$. Since all samples, no matter observed or unobserved, are subject to the same distribution, we can assume that the unobserved part $\mathbf{\Phi}_p^{(u)}$ could be expressed by a linear transformation of the observed $\mathbf{\Phi}_p^{(o)}$, i.e., $\mathbf{\Phi}_p^{(u)} = \mathbf{W}_p\mathbf{\Phi}_p^{(o)}$. Similar assumption has been widely adopted in Nyström method for large-scale kernel matrix approximation [30]. Based on this assumption, each $\mathbf{K}_p$ can be parameterized as

$$\mathbf{K}_p = \begin{pmatrix} \mathbf{K}_p^{(oo)} & \mathbf{K}_p^{(oo)}\mathbf{W}_p \\ \mathbf{W}_p^\top\mathbf{K}_p^{(oo)} & \mathbf{W}_p^\top\mathbf{K}_p^{(oo)}\mathbf{W}_p \end{pmatrix}. \tag{8}$$

In addition, an extra orthogonal constraint $\mathbf{W}_p^\top\mathbf{W}_p = \mathbf{I}_{n_p^{(u)}}$ is imposed to ensure that the whole optimization is bounded, where $n_p^{(u)}$ is the number of unobserved samples for the p-th kernel.

It is not difficult to verify that the parameterization in Eq. (8) makes both the PSD and equality constraints automatically satisfied. As a result, the imputation of incomplete kernels boils down to the learning of $\{\mathbf{W}_p\}_{p=1}^m$. With this parameterization, Eq. (7) can be equivalently written as

$$\begin{aligned} \max_{\{\mathbf{W}_p\}_{p=1}^m} \min_{\gamma\in\Delta} \max_{\mathbf{H}} \ & \langle \mathbf{K}_{\gamma,\{\mathbf{W}_p\}_{p=1}^m}, \mathbf{HH}^\top \rangle \\ s.t. \ & \mathbf{H} \in \mathbb{R}^{n\times k}, \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \\ & \mathbf{W}_p^\top\mathbf{W}_p = \mathbf{I}_{n_p^{(u)}}, \forall p, \end{aligned} \tag{9}$$

where $\mathbf{K}_{\gamma,\{\mathbf{W}_p\}_{p=1}^m} = \sum_{p=1}^m \gamma_p^2\mathbf{K}_p$ and $\mathbf{K}_p$ is parameterized in Eq. (8).

Jointly optimizing $\{\mathbf{W}_p\}_{p=1}^m$, γ and $\mathbf{H}$ in Eq. (9) is difficult. Instead, we can solve it in an alternate manner. That is, solving γ and $\mathbf{H}$ with given $\{\mathbf{W}_p\}_{p=1}^m$, and solving $\{\mathbf{W}_p\}_{p=1}^m$ with given γ and $\mathbf{H}$.

3.2.1 *Solving γ and $\mathbf{H}$ with given $\{\mathbf{W}_p\}_{p=1}^m$*

With $\{\mathbf{W}_p\}_{p=1}^m$ fixed, the missing elements of $\mathbf{K}_p$ can be imputed via Eq. (8). As a result, the formulation in Eq. (9) w.r.t. γ and $\mathbf{H}$ reduces to the one in Eq. (4), which can be readily solved via SimpleMKKM in Algorithm 3.

3.2.2 *Solving $\{\mathbf{W}_p\}_{p=1}^m$ with given γ and $\mathbf{H}$*

With γ and $\mathbf{H}$ fixed, Eq. (9) w.r.t. $\{\mathbf{W}_p\}_{p=1}^m$ can be equivalently decomposed into m independent sub-problems as follows,

$$\begin{aligned} \max_{\mathbf{W}_p} \ & \mathrm{Tr}\left(\mathbf{W}_p^\top\mathbf{K}_p^{(oo)}\mathbf{W}_p\mathbf{T}_{uu}\right) + 2\mathrm{Tr}\left(\mathbf{W}_p^\top\mathbf{K}_p^{(oo)}\mathbf{T}_{ou}\right), \\ s.t. \ & \mathbf{W}_p^\top\mathbf{W}_p = \mathbf{I}_{n_p^{(u)}} \end{aligned} \tag{10}$$

where $\mathbf{T} = \mathbf{HH}^\top = \begin{pmatrix} \mathbf{T}_{oo} & \mathbf{T}_{ou} \\ \mathbf{T}_{ou}^\top & \mathbf{T}_{uu} \end{pmatrix}$, o and u denote the indices of observed and unobserved samples for the p-th kernel, respectively. Eq. (10) can be efficiently solved via reweighed methods [31] in Algorithm 4.

Algorithm 4 Solving $\mathbf{W}_p$ with the reweighted method

```
Input: $\mathbf{K}_p^{(oo)}$, $\mathbf{T}$ and $\mathbf{s}_p$.
Output: $\mathbf{W}_p$.
Initialize $\mathbf{W}_p^{(1)}$ and $t = 1$.
repeat
    $\Sigma = 2\mathbf{K}_p^{(oo)}(\mathbf{W}_p^{(t)}\mathbf{T}_{uu} + \mathbf{T}_{ou})$.
    Update $\mathbf{W}_p^{(t+1)}$ by the optimal solution to $\max_{\mathbf{W}_p^\top\mathbf{W}_p=\mathbf{I}_{n_p^{(u)}}} \mathrm{Tr}(\mathbf{W}_p^\top\Sigma)$.
    $t = t + 1$.
until $\|\mathbf{W}_p^{(t-1)} - \mathbf{W}_p^{(t)}\|/\|\mathbf{W}_p^{(t)}\| \leq 1e^{-3}$
```

In sum, the whole algorithm solving Eq. (9) is outlined in Algorithm 5. As seen, Algorithm 5 alternately performs imputation of incomplete kernels and MKA for clustering. With the given imputation, the proposed algorithm can benefit from effective objective and advanced optimization technique, leading to better $\mathbf{H}$ and γ. The effective $\mathbf{H}$ in turns produces an imputation better serve for multiple kernel clustering. These two procedures boost each other until satisfying the stopping criterion. It usually converges in less then ten iterations in our experiments, as will be shown in Figure 6.

Algorithm 5 The Proposed Incomplete Multiple Kernel Alignment Maximization for Clustering

```
Input: $\{\mathbf{K}_p^{(oo)}\}_{p=1}^m$, $\{\mathbf{s}_p\}_{p=1}^m$ and $\epsilon_0$.
Output: $\mathbf{H}$, $\{\mathbf{K}_p\}_{p=1}^m$ and $\gamma$.
Initialize $\{\mathbf{W}_p^{(1)}\}_{p=1}^m$ and $t = 1$.
repeat
    Update $\mathbf{H}^{(t)}$ and $\gamma^{(t)}$ with fixed $\{\mathbf{W}_p^{(t)}\}_{p=1}^m$ via Algorithm 3.

    Update $\{\mathbf{W}_p^{(t+1)}\}_{p=1}^m$ with fixed $\mathbf{H}^{(t)}$ and $\gamma^{(t)}$ via Algorithm 4.
    $t = t + 1$.
until $\left(\mathrm{obj}^{(t-1)} - \mathrm{obj}^{(t)}\right)/\mathrm{obj}^{(t)} \leq \epsilon_0$
```

3.3 Discussion and Extension

In this section, we discuss the proposed algorithm from computational complexity, convergence, initialization on incomplete base kernel matrix $\mathbf{K}_p$ and other parametrization on $\mathbf{K}_p$.

Computational complexity: As seen from Algorithm 5, the proposed algorithm at each iteration needs to solve a SimpleMKKM problem, and update $\{\mathbf{W}_p\}_{p=1}^m$. The computational complexity of SimpleMKKM and updating $\{\mathbf{W}_p\}_{p=1}^m$ are $\mathcal{O}(T_t * (n^3 + m * n^3))$ and $n_p^{(o)} * n_p^{(u)} * \min\{n_p^{(o)}, n_p^{(u)}\}$, where T_t is the number of iterations to achieve convergence with given $\{\mathbf{W}_p\}_{p=1}^m$, $n_p^{(o)}$ and $n_p^{(u)}$ are the number of observed and unobserved samples of the p-th base kernel.

As a result, the whole computational complexity of the proposed algorithm at the t-th iteration is $\mathcal{O}(T_t * (n^3 + m * n^3) + n_p^{(o)} * n_p^{(u)} * \min\{n_p^{(o)}, n_p^{(u)}\})$. As observed, though the proposed algorithm does not significantly increase the computational complexity of existing MKKM algorithms, improving its efficiency to handle large-scale applications is worth further exploring.

Convergence: Algorithm 5 adopts a coordinate descent algorithm to solve the optimization problem in Eq. (7). With imputed $\{\mathbf{K}_p\}_{p=1}^m$, a clustering partition matrix $\mathbf{H}$ is generated by SimpleMKKM. It is then used for completing the missing parts of each incomplete base kernel via Algorithm 4. The convergence of this optimization procedure cannot be theoretically guaranteed. However, we empirically observe that Algorithm 5 quickly converges after several iterations in all benchmark datasets, as shown by the experimental results in Figure 6.

Initialization on $\mathbf{K}_p$ *and other variants*: In our current implementation, we initialize the missing parts of each base kernels as zeros. This initialization has well demonstrated the superiority of the proposed algorithm, as seen from the following experimental study. It is worth pointing out that other imputation methods such as mean-value filling, k-nearest neighborhood filling, EM filling, can also be taken as the initialization. More importantly, by parameterizing $\mathbf{K}_p$ as in Eq. (8), the imputation of missing parts is equivalently reduced to optimize $\{\mathbf{W}_p\}_{p=1}^m$. As seen, a better parametrization on $\mathbf{K}_p$ would produce better imputation, leading to improved clustering performance. The parametrization in our work has sufficiently demonstrated its superiority, how to incorporate prior knowledge to design more effective parametrization on $\mathbf{K}_p$ is still worth exploring.

Differences with MKKM-IK [13]: Both work handle incomplete MKC by unifying the imputation of incomplete kernels and clustering task into a single optimization framework. Nevertheless, they have the following important differences: 1) *Optimization criterion*. With imputed kernels, MKKM-IK optimizes the kernel coefficient and clustering matrix via a joint minimization procedure. Differently, our algorithm adopts a minimization-maximization procedure to optimize the kernel coefficient and clustering matrix, respectively. 2) *Optimization methods*. MKKM-IK solves the resultant optimization with coordinate descent, while our work applies the reduced gradient descent optimization. 3) *The clustering performance is different*. We empirically compare their clustering performance, and observe that the proposed algorithm consistently and significantly outperforms MKKM-IK on all benchmark datasets, as shown in Figure 1, 2 and Table 2.

4 The Generalization Analysis

In this section, we analyze the generalization error bound of the proposed algorithm by studying how its learned centroids generalizes onto unseen data. Let $\hat{\mathbf{C}} = [\hat{\mathbf{C}}_1, \cdots, \hat{\mathbf{C}}_k]$ be the k centroids, $\{\mathbf{K}_p\}_{p=1}^m$ the imputed kernel and $\hat{\gamma}$ the kernel coefficients learned by the proposed algorithm, where $\hat{\mathbf{C}}_v = \frac{1}{|\hat{\mathbf{C}}_v|}\sum_{j\in\hat{\mathbf{C}}_v} \phi_{\hat{\gamma}}(\mathbf{x}_j), 1 \leq v \leq k$. By defining $\Theta = \{\mathbf{e}_1, \cdots, \mathbf{e}_k\}$, our algorithm should make the error on unseen samples small as follows,

$$1 - \mathbb{E}_{\mathbf{x}}\left[\max\nolimits_{\mathbf{y}\in\Theta}\langle\phi_{\hat{\gamma},\mathbf{t}}(\mathbf{x}), \hat{\mathbf{C}}\mathbf{y}\rangle_{\mathcal{H}^k}\right], \tag{11}$$

where $\phi_{\hat{\gamma},\mathbf{t}}(\mathbf{x}) = [\hat{\gamma}_1 t(\mathbf{x}^{(1)})\phi_1^\top(\mathbf{x}^{(1)}), \cdots, \hat{\gamma}_m t(\mathbf{x}^{(m)})\phi_m^\top(\mathbf{x})]^\top$ is the learned feature map associated with the kernel function $K_{\hat{\gamma}}(\cdot,\cdot)$. $\mathbf{t} = [t(\mathbf{x}^{(1)}), \cdots, t(\mathbf{x}^{(m)})]^\top$ denotes the absence of $\mathbf{x}$, i.e., $t(\mathbf{x}^{(p)}) = 1$ indicates that the p-th observation of $\mathbf{x}$ is observed, otherwise its value is missing. $\mathbf{e}_1, \cdots, \mathbf{e}_k$ form the orthogonal bases of $\mathbb{R}^k$.

Intuitively, Eq. (11) says the expected alignment between test points and their closest centroid should be high. We show how the proposed algorithm achieves this goal. We define a function class first:

$$\begin{aligned}\mathcal{F} =\Big\{ f:\ \mathbf{x} \mapsto\ 1 - \max_{\mathbf{y}\in\Theta}\langle\phi_{\gamma,\mathbf{t}}(\mathbf{x}), \mathbf{C}\mathbf{y}\rangle_{\mathcal{H}^k}\Big|\gamma\in\Delta, \mathbf{C}\in\mathcal{H}^k,\\ |K_p(\mathbf{x},\tilde{\mathbf{x}})| \leq b, \forall\mathbf{x}\in\mathcal{X}\Big\},\end{aligned} \tag{12}$$

where $\mathcal{H}^k$ stands for the multiple kernel Hilbert space. Note that the orthogonal constraints on $\mathbf{W}_p$ makes the elements of $\mathbf{K}_p$ in Eq. (8) bounded. Therefore, we can assume that each entry of $\mathbf{K}_p, p \in \{1, \cdots, m\}$ is no larger than b.

Theorem 1. *For any $\delta > 0$, with probability at least $1-\delta$, the following holds for all $f \in \mathcal{F}$:*

$$\begin{aligned}\mathbb{E}_{\mathbf{x}}[f(\mathbf{x})] \leq \frac{1}{n}\sum\nolimits_{i=1}^n f(\mathbf{x}_i) + \frac{b}{n}\sqrt{2\pi}\mathcal{G}_n(\gamma,\mathbf{t})\\ + (1+b)\sqrt{\frac{\log 1/\delta}{2n}}\end{aligned}, \tag{13}$$

where $\mathcal{G}_n(\gamma,\mathbf{t}) = \mathbb{E}_\beta\left[\sup_{\gamma,\mathbf{t}}\sum_{v=1}^k\sum_{i=1}^n\sum_{p=1}^m \beta_{ivp}\gamma_p^2 t(\mathbf{x}_i^{(p)})\right]$ *and* β_{ivp} *is i.i.d. Gaussian variable with zero mean and unit standard deviation.*

The detailed proof is provided in the supplemental material due to the page limit. Note that if all kernels are observed, we have $\mathcal{G}_n(\gamma,\mathbf{t}) \leq mk\sqrt{n}$. In such case, our algorithm will have generalization bounds of order $\mathcal{O}(\sqrt{1/n})$.

According to Theorem 1, for the learned $\hat{\gamma}$ and $\hat{\mathbf{C}}$, to achieve a small $\mathbb{E}_{\mathbf{x}}[f(\mathbf{x})]$ in Eq. (13), $\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)$ should be as small as possible. Assume that γ and $\mathbf{C}$ are obtained by minimizing $\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)$ and that $\mathbf{H}$ is orthogonal, we have $\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i) \leq 1 - \frac{1}{n}\mathrm{Tr}(\mathbf{K}_\gamma\mathbf{H}\mathbf{H}^\top)$. This is because the posed orthogonal constraint $\mathbf{H}^\top\mathbf{H} = \mathbf{I}_k$ may make the corresponding centroids non-optimal for minimizing $\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)$. This implies that $\frac{1}{n}\sum_{i=1}^n f(\mathbf{x}_i)$ is upper bounded by $1 - \frac{1}{n}\mathrm{Tr}(\mathbf{K}_\gamma\mathbf{H}\mathbf{H}^\top)$. To minimize the upper bound, we may have to maximize over $\gamma, \mathbf{H}$ and $\{\mathbf{K}_p\}_{p=1}^m$, leading to $\max_{\{\mathbf{K}_p\}_{p=1}^m}\max_\gamma\max_{\mathbf{H}}\mathrm{Tr}(\mathbf{K}_\gamma\mathbf{H}\mathbf{H}^\top)$. However, the work in [9] observes that it is intractable to find a good solution under this criterion, and prone to over-fitted solutions. Instead, we take one of its lower bounds, $\max_{\{\mathbf{K}_p\}_{p=1}^m}\min_\gamma\max_{\mathbf{H}}\mathrm{Tr}(\mathbf{K}_\gamma\mathbf{H}\mathbf{H}^\top)$ as the criterion, which is exactly the objective of the proposed algorithm in Eq. (7). This considerably justifies the effectiveness of the proposed algorithm.

5 Experimental Analysis

5.1 Experimental Settings

We study the clustering performance of the proposed algorithm on several benchmark datasets, including Protein Fold[1], UCI Digit[2], Oxford Flower17[3], Oxford Flower102[4], Caltech102[5], SUNRGBDSUN [32], and NUSWIDEOBJ [33]. These datasets have been widely used to evaluate the clustering performance of MKC algorithms and can be publicly downloaded from the aforementioned websites. The number of samples, kernels and classes are listed in Table 1.

TABLE 1: Datasets used in our experiments.

Dataset	#Samples	#Kernels	#Classes
Protein Fold	694	12	27
UCI Digit	2000	3	10
Flower17	1360	7	17
Flower102	8189	4	102
Caltech102	1530	25	102
SUNRGBD	10335	2	45
NUSWIDEOBJ	12001	5	30

Along with the proposed algorithm, we run another six comparable algorithms in recent incomplete MKC literature, including: MKKM with zero-filling (MKKM+ZF), MKKM with means-filling (MKKM+MF), MKKM with KNN-filling (MKKM+KNN), MKKM with alignment-maximization filling (MKKM+AF), MKKM with incomplete kernels (MKKM-IK) [10], efficient and effective incomplete multi-view clustering (EE-IMVC) [34], multiple incomplete views clustering via weighted nonnegative matrix factorization (MIC) [17] and doubly aligned incomplete multi-view clustering (DAIMC) [35]. Among these compared algorithms, MKKM-IK and EE-IMVC are considered to be the state-of-the-art one in handling incomplete MKC. The implementations of these compared algorithms can be publicly downloaded. We run these code and report the results without revision in our experiment.

For the dataset preprocessing, we centralize and scale each incomplete base kernel to make $\kappa_p(\mathbf{x}_i, \mathbf{x}_i) = 1$ for all i and p by following the settings of [10], [34]. For each data set, we assume that the intrinsic number of clusters is known. Then, we generate incomplete kernels by following the same settings in [10], [34] and creating missing index vectors $\{\mathbf{s}_p\}_{p=1}^m$. Concretely, to simulate datasets with incomplete views, $\text{round}(\varepsilon * n)$ samples are randomly selected as samples with incomplete views, where $\text{round}(\cdot)$ is a rounding function and ε is the missing ratio. Different εs are corresponding to view missing of different extent. For each selected sample, a random vector $\mathbf{v} = (v_1, \cdots, v_m) \in [0,1]^m$ and a random scalar v_0 ($v_0 \in [0,1]$) are generated to represent the visibility of each view. The p-th view is observable for this sample if $v_p \geq v_0$. In the case that all $v_1, \cdots, v_m < v_0$ and all views are unobservable, a new $\mathbf{v}$ will be generated to ensure that at least one view is available for a sample. As to the samples without missing views, the corresponding missing indicating vector is a vector with all elements to be 1. After generating one $\mathbf{v}$ for each sample, we obtain the index vector $\mathbf{s}_p$ indicating the visibility of samples in the p-th view.

We adopt clustering accuracy (ACC), normalized mutual information (NMI) and purity as the criterion to evaluate the clustering performance of the aforementioned algorithms. We firstly give the definitions of ACC, NMI and Purity as follows. Let $\mathbf{\Omega} = [\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_k]^\top$ and $\mathbf{C} = [\mathbf{c}_1, \cdots, \mathbf{c}_l]^\top$ denote the predicted cluster labels of a clustering algorithm and the provided ground-truth labels of $\{\mathbf{x}_i\}_{i=1}^n$, respectively. The clustering accuracy (ACC) is defined as follows,

$$ACC(\mathbf{\Omega}, \mathbf{C}) = \frac{\sum_{i=1}^{n} \delta(y_i, map(r_i))}{n}, \tag{14}$$

where y_i and r_i denote the provided ground-truth label and predicted cluster label of $\mathbf{x}_i$ ($1 \leq i \leq n$), $\delta(u, v)$ is the delta function that equals one if $u = v$ and equals zero otherwise, and $map(r_i)$ is a permutation mapping function that maps each cluster label r_i to the equivalent label from data. The best mapping can be found by using the Kuhn-Munkres algorithm [36].

The mutual information between $\mathbf{\Omega}$ and $\mathbf{C}$, denoted as $\mathrm{MI}(\mathbf{\Omega}, \mathbf{C})$, is defined as follows:

$$\mathrm{MI}(\mathbf{\Omega}, \mathbf{C}) = \sum\nolimits_{\boldsymbol{\omega}_j \in \mathbf{\Omega},\, \mathbf{c}_t \in \mathbf{C}} p(\boldsymbol{\omega}_j, \mathbf{c}_t) \log_2 \frac{p(\boldsymbol{\omega}_j, \mathbf{c}_t)}{p(\boldsymbol{\omega}_j) p(\mathbf{c}_t)}, \tag{15}$$

where $p(\boldsymbol{\omega}_j)$ and $p(\mathbf{c}_t)$ are the probabilities that a sample arbitrarily selected from data belongs to the clusters $\boldsymbol{\omega}_j$ and $\mathbf{c}_t$, respectively, and $p(\boldsymbol{\omega}_j, \mathbf{c}_t)$ is the joint probability that the arbitrarily selected samples belongs to the clusters $\boldsymbol{\omega}_j$ and $\mathbf{c}_t$ at the same time. The normalized mutual information (NMI) is then defined as follows:

$$\mathrm{NMI}(\mathbf{\Omega}, \mathbf{C}) = \frac{\mathrm{MI}(\mathbf{\Omega}, \mathbf{C})}{\max\left(\mathrm{H}(\mathbf{\Omega}), \mathrm{H}(\mathbf{C})\right)}, \tag{16}$$

where $\mathrm{H}(\mathbf{\Omega})$ and $\mathrm{H}(\mathbf{C})$ are the entropies of $\mathbf{\Omega}$ and $\mathbf{C}$, respectively.

The purity is calculated as follows. Each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned documents and dividing by n. Formally,

$$\mathrm{Purity}(\mathbf{\Omega}, \mathbf{C}) = \frac{1}{n} \sum\nolimits_{j=1}^{k} \max_{1 \leq t \leq l} |\boldsymbol{\omega}_j \cap \mathbf{c}_t|. \tag{17}$$

To consider the randomness incurred by the missing indicating vectors $\{\mathbf{s}_q\}_{q=1}^m$, for different missing ratios, the "incomplete" patterns are randomly generated for 20 times via the above-mentioned procedure and their statistical results are reported. In addition, to reduce the affect of randomness caused by k-means, for each compared algorithm, we repeat the experiment for 50 times with random initialization and report the best result.

5.2 Robustness Comparison Against Missing Ratios

5.2.1 *Results on Protein Fold*

The missing ratio ε is an important parameter which could largely affect the performance of algorithms in comparison. Intuitively, the larger the value of ε is, the more information

[1] http://mkl.ucsd.edu/dataset/protein-fold-prediction
[2] http://ss.sysu.edu.cn/py/
[3] http://www.robots.ox.ac.uk/~vgg/data/flowers/17/
[4] http://www.robots.ox.ac.uk/~vgg/data/flowers/102/
[5] http://files.is.tue.mpg.de/pgehler/projects/iccv09/

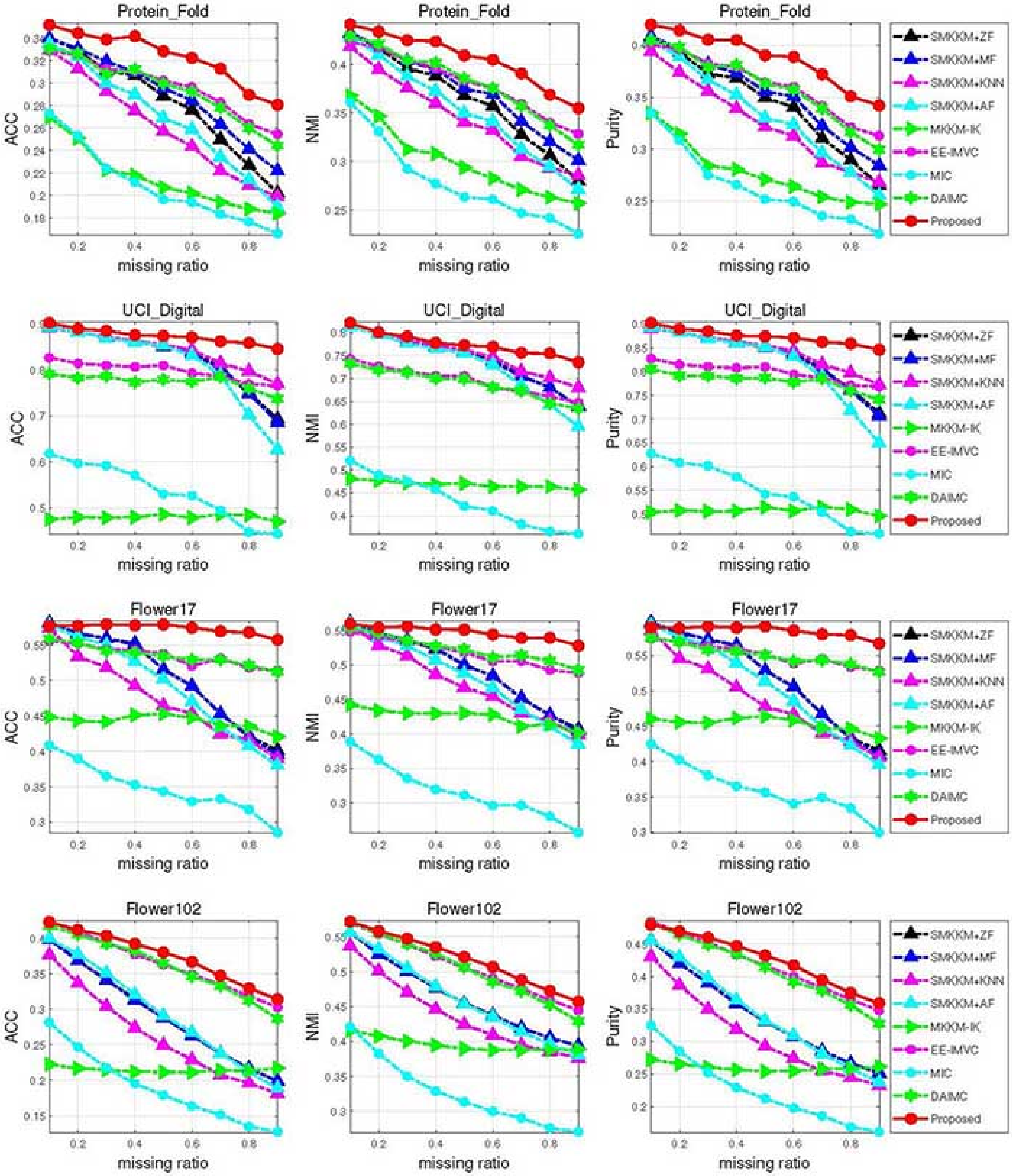

Fig. 1: The ACC, NMI and purity of different algorithms with the variation of missing ratios on Protein Fold, UCI-Digital, Flower17 and Flower102 datasets. For each missing ratio, we randomly generate the "incomplete" patterns for 20 times and report the statistical results.

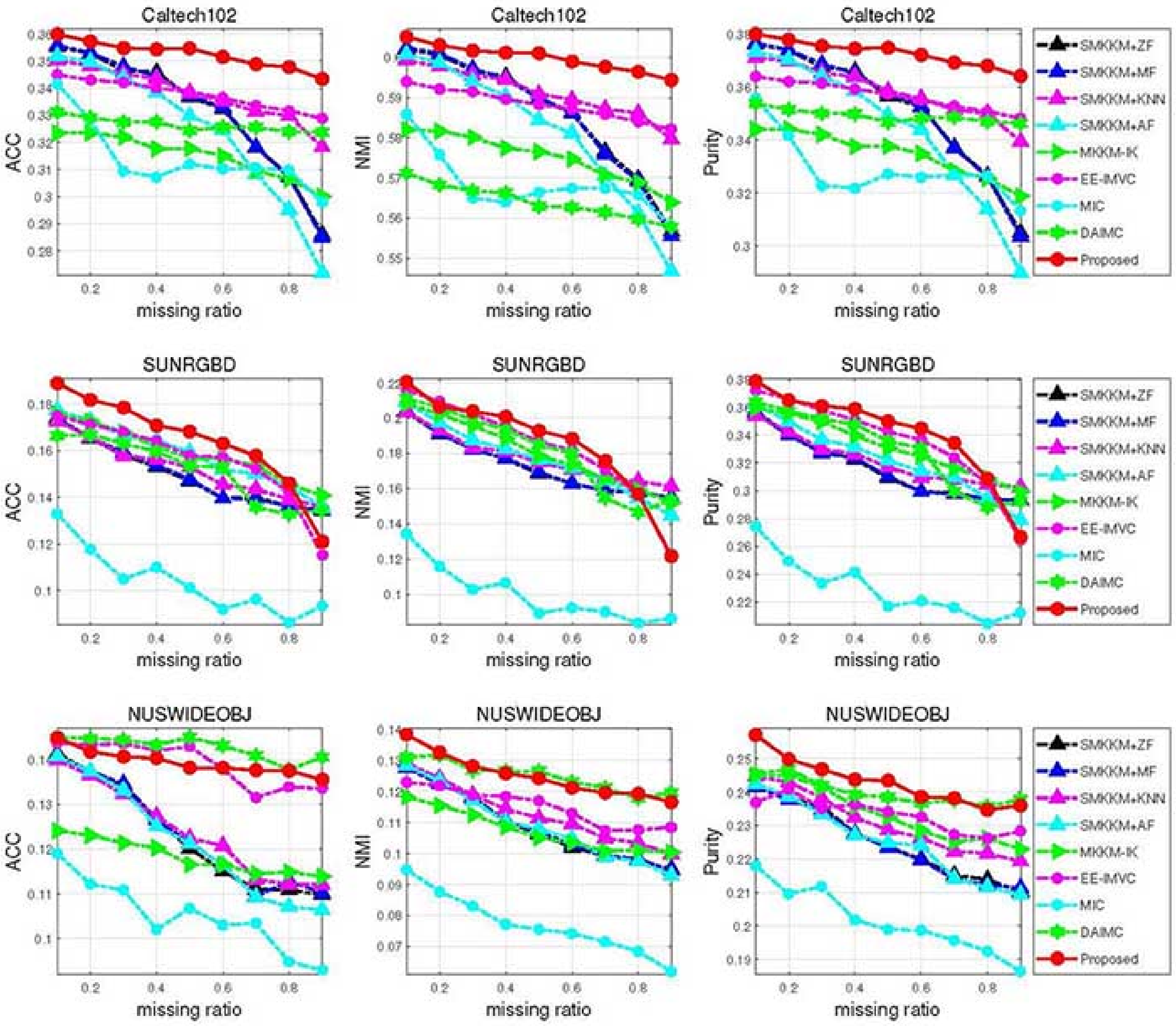

Fig. 2: The ACC, NMI and purity of different algorithms with the variation of missing ratios on Caltech102, SUNRGBD and NUSWIDEOBJ dataset. For each missing ratio, we randomly generate the "incomplete" patterns for 20 times and report the statistical results.

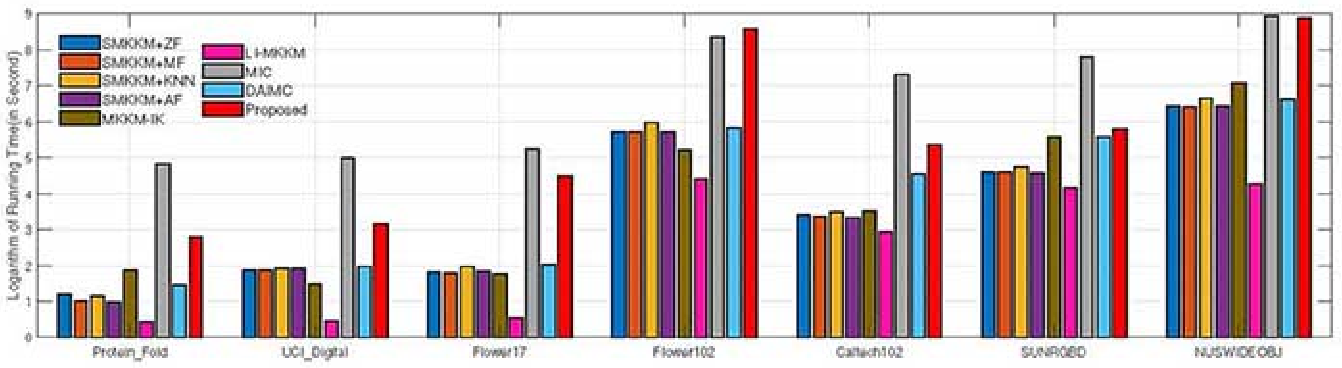

Fig. 3: Execution time comparison of different algorithms on seven benchmark datasets (in second). The experiments are conducted on a PC with Intel (R) Core (TM)-i9-10900X 3.7GHz CPU and 64G RAM in MATLAB R2020b environment.

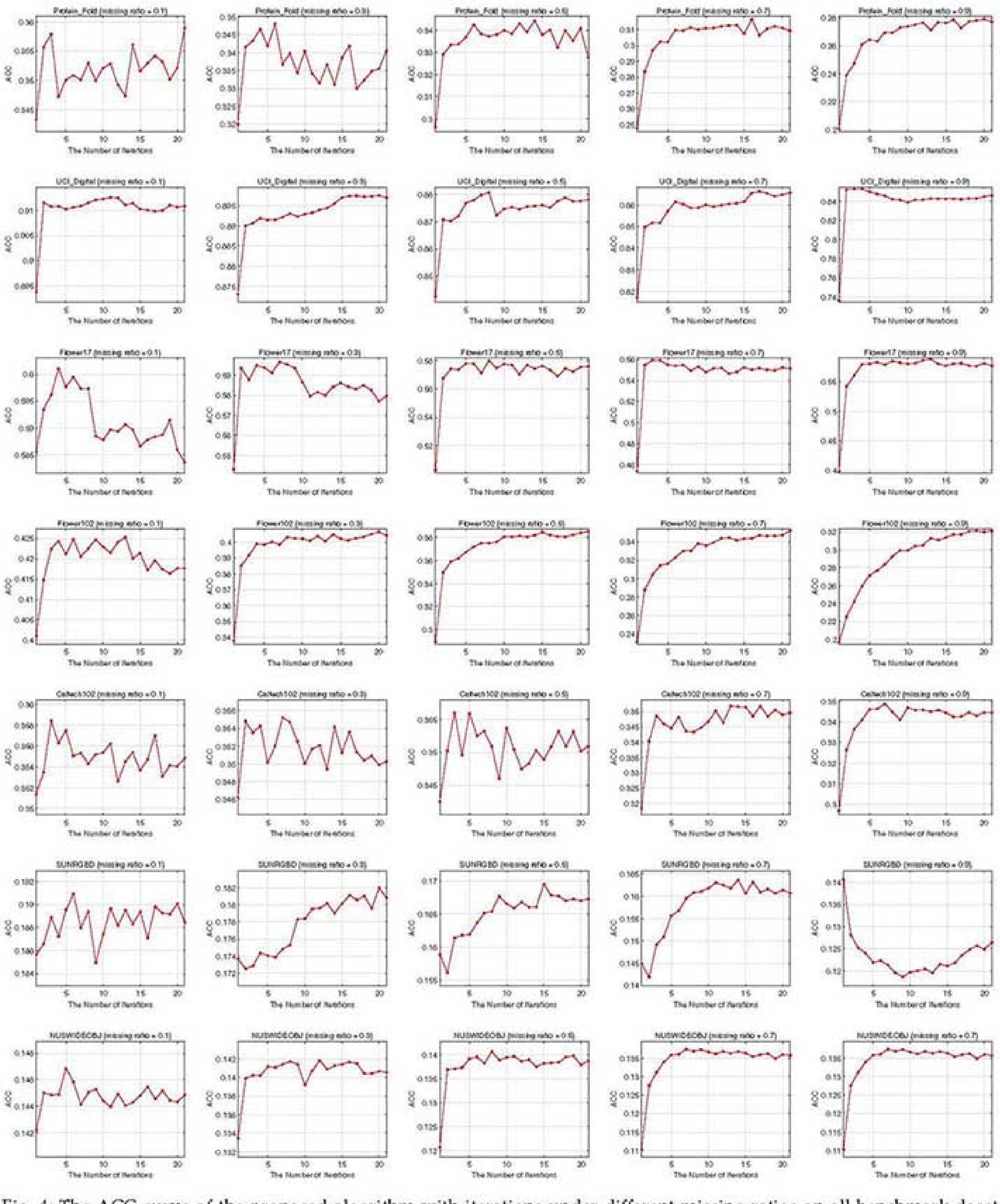

Fig. 4: The ACC curve of the proposed algorithm with iterations under different missing ratios on all benchmark dasets. The curves in terms of NMI and purity are provided in the supplemental material due to the space limit.

TABLE 2: Aggregated ACC, NMI and purity comparison (mean±std) of different clustering algorithms on all benchmark datasets. Boldface means no statistical difference from the best one.

Datasets	MKKM			MKKM-AF	MKKM-IK	EE-IMVC	MIC	DAIMC	Proposed
	+ZF	+MF	+KNN	[19]	[10]	[34]	[17]	[35]	
				ACC					
Protein_Fold	28.1 ± 1.4	29.0 ± 1.4	26.0 ± 1.2	26.8 ± 1.3	21.5 ± 1.0	29.8 ± 1.5	20.9 ± 0.8	29.5 ± 1.6	**32.4 ± 1.5**
UCI_Digital	82.8 ± 0.7	82.6 ± 0.7	84.3 ± 0.4	81.2 ± 0.9	48.0 ± 0.8	79.8 ± 0.3	53.6 ± 1.9	77.5 ± 0.9	**87.4 ± 0.1**
Flower17	50.5 ± 2.0	50.4 ± 1.9	47.4 ± 1.6	49.0 ± 1.9	44.2 ± 1.9	53.5 ± 1.6	34.7 ± 1.2	53.6 ± 1.6	**57.4 ± 1.5**
Flower102	29.2 ± 0.7	29.2 ± 0.7	26.2 ± 0.6	29.4 ± 0.7	21.5 ± 0.5	36.4 ± 1.0	18.9 ± 0.5	36.0 ± 1.0	**37.4 ± 0.9**
Caltech102	33.1 ± 1.0	33.1 ± 1.1	33.8 ± 1.0	32.4 ± 1.0	31.5 ± 1.0	33.8 ± 1.0	31.4 ± 0.6	32.7 ± 1.0	**35.3 ± 1.1**
SUNRGBD	15.0 ± 0.4	15.0 ± 0.4	15.2 ± 0.4	15.8 ± 0.5	15.9 ± 0.4	15.6 ± 0.4	10.4 ± 0.4	15.2 ± 0.4	**16.4 ± 0.5**
NUSWIDEOBJ	12.3 ± 0.3	12.3 ± 0.3	12.4 ± 0.2	12.2 ± 0.2	11.8 ± 0.3	13.9 ± 0.3	10.5 ± 0.2	**14.3 ± 0.4**	13.9 ± 0.4
mean	35.9	35.9	35.1	35.3	27.8	37.5	25.8	37.0	**40.0**
				NMI					
Protein_Fold	36.3 ± 0.9	37.3 ± 1.0	34.5 ± 0.8	35.2 ± 0.9	30.0 ± 0.8	38.1 ± 1.0	27.8 ± 0.6	38.1 ± 1.0	**40.6 ± 0.9**
UCI_Digital	74.3 ± 0.5	74.1 ± 0.5	75.3 ± 0.3	72.9 ± 0.7	46.8 ± 0.3	69.5 ± 0.3	43.1 ± 1.0	68.9 ± 0.4	**77.6 ± 0.2**
Flower17	49.3 ± 1.0	49.3 ± 1.0	47.3 ± 1.0	48.0 ± 1.0	42.5 ± 1.1	51.8 ± 0.8	31.7 ± 0.7	52.3 ± 0.8	**54.7 ± 0.8**
Flower102	46.3 ± 0.4	46.3 ± 0.4	43.8 ± 0.3	46.1 ± 0.4	39.6 ± 0.3	50.7 ± 0.4	32.6 ± 0.3	50.4 ± 0.4	**51.8 ± 0.4**
Caltech102	58.6 ± 0.6	58.6 ± 0.6	59.1 ± 0.5	58.1 ± 0.5	57.5 ± 0.6	58.8 ± 0.6	56.8 ± 0.4	56.4 ± 0.6	**60.0 ± 0.5**
SUNRGBD	17.3 ± 0.2	17.3 ± 0.3	17.8 ± 0.3	17.7 ± 0.3	18.0 ± 0.2	18.2 ± 0.2	10.0 ± 0.3	18.1 ± 0.2	**18.5 ± 0.3**
NUSWIDEOBJ	10.9 ± 0.2	10.9 ± 0.2	11.3 ± 0.2	10.9 ± 0.2	10.7 ± 0.2	11.5 ± 0.2	7.7 ± 0.1	**12.5 ± 0.2**	**12.5 ± 0.2**
mean	41.9	42.0	41.3	41.3	35.0	42.7	30.0	42.4	**45.1**
				Purity					
Protein_Fold	34.4 ± 1.2	35.3 ± 1.2	32.6 ± 1.1	33.3 ± 1.2	27.8 ± 0.9	36.2 ± 1.2	26.4 ± 0.8	36.0 ± 1.3	**38.8 ± 1.2**
UCI_Digital	83.1 ± 0.7	82.9 ± 0.6	84.4 ± 0.4	81.7 ± 0.9	50.7 ± 0.6	79.9 ± 0.3	54.7 ± 1.6	78.1 ± 0.7	**87.4 ± 0.1**
Flower17	51.9 ± 1.8	51.8 ± 1.7	48.7 ± 1.5	50.5 ± 1.7	45.4 ± 1.7	55.2 ± 1.3	36.1 ± 1.0	55.1 ± 1.3	**58.5 ± 1.4**
Flower102	34.1 ± 0.7	34.1 ± 0.6	30.9 ± 0.6	34.1 ± 0.6	26.0 ± 0.5	41.7 ± 0.7	22.4 ± 0.5	41.1 ± 0.7	**42.6 ± 0.7**
Caltech102	35.1 ± 1.0	35.1 ± 1.0	35.8 ± 0.9	34.3 ± 0.9	33.5 ± 0.9	35.7 ± 1.0	32.9 ± 0.6	34.9 ± 0.9	**37.3 ± 0.9**
SUNRGBD	31.6 ± 0.5	31.6 ± 0.5	32.2 ± 0.5	32.2 ± 0.5	33.2 ± 0.5	33.6 ± 0.5	23.0 ± 0.5	33.0 ± 0.4	**34.1 ± 0.5**
NUSWIDEOBJ	22.5 ± 0.4	22.5 ± 0.4	23.1 ± 0.3	22.5 ± 0.4	23.4 ± 0.3	23.3 ± 0.3	20.2 ± 0.2	24.0 ± 0.3	**24.3 ± 0.4**
mean	41.8	41.9	41.1	41.2	34.3	43.6	30.8	43.2	**46.1**

would be lost and the poorer the clustering performance could be resulted. To evaluate the robustness against missing ratios of different algorithms, in the first experiment, we compare the state-of-the-art algorithms with respect to different ε. The results variation of the compared algorithms on Protein Fold dataset when ε varies in the range of $[0.1 : 0.1 : 0.9]$ are illustrated in Figure 1a.

From the sub-figures, we observe that:

- Our algorithm consistently and significantly outperforms "two-stage" methods, including MKKM+ZF, MKKM+MF, MKKM+KNN, and MKKM+AF. For example, our algorithm improves these "two-stage" methods by over 4.3%, 3.4%, 6.4%, 5.6%, 10.9%, 2.6%, 11.5%, and 2.9% on Protein Fold in terms of ACC (see Figure 1a). This is not surprising since the separated imputation may hurt the subsequent MKC, leading to unsatisfying performance.
- Our algorithm consistently and significantly improves the clustering performance of MKKM-IK, which is the first work to integrate imputation for MKC. This is due to the effectiveness of our objective and optimization technique.
- Our algorithm considerably exceeds EE-IMVC, which is assumed to be the most effective algorithm in handling with incomplete MKC. For example, the ACC of our algorithm is higher than EE-IMVC by over 1.5% with missing ratio 0.1 (see Figure 1a). Moreover, the improvement is more significant with the increase of missing ratios. Our algorithm also demonstrates superior clustering performance in terms of NMI and purity, as shown in Figure 1a.

5.2.2 *Results on UCI Digital*

UCI Digital has been widely used as a benchmark to evaluate the clustering performance of multiple kernel clustering algorithms. We also test the aforementioned algorithms on this dataset with different missing ratios, and report the results in Figure 1b. From these sub-figures, we observe that the newly proposed EE-IMVC significantly improves the clustering performance of existing MKKM variants, including the "one-stage" MKKM-IK [10]. However, our algorithm further considerably improves EE-IMVC in terms of ACC, NMI and purity under different missing ratios. For example, the ACC achieved by our algorithm is higher than that of EE-IMVC by over 8 percentages with missing ratio 0.1. Moreover, this superiority is consistent under different missing ratios, indicating the effectiveness of our algorithm in handling incomplete MKC.

5.2.3 *Results on Flower17 and Flower102*

In this section, we compare the clustering performance of the above-mentioned algorithms on Flower17 and Flower102, as plotted in sub-figures 1c and 1d. From these figures, it is clearly observed that the proposed algorithm significantly exceeds the second best one, i.e., EE-IMVC, with different missing ratios. Taking the results in Figure 1c for example, our algorithm improves EE-IMVC by 2.2%, 2.5%, 3.5%, 3.5%, 4.1%, 5.4%, 3.8%, 5.0% and 4.3%

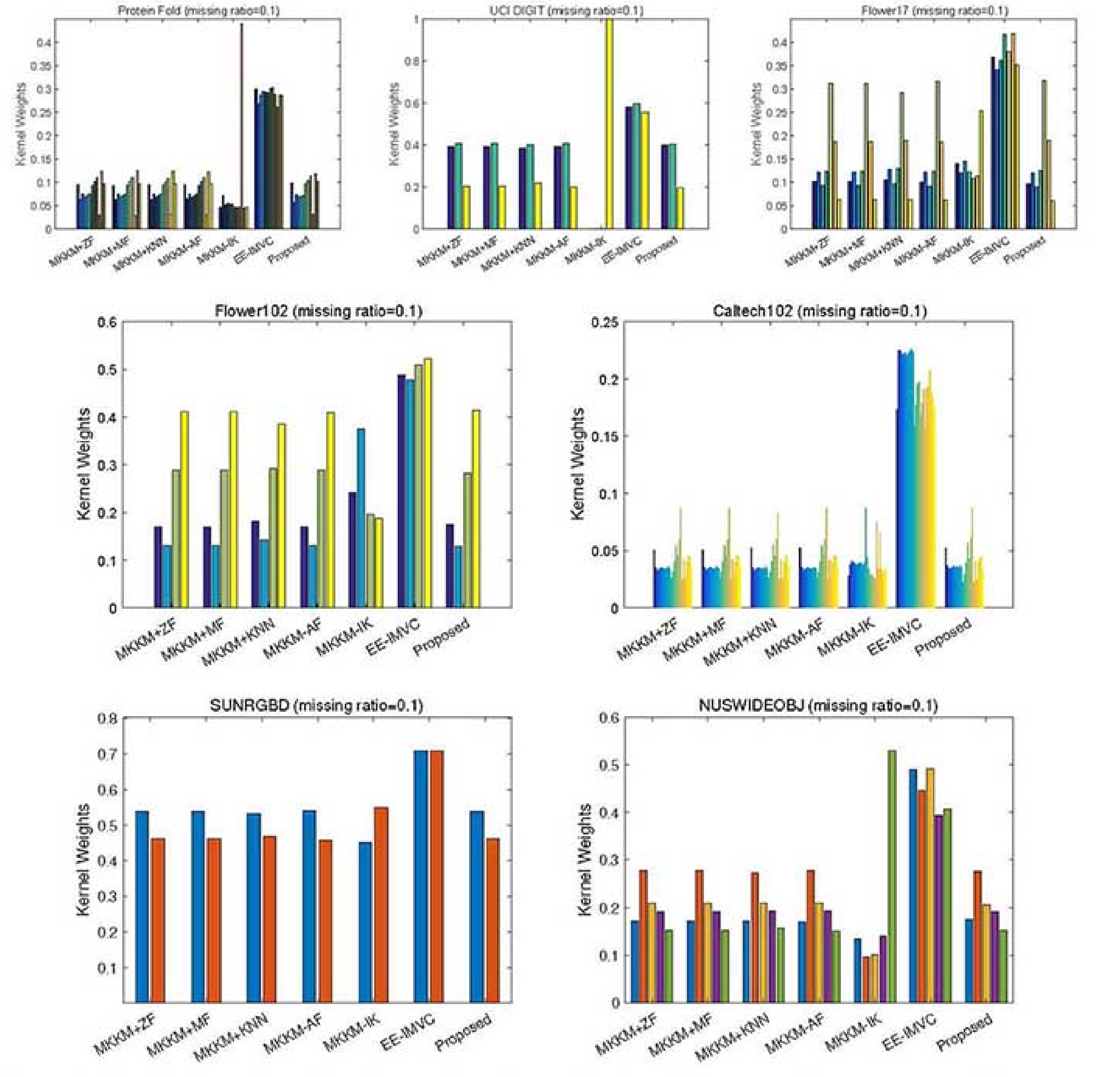

Fig. 5: The kernel weights learned by different algorithms on benchmark datasets with missing ratio 0.1. The proposed algorithm maintains reduced sparsity compared to several competitors. The kernel weights on other missing ratios are omitted due to space limit.

with different missing ratios in terms of ACC. Also, this improvement is similar in terms of NMI and purity. In addition, we can see that the proposed algorithm demonstrates comparable or slightly better clustering performance when compared with EE-IMVC on Flower102 with various missing ratios.

5.2.4 Results on Caltech102

We measure the clustering performance of the proposed algorithm on Caltech102, which is usually taken as a benchmark in the literature of multiple kernel clustering. The results are plotted in sub-figure 2a. It is observed from these sub-figures that the proposed algorithm consistently and significantly outperforms the compared one under different missing ratios, indicating its effectiveness.

5.2.5 Results on SUNRGBD and NUSWIDEOBJ

Finally, we evaluate the clustering performance of the proposed algorithm on two larger benchmark datasets, i.e., SUNRGBD [32] and NUSWIDEOBJ [33], and report the results in sub-figure 2b and 2c. As observed, the proposed algorithm demonstrates overall better or comparable clustering performance with the variation of missing ratios.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3116948, IEEE Transactions on Pattern Analysis and Machine Intelligence

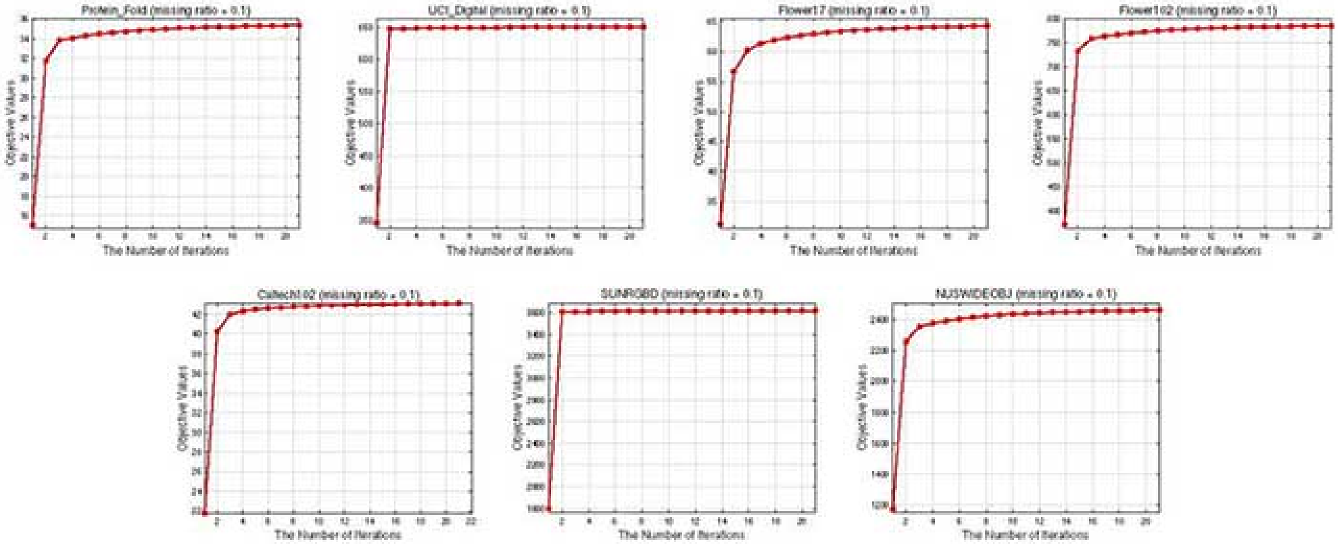

Fig. 6: The objective value of the proposed algorithm with iterations on all benchmark datasets.

5.2.6 *Overall Effectiveness Evaluation*

To illustrate the overall effectiveness of the compared algorithms against different missing ratios, we use the aggregated ACC, NMI and purity to evaluate the goodness of the algorithms. For example, the aggregated ACC is obtained by averaging the averaged ACC achieved by an algorithm over different εs. In addition, we adopt the paired *Student's t-test* to conduct a rigorous comparison, where a p-value smaller than 0.05 is considered statistically significant. The aggregated ACC, NMI and purity, and the standard deviation are reported in Table 2, where the one with the highest performance is shown in bold. Again, we observe that the proposed algorithm significantly outperforms the compared ones, which is consistent with our observations in Figure 1 and 2.

From the above experimental results, we attribute the superiority of our algorithm to: i) its effective objective and optimization, and ii) unifying imputation and clustering into a single procedure. On one hand, the effective objective and optimization contribute to learning of $\mathbf{H}$, which is in turn to guide the imputation of incomplete kernels. On the other hand, this meaningful imputation is able to better serve the MKC. These two learning processes negotiate with each other, leading to improved clustering performance. Differently, ZF+MKKM, MF+MKKM, KNN+MKKM and MKKM-AF algorithms do not considerably explore the connection between the imputation and clustering procedures. This could produce imputation that does not well serve the subsequent clustering as originally expected, affecting the clustering performance.

5.3 Evolution of the Learned $\mathbf{H}$

To investigate the clustering performance of the proposed algorithm with iterations, we take $\mathbf{H}$ at each iteration to calculate ACC, NMI and purity, and report them in Figure 4. We observe from each sub-figure that the starting point is consistently lower than the ending point. Taking the result in sub-figure 4c for example. The ACC at the first iteration is only 0.31. However, this value increases over 0.35 after several iterations. Moreover, the improvement is more significant with the increase of missing ratio. This clearly indicates the effectiveness and necessarity of joint imputation for MKC, especially in the presence of higher missing ratios. The figures in terms of NMI and purity are provided in the supplemental material due to space limit.

5.4 Kernel Weight Analysis

We next investigate the kernel weights learned by the compared algorithms. The results are plotted in Figure 5. We can see that the kernel weights learned by MKKM are extremely sparse on some datasets such as UCI-Digital, which is caused by the alternate optimization. This sparsity insufficiently exploits the multiple kernel matrices and explains the weak performance of MKKM. For example, the clustering accuracy of MKKM-IK on UCI-Digital is only 49.1% with missing ratio 0.1. However, despite the ℓ_1-norm constraint on γ, the kernel weights learned by our algorithm are all non-sparse on all datasets, which contributes to its superior clustering performance. This non-sparsity of the learned kernel weights is attributed to our new reduced gradient descent algorithm, which in turn is derived based on our new max-min-max kernel alignment objective.

5.5 Convergence

Though the convergence of the proposed algorithm cannot be theoretically guaranteed, we empirically observe that the objective value of our algorithm does monotonically increase with iterations, as shown in Figure 6. It usually converges in less than ten iterations on all datasets.

5.6 Running Time Comparison

Finally, we record the execution time of the aforementioned algorithms on all datasets, as reported in Figure 3. As observed, we can see that besides considerably improving the clustering performance, the proposed algorithm does not significantly increase the running time.

6 Conclusion

This paper proposes the incomplete multiple kernel alignment maximization for clustering to address incomplete MKC, where the kernel imputation and clustering are seamlessly integrated to achieve better clustering. The proposed algorithm effectively solves the resultant optimization problem, and it demonstrates significantly improved clustering performance via extensive experimental study. A generalization error bound is analyzed for the proposed algorithm. Many work is worth further exploration. For example, we plan to design a novel tri-level optimization framework to solve Eq. (9) more efficiently. In addition, we are going to further improve the clustering performance by considering the nonlinear transformation in Eq. (8).

Acknowledgements

We thank Prof. Timothy M. Hospedales and Prof. Marius Kloft for their constructive comments on revising the manuscript. This work was supported by the National Key R&D Program of China (project no. 2020AAA0107100) and the National Natural Science Foundation of China (project no. 61922088, 61906020 and 61773392).

References

[1] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola, "On kernel-target alignment," in *NIPS*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds., 2001, pp. 367–373.

[2] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1, pp. 131–159, Jan 2002.

[3] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for learning kernels based on centered alignment," *JMLR*, vol. 13, pp. 795–828, 2012.

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

[5] M. C. Trinidad, R. Martin-Brualla, F. Kainz, and J. Kontkanen, "Multi-view image fusion," in *ICCV 2019*, pp. 4100–4109.

[6] M. Hu and S. Chen, "One-pass incomplete multi-view clustering," in *AAAI 2019*, pp. 3838–3845.

[7] C. Xu, Z. Guan, W. Zhao, H. Wu, Y. Niu, and B. Ling, "Adversarial incomplete multi-view clustering," in *IJCAI 2019*, S. Kraus, Ed., pp. 3933–3939.

[8] X. Zhu, S. Zhang, W. He, R. Hu, C. Lei, and P. Zhu, "One-step multi-view spectral clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 10, pp. 2022–2034, 2019.

[9] X. Liu, E. Zhu, J. Liu, T. Hospedales, Y. Wang, and M. Wang, "SimpleMKKM: Simple multiple kernel k-means," *CoRR*, vol. abs/2005.04975, 2020.

[10] X. Liu, M. Li, L. Wang, Y. Dou, J. Yin, and E. Zhu, "Multiple kernel k-means with incomplete kernels," in *AAAI*, 2017, pp. 2259–2265.

[11] L. Yuan, Y. Wang, P. M. Thompson, V. A. Narayan, and J. Ye, "Multi-source feature learning for joint analysis of incomplete multiple heterogeneous neuroimaging data," *NeuroImage*, vol. 61, no. 3, pp. 622–632, 2012.

[12] J. T. Zhou, "Feature selection with multi-source transfer," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2021.

[13] X. Liu, X. Zhu, M. Li, L. Wang, E. Zhu, T. Liu, M. Kloft, D. Shen, J. Yin, and W. Gao, "Multiple kernel k-means with incomplete kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 5, pp. 1191–1204, 2020.

[14] X. Liu, X. Zhu, M. Li, L. Wang, C. Tang, J. Yin, D. Shen, H. Wang, and W. Gao, "Late fusion incomplete multi-view clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2410–2423, 2019.

[15] J. T. Zhou, M. Fang, H. Zhang, C. Gong, X. Peng, Z. Cao, and R. S. M. Goh, "Learning with annotation of various degrees," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2794–2804, 2019.

[16] X. Zhu, S. Zhang, Z. Jin, Z. Zhang, and Z. Xu, "Missing value estimation for mixed-attribute data sets," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 1, pp. 110–121, 2011.

[17] W. Shao, L. He, and S. Y. Philip, "Multiple incomplete views clustering via weighted nonnegative matrix factorization with $\ell_{2,1}$ regularization," in *PAKDD*. Springer, 2015, pp. 318–334.

[18] Z. Ghahramani and M. I. Jordan, "Supervised learning from incomplete data via an EM approach," in *NIPS*, 1993, pp. 120–127.

[19] A. Trivedi, P. Rai, H. Daumé III, and S. L. DuVall, "Multiview clustering with incomplete views," in *NIPS 2010: Machine Learning for Social Computing Workshop, Whistler, Canada*, 2010.

[20] S. Bhadra, S. Kaski, and J. Rousu, "Multi-view kernel completion," in *arXiv:1602.02518*, 2016.

[21] W. Shao, L. He, and P. S. Yu, "Multiple incomplete views clustering via weighted nonnegative matrix factorization with $\ell_{2,1}$ regularization," in *ECML PKDD*, 2015, pp. 318–334.

[22] X. Liu, M. Li, L. Wang, Y. Dou, J. Yin, and E. Zhu, "Multiple kernel k-means with incomplete kernels," in *AAAI*, S. P. Singh and S. Markovitch, Eds., pp. 2259–2265.

[23] C. Zhang, Y. Cui, Z. Han, J. T. Zhou, H. Fu, and Q. Hu, "Deep partial multi-view learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[24] Z. Huang, P. Hu, J. T. Zhou, J. Lv, and X. Peng, "Partially view-aligned clustering," *NIPS*, vol. 33, 2020.

[25] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, "lp-norm multiple kernel learning," *JMLR*, vol. 12, pp. 953–997, 2011.

[26] C. Cortes, M. Mohri, and A. Rostamizadeh, "L2 regularization for learning kernels," in *UAI*, 2009, pp. 109–116.

[27] R. Kumar, T. Chen, M. Hardt, D. Beymer, K. Brannon, and T. F. Syeda-Mahmood, "Multiple kernel completion and its application to cardiac disease discrimination," in *ISBI*. IEEE, 2013, pp. 764–767.

[28] M. Gönen and A. A. Margolin, "Localized data fusion for kernel k-means clustering with application to cancer biology," in *NIPS 2014*, pp. 1305–1313.

[29] Y. Liu, L. Fan, C. Zhang, T. Zhou, Z. Xiao, L. Geng, and D. Shen, "Incomplete multi-modal representation learning for alzheimer's disease diagnosis," *Medical Image Anal.*, vol. 69, p. 101953, 2021.

[30] P. Drineas and M. W. Mahoney, "On the nyström method for approximating a gram matrix for improved kernel-based learning," *J. Mach. Learn. Res.*, vol. 6, pp. 2153–2175, 2005.

[31] F. Nie, J. Yuan, and H. Huang, "Optimal mean robust principal component analysis," in *ICML*, vol. 32. JMLR.org, 2014, pp. 1062–1070.

[32] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *CVPR*, 2015, pp. 567–576.

[33] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng, "Nus-wide: A real-world web image database from national university of singapore," in *CIVR 09*.

[34] X. Liu, M. Li, C. Tang, J. Xia, J. Xiong, L. Liu, M. Kloft, and E. Zhu, "Efficient and effective regularized incomplete multi-view clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2634–2646, 2021.

[35] M. Hu and S. Chen, "Doubly aligned incomplete multi-view clustering," *arXiv preprint arXiv:1903.02785*, 2019.

[36] L. Lovász and M. D. Plummer, *Matching Theory*. Akadémiai Kiadó, North Holland, 1986.

Xinwang Liu received his PhD degree from National University of Defense Technology (NUDT), China. He is now Professor at School of Computer, NUDT. His current research interests include kernel learning and unsupervised feature learning. Dr. Liu has published 70+ peer-reviewed papers, including those in highly regarded journals and conferences such as IEEE T-PAMI, IEEE T-KDE, IEEE T-IP, IEEE T-NNLS, IEEE T-MM, IEEE T-IFS, ICML, NeurIPS, CVPR, ICCV, AAAI, IJCAI, etc. More information can be found at https://xinwangliu.github.io/.