Multiple Kernel Clustering

Bin Zhao*   James T. Kwok†   Changshui Zhang*

Abstract

Maximum margin clustering (MMC) has recently attracted considerable interests in both the data mining and machine learning communities. It first projects data samples to a kernel-induced feature space and then performs clustering by finding the maximum margin hyperplane over all possible cluster labelings. As in other kernel methods, choosing a suitable kernel function is imperative to the success of *maximum margin clustering*. In this paper, we propose a *multiple kernel clustering (MKC)* algorithm that simultaneously finds the maximum margin hyperplane, the best cluster labeling, and the optimal kernel. Moreover, we provide detailed analysis on the time complexity of the *MKC* algorithm and also extend *multiple kernel clustering* to the multi-class scenario. Experimental results on both toy and real-world data sets demonstrate the effectiveness and efficiency of the *MKC* algorithm.

1 Introduction

Over the decades, many clustering methods have been proposed in the literature, with popular examples including the *k-means clustering* [9], *mixture models* [9] and *spectral clustering* [4, 8, 21]. Recently, *maximum margin clustering (MMC)* has also attracted considerable interests in both the data mining and machine learning communities [26, 27, 28, 30, 31, 32]. The key idea of *MMC* is to extend the maximum margin principle of *support vector machines (SVM)* to the unsupervised learning scenario. Given a set of data samples, *MMC* performs clustering by labeling the samples such that the *SVM* margin obtained is maximized over all possible cluster labelings [27]. Recent studies have demonstrated its superior performance over conventional clustering methods.

However, while supervised large margin methods are usually formulated as convex optimization problems, *MMC* leads to a non-convex integer optimization problem which is much more difficult to solve. Recently, different optimization techniques have been used to alleviate this problem. Examples include *semi-definite programming (SDP)* [26, 27, 28], *alternating optimiza-*

tion [30] and the *cutting-plane* method [31, 32].

Moreover, like other kernel methods, *MMC* also relies on a kernel function to project the data samples to a high-dimensional kernel-induced feature space. A good choice of the kernel function is therefore imperative to the success of *MMC*. However, one of the central problems with kernel methods in general is that it is often unclear which kernel is the most suitable for a particular task [2, 5, 14, 17]. So, instead of using a single fixed kernel, recent developments in the *SVM* and other kernel methods have shown encouraging results in constructing the kernel from a number of homogeneous or even heterogeneous kernels [1, 10, 13, 14, 18, 33, 23, 24, 29]. This provides extra flexibility and also allows domain knowledge from possibly different information sources to be incorporated to the base kernels. However, previous works in this so-called *multiple kernel learning* approach have all been focused on the supervised and semi-supervised learning settings. Therefore, how to efficiently learn the kernel in unsupervised learning, or *maximum margin clustering* in particular, is still an interesting yet unexplored research topic.

In this paper, we propose a *multiple kernel clustering (MKC)* algorithm that finds the maximum margin hyperplane over all possible cluster labelings, together with the optimal kernel-induced feature map, automatically from the data. Specifically, we consider a non-negative combination of a given set of M feature maps $\Phi_1, \ldots, \Phi_M$ (corresponding to M base kernels $K_1, \ldots, K_M$):

$$(1.1) \qquad \Phi(\mathbf{x}) = \sum_{k=1}^{M} \beta_k \Phi_k(\mathbf{x}),$$

with $\beta_k \geq 0$ and $\sum_k \beta_k^p \leq 1$ for some integer p. By simultaneously optimizing the objective function in *MMC* with respect to both the hyperplane parameters (weight $\mathbf{w}$ and bias b) and the combination parameters β_k's, we can obtain the optimal feature mapping for *MMC*.

Computationally, the optimization problem in *multiple kernel clustering* can be solved by the *cutting plane* method [12]. As will be shown later in the sequel, one can construct a nested sequence of successively tighter relaxations of the original *MKC* problem, and each optimization problem in this sequence can be efficiently

---

*Department of Automation, Tsinghua University, China.

†Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong.

solved as a *second order cone program (SOCP)* [3] by using the *constrained concave-convex procedure* (*CCCP*) [22]. Experimental evaluations on toy and real-world data sets demonstrate both the effectiveness and efficiency of *multiple kernel clustering.*

The rest of this paper is organized as follows. In Section 2, we first present the principles of *multiple kernel clustering* on the simpler setting of two-class clustering. We will show that the original integer programming problem can be transformed to a sequence of convex programs which are then efficiently solved by a *cutting plane* algorithm. In Section 3, we provide theoretical analysis on the time complexity of the *MKC* algorithm. Section 4 extends *multiple kernel clustering* from the two-class to the multi-class setting. Experimental results on both toy and real-world data sets are provided in Section 5, followed by some concluding remarks in Section 6.

2 Multiple Kernel Clustering

In this section, we first present the *multiple kernel clustering* algorithm for two-class clustering. Extension to the multi-class case will be discussed in Section 4.

2.1 Maximum Margin Clustering As briefly introduced in Section 1, the key idea of *maximum margin clustering* (*MMC*) is to extend the maximum margin principle from supervised learning to unsupervised learning. In the two-cluster case, given a set of examples $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$, *MMC* aims at finding the best label combination $\mathbf{y} = \{y_1, \ldots, y_n\} \in \{-1, +1\}^n$ such that an *SVM* trained on this $\{(\mathbf{x}_i, y_i), \ldots, (\mathbf{x}_n, y_n)\}$ will yield the largest margin. Computationally, it solves the following optimization problem:

$$(2.2) \quad \min_{\mathbf{y} \in \{\pm 1\}^n} \min_{\mathbf{w}, b, \xi_i} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad \forall i \in \{1, \ldots, n\}:$$
$$y_i(\mathbf{w}^T\Phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \ \xi_i \geq 0,$$
$$-l \leq \sum_{i=1}^{n} y_i \leq l.$$

Here, the data samples $\mathbf{X}$ are mapped to a high-dimensional feature space using a possibly nonlinear feature mapping Φ. In the *support vector machine*, training is usually performed in the dual and this Φ is utilized implicitly by using the kernel trick. In cases where primal optimization with a nonlinear kernel is preferred, we can still obtain a finite-dimensional representation for each sample in the kernel-induced feature space by using *kernel principal component analysis* [20]. Alternatively, following [6], one can also compute the Cholesky

decomposition of the kernel matrix $\mathbf{K} = \hat{\mathbf{X}}\hat{\mathbf{X}}^T$, and set $\Phi(\mathbf{x}_i) = (\hat{\mathbf{X}}_{i,1}, \ldots, \hat{\mathbf{X}}_{i,n})^T$.

Moreover, the last constraint in (2.2) is the class balance constraint, which is introduced to avoid the trivially "optimal" solution that assigns all patterns to the same class and thus achieves "infinite" margin. This class balance constraint also avoids the unwanted solution of separating a single outlier or a very small group of samples from the rest of the data. Here, $l > 0$ is a constant controlling the class imbalance.

According to Eq.(2.2), *maximum margin clustering* maximizes the margin with respect to both the labeling vector $\mathbf{y}$ and the separating hyperplane parameters $(\mathbf{w}, b)$. The unknown binary vector $\mathbf{y}$ renders Eq.(2.2) an integer program, which is much more difficult to solve than the quadratic program (QP) in *SVM*. However, as shown in [31], we can equivalently formulate the *maximum margin clustering* problem as

$$(2.3) \quad \min_{\mathbf{w}, b, \xi_i} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad \forall i \in \{1, \ldots, n\}:$$
$$|\mathbf{w}^T\Phi(\mathbf{x}_i) + b| \geq 1 - \xi_i, \ \xi_i \geq 0,$$
$$-l \leq \sum_{i=1}^{n} \left[\mathbf{w}^T\Phi(\mathbf{x}_i) + b\right] \leq l.$$

Here, the labeling vector $\mathbf{y}$ is computed as $y_i = \text{sgn}(\mathbf{w}^T\phi(\mathbf{x}_i) + b)$ and a slightly relaxed class balance constraint is used [21]. This is much easier to handle than the original one in Eq.(2.2).

2.2 Multiple Kernel Maximum Margin Clustering Traditionally, *maximum margin clustering* projects the data samples to the feature space by a fixed feature mapping Φ (which is induced by a kernel K). Choosing a suitable kernel is therefore imperative to the success of *maximum margin clustering*. However, it is often unclear which kernel is the most suitable for the task at hand. In this paper, inspired by the works of *multiple kernel learning* in supervised learning [1, 10, 13, 14, 18, 33, 23, 24, 29], we propose to use a non-negative combination of several base kernels for computing the feature map in this *maximum margin clustering* setting.

Specifically, each data sample $\mathbf{x}_i$ in the input space is translated via M mappings $\Phi_k : \mathbf{x} \mapsto \Phi_k(\mathbf{x}) \in \mathbb{R}^{D_k}$, $k = 1, \ldots, M$, to M feature representations $\Phi_1(\mathbf{x}_i), \ldots, \Phi_M(\mathbf{x}_i)$. Here, D_k denotes the dimensionality of the kth feature space. For each feature mapping, there is a separate weight vector $\mathbf{w}_k$. Then one solves the following optimization problem, which is equivalent

to the *MMC* formulation in Eq.(2.3) when $M = 1$:

$$(2.4) \quad \min_{\boldsymbol{\beta},\mathbf{w},b,\xi} \quad \frac{1}{2}\sum_{k=1}^{M}\beta_k||\mathbf{w}_k||^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$

$$\text{s.t.} \quad \forall i \in \{1,\ldots,n\}:$$

$$\left|\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i)+b\right| \geq 1-\xi_i,\ \xi_i \geq 0,$$

$$\forall k \in \{1,\ldots,M\}:\ \beta_k \geq 0,$$

$$\sum_{k=1}^{M}\beta_k^p \leq 1,$$

$$-l \leq \sum_{i=1}^{n}\left[\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i)+b\right] \leq l.$$

Here, we regularize the M output functions according to their weights β_k's. The non-negativity constraints on the weights guarantee that the combined regularizer is convex, and the resulting kernel is positive semi-definite. Moreover, p here is a positive integer. In this paper, we choose $p = 2$ or, in other words, the ℓ_2 regularizer is used on $\boldsymbol{\beta} = (\beta_1,\ldots,\beta_M)^T$.

While Eq.(2.4) is quite intuitive, it has the disadvantage that both the objective function and the first and last constraints are non-convex due to the coupling of β_k and $\mathbf{w}_k$ in the output function. Therefore, we apply the following change of variables [33]

$$(2.5) \quad \forall k \in \{1,\ldots,M\}:\ \mathbf{v}_k = \beta_k\mathbf{w}_k.$$

After the above change of variables, *multiple kernel MMC* is equivalently formulated as follows.

$$(2.6) \quad \min_{\boldsymbol{\beta},\mathbf{v},b,\xi} \quad \frac{1}{2}\sum_{k=1}^{M}\frac{||\mathbf{v}_k||^2}{\beta_k} + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$

$$\text{s.t.} \quad \forall i \in \{1,\ldots,n\}:$$

$$\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right| \geq 1-\xi_i,\ \xi_i \geq 0,$$

$$\forall k \in \{1,\ldots,M\}:\ \beta_k \geq 0,$$

$$\sum_{k=1}^{M}\beta_k^2 \leq 1,$$

$$-l \leq \sum_{i=1}^{n}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right] \leq l,$$

where $\mathbf{v} = (\mathbf{v}_1,\ldots,\mathbf{v}_M)^T$. Note that the objective function and all constraints except the first one are now convex.

2.3 Cutting Plane Algorithm The *multiple kernel MMC* formulation in Eq.(2.6) has n slack variables

ξ_i's, one for each data sample. In the following, we first reformulate Eq.(2.6) to reduce the number of slack variables.

THEOREM 2.1. *Multiple kernel MMC can be equivalently formulated as:*

$$(2.7) \quad \min_{\boldsymbol{\beta},\mathbf{v},b,\xi} \quad \frac{1}{2}\sum_{k=1}^{M}\frac{||\mathbf{v}_k||^2}{\beta_k} + C\xi$$

$$\text{s.t.} \quad \forall \mathbf{c} \in \{0,1\}^n:$$

$$(2.8) \quad \frac{1}{n}\sum_{i=1}^{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right| \geq \frac{1}{n}\sum_{i=1}^{n}c_i-\xi,$$

$$\forall k \in \{1,\ldots,M\}:\ \beta_k \geq 0,$$

$$\sum_{k=1}^{M}\beta_k^2 \leq 1,\ \xi \geq 0,$$

$$-l \leq \sum_{i=1}^{n}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right] \leq l.$$

Proof. For simplicity, we denote the optimization problem shown in Eq.(2.6) as OP1 and the problem in Eq.(2.7) as OP2. To prove the theorem, we will show that OP1 and OP2 have the same optimal objective value and an equivalent set of constraints. Specifically, we will prove that for every $(\mathbf{v},b,\boldsymbol{\beta})$, the optimal ξ^* and $\{\xi_1^*,\ldots,\xi_n^*\}$ are related by $\xi^* = \frac{1}{n}\sum_{i=1}^{n}\xi_i^*$. This means that, with $(\mathbf{v},b,\boldsymbol{\beta})$ fixed, $(\mathbf{v},b,\boldsymbol{\beta},\xi^*)$ and $(\mathbf{v},b,\boldsymbol{\beta},\xi_1^*,\ldots,\xi_n^*)$ are optimal solutions to OP1 and OP2, respectively, and they result in the same objective value.

First, note that for any given $(\mathbf{v},b,\boldsymbol{\beta})$, each slack variable ξ_i in OP1 can be optimized individually as

$$(2.9) \quad \xi_i^* = \max\left\{0, 1-\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\right\}.$$

For OP2, the optimal slack variable ξ is
$$(2.10)$$
$$\xi^* = \max_{\mathbf{c}\in\{0,1\}^n}\left\{\frac{1}{n}\sum_{i=1}^{n}c_i - \frac{1}{n}\sum_{i=1}^{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\right\}.$$

Since the c_i's are independent of each other in Eq.(2.10), they can also be optimized individually and so

$$(2.11) \quad \xi^* = \sum_{i=1}^{n}\max_{c_i\in\{0,1\}}\left\{\frac{1}{n}c_i - \frac{1}{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\right\}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\max\left\{0, 1-\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\right\}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\xi_i^*.$$

Hence, the objectives of OP1 and OP2 have the same value for any $(\mathbf{v}, b, \boldsymbol{\beta})$ given the optimal ξ^* and $\{\xi_1^*, \ldots, \xi_n^*\}$. Therefore, the optima of these two optimization problems are the same. That is to say, we can solve the optimization problem in Eq.(2.7) to get the *multiple kernel MMC* solution. □

In the optimization problem shown in Eq.(2.7), the number of slack variables is reduced by $n-1$ and a single slack variable ξ is now shared across all the non-convex constraints. This greatly reduces the complexity of the non-convex optimization problem for *multiple kernel MMC*. On the other hand, the number of constraints in Eq.(2.8) is increased from n to 2^n. This exponential increase of constraints may seem intimidating at first sight. However, we will show that we can always find a small subset of constraints from the whole constraint set in (2.8) while still ensuring a sufficiently accurate solution. Specifically, we employ an adaptation of the *cutting plane* algorithm [12] to solve the *multiple kernel MMC* problem. It starts with an empty constraint subset Ω, and computes the optimal solution to problem (2.7) subject to the constraints in Ω. The algorithm then finds the most violated constraint in (2.8) and adds it to the subset Ω. In this way, we construct a series of successively tightening approximations to the original *multiple kernel MMC* problem. The algorithm stops when no constraint in (2.8) is violated by more than ϵ. The whole *cutting plane* algorithm for *multiple kernel MMC* is presented in Algorithm 1.

Algorithm 1 Cutting plane algorithm for multiple kernel maximum margin clustering.

Input: M feature mappings $\Phi_1, \ldots, \Phi_M$, parameters C, l and ϵ, constraint subset $\Omega = \phi$.
repeat
  Solve problem (2.7) for $(\mathbf{v}, b, \boldsymbol{\beta})$ under the current working constraint set Ω.
  Select the most violated constraint $\mathbf{c}$ and set $\Omega = \Omega \cup \{\mathbf{c}\}$.
until the newly selected constraint $\mathbf{c}$ is violated by no more than ϵ.

We will prove in Section 3 that one can always find a polynomially-sized subset of constraints such that the solution of the corresponding relaxed problem satisfies all the constraints in (2.8) up to a precision of ϵ. That is to say, the remaining exponential number of constraints are guaranteed to be violated by no more than ϵ, and thus do not need to be explicitly added to the optimization problem [11].

There are two remaining issues in our *cutting plane* algorithm for *multiple kernel MMC*. First, how to solve

problem (2.7) under a given constraint subset Ω? Second, how to find of the most violated constraint in (2.8)? These will be addressed in the following two subsections.

2.3.1 Optimization via the CCCP In each iteration of the *cutting plane* algorithm, we need to solve a non-convex optimization problem to obtain the optimal separating hyperplane under the current working constraint set Ω. Although the objective function in (2.7) is convex, the constraints are not. This makes problem (2.7) difficult to solve. Fortunately, the *constrained concave-convex procedure (CCCP)* is designed to solve these optimization problems with a concave-convex objective function and concave-convex constraints [22]. Specifically, the objective function in (2.7) is quadratic and all the constraints except the first one are linear. Moreover, note that although the constraint in Eq.(2.8) is non-convex, it is a difference of two convex functions which can be written as:

$$(2.12) \quad \forall \mathbf{c} \in \Omega :$$
$$\left(\frac{1}{n} \sum_{i=1}^{n} c_i - \xi \right) - \frac{1}{n} \sum_{i=1}^{n} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \leq 0.$$

Hence, we can solve problem (2.7) with the *CCCP* as follows. Given an initial estimate $(\mathbf{v}^{(0)}, b^{(0)})$, the *CCCP* computes $(\mathbf{v}^{(t+1)}, b^{(t+1)})$ from $(\mathbf{v}^{(t)}, b^{(t)})$ by replacing $\frac{1}{n} \sum_{i=1}^{n} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right|$ in the constraint (2.12) with its first-order Taylor expansion at $(\mathbf{v}^{(t)}, b^{(t)})$. Problem (2.7) then becomes:

$$(2.13) \quad \min_{\boldsymbol{\beta}, \mathbf{v}, b, \xi} \quad \frac{1}{2} \sum_{k=1}^{M} \frac{||\mathbf{v}_k||^2}{\beta_k} + C\xi$$
$$\text{s.t.} \quad \forall \mathbf{c} \in \Omega :$$
$$\frac{1}{n} \sum_{i=1}^{n} c_i \leq \xi + \frac{1}{n} \sum_{i=1}^{n} c_i z_i^{(t)} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right],$$
$$\forall k \in \{1, \ldots, M\} : \beta_k \geq 0,$$
$$\sum_{k=1}^{M} \beta_k^2 \leq 1, \; \xi \geq 0,$$
$$-l \leq \sum_{i=1}^{n} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right] \leq l,$$

where $z_i^{(t)} = \mathrm{sgn}\left(\sum_{k=1}^{M} \mathbf{v}_k^{(t)T} \Phi_k(\mathbf{x}_i) + b^{(t)} \right)$. Introducing additional variable t_k defined as the upper bound of $\frac{||\mathbf{v}_k||^2}{\beta_k}$ (i.e., adding additional constraints $\frac{||\mathbf{v}_k||^2}{\beta_k} \leq t_k$), we can formulate the above as the following *second order*

cone programming (SOCP) [3] problem:

$$(2.14)\min_{\boldsymbol{\beta},\mathbf{v},b,\xi,\mathbf{t}} \quad \frac{1}{2}\sum_{k=1}^{M} t_k + C\xi$$

$$\text{s.t.} \quad \forall \mathbf{c} \in \Omega:$$

$$\frac{1}{n}\sum_{i=1}^{n} c_i \leq \xi + \frac{1}{n}\sum_{i=1}^{n} c_i z_i^{(t)}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right],$$

$$\forall k \in \{1,\ldots,M\}:$$

$$\left\|\begin{bmatrix} 2\mathbf{v}_k \\ t_k - \beta_k \end{bmatrix}\right\| \leq t_k + \beta_k, \; \beta_k \geq 0,$$

$$\sum_{k=1}^{M}\beta_k^2 \leq 1, \; \xi \geq 0$$

$$-l \leq \sum_{i=1}^{n}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right] \leq l.$$

Here, we have used the fact that hyperbolic constraints of the form $\mathbf{s}^T\mathbf{s} \leq xy$, where $x,y \in \mathbb{R}_+$ and $\mathbf{s} \in \mathbb{R}^n$, can be equivalently transformed to the second order cone constraint [16, 25]

$$(2.15) \qquad \left\|\begin{bmatrix} 2\mathbf{s} \\ x-y \end{bmatrix}\right\| \leq x+y.$$

The above *SOCP* problem can be solved in polynomial time [15]. Following the *CCCP*, the obtained solution $(\mathbf{v},b,\boldsymbol{\beta},\xi,\mathbf{t})$ from this *SOCP* problem is then used as $(\mathbf{v}^{(t+1)},b^{(t+1)},\boldsymbol{\beta},\xi,\mathbf{t})$, and the iteration continues until convergence. The algorithm for solving problem (2.7) subject to the constraint subset Ω is summarized in Algorithm 2. As for its termination criterion, we check if the difference in objective values from two successive iterations is less than $\alpha\%$ (which is set to 0.01 in the experiments).

Algorithm 2 Solve problem (2.7) subject to constraint subset Ω via the constrained concave-convex procedure.

> Initialize $(\mathbf{v}^{(0)},b^{(0)})$.
> **repeat**
> > Obtain $(\mathbf{v}^{(t+1)},b^{(t+1)},\boldsymbol{\beta},\xi,\mathbf{t})$ as the solution of the *second order cone programming* problem (2.14).
> > Set $\mathbf{v} = \mathbf{v}^{(t+1)}$, $b = b^{(t+1)}$ and $t = t+1$.
> **until** the stopping criterion is satisfied.

2.3.2 The Most Violated Constraint The most violated constraint in (2.8) can be easily identified. Recall that the feasibility of a constraint in (2.8) is measured by the corresponding value of ξ. Therefore, the most violated constraint is the one that results in the largest ξ. Since each constraint in (2.8) is represented by a vector $\mathbf{c}$, we have the following theorem:

THEOREM 2.2. *The most violated constraint* $\mathbf{c}$ *in (2.8) can be computed as:*

$$(2.16) \qquad c_i = \begin{cases} 1 & \text{if } \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right| < 1, \\ 0 & \text{otherwise.} \end{cases}$$

Proof. The most violated constraint is the one that results in the largest ξ. In order to fulfill all the constraints in problem (2.7), the optimal ξ can be computed as:

$$(2.17)\xi^* = \sum_{i=1}^{n}\max_{c_i\in\{0,1\}}\left\{\frac{1}{n}c_i - \frac{1}{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\right\}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\max_{c_i\in\{0,1\}}\left\{c_i\left[1-\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\right]\right\}.$$

Therefore, the most violated constraint $\mathbf{c}$ corresponding to ξ^* can be obtained as in Eq.(2.16). □

The *cutting plane* algorithm iteratively selects the most violated constraint under the current hyperplane parameter and then adds it to the working constraint set Ω, until no constraint is violated by more than ϵ. Moreover, there is a direct correspondence between ξ and the feasibility of the set of constraints in problem (2.7). If a point $(\mathbf{v},b,\boldsymbol{\beta},\xi)$ fulfills all the constraints up to precision ϵ, i.e.,

$$(2.18) \forall \mathbf{c} \in \{0,1\}^n:$$

$$\frac{1}{n}\sum_{i=1}^{n} c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right| \geq \frac{1}{n}\sum_{i=1}^{n} c_i - (\xi+\epsilon),$$

then the point $(\mathbf{v},b,\boldsymbol{\beta},\xi+\epsilon)$ is feasible. Furthermore, note that in the objective function of problem (2.7), there is a single slack variable ξ measuring the clustering loss. Hence, we can simply select the stopping criterion in Algorithm 1 as being all the samples satisfying inequality (2.18). Then, the approximation accuracy ϵ of this approximate solution is directly related to the clustering loss.

2.4 Accuracy of the Cutting Plane Algorithm
The following theorem characterizes the accuracy of the solution computed by the *cutting plane* algorithm.

THEOREM 2.3. *For any* $\epsilon > 0$, *the cutting plane algorithm for multiple kernel MMC returns a point* $(\mathbf{v},b,\boldsymbol{\beta},\xi)$ *for which* $(\mathbf{v},b,\boldsymbol{\beta},\xi+\epsilon)$ *is feasible in problem (2.7).*

Proof. In the *cutting plane* algorithm, the most violated constraint $\mathbf{c}$ in (2.8), which leads to the largest value of ξ, is selected using Eq.(2.16). The *cutting*

plane algorithm terminates only when the newly selected constraint **c** is violated by no more than ϵ, i.e., $\frac{1}{n}\sum_{i=1}^{n} c_i - \frac{1}{n}\sum_{i=1}^{n} c_i \left|\sum_{k=1}^{M} \mathbf{v}_k \Phi_k(\mathbf{x}_i) + b\right| \leq \xi + \epsilon$. Since the newly selected constraint **c** is the most violated one, all the other constraints will satisfy the above inequality. Therefore, if $(\mathbf{v}, b, \boldsymbol{\beta}, \xi)$ is the solution returned by our *cutting plane* algorithm, then $(\mathbf{v}, b, \boldsymbol{\beta}, \xi+\epsilon)$ will be a feasible solution to problem (2.7). □

Based on this theorem, ϵ indicates how close one wants to be to the error rate of the best separating hyperplane. This justifies its use as the stopping criterion in Algorithm 1.

3 Time Complexity Analysis

In this section, we provide theoretical analysis on the time complexity of the *cutting plane* algorithm for *multiple kernel MMC*.

THEOREM 3.1. *The cutting plane algorithm for multiple kernel MMC takes* $O(\frac{D^{3.5}+nD}{\epsilon^2} + \frac{D^{2.5}}{\epsilon^4})$ *time, where* $D = \sum_{k=1}^{M} D_k$ *and* D_k *is the dimensionality of the kth feature space.*

To prove the above theorem, we will first obtain the time involved in each iteration of the algorithm. Next, we will prove that the total number of constraints added into the working set Ω, i.e., the total number of iterations involved in the *cutting plane* algorithm, is upper bounded. Specifically, we have the following two lemmas.

LEMMA 3.1. *Each iteration of the cutting plane algorithm for multiple kernel MMC takes* $O(D^{3.5} + nD + D^{2.5}|\Omega|)$ *time for a working constraint set size* $|\Omega|$.

Proof. In each iteration of the *cutting plane* algorithm, two steps are involved: solving problem (2.7) under the current working constraint set Ω via *CCCP* and selecting the most violated constraint. To solve problem (2.7) under the working constraint set Ω, we will need to solve a sequence of *SOCP* problems. Specifically, for an *SOCP* problem of the form

$$(3.19)\min_{\mathbf{x}} \mathbf{f}^T\mathbf{x}$$
$$s.t. \ \forall k\in\{1,\ldots,M\}: ||\mathbf{A}_k\mathbf{x}+\mathbf{b}_k||\leq\mathbf{c}_k^T\mathbf{x}+d_k,$$

where $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{f} \in \mathbb{R}^N$, $\mathbf{A}_k \in \mathbb{R}^{(N_k-1)\times N}$, $\mathbf{b}_k \in \mathbb{R}^{N_k-1}$, $\mathbf{c}_k \in \mathbb{R}^N$ and $d_k \in \mathbb{R}$, then its time complexity for each iteration is $O(N^2 \sum_k N_k)$ [15, 25]. According to the *SOCP* formulation in (2.14), we have $N = \sum_{k=1}^{M} D_k + 2M + 2 = O(D)$ and $\sum_k N_k = \sum_{k=1}^{M}(D_k + 2) + 2M +$

$|\Omega| + 3 = O(D + |\Omega|)$. Thus, the time complexity per iteration is $O(D^3 + D^2|\Omega|)$. Using the primal-dual method for solving this *SOCP*, the accuracy of a given solution can be improved by an absolute constant factor in $O(D^{0.5})$ iterations [16]. Hence, each iteration in the *CCCP* takes $O(D^{3.5} + D^{2.5}|\Omega|)$ time. Moreover, as will be seen from the numerical experiments in Section 5, each round of the *cutting plane* algorithm requires fewer than 10 iterations for solving problem (2.7) subject to Ω via *CCCP*. This is the case even on large data sets. Therefore, the time complexity for solving problem (2.7) under the working constraint set Ω via *CCCP* is $O(D^{3.5} + D^{2.5}|\Omega|)$. Finally, to select the most violated constraint using Eq.(2.16), we need to compute n inner products between $(\mathbf{v}_1,\ldots,\mathbf{v}_M)$ and $(\Phi_1(\mathbf{x}_i),\ldots,\Phi_M(\mathbf{x}_i))$. Each inner product takes $O(D)$ time and so a total of n inner products can be computed in $O(nD)$ time. Thus, the time complexity for each iteration of the *cutting plane* algorithm is $O(D^{3.5} + nD + D^{2.5}|\Omega|)$. □

LEMMA 3.2. *The cutting plane algorithm terminates after adding at most* $\frac{CR}{\epsilon^2}$ *constraints, where R is a constant independent of n and D.*

Proof. Note that $\mathbf{v} = \mathbf{0}$, $b = 0$, $\xi = 1$ with arbitrary $\boldsymbol{\beta} \geq 0$ satisfying $\sum_{k=1}^{M} \beta_k^2 \leq 1$ is a feasible solution to problem (2.7). Therefore, the optimal objective of (2.7) is upper bounded by C. In the following, we will prove that in each iteration of the *cutting plane* algorithm, the objective value will be increased by at least a constant after adding the most violated constraint. Due to the fact that the objective value is non-negative and has upper bound C, the total number of iterations will be upper bounded. For simplicity, we omit the class balance constraint in problem (2.7) and set the bias term $b = 0$. The proof for the problem with class balance constraint and non-zero bias term can be obtained similarly.

To compute the increase brought about by adding one constraint to the working constraint set Ω, we will first need to present the dual problem of (2.7). The difficulty involved in obtaining this dual problem comes from the $|\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i)+b|$ term in the constraints. Thus, we will first replace the constraints in (2.8) with

$$\forall \mathbf{c} \in \Omega : \ \frac{1}{n}\sum_{i=1}^{n} c_i t_i \geq \frac{1}{n}\sum_{i=1}^{n} c_i - \xi,$$

$$\forall i \in \{1,\ldots,n\} : \ t_i^2 \leq \mathbf{v}^T \boldsymbol{\Psi}_i \mathbf{v},$$

$$\forall i \in \{1,\ldots,n\} : \ t_i \geq 0,$$

where the $D \times D$ matrix $\boldsymbol{\Psi}_i$ is defined as

$$(3.20) \quad \boldsymbol{\Psi}_i = \begin{bmatrix} \Phi_1(\mathbf{x}_i)\Phi_1^T(\mathbf{x}_i) & \dots & \Phi_1(\mathbf{x}_i)\Phi_M^T(\mathbf{x}_i) \\ \vdots & \ddots & \vdots \\ \Phi_M(\mathbf{x}_i)\Phi_1^T(\mathbf{x}_i) & \dots & \Phi_M(\mathbf{x}_i)\Phi_M^T(\mathbf{x}_i) \end{bmatrix}.$$

Let $\boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho$ be the dual variables corresponding to the various constraints, the *Lagrangian dual function* for problem (2.7) can be obtained as

$$(3.21) \quad L(\boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho)$$

$$= \inf_{\mathbf{v}, \boldsymbol{\beta}, \xi, t} \left\{ \frac{1}{2}\sum_{k=1}^{M}\frac{||\mathbf{v}_k||^2}{\beta_k} + C\xi + \sum_{p=1}^{|\Omega|}\lambda_p\left[\frac{1}{n}\sum_{i=1}^{n}c_{pi}(1-t_i)-\xi\right] \right.$$

$$+ \sum_{i=1}^{n}\gamma_i(t_i^2 - \mathbf{v}^T\boldsymbol{\Psi}_i\mathbf{v}) - \mu\xi - \sum_{i=1}^{n}\delta_i t_i - \sum_{k=1}^{M}\alpha_k\beta_k$$

$$\left. + \rho\left(\sum_{k=1}^{M}\beta_k - 1\right) \right\}$$

$$= \inf_{\mathbf{v}, \boldsymbol{\beta}, \xi, t} \left\{ \frac{1}{2}\sum_{k=1}^{M}\frac{||\mathbf{v}_k||^2}{\beta_k} - \mathbf{v}^T\sum_{i=1}^{n}\gamma_i\boldsymbol{\Psi}_i\mathbf{v} + C\xi - \sum_{p=1}^{|\Omega|}\lambda_p\xi - \mu\xi \right.$$

$$+ \sum_{i=1}^{n}\gamma_i t_i^2 - \sum_{p=1}^{|\Omega|}\lambda_p\frac{1}{n}\sum_{i=1}^{n}c_{pi}t_i - \sum_{i=1}^{n}\delta_i t_i + \sum_{p=1}^{|\Omega|}\lambda_p\frac{1}{n}\sum_{i=1}^{n}c_{pi}$$

$$\left. - \sum_{k=1}^{M}\alpha_k\beta_k + \rho\left(\sum_{k=1}^{M}\beta_k - 1\right) \right\}$$

$$= \sum_{i=1}^{n}\left\{ -\frac{(\sum_{p=1}^{|\Omega|}\lambda_p c_{pi} + n\delta_i)^2}{4n^2\gamma_i} + \frac{1}{n}\sum_{p=1}^{|\Omega|}\lambda_p c_{pi} \right\} - \rho$$

satisfying the following constraints

$$(3.22) \quad \begin{cases} \mathbf{E}_\beta - 2\sum_{i=1}^{n}\gamma_i\boldsymbol{\Psi}_i \succeq 0, \\ \alpha_k - \rho = 0, \\ C - \sum_{p=1}^{|\Omega|}\lambda_p - \mu = 0, \\ t_i = \frac{1}{2n\gamma_i}\sum_{k=1}^{|\Omega|}\lambda_k c_{ki} + \frac{\delta_i}{2\gamma_i}, \\ \boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha} \geq 0, \end{cases}$$

where $\mathbf{E}_\beta = \mathbf{diag}\left(\frac{\mathbf{I}_{D_1 \times D_1}}{\beta_1}, \dots, \frac{\mathbf{I}_{D_M \times D_M}}{\beta_M}\right)$ and $\mathbf{I}_{D_k \times D_k}$ is the $D_k \times D_k$ identity matrix.

The *cutting plane* algorithm selects the most violated constraint $\mathbf{c}'$ and continues if the following inequality holds

$$(3.23) \quad \frac{1}{n}\sum_{i=1}^{n}c_i'(1-t_i^*) \geq \xi + \epsilon.$$

Since $\xi \geq 0$, the newly added constraint satisfies

$$(3.24) \quad \frac{1}{n}\sum_{i=1}^{n}c_i'(1-t_i^*) \geq \epsilon.$$

Let $L_*^{(t+1)}$ be the optimal value of the *Lagrangian dual function* subject to $\Omega^{(t+1)} = \Omega^{(t)} \cup \{\mathbf{c}'\}$, and $\gamma_i^{(t)}$ be the value of γ_i which results in the largest $L_*^{(t)}$. The addition of a new constraint to the primal problem is equivalent to adding a new variable λ_{t+1} to the dual problem, and so

$$(3.25) \quad L_*^{(t+1)}$$

$$= \max_{\boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho} \sum_{i=1}^{n}\left\{ -\frac{(\sum_p\lambda_p c_{pi} + \lambda_{t+1}c_i' + n\delta_i)^2}{4n^2\gamma_i} \right.$$

$$\left. + \frac{1}{n}\left[\sum_{p=1}^{t}\lambda_p c_{pi} + \lambda_{t+1}c_i'\right] \right\} - \rho$$

$$\geq L_*^{(t)} + \max_{\lambda_{t+1}\geq 0}\sum_{i=1}^{n}\left\{ -\frac{\lambda_{t+1}c_i'\sum_{p=1}^{t}\lambda_p^{(t)}c_{pi}}{2\gamma_i^{(t)}n^2} \right.$$

$$\left. - \frac{\lambda_{t+1}c_i'\delta_i^{(t)}}{2\gamma_i^{(t)}n} - \frac{(\lambda_{t+1}c_i')^2}{4\gamma_i^{(t)}n^2} + \frac{1}{n}\lambda_{t+1}c_i' \right\}.$$

According to inequality (3.24) and the constraint $\lambda_{t+1} \geq 0$, we have

$$\sum_{i=1}^{n}\left[\frac{\lambda_{t+1}c_i'\sum_{p=1}^{t}\lambda_p^{(t)}c_{pi}}{2\gamma_i^{(t)}n^2} + \frac{\lambda_{t+1}c_i'\delta_i^{(t)}}{2\gamma_i^{(t)}n}\right] \leq \frac{1}{n}\sum_{i=1}^{n}\lambda_{t+1}c_i' - \epsilon\lambda_{t+1}.$$

Substituting the above inequality into (3.25), we get the following lower bound of $L_*^{(t+1)}$:

$$(3.26) \quad L_*^{(t+1)} \geq L_*^{(t)} + \max_{\lambda_{t+1}\geq 0}\left\{ -\frac{1}{n}\sum_{i=1}^{n}\lambda_{t+1}c_i' + \epsilon\lambda_{t+1} \right.$$

$$\left. - \sum_{i=1}^{n}\frac{(\lambda_{t+1}c_i')^2}{4\gamma_i^{(t)}n^2} + \sum_{i=1}^{n}\frac{1}{n}\lambda_{t+1}c_i' \right\}$$

$$= L_*^{(t)} + \max_{\lambda_{t+1}\geq 0}\left\{ \epsilon\lambda_{t+1} - \sum_{i=1}^{n}\frac{(\lambda_{t+1}c_i')^2}{4\gamma_i^{(t)}n^2} \right\}$$

$$= L_*^{(t)} + \frac{\epsilon^2}{\sum_{i=1}^{n}(c_i'^2/\gamma_i^{(t)}n^2)}.$$

By maximizing the *Lagrangian dual function* shown in Eq.(3.21), $\gamma^{(t)}$ can be obtained as:

$$(\boldsymbol{\lambda}^{(t)}, \boldsymbol{\gamma}^{(t)}, \mu^{(t)}, \boldsymbol{\delta}^{(t)}, \boldsymbol{\alpha}^{(t)}, \rho^{(t)})$$

$$= \arg\max_{\boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho} \sum_{i=1}^{n}\left\{ -\frac{(\sum_{p=1}^{t}\lambda_p c_{pi} + n\delta_i)^2}{4n^2\gamma_i} + \frac{1}{n}\sum_{p=1}^{t}\lambda_p c_{pi} \right\} - \rho$$

$$= \arg\max_{\boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho} \sum_{i=1}^{n}(\gamma_i - \delta_i)$$

subject to the following equation

$$(3.27) \quad 2n\gamma_i = \sum_{p=1}^{t}\lambda_p c_{pi} + n\delta_i.$$

The only constraint on δ_i is $\delta_i \geq 0$. Therefore, to maximize $\sum_{i=1}^{n}(\gamma_i - \delta_i)$, the optimal value for δ_i is 0. Hence, the following equation holds

$$(3.28) \qquad 2n\gamma_i^{(t)} = \sum_{p=1}^{t} \lambda_p^{(t)} c_{pi}.$$

Thus, $n\gamma_i^{(t)}$ is a constant independent of n. Moreover, $\sum_{i=1}^{n} \frac{(c_i')^2}{n}$ measures the fraction of non-zero elements in the constraint vector $\mathbf{c}'$, and therefore is a constant related only to the newly added constraint, also independent of n. Hence, $\sum_{i=1}^{n} \frac{(c_i')^2}{\gamma_i^{(t)} n^2}$ is a constant independent of n and D, and we denote it with $Q^{(t)}$. Moreover, define $R = \max_t\{Q^{(t)}\}$ as the maximum of $Q^{(t)}$ throughout the whole *cutting plane* process. Therefore, the increase of the objective function of the *Lagrangian dual problem* after adding the most violated constraint $\mathbf{c}'$ is at least $\frac{\epsilon^2}{R}$. Furthermore, denote with $G^{(t)}$ the value of the objective function in problem (2.7) subject to $\Omega^{(t)}$ after adding t constraints. Due to weak duality [3], at the optimal solution $L_*^{(t)} \leq G_*^{(t)} \leq C$. Since the *Lagrangian dual function* is upper bounded by C, the *cutting plane* algorithm terminates after adding at most $\frac{CR}{\epsilon^2}$ constraints. □

Recall that Lemma 3.2 bounds the number of iterations in our *cutting plane* algorithm by a constant $\frac{CR}{\epsilon^2}$, which is independent of n and D. Moreover, each iteration of the algorithm takes $O(D^{3.5} + nD + D^{2.5}|\Omega|)$ time. Therefore, the *cutting plane* algorithm for *multiple kernel MMC* has a time complexity of $\sum_{|\Omega|=1}^{CR/\epsilon^2} O(D^{3.5} + nD + D^{2.5}|\Omega|) = O(\frac{D^{3.5}+nD}{\epsilon^2} + \frac{D^{2.5}}{\epsilon^4})$. Hence, we have proved theorem 3.1.

4 Multi-Class Multiple Kernel Clustering

In this section, we extend the *multiple kernel MMC* algorithm to multi-class clustering.

4.1 Multi-Class Formulation For the multi-class scenario, we will start with an introduction to the *multi-class support vector machine* formulation proposed in [7]. Given a point set $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ and their labels $\mathbf{y} = (y_1, \ldots, y_n) \in \{1, \ldots, m\}^n$, the *SVM* defines a weight vector $\mathbf{w}^p$ for each class $p \in \{1, \ldots, m\}$ and classifies sample $\mathbf{x}$ by $p^* = \arg\max_{p \in \{1,\ldots,k\}} \mathbf{w}^{pT}\mathbf{x}$. The weight vectors are obtained as follows:

$$(4.29)\min_{\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\sum_{p=1}^{m} ||\mathbf{w}^p||^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i$$
$$\text{s.t.} \quad \forall i \in \{1,\ldots,n\}, r \in \{1,\ldots,m\}:$$
$$\mathbf{w}^{y_i\,T}\Phi(\mathbf{x}_i) + \delta_{y_i,r} - \mathbf{w}^{r\,T}\Phi(\mathbf{x}_i) \geq 1 - \xi_i,$$
$$\forall i \in \{1,\ldots,n\}: \xi_i \geq 0.$$

Instead of a single feature mapping Φ, we consider the non-negative combination of M feature mappings as shown in Eq.(1.1). The *multiple kernel multi-class SVM* can therefore be formulated as:

$$(4.30)\min_{\boldsymbol{\beta},\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\sum_{k=1}^{M}\beta_k\sum_{p=1}^{m} ||\mathbf{w}_k^p||^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad \forall i \in \{1,\ldots,n\}, r \in \{1,\ldots,m\}:$$
$$\sum_{k=1}^{M}\beta_k(\mathbf{w}_k^{y_i} - \mathbf{w}_k^r)^T\Phi_k(\mathbf{x}_i) + \delta_{y_i,r} \geq 1 - \xi_i,$$
$$\forall i \in \{1,\ldots,n\}: \xi_i \geq 0,$$
$$\forall k \in \{1,\ldots,M\}: \beta_k \geq 0,$$
$$\sum_{k=1}^{M}\beta_k^2 \leq 1,$$

where the superscript p in $\mathbf{w}_k^p$ denotes the pth class and the subscript k denotes the kth feature mapping. Instead of finding a large margin classifier given labels on the data as in *SVM*, *MMC* targets to find a labeling that will result in a large margin classifier. The *multiple kernel multi-class maximum margin clustering* can therefore be formulated as:

$$(4.31) \quad \min_{\mathbf{y},\boldsymbol{\beta},\mathbf{v},\boldsymbol{\xi}} \quad \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k} + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad \forall i \in \{1,\ldots,n\}, r \in \{1,\ldots,m\}:$$
$$\sum_{k=1}^{M}(\mathbf{v}_k^{y_i} - \mathbf{v}_k^r)^T\Phi_k(\mathbf{x}_i) + \delta_{y_i,r} \geq 1 - \xi_i,$$
$$\forall i \in \{1,\ldots,n\}: \xi_i \geq 0,$$
$$\forall k \in \{1,\ldots,M\}: \beta_k \geq 0,$$
$$\sum_{k=1}^{M}\beta_k^2 \leq 1,$$
$$\forall p,q \in \{1,\ldots,m\}:$$
$$-l \leq \sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p - \mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i) \leq l,$$

where we have applied the following change of variables

$$(4.32) \quad \forall i \in \{1,\ldots,n\}, k \in \{1,\ldots,M\}: \mathbf{v}_k^p = \beta_k\mathbf{w}_k^p$$

to ensure that the objective function and the last constraint are convex. Similar to two-class clustering, we have also added class balance constraints (where $l > 0$) in the formulation to control class imbalance. Again, the above formulation is an integer program, and is much more complex than the *QP* problem in *multi-class SVM*. Fortunately, similar to the two-class case, we have the following theorem.

THEOREM 4.1. *Problem (4.31) is equivalent to*

$$(4.33) \min_{\boldsymbol{\beta},\mathbf{v},\boldsymbol{\xi}} \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k}+\frac{C}{n}\sum_{i=1}^{n}\xi_i$$

$$s.t. \ \forall i\in\{1,\ldots,n\}, r\in\{1,\ldots,m\}:$$

$$\sum_{k=1}^{M}\left(\sum_{p=1}^{m}z_{ip}\mathbf{v}_k^p-\mathbf{v}_k^r\right)^T\Phi_k(\mathbf{x}_i)+z_{ir}\geq 1-\xi_i,$$

$$\forall i\in\{1,\ldots,n\}:\ \xi_i\geq 0,$$

$$\forall k\in\{1,\ldots,M\}:\ \beta_k\geq 0,$$

$$\sum_{k=1}^{M}\beta_k^2\leq 1,$$

$$\forall p,q\in\{1,\ldots,m\}:$$

$$-l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l,$$

where z_{ip} is defined as $\forall i\in\{1,\ldots,n\}, p\in\{1,\ldots,m\}$:

$$z_{ip}=\prod_{q=1,q\neq p}^{m}I_{[\sum_{k=1}^{M}\mathbf{v}_k^{r\,T}\Phi_k(\mathbf{x}_i)>\sum_{k=1}^{M}\mathbf{v}_k^{q\,T}\Phi_k(\mathbf{x}_i)]},$$

with $I(\cdot)$ being the indicator function *and the label for sample $\mathbf{x}_i$ is determined as $y_i=\arg\max_p\sum_{k=1}^{M}\mathbf{v}_k^{p\,T}\Phi_k(\mathbf{x}_i)=\sum_{p=1}^{m}pz_{ip}$.*

The *multiple kernel clustering* formulation shown in Eq.(4.33) has n slack variables $\{\xi_1,\ldots,\xi_n\}$ in the non-convex constraints. We propose the following theorem to reduce the number of slack variables in Eq.(4.33).

THEOREM 4.2. *Problem (4.33) can be equivalently formulated as problem (4.34), with $\xi^*=\frac{1}{n}\sum_{i=1}^{n}\xi_i^*$.*

$$(4.34) \min_{\boldsymbol{\beta},\mathbf{v},\boldsymbol{\xi}} \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k}+C\xi$$

$$s.t. \ \forall\mathbf{c}_i\in\{\mathbf{e}_0,\mathbf{e}_1,\ldots,\mathbf{e}_k\}, i\in\{1,\ldots,n\}:$$

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}(\mathbf{c}_i^T\mathbf{e}z_{ip}-c_{ip})\mathbf{v}_k^{p\,T}\Phi_k(\mathbf{x}_i)$$

$$+\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}\geq\frac{1}{n}\sum_{i=1}^{n}\mathbf{c}_i^T\mathbf{e}-\xi,$$

$$\forall k\in\{1,\ldots,M\}:\ \beta_k\geq 0,$$

$$\sum_{k=1}^{M}\beta_k^2\leq 1,\ \xi\geq 0,$$

$$\forall p,q\in\{1,\ldots,m\}:$$

$$-l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l.$$

where we define $\mathbf{e}_p$ as the $k\times 1$ vector with only the pth element being 1 and others 0, $\mathbf{e}_0$ as the $k\times 1$ zero vector and $\mathbf{e}$ as the vector of ones.

After the above reformulation, a single slack variable ξ is shared across all the non-convex constraints. We propose the use of the *cutting plane* algorithm to handle the exponential number of constraints in problem (4.34).

Algorithm 3 Cutting plane algorithm for multiple kernel multi-class maximum margin clustering.

Input: M feature mappings $\Phi_1,\ldots,\Phi_M$, parameters C, l and ϵ, constraint subset $\Omega=\phi$.
repeat
  Solve problem (4.34) for $(\mathbf{v}_k^p,\boldsymbol{\beta})$, $k=1,\ldots,M$ and $p=1,\ldots,m$, under the current working constraint set Ω.
  Select the most violated constraint $\mathbf{c}$ and set $\Omega=\Omega\cup\{\mathbf{c}\}$.
until the newly selected constraint $\mathbf{c}$ is violated by no more than ϵ.

4.2 Optimization via the *CCCP* Given an initial point $\mathbf{v}^{(0)}$, the *CCCP* computes $\mathbf{v}^{(t+1)}$ from $\mathbf{v}^{(t)}$ by replacing $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}\mathbf{c}_i^T\mathbf{e}z_{ip}\mathbf{v}_k^{p\,T}\Phi_k(\mathbf{x}_i)+\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}$ in the constraint with its first-order Taylor expansion at $\mathbf{v}^{(t)}$.

$$(4.35) \min_{\boldsymbol{\beta},\mathbf{v},\xi} \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k}+C\xi$$

$$s.t. \ \forall[\mathbf{c}_1,\ldots,\mathbf{c}_n]\in\Omega, i\in\{1,\ldots,n\}:$$

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}(\mathbf{c}_i^T\mathbf{e}z_{ip}^{(t)}-c_{ip})\mathbf{v}_k^{p\,T}\Phi_k(\mathbf{x}_i)$$

$$+\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}^{(t)}\geq\frac{1}{n}\sum_{i=1}^{n}\mathbf{c}_i^T\mathbf{e}-\xi,$$

$$\forall k\in\{1,\ldots,M\}:\ \beta_k\geq 0,$$

$$\sum_{k=1}^{M}\beta_k^2\leq 1,\ \xi\geq 0,$$

$$\forall p,q\in\{1,\ldots,m\}:$$

$$-l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l.$$

Similar to two-class clustering, the above problem can be formulated as an *SOCP* and solved efficiently. According to the procedure of *CCCP*, we solve problem (4.34) under the constraint set Ω with Algorithm 4.

Algorithm 4 Solve problem (4.34) subject to constraint subset Ω via the CCCP.

> Initialize $\mathbf{v}^{(0)}$.
> **repeat**
>> Obtain $(\mathbf{v}^{(t+1)}, \boldsymbol{\beta}, \xi)$ as the solution to the *second order cone programming* problem (4.35).
>> Set $\mathbf{v} = \mathbf{v}^{(t+1)}$ and $t = t + 1$.
> **until** the stopping criterion is satisfied.

4.3 The Most Violated Constraint The most violated constraint is the one that results in the largest ξ, and can be obtained by the following theorem.

THEOREM 4.3. *The most violated constraint can be obtained as*

$$\mathbf{c}_i = \begin{cases} \mathbf{e}_{r^*} & \text{if } \left[\sum_{k=1}^M \mathbf{v}_k^{p*T}\Phi_k(\mathbf{x}_i) - \sum_{k=1}^M \mathbf{v}_k^{r*T}\Phi_k(\mathbf{x}_i)\right] < 1, \\ \mathbf{0} & \text{otherwise,} \end{cases}$$

where $p^* = \arg\max_p \sum_{k=1}^M \mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i)$ and $r^* = \arg\max_{r \neq p^*} \sum_{k=1}^M \mathbf{v}_k^{rT}\Phi_k(\mathbf{x}_i)$.

5 Experiments

In this section, we demonstrate the accuracy and efficiency of the *multiple kernel clustering* algorithms on several toy and real-world data sets. All the experiments are performed with MATLAB 7.0 on a 1.66GHZ Intel Core[TM]2 Duo PC running Windows XP and with 1.5GB memory. In the following, we will simply refer to the proposed algorithms as *MKC*.

5.1 Data Sets We use seven data sets which are intended to cover a wide range of properties: **ionosphere, digits, letter** and **satellite** (from the UCI repository), **svmguide1-a** (from the LIBSVM data[1]), **ringnorm**[2] and **mnist**[3]. The two-class data sets are created following the same setting as in [30]. We also create several multi-class data sets from the **digits, letter** and **mnist** data. We summarize all of these in Table 1.

Table 1: Descriptions of the data sets.

Data	Size	Dimension	Class
digits1v7	361	64	2
digits2v7	356	64	2
ionosphere	354	64	2
svmguide1-a	1000	4	2
ringnorm	1000	20	2
letterAvB	1555	16	2
satellite	2236	36	2
digits0689	713	64	4
digits1279	718	64	4
letterABCD	3096	16	4
mnist01234	28911	196	5

[1]http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/data sets/
[2]http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm
[3]http://yann.lecun.com/exdb/mnist/

5.2 Comparison of Clustering Accuracy We will first study the clustering accuracy of the *MKC* algorithm. Specifically, we use a kernel matrix $\mathbf{K} = \sum_{k=1}^3 \beta_k \mathbf{K}_k$, where the $\mathbf{K}_k$'s are initial "guesses" of the kernel matrix. We use a linear kernel function $k_1(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T\mathbf{x}_2$ for $\mathbf{K}_1$, a polynomial kernel function $k_2(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_1^T\mathbf{x}_2)^d$ for $\mathbf{K}_2$ and a Gaussian kernel function $k_3(\mathbf{x}_1, \mathbf{x}_2) = \exp(-(\mathbf{x}_1 - \mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2)/2\sigma)$ for $\mathbf{K}_3$. For comparison, we use *k-means clustering (KM)* and *normalized cut (NC)* as baselines. We also compare with *IterSVR* [30] which performs MMC on two-class data. For *KM*, the cluster centers are initialized randomly, and the performance reported are summarized over 50 independent runs. For *NC*, the width of the Gaussian kernel is set by an exhaustive search from the grid $\{0.1\sigma_0, 0.2\sigma_0, \ldots, \sigma_0\}$, where σ_0 is the range of distance between any two data points in the data set. Finally, for *IterSVR*, the initialization is based on *k-means* with randomly selected initial cluster centers. The Gaussian kernel is used and its width is set in the same way as in *NC*.

In all the clustering algorithms, we set the number of clusters to the true number of classes (m). To assess the clustering accuracy, we follow the strategy used in [27]: We first take a set of labeled data, remove all the labels and run the clustering algorithms; then we label each of the resulting clusters with the majority class according to the original labels, and finally measure the number of correct classifications. Moreover, we also calculate the *Rand Index*[4] [19] for each clustering result.

Results on the various data sets are summarized in Table 2. As can be seen, the clustering accuracy and *Rand Index* of *MKC* are comparable to those attained by the best base kernel. In most cases, it is even better than the other competing clustering algorithms.

5.3 Speed A potential concern about *multiple kernel clustering* is that it might be much slower than *maximum margin clustering*. Figure 1 compares the CPU-time[5] of *MKC* with the total CPU-time of *MMC* with $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{K}_3$. As can be seen, the speed of *MKC* is comparable to *MMC*. Indeed, *MKC* even converges faster than *MMC* on several data sets. However, unlike *MMC* which requires a carefully defined kernel matrix, *MKC* has the strong advantage that it can automatically choose a good combination of base kernels.

---

[4]The Rand index has a value between 0 and 1, with 0 indicating that the data clustering does not agree on any pair of points with the true clustering, and 1 indicating that the two clustering results are exactly the same.

[5]The CPU-time consists of the computational time of *kernel PCA* (to obtain the feature representations corresponding to nonlinear kernels) and that of *MKC* or *MMC*.

Table 2: Clustering accuracies (%) and Rand Indices on the various data sets. For each method, the number on the left denotes the clustering accuracy, and the number on the right stands for the Rand Index. The symbol '-' means that the corresponding algorithm cannot handle the data set in reasonable time. Moreover, note that *IterSVR* can only be used on two-class data sets.

| Data	K_1		K_2		K_3		MKC		KM		NC		IterSVR	
digits1v7	**100.00**	**1.00**	95.52	0.915	95.24	0.910	**100.00**	**1.00**	99.45	0.995	55.00	0.504	99.45	0.995
digits2v7	**100.00**	**1.00**	97.47	0.951	99.16	0.989	**100.00**	**1.00**	96.91	0.940	66.00	0.550	**100.00**	**1.00**
ionosphere	72.36	0.599	62.51	0.531	86.52	0.767	**91.01**	**0.839**	68.00	0.564	75.00	0.626	67.70	0.562
svmguide1-a	78.40	0.661	77.60	0.653	84.30	0.735	85.50	0.752	76.50	0.640	87.50	0.781	**93.20**	**0.873**
ringnorm	76.70	0.642	58.10	0.513	98.40	0.969	**99.00**	**0.980**	76.00	0.635	77.70	0.653	80.70	0.831
letterAvB	93.12	0.873	90.35	0.826	93.38	0.877	**93.83**	**0.885**	82.06	0.706	76.80	0.644	92.80	0.867
satellite	98.48	0.971	76.79	0.644	88.68	0.799	**99.37**	**0.992**	95.93	0.922	95.79	0.919	96.82	0.939
digits0689	96.63	0.968	94.11	0.946	84.57	0.887	**97.77**	**0.978**	42.33	0.696	93.13	0.939		
digits1279	94.01	0.943	90.11	0.911	90.39	0.914	**94.43**	**0.948**	40.42	0.681	90.11	0.909		
letterABCD	70.77	0.804	65.05	0.761	53.55	0.684	**71.89**	**0.821**	66.18	0.782	68.19	0.811		
minst01234	89.98	0.901	87.34	0.882	90.12	0.907	**91.55**	**0.922**	67.63	0.898	-	-		
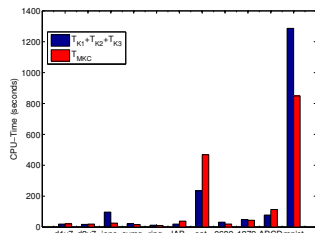
Figure 1: CPU-time (seconds) of *MKC* and *MMC* as a function of the data set size n.

5.4 Scaling Properties of MKC In Section 3, we showed that the time complexity of *MKC* scales linearly with the number of samples. Figure 2 shows a log-log plot of the empirical results. Note that lines in a log-log plot correspond to polynomial growth $O(n^d)$, where d is the slope of the line. As can be seen, the CPU-time of *MKC* scales roughly as $O(n)$, and is thus consistent with Theorem 3.1.

Moreover, as mentioned in Section 3, each round of *MKC* requires fewer than 10 iterations for solving problem (2.7) or (4.34) subject to the constraints in Ω. Again, this is confirmed by the experimental results in Figure 2, which shows how the number of *CCCP* iterations (averaged over all the cutting plane iterations) varies with sample size on the various data sets.
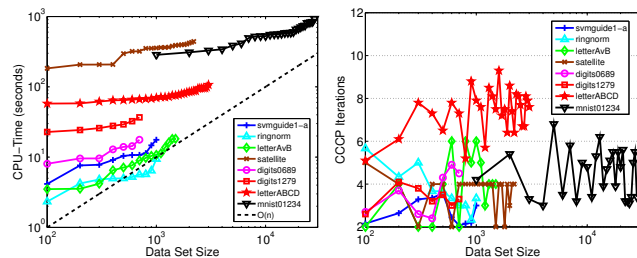

Figure 2: Left: CPU-time of MKC vs. data set size. Right: Average number of CCCP iterations in MKC vs. data set size.

Next, we study how the CPU-time of *MKC* varies

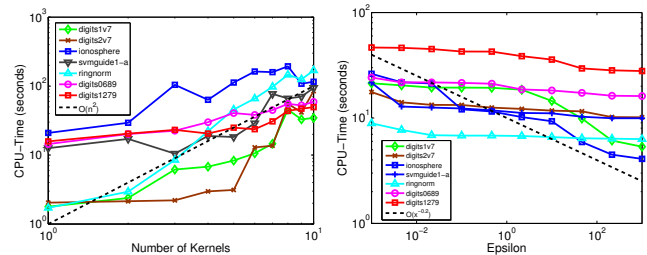Figure 3: Left: CPU-time of MKC vs. the number of kernels. Right: CPU-time of MKC vs. ϵ.

when the number of base kernels[6] is increased from one to ten. As can be seen from Figure 3, the CPU-time of *MKC* scales roughly quadratically with the number of base kernels. This is much better than the bound of $O(M^{3.5})$ in Section 3.

Finally, Section 3 states that the total number of iterations involved in *MKC* is at most $\frac{CR}{\epsilon^2}$. This means that with a higher ϵ, the algorithm might converge faster. Figure 3 shows how the CPU-time of *MKC* scales with ϵ. As can be seen, the empirical scaling is roughly $O(\frac{1}{\epsilon^{0.2}})$, which is much better than $O(\frac{D^{3.5}+nD}{\epsilon^2} + \frac{D^{2.5}}{\epsilon^4})$ shown in the bound.

5.5 Generalization Ability of MKC Recall that *maximum margin clustering* adopts the maximum margin principle of *SVM*, which often allows good generalization on unseen data. In this experiment, we also examine the generalization ability of *MKC* on unseen data samples. We first learn the *multiple kernel clustering* model on a data subset randomly drawn from the whole data set. Then we use the learned model to cluster the whole data set. As can be seen in Table 3, the clustering performance of the model learned on the data subset is comparable with that of the model learned on the whole data set. This suggests an important application scenario for *multiple kernel clustering*, namely that

---

[6]Gaussian kernels are used here.

for a large data set, we can simply perform the clustering process on a small subset of the data and then use the learned model to cluster the remaining data points.

Table 3: Generalization ability of *MKC*.

Data	from whole set		from data subset		
	Acc	RIndex	subset size	Acc	RIndex
letterAvB	93.83	0.885	500	93.27	0.874
satellite	99.37	0.992	500	98.47	0.984
letterABCD	71.89	0.821	500	70.00	0.781
mnist01234	91.55	0.922	1000	91.68	0.920

6 Conclusions

In this paper, we extend multiple kernel learning to unsupervised learning. In particular, we propose the *multiple kernel clustering (MKC)* algorithm that simultaneously finds the maximum margin hyperplane, the best cluster labeling and the optimal kernel. Experimental results on both toy and real-world data sets demonstrate the effectiveness and efficiency of the algorithm.

Acknowledgements

References

[1] F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *ICML*, 2004.

[2] O. Bousquet and D. Herrmann. On the complexity of learning the kernel matrix. In *NIPS*, 2003.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] P. K. Chan, D. F. Schlag, and J. Y. Zien. Spectral *k*-way ratio-cut partitioning and clustering. *IEEE Trans. Computer-Aided Design*, 13:1088–1096, 1994.

[5] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46:131–159, 2002.

[6] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *AISTATS*, 2005.

[7] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *JMLR*, 2:265–292, 2001.

[8] C. Ding, X. He, H. Zha, M. Gu, and H. D. Simon. A min-max cut algorithm for graph partitioning and data mining. In *ICDM*, pages 107–114, 2001.

[9] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., 2001.

[10] M. Gonen and E. Alpaydin. Localized multiple kernel learning. In *ICML*, 2008.

[11] T. Joachims. Training linear SVMs in linear time. In *SIGKDD*, 2006.

[12] J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial Applied Mathematics*, 8:703–712, 1960.

[13] G. Lanckriet, T. De Bie, N. Cristianini, M. Jordan, and W. Noble. A statistical framework for genomic data fusion. *Bioinfomatics*, 20(16):2626–2635, 2004.

[14] G. Lanckriet, N. Cristianini, L. Ghaoui, P. Bartlett, and M. Jordan. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004.

[15] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra Appl.*, 284:193–228, 1998.

[16] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. 1994.

[17] C. Ong, A. Smola, and R. Williamson. Learning the kernel with hyperkernels. *JMLR*, 6:1043–1071, 2005.

[18] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. More efficiency in multiple kernel learning. In *ICML*, 2007.

[19] W. Rand. Objective criteria for the evaluation of clustering methods. *JASA*, 66:846–850, 1971.

[20] B. Schölkopf, A. J. Smola, and K. R. Müller. Kernel principal component analysis. *Advances in kernel methods: Support vector learning*, pages 327–352, 1999.

[21] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[22] A. J. Smola, S.V.N. Vishwanathan, and T. Hofmann. Kernel methods for missing variables. In *AISTATS*, 2005.

[23] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *JMLR*, 7:1531–1565, 2006.

[24] P. Sun and X. Yao. Boosting kernel models for regression. In *ICDM*, 2006.

[25] I. Tsang and J. Kwok. Efficient hyperkernel learning using second-order cone programming. *IEEE Transactions on Neural Networks*, 17(1):48–58, 2006.

[26] H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. In *NIPS*, 2007.

[27] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *NIPS*, 2004.

[28] L. Xu and D. Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *AAAI*, 2005.

[29] J. Ye, S. Ji, and J. Chen. Learning the kernel matrix in discriminant analysis via quadratically constrained quadratic programming. In *SIGKDD*, 2007.

[30] K. Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. In *ICML*, 2007.

[31] B. Zhao, F. Wang, and C. Zhang. Efficient maximum margin clustering via cutting plane algorithm. In *SDM*, 2008.

[32] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *ICML*, 2008.

[33] A. Zien and C. Ong. Multiclass multiple kernel learning. In *ICML*, 2007.