学号 1206010413

年级 12 级

河 海 大 學

本科毕业论文

多核聚类研究

专　　业 计算机科学与技术

姓　　名 朱 雪 林

指导教师 王敏、薛晖

评 阅 人

2016年4月

中国　南京

BACHELOR'S DEGREE THESIS
OF HOHAI UNIVERSITY

Multiple Kernel Clustering Research

College	:	College of Computer and Information
Subject	:	Computer Science and Technology
Name	:	XueLin Zhu
Directed by	:	Min Wang、Hui Xue associate professor

NANJING CHINA

郑 重 声 明

本人呈交的毕业论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计（论文）的研究成果不包含他人享有著作权的内容。对本设计（论文）所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计（论文）的知识产权归属于培养单位。

本人签名：_____　　日期：_____

摘　要

支持向量机 SVM 以间隔最大化为学习策略，并引入核技巧实现非线性分类，被公认为目前最好的分类模型。受到间隔最大化学习策略和核技巧在 SVM 中所取得的卓越性能的启发，本文考虑将间隔最大化学习策略和核技巧从有监督的学习推广到无监督的学习中。

最大间隔聚类 MMC 的核心思想是首先使用核函数将数据映射到高维的核特征空间，然后寻找一组最优的类标记组合，使得 SVM 在新的特征空间上训练产生最大的间隔。在此基础上，MMC 通过对约束条件进行松弛变换，得到半定规划 (SDP) 模型。由于最大间隔聚类仅仅依赖于由样本数据生成的核矩阵，因此很容易实现非线性聚类。但最大间隔聚类与其它的核方法一样，核函数的选择将直接决定模型最终性能的好坏，而对于当前特定的任务，如何选择合适的核函数还是一个尚未解决的问题。

多核聚类 MKC 在此基础上利用有监督和半监督中多核学习的思想，提出对多个核函数进行非负线性组合并得到一个新的基核，再使用这个基核进行训练。MKC 在此基础上构造出二阶锥规划 (SOCP) 模型，并使用割平面算法和凹凸规划对模型进行优化和求解，并同时得到最优的核函数组合、最适当的类标记组合以及最大间隔超平面。

关键字：聚类；最大间隔聚类；多核聚类；核技巧

Abstract

The learning strategy of support vector machine is finding maximum margin hyperplane on training dataset, and it achieves nonlinear classification by using kernel trick. SVM is considered the best classified model at present. Inspired by the outstanding performance of the maximum margin learning strategy and kernel trick in SVM, this paper considers to apply the maximum margin learning strategy and kernel trick from supervised learning to unsupervised learning.

The key idea of maximum margin clustering (MMC) is mapping the sample dataset to high-dimensional feature space at first, and then finds a set of optimal class labels so that SVM can find maximum margin by training in new feature space. MMC obtains semidefinite programming model by relaxing constraint condition. MMC is so easy to achieve nonlinear classification by using kernel trick. However, as in other kernel methods, choosing a suitable kernel function is imperative to the success of MMC. But for a current specific task, it is still a unresolved problem to find a suitable kernel function.

Inspired by the works called multiple kernel learning in supervised and semi-supervised leaning, multiple kernel clustering (MKC) proposes a non-negative linear combination of many kernel functions to generate a new base kernel, and then uses this new base kernel to train. MKC constructs second-order cone programing model base on previous idea, using cutting plane algorithm and concave-convex procedure to optimize and solve the model. Finally, it can find the maximum margin hyperplane, the best cluster labeling, and the optimal kernel.

Key words: Clustering; Maximum margin clustering; Multiple kernel clustering; kernel trick

目　录

第一章　绪论

1.1　研究背景与意义

聚类是一种无监督的学习，其训练样本的标记信息是未知的，目标是通过对无标记训练样本的学习来揭示数据的内在性质规律，为进一步的数据分析提供基础。聚类试图将数据集中的样本划分为若干个通常是不想交的子集，每个子集称为一个“簇”，使得簇内样本差异很小，簇间样本差异较大。聚类不但能作为一个单独的过程，用于寻找样本数据内部的分布情况，也可以作为分类等其他学习任务的前驱过程[1]。

在数据挖掘和机器学习等领域，聚类是十分重要的研究课题。尤其是在数据挖掘领域，聚类的应用极为广泛。在电子商务上，聚类能够将具有相似浏览行为的客户进行分组，并分析客户相同的行为特征，能够更好的帮助工作人员了解自己的客户，以便向客户提供更加精确的服务；在生物学上，聚类能对动植物的基因进行分类，分析种群固有的结构知识等等；在搜索引擎上，聚类能对网页和文档等进行分类，挖掘其中的信息等；尤其是在社交网络挖掘等领域，聚类技术的应用极为广泛。

但是，传统的聚类方法只能应用于一些维度较低的数据。由于在实际应用中，数据往往十分复杂，维度较高，比如基因表达数据、图像特征、文档词频数等，其维度通常能达到成百上千，甚至更高。在高维数据集中进行聚类时，传统的聚类方法主要遇到两个问题：

1. 高维数据集中存在许多与类别无关的特征，这就使得在所有特征维度下，存在簇的可能性非常低；

2. 在高维空间中，数据的分布会比低维空间更加稀疏，这造成各个数据点之间的距离几乎相等，差距不大。而传统的聚类方法基本上都是以数据点之间的距离作为相似度的衡量标准，因此很难在高维空间中进行聚类。

在传统的聚类技术很难解决高维数据集聚类问题时，核技巧在有监督的分类学习任务中展现出它的优越的性能，其中应用最为广泛的分类模型就是支持向量机 SVM。得

益于核技巧的应用，以间隔最大化为学习策略的 SVM 只依赖于由样本数据生成的核矩阵，从而有效解决高维数据集问题，并且很容易的实现了非线性分类。

受到 SVM 卓越的分类性能的启发，将间隔最大化学习策略和核技巧从有监督的分类学习任务推广到无监督的聚类学习任务中，其学习得到的模型必然会比传统的聚类模型具有更多的优越之处。一方面核技巧能有效解决高维数据集问题，另一方面间隔最大化学习策略保证了模型良好的泛化性能，如何将其应用到无监督的聚类学习任务中，具有非常重大的研究意义。

1.2 聚类方法研究现状

传统的聚类算法主要包括原型聚类、层次聚类以及密度聚类等。原型聚类中最简单的当属 k 均值算法，该算法存在大量的变体，比如 k-medoids 算法[2] 强制原型向量必为训练样本，k-modes 算法[3] 可处理离散属性，Fuzzy C-means (FCM) [4] 则是“软聚类”算法，允许每个样本以不同程度同时属于多个原型。需要注意的是，k 均值类算仅在凸形簇结构上效果很好。最近研究表明，若采用某种 Bregman 距离，则可显著增强此类算法对更多类型簇结构的适用性[5]。引入核技巧则可得到核 k 均值算法[6]，这与谱聚类[7] 有模切联系。谱聚类可以看作在拉普拉斯特征映射降维后执行 k 均值聚类。聚类簇数 k 通常需要用户提供，有一些启发式用于自动确定 k[8]，但常用的仍然是基于不同 k 值多次运行后选取最佳结果。

采用不同方式表征样本分布的紧密程度，可设计出不同的密度聚类算法，除 DBSCAN[9] 外，较常用的还有 OPTICS[10]、DENCLUE[11] 等。AGNES[12] 采用自底向上的聚类策略来产生层次聚类结构，与之相反，DIANA[12] 则采用自顶向下的分拆策略。AGNES 和 DIANA 都不能对已合并或已分拆的聚类簇进行回溯调整，常用的层次聚类如 BIRCH[13]、ROCK[14] 等对此进行了改进。

近些年来核技巧开始逐渐被用在聚类分析中。Tax 通过使用支持向量方法来描述数据域，并提出了一种基于高斯核的 SVDD (Support Vector Domain Description) 算法[15]。

Ben-Hen 提出了支持向量聚类 (SVC)[16] 算法。作为一种新的无监督非参数型算法，SVC 主要分为两个过程：SVM 训练过程和标记分配过程，其中 SVC 训练过程主要包括确定高斯核宽度系数、计算核矩阵和拉格朗如乘子、选取支持向量并计算在高维特征空间中特征球的半径；标记分配过程首先会生成关联矩阵，然后根据关联矩阵进行标记分配[17]。

就目前的聚类方法而言，大多数都只能在特定的样本数据集上表现出较好的性能，包括传统的 k-means 聚类算法和模糊 C 均值聚类算法等等，这些算法都是直接基于样本数据特征进行聚类，却没有对样本数据的特征进行优化，因此上面这些方法的聚类性能往往在很大程度上取决于样本数据内部的分布情况[18]。但是，通过使用核技巧，将输入空间的数据映射到高维的特征空间中，增加各类样本数据之间的差异，以达到在高维特征空间中线性可聚的目的，从而提高聚类的精确度[18]。随着近些年来对核聚类的探索，涌现出许多基于核的聚类算法。比如支持向量聚类，基于核的模糊聚类算法，基于模糊核聚类的 SVM 多类分类方法等等。核聚类研究的进展与突破，为有效处理非线性样本数据带来了突破口，同时也拓展了聚类课题的研究范围。

1.3 研究目标及内容

支持向量机以间隔最大化为学习策略，并引入核技巧实现非线性分类器。其优越的分类精度，尤其在文本分类任务中显示出卓越性能[19]，被业界公认为最好的分类模型。间隔最大化学习策略能保证模型具有较好的泛化能力，核技巧能轻松实现非线性分类，并且能有效避免“维数灾难”。考虑到间隔最大化学习策略和核技巧在分类学习任务中的卓越性能，那么将其应用在无监督学习的聚类任务中，取得的性能让人十分期待。

本文的研究目标是：基于间隔最大化学习策略和核技巧的思想，着重研究其在最大间隔聚类 (MMC)[20] 算法和多核聚类 (MKC)[21] 算法的应用技巧，详细探讨间隔最大化学习策略和核技巧在聚类中所取得的性能。

本文的主要工作内容：

1. 分析间隔最大化学习策略和核技巧的优越之处。分别从 SVM 主问题以及对偶问

题出发研究间隔最大化学习策略和核技巧在 SVM 中的应用方法。

2. 分析最大间隔聚类算法原理和模型推导过程。针对分类模型 SVM，将其推广到聚类分析中，并通过凸优化技巧进行建模。

3. 分析多核聚类算法原理和模型推导过程。针对最大间隔聚类算法的局限性，对其进行改进，由单核学习到多核学习。并引入割平面算法和凹凸规划，提高算法的收敛速度。

4. 针对最大间隔聚类算法和多核聚类算法，使用 UCI 上的数据集，通过实验来验证算法的性能，并与 k 均值聚类和谱聚类等算法进行对比。

1.4 论文结构

本文共分为六章。第一章介绍了本文工作的研究背景与意义，聚类的研究现状，研究动机以及主要的研究内容和目标；第二章介绍了 SVM 模型的基本内容，分别从 SVM 主问题以及对偶问题出发，分析 SVM 中的间隔最大化学习策略和核技巧的应用，为后续章节提供必要的背景知识；第三章分析了 MMC 算法的模型推导过程，并提出该算法的局限性；第四章分析了 MKC 算法的模型推导过程，并提出该算法的优越之处；第五章进行相关实验来验证 MMC 算法和 MKC 算法的性能，并与传统的聚类算法进行对比；第六章对本文的工作进行了总结。

第二章　支持向量机 SVM

SVM 是 Vapnik[22] 在 1995 年提出的一种非常有效的算法，由于其卓绝的分类性能，很快成为机器学习的主流技术。以统计学习理论为基础，SVM 提出了间隔 (Margin) 的概念，不但提高了分类精确率，而且保证其训练得到的分类器具有较好的泛化能力。并且在此基础上引入核技巧，在有效解决维数灾难问题的同时，也成为更具实用性的非线性分类模型。本章将重点介绍 SVM 模型的相关基础知识，分别从 SVM 的主问题和对偶问题出发，分析其间隔最大化学习策略和核技巧的应用。

2.1　SVM 模型

2.1.1　硬间隔最大化分类器

SVM 是定义在特征空间上间隔最大的分类器。考虑图2.1所示的二维二类线性可分的情况：

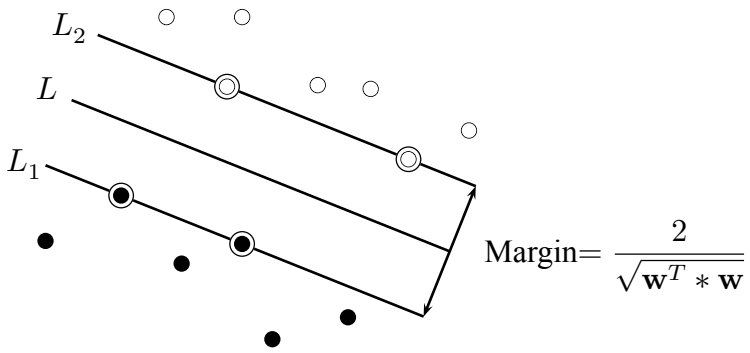

图 2.1: 二维二类分类问题

图中分别使用实心点和空心点表示两类训练样本。其中 L 是将两类样本正确分类的直线，也即分类线，L_1 和 L_2 分别为过各类样本集合中离分类线最近的样本点且平行于分类线 L 的直线，将两类样本之间的间隔定义为直线 L_1 和 L_2 之间的距离。如果分类线 L 不但能够将两类样本完全正确的分开，并且使得两类样本之间的间隔最大，那

么就认为分类线 L 是最优分类线。最优分类线不但能最小化经验风险，而且还能使泛化性能最优, 从而使得结构风险最小。若将其推广到高维空间，那么最优分类线就变成最优分类面。

考虑两类分类问题, 给定训练集 $T=\{(\mathbf{x}_i,y_i)\}_{i=1}^m$, 其中 $\mathbf{x}_i\in X=\mathbb{R}^n, y_i\in\{-1,+1\}$ 是训练样本所对应的类别标记。则其相应的分类决策函数为 $f(x)=\text{sign}(\mathbf{w}^T\mathbf{x}+b)$，分离超平面为:

$$\mathbf{w}^T\mathbf{x}+b=0 \tag{2.1}$$

其中，$\mathbf{w}$ 是权值向量，b 是偏置。定义超平面 $(\mathbf{w},b)$ 关于训练数据集 T 的函数间隔:

$$\hat{\gamma}=\min_{1,\cdots,m} y_i(\mathbf{w}^T\mathbf{x}_i+b) \tag{2.2}$$

进一步规范化分离超平面的法向量 $\mathbf{w}$，也即令 $\|\mathbf{w}\|=1$，这时便得到超平面 $(\mathbf{w},b)$ 关于训练数据集 T 的几何间隔:

$$\gamma=\frac{\hat{\gamma}}{\|\mathbf{w}\|} \tag{2.3}$$

那么，求解使得几何间隔最大的分离超平面问题就可以公式化的表示为下面的约束最优化问题:

$$\begin{aligned}&\max_{\mathbf{w},b} \quad \gamma \\ &s.t. \quad y_i\left(\frac{\mathbf{w}^T}{\|\mathbf{w}\|}\mathbf{x}_i+\frac{b}{\|\mathbf{w}\|}\right)\geq\gamma, \quad i=1,2,\cdots,m\end{aligned} \tag{2.4}$$

考虑函数间隔和几何间隔之间的关系式 (2.3)，便可得到线性可分支持向量机学习

的最优化问题：

$$\begin{aligned} \max_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ s.t. \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \geq 0, \quad i = 1, 2, \cdots, m \end{aligned} \tag{2.5}$$

2.1.2 软间隔最大化分类器

在训练数据线性不可分的情况下，上面的线性可分问题的 SVM 学习方法是不适用的，这就需要对硬间隔最大化进行松弛，得到软间隔最大化。假设训练数据集 T 中存在一些特异点 (outlier)，使得训练集 T 是线性不可分的。对其中的每个样本点 $(\mathbf{x}_i, y_i)$ 引进一个松弛变量 $\xi_i \geq 0$，使得函数间隔加上松弛变量后大于等于 1。同时对每个松弛变量 ξ_i 都要付出一定的代价，便得到线性不可分的线性支持向量机的学习问题：

$$\begin{aligned} \max_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ s.t. \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \cdots, m \\ & \xi_i \geq 0, \quad i = 1, 2, \cdots, m \end{aligned} \tag{2.6}$$

其中，$C > 0$ 是惩罚参数，用来对错误分类的样本进行惩罚。目标函数包含两层含义：几何间隔尽可能的大，同时使错误分类的样本个数尽可能的小。C 是调和二者的系数。

2.2 对偶问题

针对上面的软间隔 SVM 学习问题：

$$\begin{aligned} \max_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ s.t. \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \cdots, m \\ & \xi_i \geq 0, \quad i = 1, 2, \cdots, m \end{aligned} \tag{2.7}$$

求解对偶问题 (2.7) 常用的方法是引入拉格朗日函数：

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{m}\mu_i\xi_i \quad (2.8)$$

其中 $\boldsymbol{\alpha}$ 和 $\boldsymbol{\mu}$ 是拉格朗日乘子，且满足 $\boldsymbol{\xi} \geq 0, \boldsymbol{\mu} \geq 0$。进一步求 $L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu})$ 对 $\mathbf{w}, b, \boldsymbol{\xi}$ 的极小，即：

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i \quad (2.9)$$

$$\nabla_b L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = 0 \Rightarrow \sum_{i=1}^{m}\alpha_i y_i = 0 \quad (2.10)$$

$$\nabla_{\xi_i} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = 0 \Rightarrow C - \alpha_i - \mu_i = 0 \quad (2.11)$$

将式 (2.9)∼(2.11) 代入式 (2.8) 中，则可将 SVM 主问题转化为下面的对偶问题：

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i^T\mathbf{x}_j) - \sum_{i=1}^{m}\alpha_i \\ s.t. \quad & \sum_{i=1}^{m}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad i = 1, 2, \cdots, m \end{aligned} \quad (2.12)$$

SVM 学习问题最终转化为其对偶问题 (2.12) 进行求解。常用的优化求解算法主要包括：序列最小最优化方法 (SMO)[23]、选块算法[23] 和分解算法[23]。

2.3 核技巧

核技巧是一种用线性分类方法求解非线性分类问题的技术，首先使用一个变换将原空间的数据映射到新空间，然后在新空间里用线性分类学习方法从训练数据中学习分类模型[24]。

在 SVM 中应用核技巧，其基本想法就是通过一个非线性变换将输入空间 (欧式空间 $\mathbb{R}^n$ 或离散集合) 对应于一个特征空间 (希尔伯特空间 $\mathcal{H}$)，使得在输入空间 $\mathbb{R}^n$ 中的

超曲面模型对应于特征空间 $\mathbb{H}$ 中的超平面模型 (支持向量机)，这样，分类问题的学习任务通过在特征空间中求解线性 SVM 就可以完成[24]。下面首先定义核函数的概念：

定义 2.1 设 $\mathcal{X}$ 是输入空间 (欧式空间 $\mathbb{R}^n$ 的子集或离散集合)，又设 $\mathcal{H}$ 为特征空间 (希尔伯特空间)，如果存在一个从 $\mathcal{X}$ 到 $\mathcal{H}$ 的映射：

$$\phi(x) : \mathcal{X} \to \mathcal{H} \tag{2.13}$$

使得对所有 $x, z \in \mathcal{X}$，函数 $K(x, z)$ 满足条件：

$$K(x, z) = \phi(x) \cdot \phi(z) \tag{2.14}$$

则称 $K(x, z)$ 为核函数，$\phi(x)$ 为映射函数，式中 $\phi(x) \cdot \phi(z)$ 为 $\phi(x)$ 与 $\phi(z)$ 的内积[24]。

核技巧的想法是在学习和预测过程中，只显示定义核函数 $K(x, z)$，而不是显示定义映射函数 ϕ。观察 SVM 的对偶学习问题 (2.12) 的目标函数可以发现 $(\mathbf{x}_i^T \mathbf{x}_j)$ 是两个样本的内积，可以使用核函数 $K(\mathbf{x}_i, \mathbf{x}_j)$ 来代替，便得到非线性 SVM 的最优化问题：

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{m} \alpha_i \\ s.t. \quad & \boldsymbol{\alpha}^T \mathbf{y} = 0 \\ & 0 \le \alpha_i \le C \quad i = 1, 2, \cdots, m \end{aligned} \tag{2.15}$$

这等价于将原来的输入空间经过映射函数 ϕ 转换到一个新的特征空间，使用特征空间中的内积 $\phi(x_i) \cdot \phi(x_j)$ 来代替输入空间中的内积 $(\mathbf{x}_i^T \mathbf{x}_j)$，在训练样本的新的特征空间中学习线性 SVM。当映射函数为非线性函数时，学习得到的 SVM 模型是非线性模型。

核技巧的使用让 SVM 有效解决了维数灾难问题。此外，根据具体问题的数据样本的分布特点，选择合适的核函数更加有利于向学习问题嵌入其先验知识。

2.4 本章小结

本章主要介绍了 SVM 模型的相关基础知识，分别从 SVM 的主问题和其对偶问题出发，探讨了模型的构建、约束最优化问题的推导以及核技巧的引入等问题。详细介绍了 SVM 中的间隔最大化学习策略，以及核技巧的应用方法，为本文后续的研究工作提供了必要的知识背景。鉴于 SVM 在分类领域所取得的巨大的成功，那么，如果将间隔最大化学习策略和核技巧应用在无监督的聚类学习中，会取得怎样的效果呢？受这一灵感的启发，接下来一章开始探讨最大间隔聚类 (MMC) 算法的原理和模型推导过程。

第三章　最大间隔聚类 MMC

由第二章的分析可知，SVM 以间隔最大化为学习策略，通过对偶问题引入核技巧，实现高性能的非线性分类模型。因此在本章中，将间隔最大化学习策略和核技巧应用到无监督的聚类学习中，将聚类问题公式化的描述为凸整形规划问题，并对该问题进行松弛变化，得到最终 MMC 的半定规划模型。

3.1　MMC 模型

MMC 模型将有监督的软间隔 SVM 学习过程推广到无监督的学习过程，为无标记的训练数据添加标记，使得添加标记后的训练数据经过软间隔 SVM 学习后，能够得到最大的间隔。也就是说，给定训练数据 $T = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m\}$，其中 $\mathbf{x}_i \in X = \mathbb{R}^m$，MMC 将每个样本标记为 $y_i \in \{-1, +1\}$，使得这两个正负类别之间的间隔是最大的。

很显然，这样的样本类别标记有 2^m 种组合，因此这个问题在计算上十分困难。但若能将该问题转化为凸整形规划 (convex integer program) 问题，就有可能得到解析解。在此基础上，通过松弛整形约束将问题转化为半定规划 (semidefinite program) 问题，通过目前已有的半定规划求解工具包，就能很容易计算得到近似最大间隔的样本类别标记组合。

在进行主要工作之前，需要注意几个问题：

1. 需要在约束中添加类平衡约束。这样做不仅是为了防止所有的样本数据都被分配到相同的类标记中，更重要的是避免受到噪音的干扰，防止模型最终将一个或几个特异样本点分配到一个类中以满足最大间隔的要求。

2. 由于可能存在一些噪音数据具有相同的类别标记，为了提高 MMC 的性能，在这里使用软间隔最大化准则。

3. 尽管理论上将 MMC 模型推广到多类聚类是可能的，但为了简单起见，这里只关注于二类聚类情况。

4. 由于 SVM 中的参数 b，也就是分类器的偏置，会导致非凸问题的出现，而当前还没找到有效的方法解决这个问题，因此只考虑其次分类器，也即令 $b=0$，这样问题 (2.12) 中的约束 $\boldsymbol{\alpha}^T\mathbf{y}=0$ 就可以去掉。尽管这样的限制看起来十分严重，但可以通过将训练数据进行中心化来降低该限制的影响。

3.2 模型推导

结合上述提到的处理方法，将软间隔 SVM 对偶问题 (2.15) 推广为 MMC 模型，便得到下面的最优化问题：

$$\begin{aligned}\min_{\mathbf{y}\in\{-1,+1\}^m}\max_{\boldsymbol{\alpha}}\quad & 2\boldsymbol{\alpha}^T\mathbf{e}-\langle K\circ\boldsymbol{\alpha}\boldsymbol{\alpha}^T,\mathbf{y}\mathbf{y}^T\rangle\\ s.t.\quad & 0\le\boldsymbol{\alpha}\le C\\ & -l\le\mathbf{e}^T\mathbf{y}\le l\end{aligned}\tag{3.1}$$

其中，K 表示由特征向量 $\Phi=[\phi(\mathbf{x}_1),\cdots,\phi(\mathbf{x}_m)]$ 的内积得到 $m\times m$ 的核矩阵，即 $K=\Phi^T\Phi$，$k_{ij}=\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$，$\mathbf{e}$ 表示全 1 的向量。令 $A\circ B$ 表示矩阵的分量乘法，$\langle A,B\rangle=\sum_{ij}a_{ij}b_{ij}$。约束 $-l\le\mathbf{e}^T\mathbf{y}\le l$ 表示类平衡约束。由于最优化问题 (3.1) 的目标函数不是凸函数，因此无法使用有效的算法去解决。实际上，为了能有效的求解最优化问题 (3.1)，需要下面两个步骤：

(1) 重新描述最优化问题 (3.1)，记类标记核矩阵 $M=\mathbf{y}\mathbf{y}^T$。这样目标函数变成在关于 M 的线性函数上求最大值，因此目标函数是凸函数。这样虽然保证了目标函数具有凸性，但同时也产生了非凸约束 $M=\mathbf{y}\mathbf{y}^T$。因此必须寻找一种方法来约束 M，从而保证 $M=\mathbf{y}\mathbf{y}^T$。

(2) 添加一系列的线性约束条件来约束 M，从而保证 $M=\mathbf{y}\mathbf{y}^T$。注意到对于任意

的 $\mathbf{y} \in \{-1,+1\}^m$，$M = \mathbf{y}\mathbf{y}^T$ 一定有：

$$m_{ij} = \begin{cases} 1 & \text{if } y_i = y_j \\ -1 & \text{if } y_i \neq y_j \end{cases}$$

因此 M 具有传递性、自反性和对称性，并且 M 中有且仅有两种相等的类标记。考虑到这两个性质，下面通过添加一系列线性约束条件来满足这些要求：

$$\begin{aligned} \mathcal{L}_1: &\quad m_{ii} = 1; m_{ij} = m_{ji}; m_{ik} \geq m_{ij} + m_{jk} - 1; \quad \forall_{ijk} \\ \mathcal{L}_2: &\quad m_{jk} \geq -m_{ij} - m_{ik} - 1; \quad \forall_{ijk} \\ \mathcal{L}_3: &\quad \sum_i m_{ij} \leq m - 2; \quad \forall_j \end{aligned}$$

这些关于 M 的线性约束能够满足条件 $M = \mathbf{y}\mathbf{y}^T$。最后添加约束：

$$\mathcal{L}_4: \quad -l \leq \sum_i m_{ij} \leq l; \quad \forall_j$$

来替换类平衡约束，同时这个约束也能满足 $\mathcal{L}_3$。

经过上述两个步骤，问题 (3.1) 被重新公式化描述为下面的凸整形规划问题：

$$\begin{aligned} \min_{M \in \{-1,+1\}^{m \times m}} \quad & \max_{\boldsymbol{\alpha}} 2\boldsymbol{\alpha}^T e - \langle K \circ \boldsymbol{\alpha}\boldsymbol{\alpha}^T, M \rangle \\ s.t. \quad & 0 \leq \boldsymbol{\alpha} \leq C, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_4 \end{aligned} \tag{3.2}$$

但是问题 (3.2) 并不适用，因为凸整形规划问题的计算仍然十分困难。因此，需要进一步松弛关于 M 的整形约束，从而得到连续参数空间上的凸优化问题：

$$\begin{aligned} \min_{M \in \{-1,+1\}^{m \times m}} \quad & \max_{\boldsymbol{\alpha}} 2\boldsymbol{\alpha}^T e - \langle K \circ \boldsymbol{\alpha}\boldsymbol{\alpha}^T, M \rangle \\ s.t. \quad & 0 \leq \boldsymbol{\alpha} \leq C, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_4, M \succeq 0 \end{aligned} \tag{3.3}$$

问题 (3.3) 与下列问题等价:

$$\begin{aligned} &\min_{M.\delta} \quad \delta \\ &s.t. \quad \delta \leq \max_{\boldsymbol{\alpha}} 2\boldsymbol{\alpha}^T\mathbf{e} - \langle K \circ \boldsymbol{\alpha}\boldsymbol{\alpha}^T, M\rangle, 0 \leq \boldsymbol{\alpha} \leq C, \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_4, M \succeq 0 \end{aligned} \tag{3.4}$$

令 $G(K) = M \circ K$，上述问题的拉格朗日函数为:

$$L(\boldsymbol{\alpha}, \boldsymbol{\mu}, \boldsymbol{\nu}) = 2\boldsymbol{\alpha}^T\mathbf{e} - \boldsymbol{\alpha}^T G(K)\boldsymbol{\alpha} + 2\boldsymbol{\mu}\boldsymbol{\alpha} + 2\boldsymbol{\nu}(C - \boldsymbol{\alpha}) \tag{3.5}$$

拉格朗日函数 L 对 $\boldsymbol{\alpha}$ 求偏导可得:

$$\frac{\partial L}{\partial \boldsymbol{\alpha}} = 0 \Longrightarrow \boldsymbol{\alpha} = G(K)^{-1}(\mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu}) \tag{3.6}$$

将上式代入拉格朗日函数可得

$$\begin{aligned} W(\boldsymbol{\mu}, \boldsymbol{\nu}) &= \max_{\boldsymbol{\alpha}} \min_{\boldsymbol{\mu}\geq 0, \boldsymbol{\nu}\geq 0} L \\ &= \min_{\boldsymbol{\mu}\geq 0, \boldsymbol{\nu}\geq 0} \max_{\boldsymbol{\alpha}} L \\ &= \min_{\boldsymbol{\mu}\geq 0, \boldsymbol{\nu}\geq 0} (\mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu})^T G(K)^{-1}(\mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu}) + 2C\mathbf{v}^T\mathbf{e} \end{aligned} \tag{3.7}$$

那么要使得 $W(\boldsymbol{\mu}, \boldsymbol{\nu}) \leq \delta$，必存在 $\boldsymbol{\mu} \geq 0, \boldsymbol{\nu} \geq 0$，使得:

$$(\mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu})^T G(K)^{-1}(\mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu}) + 2C\mathbf{v}^T\mathbf{e} \leq \delta$$

由 Schur 补引理即可得到等价的问题 (3.3)：

$$\begin{aligned} \min_{M,\delta,\boldsymbol{\mu},\boldsymbol{\nu}} \quad & \delta \\ s.t. \quad & \mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_4, \boldsymbol{\mu} \geq 0, \boldsymbol{\nu} \geq 0, M \succeq 0 \\ & \begin{bmatrix} M \circ K & \mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu} \\ (\mathbf{e} + \boldsymbol{\mu} - \boldsymbol{\nu})^T & \delta - 2C\mathbf{v}^T\mathbf{e} \end{bmatrix} \succeq 0 \end{aligned} \tag{3.8}$$

求解半定规划问题 (3.8) 得到最优的矩阵 M^*，训练样本的类标记 $\mathbf{y} = \sqrt{\lambda}\mathbf{v}$，其中 $\lambda, \mathbf{v}$ 分别是矩阵 M^* 最大的特征值和相应的特征向量。

3.3 本章小结

MMC 将 SVM 中间隔最大化学习策略和核技巧推广到无监督的聚类学习中，其最终目标是为无标记的训练数据添加类别标记，使得在该组类别标记下的训练数据经过 SVM 的训练学习后，能够得到最大的间隔。但是在模型构造的过程中，需要去掉偏置参数 b 以避免非凸问题的出现，并且为了得到 SDP 问题松弛类标记核矩阵约束，这会导致损失部分参数空间。同时 MMC 与其它核方法一样，难以选择合适的核函数，这些都对 MMC 的聚类性能和适用性造成一定的影响。就 MMC 过程中的非凸整形优化问题而言，目前已经有一些不同的优化方法能够解决这个问题，除了这里提到的 SDP，还有替代优化以及割平面算法。针对 MMC 的局限性和新方法的出现，接下来一章将会介绍使用割平面算法，并引入多核学习的多核聚类算法。

第四章　多核聚类 MKC

经过上一章的分析，MMC 的基本原理和模型推导过程都已经十分清晰。但很容易想到，使用半定规划 (SDP) 解决凸整形优化问题 (3.1) 的过程损失了部分参数空间，使得最终求解的最优类标记与实际的类标记之间有一定的偏差。并且 MMC 与其它核方法一样，核函数的选择将直接决定模型性能的好坏，但目前如何选择合适的核函数仍然是未解决的问题。针对 MMC 的适用性，在本章中，受到有监督学习中多核学习工作[25] 的鼓舞，在 MMC 模型的基础上，引入多核学习的思想得到多核聚类 (MKC)。MKC 针对 MMC 中的非凸整形优化问题 (3.1)，使用割平面算法[26] 和凹凸规划进行求解。最终得到最大间隔超平面、最适当的类标记组合以及最优的核函数组合。

4.1　MKC 模型

4.1.1　多核学习

MMC 与其它核方法一样，都依赖于将数据样本映射到高维的特征空间。但它们都面临一个核心的问题，对于一个特定的任务，不清楚哪个核函数最适合。因此，最近 SVM 和其它核方法从一系列性质相同或者不同的核中构造出一个核，而不是使用单一固定的核，这样的工作已经显示出令人激动的结果[25]。允许数据样本在不同特征空间的映射合并为一个基核，这样带来很大的灵活性。基于这些启发，MKC 将 MMC 的单核学习改进为多核学习，详细的说，MKC 考虑将 M 个特征映射 $\Phi_1, \cdots, \Phi_M$ (对应于 M 个基核 $K_1, \cdots, K_M$) 进行非负线性组合，公式化描述如下：

$$\Phi(\mathbf{x}) = \sum_{k=1}^{M} \beta_k \Phi_k(\mathbf{x}) \tag{4.1}$$

其中，$\beta_k \geq 0$，并且存在整数 p，使得 $\sum_k \beta_k^p \leq 1$。通过同时考虑到超平面参数 (权值 $\mathbf{w}$ 和偏置 b) 和权重参数 β_k 来优化 MMC 中的目标函数，就能得到最优的 MMC 特征映射。

4.1.2 优化过程

论文的后面将会看到，原始 MKC 最优化问题的约束条件数非常之多，难以在有限的时间内求解，因此可以通过割平面法[27] 求解。通过构造一个可嵌套的、逐渐逼近原始 MKC 最优化问题的松弛序列，序列中的每个最优化问题都可以看作二阶锥规划 (SOCP)[28]，并能通过凹凸规划 (CCCP)[29] 来求解。

4.2 模型推导

4.1.1 单核最大间隔聚类

在第三章中已经介绍 MMC 的核心思想，就是将最大间隔标准从有监督的学习推广到无监督的学习中。在二类聚类情况下，给予数据集：$X = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$，MMC 的目标是寻找最优的标签集合 $\mathbf{y} = \{y_1, \cdots, y_m\} \in \{-1, +1\}^m$，使得 SVM 在数据集 $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$ 上训练并产生最大间隔。对最优化问题 (3.1) 重新公式化描述如下：

$$\begin{aligned} \min_{y\in\{\pm1\}^m} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ s.t. \quad & \forall i \in \{1, \cdots, m\}: \\ & y_i(\mathbf{w}^T\Phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \xi_i \geq 0, \\ & -l \leq \sum_{i=1}^{n} y_i \leq l. \end{aligned} \tag{4.2}$$

其中，数据样本 X 被映射到高维特征空间，Φ 是线性或非线性特征映射。在支持向量机中，通常是使用其对偶形式进行训练的，借助核方法来隐含使用 Φ。$\Phi(\mathbf{x})$ 可以通过计算相应的核矩阵 K 的乔姆斯基 (Cholesky) 分解得到，也即 $K = \hat{X}\hat{X}^T, \Phi(\mathbf{x}_i) = (\hat{X}_{i,1}, \cdots, \hat{X}_{i,m})^T$，或者使用特征分解也能得到相同的结果。

此外，问题 (4.2) 中的最后一个约束是类平衡约束，其目的是避免所有的训练样本点都被分配相同的类标签。这里 $l > 0$ 是控制类平衡的常数。

根据最优化问题 (4.2)，单核最大间隔聚类在最大化间隔的同时考虑类标记向量 $\mathbf{y}$ 和分离超平面参数 $(\mathbf{w}, b)$。未知的二值向量 $\mathbf{y}$ 使最优化问题 (4.2) 变成整形规划，比 SVM 中的二次规划 (QP) 问题更难解决。然而，根据参考文献 [26]，我们能将单核最大间隔聚类问题 (4.2) 等价的公式化描述为：

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ s.t. \quad & \forall i \in \{1,\cdots,m\}: \\ & \left|\mathbf{w}^T\Phi(\mathbf{x}_i) + b\right| \geq 1 - \xi_i, \xi_i \geq 0, \\ & -l \leq \sum_{i=1}^{n}\left[\mathbf{w}^T\Phi(\mathbf{x}_i) + b\right] \leq l. \end{aligned} \tag{4.3}$$

其中，标签向量 $\mathbf{y}$ 由 $y_i = \mathrm{sign}(\mathbf{w}^T\phi(\mathbf{x}_i) + b)$ 计算得到，最后一个约束是松弛类平衡约束。与问题 (4.2) 相比，问题 (4.3) 更加容易处理。

4.2.2 多核最大间隔聚类

由于对某个特定的问题，核函数的适用性是未知的。因此这里利用多个核函数的非负线性组合构造基核，再将这个基核来代替单核最大间隔聚类中的核函数进行训练。详细的说，就是将输入空间中的每个数据样本 $\mathbf{x}_i$ 通过 M 个映射 $\Phi_k : \mathbf{x} \mapsto \Phi(\mathbf{x}) \in \mathbb{R}^{D_k}, k = 1,\cdots,M$ 转换为 M 个特征向量 $\Phi_1(\mathbf{x}_i),\cdots,\Phi_M(\mathbf{x}_i)$。这里 D_k 表示第 k 个特征空间的维度。对于每个特征映射来说，都有一个独立的权值向量 $\mathbf{w}_k$。

从而得到下面的优化问题，当 $M=1$ 时与问题 (4.3) 等价。

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\sum_{k=1}^{M}\beta_k\|\mathbf{w}_k\|^2+\frac{C}{m}\sum_{i=1}^{m}\xi_i \\
s.t. \quad & \forall i\in\{1,\cdots,m\}: \\
& \left|\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i)+b\right|\geq 1-\xi_i,\xi_i\leq 0, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^p\leq 1, \\
& -l\leq\sum_{i=1}^{n}\left[\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i)+b\right]\leq l
\end{aligned} \tag{4.4}$$

其中，权值 β_k 是用来规则化 M 个输出函数，权值的非负性约束是为了保证核函数之间的线性组合具有凸性，并且能得到基核是半正定的。此外，这里的 p 是一个正整数，这里设定 $p=2$，也就是说，使用 l_2 范数对 $\boldsymbol{\beta}=(\beta_1,\cdots,\beta_M)^T$ 进行规则化。

从问题 (4.4) 中很容易看到，由于成对的参数 β_k 和 $\mathbf{w_k}$ 使得目标函数、第一个约束以及最后一个约束都是非凸的。因此，这里需要对变量作一些调整：

$$\forall k\in\{1,\cdots,M\}:\mathbf{v}_k=\beta_k\mathbf{w}_k. \tag{4.5}$$

进过上面的转换，得到与问题 (4.4) 等价的多核 MMC 公式化描述：

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{v},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\sum_{k=1}^{M}\frac{\|\mathbf{v}_k\|^2}{\beta_k}+\frac{C}{m}\sum_{i=1}^{m}\xi_i \\
s.t. \quad & \forall i\in\{1,\cdots,m\}: \\
& \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\geq 1-\xi_i,\xi_i\leq 0, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^p\leq 1, \\
& -l\leq\sum_{i=1}^{n}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right]\leq l
\end{aligned} \tag{4.6}$$

其中，$\mathbf{v}=(\mathbf{v}_1,\cdots,\mathbf{v}_M)^T$。现在除了第一个约束条件外，其余的约束条件以及目标函数都具有凸性。

4.2.3 割平面算法

问题 (4.6) 中有 m 个松弛变量 ξ_i，对应于每个数据样本。接下来首先对问题 (4.6) 重新公式化描述，减少松弛变量的数量。

定理 4.1 多核 MMC 可以被等价的公式化描述为：

$$\min_{\boldsymbol{\beta},\mathbf{v},b,\xi} \quad \frac{1}{2}\sum_{k=1}^{M}\frac{\|\mathbf{v}_k\|^2}{\beta_k}+C\xi \tag{4.7}$$

$$\begin{aligned}
s.t. \quad & \forall\mathbf{c}\in\{0,1\}^m: \\
& \frac{1}{m}\sum_{i=1}^{m}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\geq\frac{1}{m}\sum_{i=1}^{m}c_i-\xi, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^p\leq 1,\xi\geq 0, \\
& -l\leq\sum_{i=1}^{m}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right]\leq l
\end{aligned} \tag{4.8}$$

证明 为了简单起见，用 OP1 表示最优化问题 (4.6)，用 OP2 表示最优化问题 (4.7)。证明理论 (4.1) 成立等价于证明 OP1 和 OP2 有相同的最优目标值和等价的约束条件。详细的说，需要证明对于每个 $(\mathbf{v}, b, \boldsymbol{\beta})$，最优的 ξ^* 和 $\{\xi_1^*, \cdots, \xi_m^*\}$ 之间满足 $\xi^* = \frac{1}{m}\sum_{i=1}^{m}\xi_i^*$。这就意味着，当 $(\mathbf{v}, b, \boldsymbol{\beta})$ 固定，$(\mathbf{v}, b, \boldsymbol{\beta}, \xi^*)$ 和 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi_1^*, \cdots, \xi_m^*)$ 分别是 OP1 和 OP2 的最优解，最终得到相同的目标值。

首先，注意到对于任意的 $(\mathbf{v}, b, \boldsymbol{\beta})$，OP1 中的每个松弛变量 ξ_i 都能被单独的优化：

$$\xi_i^* = \max\left\{0, 1 - \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \tag{4.9}$$

对于 OP2 来说，最优的松弛变量 ξ 是：

$$\xi^* = \max_{c\in\{0,1\}^m}\left\{\frac{1}{m}\sum_{i=1}^{m}c_i - \frac{1}{m}\sum_{i=1}^{m}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \tag{4.10}$$

因为在等式 (4.9) 中 c_i 是互不相关的，因此它们也能被单独的优化：

$$\begin{aligned}\xi^* &= \sum_{i=1}^{m}\max_{c_i\in\{0,1\}^m}\left\{\frac{1}{m}c_i - \frac{1}{m}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \\ &= \frac{1}{m}\sum_{i=1}^{m}\max\left\{0, 1 - \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \\ &= \frac{1}{m}\sum_{i=1}^{m}\xi_i^*.\end{aligned} \tag{4.11}$$

因此，对于任意的 $(\mathbf{v}, b, \boldsymbol{\beta})$，给予最优的 ξ^* 和 $\{\xi_1^*, \cdots, \xi_m^*\}$，OP1 和 OP2 有相同的目标值。所以，这两个优化问题最优值是相同的。也就是说，我们能通过求解最优化问题 (4.7) 来得到多核 MMC 的解。定理4.1 得证。

在最优化问题 (4.7) 中，松弛变量减少了 $m-1$ 个，所有的非凸约束都共用同一个松弛变量 ξ，这在很大程度上降低了多核 MMC 中非凸最优化问题的复杂度。另一方面，方程 (4.8) 中约束条件的数量从 m 增加到 2^m，这种指数级增长十分惊人。然而，割平面算法总能找到整个约束集合的一个小的子集，并且在此约束子集上求解得到的结果

仍然能保证足够的精度，从而解决多核 MMC 问题。详细的说，首先初始化一个空的约束子集 Ω，计算在满足约束 Ω 下问题 (4.7) 的最优解；然后割平面算法会寻找 (4.8) 中最违背的约束并添加到约束子集 Ω 中。通过这种思想，割平面算法构造出一系列逐渐逼近原始多核 MMC 问题的近似值。如果约束集合 (4.8) 中在 ϵ 范围内没有约束是违背的，那么算法终止。算法 1 是完整的多核 MMC 割平面算法。

算法 1 多核最大间隔聚类的割平面算法

Input：M 个特征映射 $\Phi_1, \cdots, \Phi_M$，参数 C，l 和 ϵ，约束子集 $\Omega = \phi$
repeat
  在当前约束子集 Ω 下求解问题 (4.7)，得到 $(\mathbf{v}, b, \boldsymbol{\beta})$
  选择最违背的约束 $\mathbf{c}$，令 $\Omega = \Omega \cup \{\mathbf{c}\}$
until
  新选择的约束 $\mathbf{c}$ 违背的程度小于 ϵ

在上述割平面算法中还存在两个问题：

1. 在给予的约束子集 Ω 下如何求解问题 (4.7)?

2. 如何在约束集合 (4.8) 中找出最违背的约束?

这些将在接下来的两个小节中讨论。

4.2.3.1 通过 CCCP 优化

在割平面算法的每一轮迭代中，需要在当前的约束子集 Ω 下求解非凸最优化问题 (4.7)，从而得到最优的分离超平面。尽管问题 (4.7) 中的目标函数是凸的，但约束并不是凸的，这使得问题很难求解。幸运的是，凹凸规划[29](CCCP) 能解决这类最优化问题。详细的说，问题 (4.7) 中的目标函数是二次的，并且除了第一个约束外所有的约束都是线性的。而且，注意到尽管约束集合 (4.8) 中的约束是非凸的，但能够写成两个凸函数的差值：

$$\begin{aligned} &\forall \mathbf{c} \in \Omega: \\ &\left(\frac{1}{m}\sum_{i=1}^{m} c_i - \xi\right) - \frac{1}{m}\sum_{i=1}^{m} c_i \left|\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right| \leq 0. \end{aligned} \tag{4.12}$$

因此，可以通过下面的步骤利用 CCCP 求解最优化问题 (4.7)。首先初始化 $(\mathbf{v}^{(0)}, b^{(0)})$，通过将 $\frac{1}{m}\sum_{i=1}^{m} c_i \left|\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right|$ 替换为其在 $(\mathbf{v}^{(t)}, b^{(t)})$ 处的一阶泰勒

展开式，CCCP 能够从 $(\mathbf{v}^{(t)}, b^{(t)})$ 计算得到 $(\mathbf{v}^{(t+1)}, b^{(t+1)})$。问题 (4.7) 变成：

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{v},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\sum_{k=1}^{M}\frac{\|\mathbf{v}_k\|^2}{\beta_k} + C\xi \\
s.t. \quad & \forall \mathbf{c} \in \Omega : \\
& \frac{1}{m}\sum_{i=1}^{m} c_i \le \xi + \frac{1}{m}\sum_{i=1}^{m} c_i z_i^{(t)} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right] \\
& \forall k \in \{1, \cdots, M\} : \beta_k \ge 0, \\
& \sum_{k=1}^{M} \beta_k^2 \le 1, \xi \ge 0, \\
& -l \le \sum_{i=1}^{m} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right] \le l
\end{aligned} \tag{4.13}$$

其中 $z_i^{(t)} = \text{sign}(\sum_{k=1}^{M} \mathbf{v}_k^{(t)T} \Phi_k(\mathbf{x}_i) + b^{(t)})$。引入额外的变量 t_k 作为 $\frac{\|\mathbf{v}_k\|^2}{\beta_k}$ 的上界，我们能将上面的问题公式化描述为二阶锥规划 (SOCP) 问题：

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{v},b,\boldsymbol{\xi},\mathbf{t}} \quad & \frac{1}{2}\sum_{k=1}^{M} t_k + C\xi \\
s.t. \quad & \forall \mathbf{c} \in \Omega : \\
& \frac{1}{m}\sum_{i=1}^{m} c_i \le \xi + \frac{1}{m}\sum_{i=1}^{m} c_i z_i^{(t)} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right] \\
& \forall k \in \{1, \cdots, M\} : \\
& \left\| \begin{bmatrix} 2\mathbf{v}_k \\ t_k - \beta_k \end{bmatrix} \right\| \le t_k + \beta_k, \beta_k \ge 0, \\
& \sum_{k=1}^{M} \beta_k^2 \le 1, \xi \ge 0, \\
& -l \le \sum_{i=1}^{m} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right] \le l
\end{aligned} \tag{4.14}$$

这里应用了一个定理，就是形如 $\mathbf{s}^T\mathbf{s} \le xy$, $(x, y \in \mathbb{R}_+, \mathbf{s} \in \mathbb{R}^n)$ 的双曲线约束能被

等价地转换为二阶锥约束[30, 31]：

$$\left\| \begin{bmatrix} 2\mathbf{s} \\ x-y \end{bmatrix} \right\| \leq x+y, \tag{4.15}$$

由 CCCP 可知，SOCP 问题求得的解 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi, \mathbf{t})$ 将作为 $(\mathbf{v}^{(t+1)}, b^{(t+1)}, \boldsymbol{\beta}, \xi, \mathbf{t})$，然后一直迭代直到收敛为止。算法 2 总结了在满足约束子集 Ω 下求解问题 (4.14) 的方法，检查前后两次迭代之间目标函数的差值是否小于 $\alpha\%$ (实验中通常将其设置为 0.01) 作为其终止条件。

算法 2 满足约束子集 Ω 条件下通过 CCCP 求解问题 (4.7)

初始化 $(\mathbf{v}^{(0)}, b^{(0)})$
repeat
　求得 $(\mathbf{v}^{(t+1)}, b^{(t+1)}, \boldsymbol{\beta}, \xi, \mathbf{t})$ 作为 SOCP 问题 (4.14) 的解。
　令 $\mathbf{v} = \mathbf{v}^{(t+1)}, b = b^{(t+1)}, t = t+1$。
until
　满足停止准则。

4.2.3.2 最违背的约束

问题 (4.7) 中最违背的约束很容易定义，其约束的可行性由相应的值 ξ 来度量。因此，最违背的约束条件也就是其相应值 ξ 最大。我们用向量 c 来表示约束集合 (4.8) 中的每个约束，因此有下面的定理：

定理 4.2 问题 (4.7) 中最违背的约束可以按下面方式计算：

$$c_i = \begin{cases} 1 & \text{if } \left|\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.16}$$

证明 最违背的约束条件也就是其相应值 ξ 最大，为了满足问题 (4.7) 中所有的约束，最优的 ξ 值计算如下：

$$\begin{aligned} \xi^* &= \sum_{i=1}^{m} \max_{c_i \in \{0,1\}} \left\{ \frac{1}{m} c_i - \frac{1}{m} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \right\} \\ &= \frac{1}{m} \sum_{i=1}^{m} \max_{c_i \in \{0,1\}} \left\{ c_i \left[1 - \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \right] \right\} \end{aligned} \tag{4.17}$$

因此，最违背的约束 c 相应的 ξ^* 能够通过等式 (4.16) 获得。

割平面算法在迭代时，会选择在当前超平面参数下最违背的约束并将其添加到约束子集 Ω，直到在 ϵ 范围内没有约束是违背的。此外，ξ 与问题 (4.7) 中的约束集合的可行性有着直接的对应关系，如果点 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi)$ 在精度 ϵ 内满足所有的约束，也就是说：

$$\begin{aligned} &\forall \mathbf{c} \in \{0,1\}^n : \\ &\frac{1}{m} \sum_{i=1}^{m} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \geq \frac{1}{m} \sum_{i=1}^{m} c_i - (\xi + \epsilon) \end{aligned} \tag{4.18}$$

那么点 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi + \epsilon)$ 也是可行的。此外，注意到在问题 (4.7) 的目标函数中，松弛变量 ξ 度量了其聚类损失。因此，可以将所有的训练数据都满足不等式 (4.18) 作为算法 1 的终止条件。

4.3 本章小结

MKC 对 MMC 的局限性进行改进，引入多核学习的思想，使用多个核函数的非负线性组合得到的基核进行训练，并使用割平面算法构造一系列逐渐逼近原始多核 MMC 问题的序列，并且序列中的每个问题都可以通过 CCCP 进行求解。MKC 最终能在训练数据上寻找到最大间隔超平面、最适合的类标记组合以及最优的核函数组合。

第五章　实验

通过前面几章对 MMC、MKC 模型的分析，本章将使用 UCI 上的数据集验证模型的性能。所有实验的运行环境是 MATLAB 8.6，操作系统为 Windows 10，硬件环境为 3.4GHz Inter Core i7 处理器、8GB 内存的 PC 上，使用的凸优化工具包为 CVX 和 Mosek 的组合。

5.1　数据集

本次实验所使用的数据集全部来自 UCI 仓库，受计算资源等因素的限制，只选择了 digits1v7、ionosphere 和 ringnorm 三个数据集，数据描述如表5-1 所示。

表 5-1: 数据集描述

数据	大小	维度	类别
digits1v7	776	64	2
ionosphere	354	64	2
ringnorm	1000	20	2

5.2　聚类精度

首先，本文从聚类精度来验证 MMC 和 MKC 的性能。这里的聚类精度是指在最终的聚类结果中，类标记正确的数据样本个数占数据样本总数的百分比。对于 MKC，这里使用了三个核函数，分别是：线性核函数 $k_1(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T\mathbf{x}_2$，多项式核函数 $k_2(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1^T\mathbf{x}_2 + 1)^d$，高斯核函数 $k_3(\mathbf{x}_1, \mathbf{x}_2) = \exp(-(\mathbf{x}_1 - \mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2)/2\sigma)$。使用 iterSVM[32] 运行 MMC 算法，同时为了方便对比，本文使用 k 均值聚类和谱聚类作为基准。至于 MKC 模型中的多个参数，本文采用网格搜索，选择其使得聚类结果最优的一组参数建立模型，待搜索的参数空间见表5-2。

为了评估聚类精度，本文采用的策略是：首先采用有类标记的数据集，去掉所有的类标记并运行聚类算法，然后将聚类得到的结果与类标记进行对比，计算聚类精度。依

据此策略，实验最终运行得到的聚类结果见表5-3。

表 5-2: MKC 的参数空间

参数	取值空间
C	2^{-6} 2^{-5} 2^{-4} 2^{-3} 2^{-2} 2^{-1} 2^{0} 2^{1} 2^{2} 2^{3} 2^{4} 2^{5} 2^{6}
d	2^{0} 2^{1} 2^{2} 2^{3} 2^{4} 2^{5} 2^{6} 2^{7} 2^{8} 2^{9} 2^{10}
σ	0.5 1 2 5 7 10 12 15 17 20

表 5-3: 聚类精度 (%)

数据	K_1	K_2	K_3	MKC	iterSVM	KM	PC
digits1v7	99.61	99.61	97.62	99.74	99.36	99.23	50.26
ionosphere	77.78	81.48	78.06	87.46	63.84	71.23	72.60
ringnorm	74.50	96.90	98.70	96.50	79.11	75.55	92.27

从表5-3 可以看出，MKC 算法的聚类精度确实比 MMC、k 均值和谱聚类高。在 digits1v7、ionosphere 和 ringnorm 三个数据集中，k 均值和谱聚类的精度明显受到数据集内部分布的影响，存在忽高忽低的情况。而 MKC 相比较 k 均值和谱聚类而言，其聚类结果更加稳定，而且受到数据集内部分布的影响程度不大。但是 MKC 算法的稳定性不如 k 均值和谱聚类，从图5.1、图5.2 和图5.3 可以看出，MKC 算法在割平面迭代过程中，其聚类精度是十分不稳定的。随着迭代次数的增加，其聚类精度呈现出上下跳跃的状态，因此，即使割平面最终收敛，但得到的模型的聚类精度可能并不高。所以本文以割平面过程中使得聚类精度最高的参数生成 MKC 模型，而非最终迭代终止时的参数。另外，MKC 的模型参数在很大程度上直接决定其聚类精度的高低，因此，搜索参数空间，选取最优的模型参数是 MKC 模型必不可少的过程。

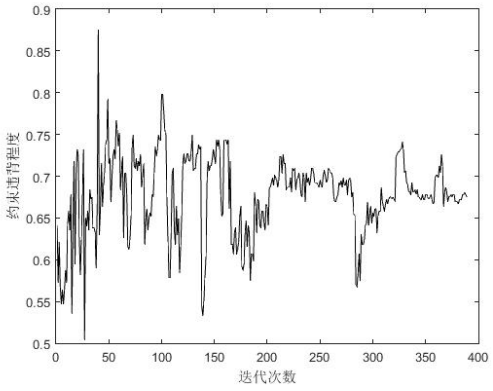

图 5.1: ionosphere 割平面收敛过程

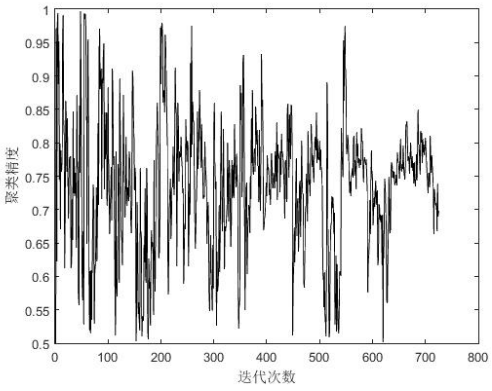

图 5.2: digits1v7 割平面收敛过程

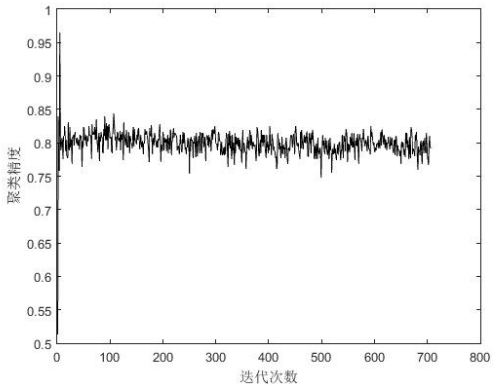

图 5.3: ringnorm 割平面收敛过程

5.3 速度

关于 MKC 的潜在担忧可能是时间复杂度问题。显然，MKC 算法中割平面过程的收敛时间由数据规模和终止条件 ϵ 共同决定。因此论文从这两个方面进行了相关实验，结果见图5.4。从图5.4 可以看出，随着数据规模的增大，MKC 算法的收敛时间随着终止条件 ϵ 的减小而迅速增大，甚至在有限的时间内无法收敛。另外从图5.5、图5.6 和图5.7 可以看到，MKC 的割平面过程收敛十分缓慢，并且存在上下波动的情况，但其总体趋势是下降的，这也符合割平面方法的思想。

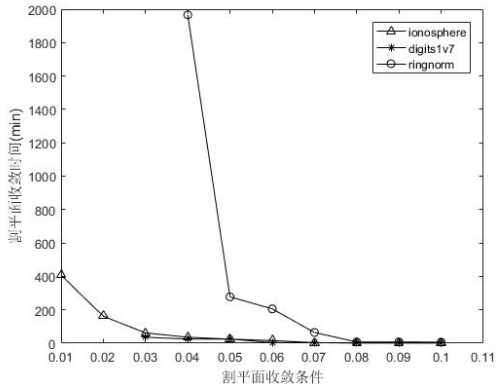

图 5.4: MKC 算法的收敛情况

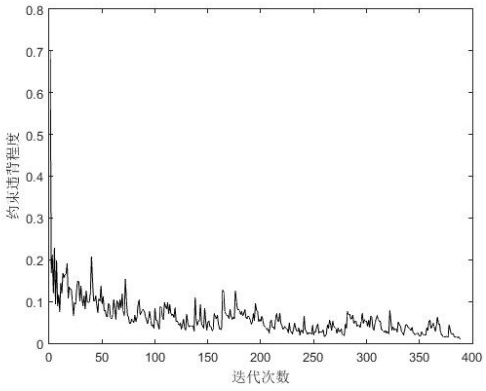

图 5.5: ionosphere 割平面收敛过程

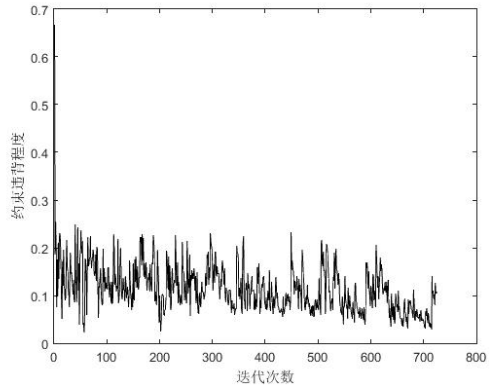

图 5.6: digits1v7 割平面收敛过程

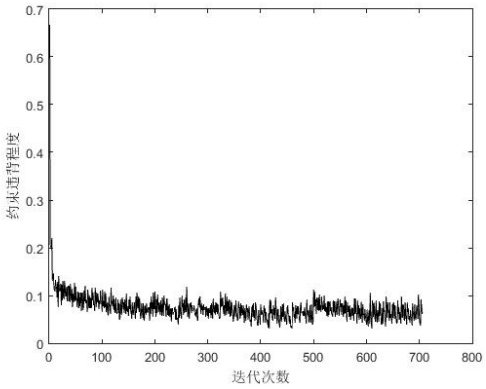

图 5.7: ringnorm 割平面收敛过程

5.4 泛化能力

MKC 采用了 SVM 中的间隔最大化学习策略，因此本文实验也测试了 MKC 在未知数据上的泛化能力。首先从总数据集上随机获取一定规模的子集作为训练数据集，剩下的数据集作为测试数据集。然后利用训练数据集学习得到 MKC 模型，并使用此模型在测试数据集上进行聚类。

表 5-4: MKC 的泛化能力 (%)

数据	训练数据集	测试数据集
digits1v7	99.74	98.92
ringnorm	87.46	87.19

观察表5-4，发现对数据集的子集进行训练得到的模型在测试数据集上也能取得较好的效果，也即泛化性能较好。受这个结果的启发，在对数据集进行学习训练的过程中，若数据集规模较大，可以以一定的策略 (比如随机) 选择数据集的一个小的子集作为训练集，再在这个训练集上进行训练，得到的 MKC 模型在整个数据集上同样能得到很好的聚类效果。

5.5 本章小结

本章使用 UCI 上的数据集进行实验来验证 MKC 模型的性能，并与传统的 k 均值聚类和谱聚类进行比较。从前面的实验可以看出，与传统的 k 均值和谱聚类相比，MKC 算法的聚类精度较高，较好的克服了数据集内部结构对模型聚类精度的影响。但 MKC 模型的训练过程需要较长的时间，并且其割平面迭代过程很不稳定，每轮割平面得到的模型进行聚类时，得到的精确度跳跃特别大，上下剧烈抖动，造成割平面收敛时得到的模型不一定是最优的，本文以迭代过程中聚类精度最高的参数作为最优模型的参数。此外，MKC 算法具有较好的泛化性能，这使得 MKC 算法能轻松应对大规模数据集。

第六章　结束语

6.1　本文工作小结

本文就如何将 SVM 中的间隔最大化学习策略和核技巧应用到无监督的聚类学习中进行了深入研究。首先，分别从 SVM 的主问题和对偶问题出发，分析了 SVM 的学习问题，讨论了模型的构建、优化问题的推导以及二者之间的关系等问题，并研究了间隔最大化学习策略和核技巧在 SVM 中的应用方法。

其次，将 SVM 推广到无监督学习中，为无类别标记的训练数据添加一组标记，使得其经过 SVM 训练学习后，能得到最大的间隔，这就是最大间隔聚类 MMC。MMC 通过使用一系列线性约束来替换原始非凸整形规划问题的非凸约束，得到凸整形规划问题，然后再松弛其中的整形约束，从而得到最终的 SDP 问题，并能通过现有的半定规划工具包进行求解。但 MMC 在松弛约束的过程中会损失部分参数空间，并且与其它核方法一样，寻找合适的核函数是个比较困难的问题。

进一步，在 MMC 的基础上引入有监督和半监督学习中多核学习的思想，使用多个核函数的非负线性组合得到新的基核，并使用此基核进行训练、学习。并针对 MMC 中出现的非凸整形规划问题，MKC 应用割平面算法，构造一系列逐渐逼近该问题的序列，并且序列中的每个问题都可以使用 CCCP 求解。MKC 最终能在训练数据上找到最大间隔超平面、最适当的类标记组合以及最优的核函数组合。

最后，在对 MMC 和 MKC 模型进行详细探讨之后，通过实验来验证其性能的优越性。本文使用 UCI 上的数据集，对 MMC、MKC、k 均值和谱聚类的聚类精度进行比较，并对 MKC 的聚类性能作出评价。

6.2　进一步的工作

本文深入分析了 MMC 和 MKC 的聚类原理以及模型推导过程，实现了间隔最大化学习策略和核技巧向聚类的推广，并引入有监督和半监督学习中的多核思想，最终实现

基于间隔最大化学习策略和核技巧的多核学习聚类模型。在此基础上，本文还有以下工作可以进一步拓展和完善：

1. 选用计算能力更强的设备，使用更多的数据集对模型性能进行测试，以便客观的评价模型的性能。

2. 分析 MKC 算法的时间复杂度和聚类精度，为 MKC 算法的收敛速度和聚类精度提供理论支持。

3. 考虑多分类 SVM 模型，并在此基础上尝试将 MKC 推广为多类分类模型，再进行相关实验验证多类 MKC 模型的性能。

致　谢

时光如白驹过隙，转眼间我即将完成四年的本科生生涯。这三年多的时间见证了我的成长，一路走来，收获太多。此时此刻我的心中充满了感动，在这里我要感谢所有师长、同学和家人的指导、支持和帮助。

首先我要感谢王敏老师和东南大学的薛晖老师以及李森学长。历时半年之久的毕业设计，从最初的选题、写开题报告，到后来的初稿、定稿，都不是一帆风顺，因为她(他) 们的严格要求、督促和指导，让我顺利完成本科毕设设计。在这里我要对她 (他) 们给予我的指导和帮助致以最衷心的感谢。

其次，我要感谢河海大学计算机与信息学院 12 级辅导员姜彬彬老师对我的教诲和关怀，以及所有授课教师的授业解惑，他们在我人生的成长道路上留下了浓墨重彩的一笔，在这里我要对他们给予我生活和学习上的帮助致以最崇高的敬意。

此外，我要感谢我的室友在这三年多的时间里给予我的关怀与帮助，是他们为我营造了积极和睦的生活环境。

最后，在我即将走出校门，踏入研究生生涯之际，我要感谢我的父母和亲人。没有他们这么多年来对我的关怀、支持和鼓励，就不会有今天的我。

仅以此文献给所有关心和帮助过我的人。

参考文献

[1] 周志华. *机器学习*. 清华大学出版社, 2016.

[2] Leonard Kaufman and Peter Rousseeuw. *Clustering by means of medoids*. North-Holland, 1987.

[3] Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data mining and knowledge discovery*, 2(3):283–304, 1998.

[4] James C Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013.

[5] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *The Journal of Machine Learning Research*, 6:1705–1749, 2005.

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

[7] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[8] Dan Pelleg, Andrew W Moore, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML*, volume 1, 2000.

[9] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[10] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. In *ACM Sigmod Record*, volume 28, pages 49–60. ACM, 1999.

[11] Alexander Hinneburg and Daniel A Keim. An efficient approach to clustering in large multimedia databases with noise. In *KDD*, volume 98, pages 58–65, 1998.

[12] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[13] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM Sigmod Record*, volume 25, pages 103–114. ACM, 1996.

[14] Saikat Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 512–521. IEEE, 1999.

[15] David MJ Tax and Robert PW Duin. Support vector domain description. *Pattern recognition letters*, 20(11):1191–1199, 1999.

[16] Asa Ben-Hur, David Horn, Hava T Siegelmann, and Vladimir Vapnik. Support vector clustering. *The Journal of Machine Learning Research*, 2:125–137, 2002.

[17] 吕常魁, 姜澄宇, and 王宁生. 一种支持向量聚类的快速算法. *华南理工大学学报*, 33(1):6–9, 2005.

[18] Yuichiro Anzai. *Pattern Recognition & Machine Learning*. Elsevier, 2012.

[19] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Leaning(ECML)*, pages 137–142. Chemnitz,Germany, 1998.

[20] Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *Advances in neural information processing systems*, pages 1537–1544, 2004.

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

[22] Vapnik N Vladimir and V Vapnik. The nature of statistical learning theory, 1995.

[23] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[24] 李航. *统计学习方法*. 清华大学出版社, 2012.

[25] Alexander Zien and Cheng Soon Ong. Multiclass multiple kernel learning. *Icml Acm*, 7(2006):2007, 2007.

[26] Bin Zhao, Fei Wang, and Changshui Zhang. Efficient maximum margin clustering via cutting plane algorithm. In *SDM*, pages 751–762. SIAM, 2008.

[27] James E Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[28] Boyd, Vandenberghe, and Faybusovich. Convex optimization. *IEEE Transactions on Automatic Control*, 51(11):1859–1859, 2006.

[29] Alan L Yuille and Anand Rangarajan. The concave-convex procedure. *Neural computation*, 15(4):915–936, 2003.

[30] Yurii Nesterov, Arkadii Nemirovskii, and Yinyu Ye. *Interior-point polynomial algorithms in convex programming*, volume 13. SIAM, 1994.

[31] Ivor Wai-Hung Tsang and James Tin-Yau Kwok. Efficient hyperkernel learning using second-order cone programming. *Neural Networks, IEEE Transactions on*, 17(1):48–58, 2006.

[32] Kai Zhang, Ivor W Tsang, and James T Kwok. Maximum margin clustering made practical. *Neural Networks, IEEE Transactions on*, 20(4):583–596, 2009.

算法源码

本文中的 MKC 算法的核心 Matlab 代码如下:

```
% 参数介绍
% data为待输入的训练数据，矩阵类型，每行表示一条训练数据，每列表示训
    练数据的属性
% label为训练数据的标签，列向量，元素取值为-1或1
% C为松弛参数
% d为多项式核函数的参数
% sigma为高斯核函数的参数
% epsilon为割平面过程的终止条件，这里默认为0.05
% alpha为CCCP过程的终止条件，这里默认为0.01
% best_v为在当前模型参数下，割平面过程中使得聚类精度最高的权值参数
% best_b为在当前模型参数下，割平面过程中使得聚类精度最高的偏置参数
% max_acc为在当前模型参数下的最高聚类精确度
function [ best_v, best_b, max_acc ] = mkc( data, label, C, d, sigma,
    epsilon, alpha )
if nargin == 5  % 设置默认值
    epsilon = 0.05;
    alpha = 0.01;
end
[m,n] = size(data);
l = m/3;        % 类平衡约束条件
M = 3;     % 核函数个数,这里包括线性核函数、多项式核函数、高斯核函数
linear_kernel = data * data';   % 线性核函数生成核矩阵
polynomial_kernel = (data * data'+1).^d; % 多项式核函数生成核矩阵
guass_kernel = zeros(m,m);
for i = 1:m
    for j = 1:m
        data_temp = data(i,:) - data(j,:);
        guass_kernel(i,j) = exp(-data_temp*data_temp'/(2*sigma));
                % 高斯核函数生成核矩阵
```

```
    end
end
X = {};
X{1} = getFeature(linear_kernel);       % 根据核矩阵获取其特征
X{2} = getFeature(polynomial_kernel);   % 根据核矩阵获取其特征
X{3} = getFeature(guass_kernel);        % 根据核矩阵获取其特征
C_Omega = {};           % 保存所有最违背的约束，也即约束子集Omega
max_acc = -inf;
% 开始割平面过程，在每轮割平面中进行CCCP过程，并求解每轮CCCP中的最优
    化问题
while 1 % 割平面迭代过程
    v_k = ones(m,M);
    b_k = 1;
    res_temp = inf;
    while 1             % CCCP迭代过程
        cvx_begin sdp quiet             % 使用cvx优化包，一次CCCP求解
            过程
            cvx_solver mosek            % 使用mosek求解器
            variables betta(M) T(M)
            variables xi(1) b(1)
            variable v(m,M)
            minimize(0.5*sum(T)+C*xi)   % 最小化目标函数
            subject to          % 所有约束条件
                % 基础约束
                for i=1:M
                    norm([2*v(:,i);T(i)-betta(i)]) <= T(i)+betta(i);
                    betta(i) >= 0;
                end
                sum(betta.^2) <= 1;
                xi >= 0;
                l_temp = zeros(1,m);
                for i=1:M
                    l_temp = l_temp + v(:,i)'*X{i}';
```

```
            end
            -l <= sum(l_temp+b) <= l;
            % 由最违背约束子集Omega生成的约束条件
            if numel(C_Omega) > 0
                z_temp = zeros(1,m);
                for i=1:M
                    z_temp = z_temp + v_k(:,i)'*X{i}';
                end
                z = sign(z_temp+b_k)';
                for i=1:numel(c_arr)
                    1/m*sum(C_Omega{i}) <= xi+1/m*((C_Omega{i}.*z
                        )'*(l_temp+b)');
                end
            end
    cvx_end         % 最优化问题求解完毕
    res = cvx_optval;
    % 判断是否满足CCCP迭代终止条件
    if abs((res-res_temp)/res) <= alpha
        break;
    end
    v_k = v;
    b_k = b;
    res_temp = res;
end % CCCP迭代终止，进入割平面过程
temp = zeros(1,m);
for i=1:M
    temp = temp + v(:,i)' * X{i}';
end
temp = (temp + b)';
c = zeros(m,1);     % 初始化最违背约束对应的c向量
xi_m = 0;
for i = 1:m     % 寻找最违背的约束
    if abs(temp(i)) < 1
```

```
            c(i) = 1;
        else
            c(i) = 0;
        end
        xi_m = xi_m + c(i) * (1 - abs(temp(i)));
    end
    xi_m = xi_m / m;
    cluster = sign(temp);       % 聚类生成的样本类别标记
    acc = sum(label==cluster)/m;        % 计算聚类准确率
    if acc < 0.5                % 保证多数原则
        acc = 1 - acc;
    end
    if max_acc < acc            % 判断本次迭代的聚类准确率是否最优
        max_acc = acc;
        best_v = v_k;
        best_b = b_k;
    end
    if xi_m-xi <= epsilon       % 判断是否满足割平面过程的终止条件
        break;
    end
    % 将最违背的约束加入约束子集，再进入CCCP过程
    C_Omega{numel(C_Omega)+1} = c;
end
end

% 特征提取函数参数介绍
% K为由核函数生成的样本矩阵，也即核矩阵
% X为对核矩阵进行特征分解得到的特征向量
function [ X ] = getFeature( K )
n = length(K);
[p,x] = eig(K);
for i=1:n
    if x(i,i) < 0
```

```
        x(i,i) = 0;
    end
    x(i,i) = x(i,i)^0.5;
end
X = p * x;
end
```

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 optimization* [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 $K_1, \ldots, K_M$):

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

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.

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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:

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

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

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

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

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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

$$
\begin{aligned}
(2.4)\min_{\boldsymbol{\beta},\mathbf{w},b,\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\beta_k||\mathbf{w}_k||^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\
\text{s.t.} \quad & \forall i \in \{1,\ldots,n\}: \\
& \left|\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i)+b\right| \geq 1-\xi_i,\ \xi_i \geq 0, \\
& \forall k \in \{1,\ldots,M\}:\ \beta_k \geq 0, \\
& \sum_{k=1}^{M}\beta_k^p \leq 1, \\
& -l \leq \sum_{i=1}^{n}\left[\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i)+b\right] \leq l.
\end{aligned}
$$

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]

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

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

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

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:*

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

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

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

For OP2, the optimal slack variable ξ is

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

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

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

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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:

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

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:

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

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

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

cone programming (SOCP) [3] problem:

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

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:

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

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.,

$$
\begin{aligned}
(2.18)\ & \forall\ \mathbf{c} \in \{0,1\}^n : \\
& \frac{1}{n} \sum_{i=1}^{n} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \geq \frac{1}{n} \sum_{i=1}^{n} c_i - (\xi + \epsilon),
\end{aligned}
$$

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*

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

plane algorithm terminates only when the newly selected constraint $\mathbf{c}$ 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 $\mathbf{c}$ 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* k*th 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

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

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 $\mathbf{d}_k \in \mathbb{R}$, 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 $\beta \geq 0$ 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

$$\begin{aligned} &\forall \mathbf{c} \in \Omega: \ \frac{1}{n}\sum_{i=1}^{n} c_i t_i \geq \frac{1}{n}\sum_{i=1}^{n} c_i - \xi, \\ &\forall i \in \{1,\ldots,n\}: \ t_i^2 \leq \mathbf{v}^T \boldsymbol{\Psi}_i \mathbf{v}, \\ &\forall i \in \{1,\ldots,n\}: \ t_i \geq 0, \end{aligned}$$

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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

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

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

$$\begin{aligned} &L(\boldsymbol{\lambda}, \boldsymbol{\gamma}, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho) \\ &= \inf_{\mathbf{v},\boldsymbol{\beta},\xi,t} \left\{ \frac{1}{2}\sum_{k=1}^{M} \frac{||\mathbf{v}_k||^2}{\beta_k} + C\xi + \sum_{p=1}^{|\Omega|} \lambda_p \left[\frac{1}{n}\sum_{i=1}^{n} c_{pi}(1-t_i) - \xi \right] \right. \\ &\quad + \sum_{i=1}^{n} \gamma_i (t_i^2 - \mathbf{v}^T \mathbf{\Psi}_i \mathbf{v}) - \mu\xi - \sum_{i=1}^{n} \delta_i t_i - \sum_{k=1}^{M} \alpha_k \beta_k \\ &\quad \left. + \rho \left(\sum_{k=1}^{M} \beta_k - 1 \right) \right\} \\ &= \inf_{\mathbf{v},\beta,\xi,t} \left\{ \frac{1}{2}\sum_{k=1}^{M} \frac{||\mathbf{v}_k||^2}{\beta_k} - \mathbf{v}^T \sum_{i=1}^{n} \gamma_i \mathbf{\Psi}_i \mathbf{v} + C\xi - \sum_{p=1}^{|\Omega|} \lambda_p \xi - \mu\xi \right. \\ &\quad + \sum_{i=1}^{n} \gamma_i t_i^2 - \sum_{p=1}^{|\Omega|} \lambda_p \frac{1}{n} \sum_{i=1}^{n} c_{pi} t_i - \sum_{i=1}^{n} \delta_i t_i + \sum_{p=1}^{|\Omega|} \lambda_p \frac{1}{n} \sum_{i=1}^{n} c_{pi} \\ &\quad \left. - \sum_{k=1}^{M} \alpha_k \beta_k + \rho \left(\sum_{k=1}^{M} \beta_k - 1 \right) \right\} \\ &= \sum_{i=1}^{n} \left\{ -\frac{(\sum_{p=1}^{|\Omega|} \lambda_p c_{pi} + n\delta_i)^2}{4n^2\gamma_i} + \frac{1}{n}\sum_{p=1}^{|\Omega|} \lambda_p c_{pi} \right\} - \rho \end{aligned} \tag{3.21}$$

satisfying the following constraints

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

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

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

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

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

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

$$\begin{aligned} &L_*^{(t+1)} \\ &= \max_{\boldsymbol{\lambda},\boldsymbol{\gamma},\mu,\boldsymbol{\delta},\boldsymbol{\alpha},\rho} \sum_{i=1}^{n} \left\{ -\frac{(\sum_p \lambda_p c_{pi} + \lambda_{t+1} c_i' + n\delta_i)^2}{4n^2\gamma_i} \right. \\ &\quad \left. + \frac{1}{n}\left[\sum_{p=1}^{t} \lambda_p c_{pi} + \lambda_{t+1} c_i' \right] \right\} - \rho \\ &\geq L_*^{(t)} + \max_{\lambda_{t+1} \geq 0} \sum_{i=1}^{n} \left\{ -\frac{\lambda_{t+1} c_i' \sum_{p=1}^{t} \lambda_p^{(t)} c_{pi}}{2\gamma_i^{(t)} n^2} \right. \\ &\quad \left. - \frac{\lambda_{t+1} c_i' \delta_i^{(t)}}{2\gamma_i^{(t)} n} - \frac{(\lambda_{t+1} c_i')^2}{4\gamma_i^{(t)} n^2} + \frac{1}{n}\lambda_{t+1} c_i' \right\}. \end{aligned} \tag{3.25}$$

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 $L_*^{(t+1)}$:

$$\begin{aligned} L_*^{(t+1)} &\geq L_*^{(t)} + \max_{\lambda_{t+1} \geq 0} \left\{ -\frac{1}{n}\sum_{i=1}^{n} \lambda_{t+1} c_i' + \epsilon\lambda_{t+1} \right. \\ &\quad \left. - \sum_{i=1}^{n} \frac{(\lambda_{t+1} c_i')^2}{4\gamma_i^{(t)} n^2} + \sum_{i=1}^{n} \frac{1}{n}\lambda_{t+1} c_i' \right\} \\ &= L_*^{(t)} + \max_{\lambda_{t+1} \geq 0} \left\{ \epsilon\lambda_{t+1} - \sum_{i=1}^{n} \frac{(\lambda_{t+1} c_i')^2}{4\gamma_i^{(t)} n^2} \right\} \\ &= L_*^{(t)} + \frac{\epsilon^2}{\sum_{i=1}^{n} (c_i'^2 / \gamma_i^{(t)} n^2)}. \end{aligned} \tag{3.26}$$

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

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

subject to the following equation

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

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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

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

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:

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

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:

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

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:

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

where we have applied the following change of variables

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

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.

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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

$$
\begin{aligned}
(4.33)\min_{\boldsymbol{\beta},\mathbf{v},\boldsymbol{\xi}}\ & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k}+\frac{C}{n}\sum_{i=1}^{n}\xi_i \\
s.t.\ & \forall i\in\{1,\ldots,n\}, r\in\{1,\ldots,m\}: \\
& \sum_{k=1}^{M}\left(\sum_{p=1}^{m}z_{ip}\mathbf{v}_k^p-\mathbf{v}_k^r\right)^T\Phi_k(\mathbf{x}_i)+z_{ir}\geq 1-\xi_i, \\
& \forall i\in\{1,\ldots,n\}:\ \xi_i\geq 0, \\
& \forall k\in\{1,\ldots,M\}:\ \beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1, \\
& \forall p,q\in\{1,\ldots,m\}: \\
& -l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l,
\end{aligned}
$$

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

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

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

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

$$
\begin{aligned}
(4.34)\ \min_{\boldsymbol{\beta},\mathbf{v},\boldsymbol{\xi}}\ & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k}+C\xi \\
s.t.\ & \forall\mathbf{c}_i\in\{\mathbf{e}_0,\mathbf{e}_1,\ldots,\mathbf{e}_k\}, i\in\{1,\ldots,n\}: \\
& \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}(\mathbf{c}_i^T\mathbf{e}z_{ip}-c_{ip})\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i) \\
& +\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}\geq\frac{1}{n}\sum_{i=1}^{n}\mathbf{c}_i^T\mathbf{e}-\xi, \\
& \forall k\in\{1,\ldots,M\}:\ \beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1,\ \xi\geq 0, \\
& \forall p,q\in\{1,\ldots,m\}: \\
& -l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l.
\end{aligned}
$$

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 $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}\mathbf{c}_i^T\mathbf{e}z_{ip}\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i)+\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}$ in the constraint with its first-order Taylor expansion at $\mathbf{v}^{(t)}$.

$$
\begin{aligned}
(4.35)\ \min_{\boldsymbol{\beta},\mathbf{v},\xi}\ & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{||\mathbf{v}_k^p||^2}{\beta_k}+C\xi \\
s.t.\ & \forall[\mathbf{c}_1,\ldots,\mathbf{c}_n]\in\Omega, i\in\{1,\ldots,n\}: \\
& \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}(\mathbf{c}_i^T\mathbf{e}z_{ip}^{(t)}-c_{ip})\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i) \\
& +\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}^{(t)}\geq\frac{1}{n}\sum_{i=1}^{n}\mathbf{c}_i^T\mathbf{e}-\xi, \\
& \forall k\in\{1,\ldots,M\}:\ \beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1,\ \xi\geq 0, \\
& \forall p,q\in\{1,\ldots,m\}: \\
& -l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l.
\end{aligned}
$$

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.

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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$^{\text{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*.

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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	-	-		

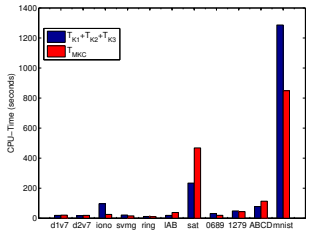

Figure 1: CPU-time (seconds) of *MKC* and *MMC* as a function of the data set size n.

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.

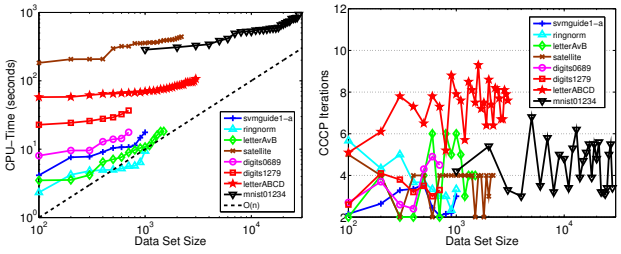

Figure 2: Left: CPU-time of MKC vs. data set size. Right: Average number of CCCP iterations in MKC vs. data set size.

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

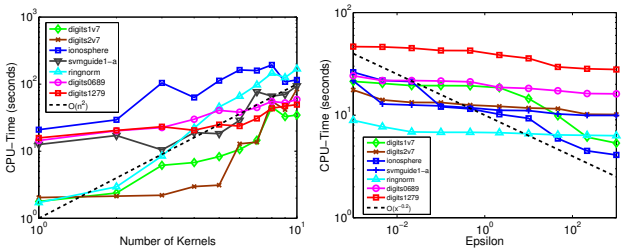

Figure 3: Left: CPU-time of MKC vs. the number of kernels. Right: CPU-time of MKC vs. ϵ.

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.

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

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

Supported by the projects (60835002) and (60675009) of NSFC, and Competitive Earmarked Research Grant (CERG) 614508 from the Research Grants Council of the Hong Kong Special Administrative Region, China.

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.

Copyright © by SIAM.
Unauthorized reproduction of this article is prohibited.

多核聚类

摘要

最近在数据挖掘和机器学习领域，最大间隔聚类 (MMC) 受到了极大的关注。最大间隔聚类首先将数据样本映射到核特征空间，然后在所有可能的类标记组合上寻找最大间隔超平面，完成聚类工作。与其它的核方法一样，要想使得最大间隔聚类算法十分成功，选择一个合适的核函数是必不可少的。在这篇论文中，我们提出了多核聚类算法 (MKC)，能同时寻找到最大间隔超平面、最适当的类标记组合以及最优的核组合。此外我们对 MKC 算法的时间复杂度进行了详细的分析，并且将二类多核聚类推广到多类情况。最后，我们使用玩具数据集和现实数据集进行了相关实验，实验结果都证实了多核聚类较高的性能。

1 简介

过去十年间，在相关文献中已经提出了一些聚类算法，包括 k 均值聚类[1]、混合模型[1] 和谱聚类[2, 3, 4]。但最近在数据挖掘和机器学习领域，最大间隔聚类 (MMC) 受到了极大的关注[5, 6, 7, 8, 9, 10]。MMC 的主要思想是将支持向量机 (SVM) 中的最大间隔标准推广到无监督学习方法中。详细的说，在给予的数据样本集中，MMC 在所有可能的类标记组合上为样本添加一组标记，使得经过 SVM 训练后得到最大的间隔，从而达到聚类目的。最近的研究表明，MMC 算法比传统的聚类方法更加优越。

然而，有监督的最大间隔方法通常都能被公式化的描述为凸优化问题，但 MMC 会导致非凸整形优化问题的出现，使得问题更难去解决。但是，最近已经有一些不同的优化方法能够解决这个问题，例如半定规划 (SDP)[5, 6, 7]、替代优化[8] 以及割平面法[9, 10]。

此外，MMC 与其它核方法一样，依赖于核函数对数据样本的映射。因此，选择一个适当的核函数将直接决定 MMC 的性能好坏。但一般而言，使用核方法都会遇到一个

核心问题，就是对于一个特别的任务而言，不清楚哪个核函数是最适合的[11, 12, 13, 14]。因此，最近 SVM 和其它核方法从一系列性质相同或不同的核中构造出一个核，而不是使用单一固定的核，这样的改进已经显示了令人激动的结果[15, 16, 17, 13, 18, 19, 20, 21, 22]。这样能产生额外的灵活性，允许将不同领域的先验知识合并为基核。然而，目前这些工作都只关注于有监督和半监督学习。因此，如何在无监督学习，尤其是最大间隔聚类中有效的学习出适当的核函数，仍然是一个十分有趣但尚未被考虑的主题。

在这篇论文中，我们提出了多核聚类算法，其原理就是在所有可能的类标记组合中，考虑最优的核特征映射，并寻找最大间隔超平面。详细的说，我们考虑将 M 个特征映射 $\Phi_1, \cdots, \Phi_M$ (对应于 M 个基核 $K_1, \cdots, K_M$) 结合起来：

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

其中，$\beta_k \geq 0$，存在整数 p，使得 $\sum_k \beta_k^p \leq 1$。通过同时考虑超平面的参数 (权值 $\mathbf{w}$ 和偏置 b) 和核函数的组合参数 β_k 来优化 MMC 中的目标函数，我们能得到最优的 MMC 特征映射。

多核聚类中的最优化问题能通过割平面算法[23] 求解。后面我们将会看到，通过构造一个可嵌套的松弛问题序列来逐渐逼近原 MKC 问题，并且序列中每个最优化问题都可以看作二阶锥规划 (SOCP)[24]，并能通过凹凸规划 (CCCP)[25] 来求解。最终在玩具数据集和现实数据集的实验结果都证实了多核聚类优越的性能。

这篇论文的剩余部分组织如下：在第二部分，我们首先介绍在二类聚类的前提下多核聚类的原理，并将原始整数规划转化为一组凸问题序列，从而能够通过割平面算法有效求解。在第三部分，我们对 MKC 算法的时间复杂度进行理论分析。在第四部分将多核聚类从二类推广到多类。第五部分是玩具数据集和现实数据集的实验结果。第六部分是一些总结和评论。

2 多核聚类

在这个部分，我们首先介绍多核聚类中的二类聚类过程，在第四部分将会讨论多类情况。

2.1 最大间隔聚类

在第一部分我们已经短暂的介绍了 MMC 的核心思想，就是将最大间隔标准从有监督的学习推广到无监督的学习中。在二类聚类情况下，给予数据集：$\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$，MMC 的目标是寻找最优的类标记组合 $\mathbf{y} = \{y_1, \cdots, y_n\} \in \{-1, +1\}^n$，使得 SVM 在数据集 $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$ 上训练并产生最大间隔。公式化描述如下：

$$\begin{aligned} \min_{y\in\{\pm 1\}^n} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & \forall i \in \{1, \cdots, n\}: \\ & y_i(\mathbf{w}^T\Phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \xi_i \geq 0, \\ & -l \leq \sum_{i=1}^{n} y_i \leq l. \end{aligned} \tag{2.1}$$

其中，数据样本 $\mathbf{X}$ 被映射到高维特征空间，Φ 可能是非线性特征映射。在 SVM 中，通常是使用其对偶形式进行训练的，并借助核方法来隐含使用 Phi。为了防止原始的优化问题 (2.2) 更倾向于非线性核，我们通过核标准组件分析[26]，仍然能得到核特征空间中每个样本有穷维的描述。或者计算核矩阵的乔姆斯基 (Cholesky) 分解[27]，即 $\mathbf{K} = \hat{\mathbf{X}}\hat{\mathbf{X}}^T$，然后设置 $\Phi(\mathbf{x}_i) = (\hat{\mathbf{X}}_{i,1}, \cdots, \hat{\mathbf{X}}_{i,n})^T$。

此外，问题 (2.1) 中的最后一个约束是类平衡约束，目的是避免所有的训练样本点都被分配相同的类标记。这里 $l > 0$ 是控制类平衡的常数。

根据问题 (2.1)，最大间隔聚类在最大化间隔时同时考虑类标记向量 $\mathbf{y}$ 和分离超平面参数 $(\mathbf{w}, b)$。未知的二值向量 $\mathbf{y}$ 使问题 (2.1) 变成整形规划，这比 SVM 中的二次规划 (QP) 问题更难解决。但是参考[9]，我们能将最大间隔聚类问题等价的公式化描述为：

$$\begin{aligned} \min_{\mathbf{w},b,\xi_i} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & \forall i \in \{1, \cdots, n\}: \\ & |\mathbf{w}^T\Phi(\mathbf{x}_i) + b| \geq 1 - \xi_i, \xi_i \geq 0, \\ & -l \leq \sum_{i=1}^{n}[\mathbf{w}^T\Phi(\mathbf{x}_i) + b] \leq l. \end{aligned} \tag{2.2}$$

其中，标签向量 $\mathbf{y}$ 由 $y_i = \text{sign}(\mathbf{w}^T\phi(\mathbf{x}_i) + b)$ 计算得到，最后一个约束是紧松弛类

平衡约束[4]。与问题 (2.1) 相比，问题 (2.2) 更加容易处理。

2.2 多核最大间隔聚类

传统的 MMC 通过一个固定的特征映射 Φ 将数据映射到高维特征空间。因此选择一个合适的核函数是将直接决定 MMC 的性能好坏。但是，通常情况下我们不知道哪个核函数最适合于手头的工作。因此在这篇论文中，受到有监督学习中多核学习[15, 16, 17, 13, 18, 19, 20, 21, 22] 工作的鼓舞，我们提出使用几个基核的非负组合来作为 MMC 中的特征映射。

详细的说，就是输入空间中的每个数据样本 $\mathbf{x}_i$ 通过 M 个映射 $\Phi_k : \mathbf{x} \mapsto \Phi(\mathbf{x}) \in \mathbb{R}^{D_k}, k = 1, \cdots, M$ 转换为 M 个特征空间 $\Phi_1(\mathbf{x}_i), \cdots, \Phi_M(\mathbf{x}_i)$。这里 D_k 表示第 k 个特征空间的维度。对于每个特征映射来说，都有一个独立的权值向量 $\mathbf{w}_k$ 与之对应。从而得到下面的优化问题，当 $M = 1$ 时与问题 (2.2) 等价。

$$
\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{w},b,\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\beta_k\|\mathbf{w}_k\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\
s.t. \quad & \forall i \in \{1, \cdots, n\} : \\
& \left|\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i) + b\right| \geq 1 - \xi_i, \xi_i \leq 0, \\
& \forall k \in \{1, \cdots, M\} : \beta_k \geq 0, \\
& \sum_{k=1}^{M}\beta_k^p \leq 1, \\
& -l \leq \sum_{i=1}^{n}\left[\sum_{k=1}^{M}\beta_k\mathbf{w}_k^T\Phi_k(\mathbf{x}_i) + b\right] \leq l
\end{aligned}
\tag{2.3}
$$

其中，我们根据权值 β_k 来调整 M 个特征映射的输出。权值的非负性约束是为了保证特征映射的组合具有凸性，并且能得到半正定核。此外，这里的 p 是一个正整数。在这篇论文中，我们选择 $p = 2$。

从问题 (2.3) 中我们很直观的看到一个缺点，由于变量 β_k 和 $\mathbf{w}_k$ 同时存在，使得目标函数和第一个约束以及最后一个约束都是非凸的。因此，我们对变量进行下面的变换：

$$
\forall k \in \{1, \cdots, M\} : \mathbf{v}_k = \beta_k\mathbf{w}_k. \tag{2.4}
$$

经过上面的变换后，多核 MMC 等价的公式化描述如下：

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{v},b,\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\frac{\|\mathbf{v}_k\|^2}{\beta_k}+\frac{C}{n}\sum_{i=1}^{n}\xi_i \\
s.t. \quad & \forall i\in\{1,\cdots,n\}: \\
& \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\geq 1-\xi_i,\xi_i\leq 0, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^p\leq 1, \\
& -l\leq\sum_{i=1}^{n}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right]\leq l
\end{aligned} \tag{2.5}$$

其中，$\mathbf{v}=(\mathbf{v}_1,\cdots,\mathbf{v}_M)^T$。现在除了第一个约束条件，其余的约束条件以及目标函数都是凸的。

2.3 割平面算法

最优化问题 (2.5) 中有 n 个松弛变量 ξ_i，与每个数据样本相对应。接下来，我们首先对问题 (2.5) 重新公式化描述，减少松弛变量的数量。

定理 2.1 多核 MMC 能被等价的公式化描述为：

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

$$\begin{aligned}
s.t. \quad & \forall\mathbf{c}\in\{0,1\}^n: \\
& \frac{1}{n}\sum_{i=1}^{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right|\geq\frac{1}{n}\sum_{i=1}^{n}c_i-\xi, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^p\leq 1,\xi\geq 0, \\
& -l\leq\sum_{i=1}^{n}\left[\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i)+b\right]\leq l
\end{aligned} \tag{2.7}$$

证明 为了简单起见，我们用 OP1 表示最优化问题 (2.5)，用 OP2 表示最优化问题 (2.6)。为了证明该理论，我们需要证明 OP1 和 OP2 有相同的最优目标值以及等价的约束条件。详细的说，对于每个 $(\mathbf{v}, b, \boldsymbol{\beta})$，我们需要证明最优的 ξ^* 和 $\{\xi_1^*, \cdots, \xi_n^*\}$ 具有 $\xi^* = \frac{1}{n}\sum_{i=1}^{n}\xi_i^*$ 的等价关系。也就意味着，当 $(\mathbf{v}, b, \boldsymbol{\beta})$ 固定，$(\mathbf{v}, b, \boldsymbol{\beta}, \xi^*)$ 和 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi_1^*, \cdots, \xi_n^*)$ 分别是 OP1 和 OP2 的最优解，从而最终得到相同的目标值。

首先，注意到对于任意的 $(\mathbf{v}, b, \boldsymbol{\beta})$，OP1 中的每个松弛变量 ξ_i 都能被单独的优化：

$$\xi_i^* = \max\left\{0, 1 - \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \tag{2.8}$$

对于 OP2 来说，最优的松弛变量 ξ 是：

$$\xi^* = \max_{\mathbf{c}\in\{0,1\}^n}\left\{\frac{1}{n}\sum_{i=1}^{n}c_i - \frac{1}{n}\sum_{i=1}^{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \tag{2.9}$$

因为在等式 (2.9) 中 c_i 是互不相关的，因此它们也能被单独的优化：

$$\begin{aligned}\xi^* &= \sum_{i=1}^{n}\max_{c_i\in\{0,1\}}\left\{\frac{1}{n}c_i - \frac{1}{n}c_i\left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \\ &= \frac{1}{n}\sum_{i=1}^{n}\max\left\{0, 1 - \left|\sum_{k=1}^{M}\mathbf{v}_k^T\Phi_k(\mathbf{x}_i) + b\right|\right\} \\ &= \frac{1}{n}\sum_{i=1}^{n}\xi_i^*.\end{aligned} \tag{2.10}$$

因此，对于任意的 $(\mathbf{v}, b, \boldsymbol{\beta})$，给予最优的 ξ^* 和 $\{\xi_1^*, \cdots, \xi_n^*\}$，OP1 和 OP2 有相同的目标值。因此这两个优化问题的最优值是相同的。也就是说，我们能通过求解最优化问题 (2.6) 来得到多核 *MMC* 的解。

在最优化问题 (2.6) 中，松弛变量的数量减少了 $n-1$ 个，所有的非凸约束都共用同一个松弛变量 ξ。这极大地降低了多核 MMC 中非凸最优化问题的复杂度。另一方面，方程 (2.7) 中约束条件的数量从 n 增加到 2^n，这种约束条件的指数级增长看起来十分吓人，但是，我们会发现总能找到整个约束集合的一个小的约束子集，并且仍然能确保得到足够精确的解。详细的说，我们使用自适应的割平面算法[23] 来解决多核 MMC 问题。首先初始化一个空的约束子集 Ω，在满足约束 Ω 下求解问题 (2.6) 的最优解。然后算法会寻找约束集合 (2.7) 中最违背的约束并添加到约束子集 Ω 中。通过这种方法，我们构

算法 1　多核最大间隔聚类的割平面算法

输入：M 个特征映射 $\Phi_1, \cdots, \Phi_M$，参数 C，l 和 ϵ，约束子集 $\Omega = \phi$

repeat

　　在当前工作约束集 Ω 下求解问题 (2.6)，得到 $(\mathbf{v}, b, \boldsymbol{\beta})$

　　选择最违背的约束 $\mathbf{c}$，令 $\Omega = \Omega \cup \{\mathbf{c}\}$

until

　　新选择的约束 $\mathbf{c}$ 违背的差值小于 ϵ

造出一系列逐渐逼近原始多核 MMC 的问题序列。如果约束集合 (2.7) 中在 ϵ 范围内没有约束是违背的，那么算法终止。算法 1 完整的描述了多核 MMC 割平面算法。

我们将会在第三部分证明总能寻找到多项式大小的约束子集，使得内相应的满足 (2.7) 中所有约束的松弛问题的解满足精度 ϵ 的要求。也就是说，保证剩余的约束在不超过 ϵ 范围内被违背，因此不需要将其隐含的加入最优化问题中[28]。

在多核聚类的割平面算法中还剩下两个问题，一：在给予的约束子集 Ω 下如何求解问题 (2.6)？二：怎么在约束集合 (2.7) 中找出最违背的约束？这些将在接下来的两个小节中讨论。

2.3.1　通过 CCCP 求解

在割平面算法的每一轮迭代中，我们需要在当前的工作约束集 Ω 下求解非凸最优化问题，从而得到最优的分离超平面。尽管问题 (2.6) 中的目标函数是凸的，但约束并不是凸的，这使得问题很难求解。幸运的是，凹凸规划 (CCCP) 能解决这一类目标函数凹凸和约束凹凸[25] 的最优化问题。详细的说，问题 (2.6) 中的目标函数是二次的，并且除了第一个约束外所有的约束都是线性的。而且，注意到尽管 (2.7) 中的约束是非凸的，但能够写成两个凸函数的差值：

$$\begin{aligned} &\forall \mathbf{c} \in \Omega : \\ &\left(\frac{1}{n}\sum_{i=1}^{n} c_i - \xi\right) - \frac{1}{n}\sum_{i=1}^{n} c_i \left|\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right| \leq 0. \end{aligned} \tag{2.11}$$

因此，我们按下面的方法使用 CCCP 来求解问题 (2.6)。初始化 $(\mathbf{v}^{(0)}, b^{(0)})$，通过将 $\frac{1}{n}\sum_{i=1}^{n} c_i \Big| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \Big|$ 替换为其在 $(\mathbf{v}^{(t)}, b^{(t)})$ 处的一阶泰勒展开式，CCCP 能够通过 $(\mathbf{v}^{(t)}, b^{(t)})$ 计算得到 $(\mathbf{v}^{(t+1)}, b^{(t+1)})$。问题 (2.6) 变成：

$$
\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{v},b,\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\frac{\|\mathbf{v}_k\|^2}{\beta_k} + C\xi \\
s.t. \quad & \forall \mathbf{c} \in \Omega : \\
& \frac{1}{n}\sum_{i=1}^{n} c_i \le \xi + \frac{1}{n}\sum_{i=1}^{n} c_i z_i^{(t)} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right] \\
& \forall k \in \{1, \cdots, M\} : \beta_k \ge 0, \\
& \sum_{k=1}^{M} \beta_k^2 \le 1, \xi \ge 0, \\
& -l \le \sum_{i=1}^{n} \left[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\right] \le l
\end{aligned}
\tag{2.12}
$$

其中，$z_i^{(t)} = \mathrm{sign}(\sum_{k=1}^{M} \mathbf{v}_k^{(t)T} \Phi_k(\mathbf{x}_i) + b^{(t)})$。引入额外的变量 t_k 作为 $\frac{\|\mathbf{v}_k\|^2}{\beta_k}$ 的上界，我们能将上面的问题公式化描述为二阶锥规划 (SOCP)[24] 问题：

$$
\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{v},b,\xi,\mathbf{t}} \quad & \frac{1}{2}\sum_{k=1}^{M} t_k + C\xi \\
s.t. \quad & \forall \mathbf{c} \in \Omega : \\
& \frac{1}{n}\sum_{i=1}^{n} c_i \le \xi + \frac{1}{n}\sum_{i=1}^{n} c_i z_i^{(t)} \Big[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\Big] \\
& \forall k \in \{1, \cdots, M\} : \\
& \left\| \begin{bmatrix} 2\mathbf{v}_k \\ t_k - \beta_k \end{bmatrix} \right\| \le t_k + \beta_k, \beta_k \ge 0, \\
& \sum_{k=1}^{M} \beta_k^2 \le 1, \xi \ge 0, \\
& -l \le \sum_{i=1}^{n} \Big[\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b\Big] \le l
\end{aligned}
\tag{2.13}
$$

这里我们应用了一个定理，就是形如 $\mathbf{s}^T\mathbf{s} \le xy$, $(x, y \in \mathbb{R}_+, \mathbf{s} \in \mathbb{R}^n)$ 的双曲线约束

能被等价地转换为二阶锥约束[29, 30]：

$$\left\| \begin{bmatrix} 2\mathbf{s} \\ x-y \end{bmatrix} \right\| \leq x+y, \tag{2.14}$$

上面的 SOCP 问题能够在多项式时间内求解[31]。由 CCCP 可知，从 SOCP 问题求得的解 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi, \mathbf{t})$ 被用作 $(\mathbf{v}^{(t+1)}, b^{(t+1)}, \boldsymbol{\beta}, \xi, \mathbf{t})$，然后一直迭代直到收敛为止。算法 2 总结了在满足约束子集 Ω 下求解问题 (2.6) 的方法。至于它的终止条件，我们检查前后两次迭代的目标值之间的差值是否小于 $\alpha\%$ (实验中通常将其设置为 0.01)。

算法 2　满足约束子集 Ω 条件下通过 CCCP 求解问题 (2.6)

初始化 $(\mathbf{v}^{(0)}, b^{(0)})$

repeat

　　求得 $(\mathbf{v}^{(t+1)}, b^{(t+1)}, \boldsymbol{\beta}, \xi, \mathbf{t})$ 作为二阶锥规划问题 (2.13) 的解。

　　令 $\mathbf{v} = \mathbf{v}^{(t+1)}, b = b^{(t+1)}, t = t+1$。

until

　　满足停止准则。

2.3.2　最违背的约束

最优化问题 (2.7) 中最违背的约束很容易定义，其约束的可行性由相应的值 ξ 来度量。因此，最违背的条件也就是其相应值 ξ 最大。我们使用向量 c 来表示问题 (2.7) 中每个约束，因此有下面的定理：

定理 2.2 最优化问题 (2.7) 中最违背的约束可以按下面方式计算：

$$c_i = \begin{cases} 1 & \text{if } \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.15}$$

证明 最违背的条件也就是其相应值 ξ 最大，为了满足问题 (2.6) 中所有的约束，最优的 ξ 值计算如下：

$$\begin{aligned}\xi^* &= \sum_{i=1}^{n} \max_{c_i \in \{0,1\}} \left\{ \frac{1}{n} c_i - \frac{1}{n} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \right\} \\ &= \frac{1}{n} \sum_{i=1}^{n} \max_{c_i \in \{0,1\}} \left\{ c_i \left[1 - \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \right] \right\}\end{aligned} \tag{2.16}$$

因此，最违背的约束 c 相应的 ξ^* 能够通过等式 (2.15) 获得。

割平面算法在迭代时，会选择在当前超平面参数下最违背的约束并将其添加到约束子集 Ω，直到在 ϵ 范围内没有约束是违背的。此外，ξ 与问题 (2.6) 中的约束集合的可行性有着直接的对应关系，如果点 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi)$ 在精度 ϵ 内满足所有的约束，也就是说：

$$\begin{aligned}&\mathbf{c} \in \{0,1\}^n : \\ &\frac{1}{n} \sum_{i=1}^{n} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \geq \frac{1}{n} \sum_{i=1}^{n} c_i - (\xi + \epsilon)\end{aligned} \tag{2.17}$$

那么点 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi + \epsilon)$ 也是可行的。此外，注意到在问题 (2.6) 的目标函数中，有一个单松弛变量 ξ 来度量聚类损失。因此，我们可以将样本都满足不等式 (2.17) 作为算法 1 终止条件。

2.4 割平面算法的精度

接下来的定理定义了通过割平面算法计算得到的最优解的精度。

定理 2.3 对于任意的 $\epsilon > 0$，多核 MMC 的割平面算法求解得到 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi)$，那么相应的点 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi + \epsilon)$ 是问题 (2.6) 的可行解。

证明 割平面算法中，在约束集合 (2.7) 中，通过使用等式 (2.15) 选择使得 ξ 值最大的最违背的约束 $\mathbf{c}$。若最新选择的约束 $\mathbf{c}$ 在不超过 ϵ 范围内违背，也即 $\frac{1}{n}\sum_{i=1}^{n} c_i - \frac{1}{n}\sum_{i=1}^{n} c_i \left| \sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b \right| \leq \xi + \epsilon$，那么割平面算法终止。因为新选择的约束 c 已经是最违背的约束，那么所有其它的约束都将满足上述的不等式。因此，如果 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi)$ 是割平面算法返回的解，那么 $(\mathbf{v}, b, \boldsymbol{\beta}, \xi + \epsilon)$ 是问题 (2.6) 的可行解。

在这个定理的基础上，ϵ 表明了与最好的分离超平面的错误率之间的接近程度。这解释了其作为算法 1 的终止条件的原因。

3 时间复杂度分析

在这一部分，我们将深入分析多核 MMC 割平面算法的时间复杂度。

定理 3.1 多核 *MMC* 的割平面算法时间复杂度为 $O(\frac{D^{3.5}+nD}{\epsilon^2}+\frac{D^{2.5}}{\epsilon^4})$，其中 $D=\sum_{k=1}^{M}D_k$，D_k 是第 k 个特征空间的维度。

为了证明上述定理，我们首先求得该算法每次迭代花费的时间。接下来，我们要求解加入工作约束集 Ω 的总的约束数量，也即割平面算法总的迭代次数的上界。特别的，我们需要下面两个引理：

引理 3.1 多核 MMC 割平面算法每轮迭代花费时间为 $O(D^{3.5}+nD+D^{2.5}|\Omega|)$，其中 $|\Omega|$ 是工作约束集大小。

证明 割平面算法的每轮迭代涉及到两个步骤：在当前工作约束集 Ω 下通过 CCCP 求解问题 (2.6) 和选择最违背的约束。为了在当前工作约束集 Ω 下求解问题 (2.6)，我们需要求解一系列的 SOCP 问题。特别对于形式如下的 SOCP 问题：

$$\begin{aligned}\min_{x}\quad & \mathbf{f}^T x \\ s.t.\quad & \forall k\in\{1,\cdots,M\}:\|A_k x+b_k\|\le c_k^T x+d_k,\end{aligned} \tag{3.1}$$

其中，$\mathbf{x}\in\mathbb{R}^N,\mathbf{f}\in\mathbb{R}^N,\mathbf{A}_k\in\mathbb{R}^{(N_k-1)\times N},\mathbf{b}_k\in\mathbb{R}^{N_k-1},\mathbf{c}_k\in\mathbb{R}^N,\mathbf{d}_k\in R$，其每轮迭代的时间复杂度为 $O(N^2\sum_k N_k)$[31, 30]。根据问题 (2.13) 中 SOCP 问题的公式化描述，我们有：$N=\sum_{k=1}^{M}D_k+2M+2=O(D)$ 和 $\sum_k N_k=\sum_{k=1}^{M}(D_k+2)+2M+|\Omega|+3=O(D+|\Omega|)$。因此，每轮迭代的时间复杂度为 $O(D^3+D^2|\Omega|)$。使用原始的对偶方法求解这个 SOCP 问题时，通过一个绝对常量因子，并且每轮迭代增加 $O(D^{0.5})$ 的时间消耗来提高解的精度[29]。因此，CCCP 的每一轮迭代花费时间为 $O(D^{3.5}+D^{2.5}|\Omega|)$。此外，在第五部分的实验中我们能看到，在满足工作约束集 Ω 下，通过 CCCP 求解问题 (2.6) 时，割平面算法的每轮迭代的次数不超过 10 次，即使在数据集很大时也一样。因此，在满足工作约束集 Ω 下通过 CCCP 求解问题 (2.6) 的时间复杂度是 $O(D^{3.5}+D^{2.5}|\Omega|)$。最后，在使用等式 (2.15) 选择最违背的约束时，我们需要计算 n 次 $(\mathbf{v}_1,\cdots,\mathbf{v}_M)$ 和 $(\Phi_1(\mathbf{x}_i),\cdots,\Phi_M(\mathbf{x}_i))$

之间的内积，需要消耗 $O(nD)$ 的时间。因此，割平面算法每轮迭代的时间复杂度为 $O(D^{3.5} + nD + D^{2.5}|\Omega|)$。

引理 3.2 在最多添加 $\frac{CR}{\epsilon^2}$ 个约束后割平面算法会终止，其中 R 是一个与 n 和 D 无关的常数。

证明 注意到 $\mathbf{v} = 0, b = 0, \xi = 1$ 并且满足 $\sum_{k=1}^{M} \beta_k^2 \leq 1$ 的任意 $\boldsymbol{\beta} \leq 0$ 是问题 (2.6) 的一组可行解，因此，问题 (2.6) 的最优目标值的上界是 C。接下来，我们将要证明在割平面算法的每轮迭代中，在添加最违背的约束后目标值将至少增加一个常量。由于目标值是非负的并且上界为 C，因此总的迭代数量会有一个上界。简单起见，我们忽略问题 (2.6) 中的类平衡约束，并设置偏置 $b = 0$。带有类平衡约束和非零偏置项的问题的证明类似。

为了计算在向工作约束集 Ω 添加约束时带来的增加值，我们首先需要得到问题 *(2.6)* 的对偶形式。但这个过程有一个困难，就是约束中的 $|\sum_{k=1}^{M} \mathbf{v}_k^T \Phi_k(\mathbf{x}_i) + b$ 项。因此，我们首先替换该约束如下：

$$\begin{aligned}
&\forall c \in \Omega : \frac{1}{n}\sum_{i=1}^{n} c_i t_i \geq \frac{1}{n}\sum_{i=1}^{n} c_i - \xi, \\
&\forall i \in \{1, \cdots, n\} : t_i^2 \leq \mathbf{v}^T \Psi_i \mathbf{v}, \\
&\forall i \in \{1, \cdots, n\} : t_i \leq 0,
\end{aligned}$$

其中，Ψ_i 是 $D \times D$ 的矩阵，定义如下：

$$\Psi_i = \begin{bmatrix}
\Phi_1(\mathbf{x}_i)\Phi_1^T(\mathbf{x}_i) & \cdots & \Phi_1(\mathbf{x}_i)\Phi_M^T(\mathbf{x}_i) \\
\vdots & \ddots & \vdots \\
\Phi_M(\mathbf{x}_i)\Phi_1^T(\mathbf{x}_i) & \cdots & \Phi_M(\mathbf{x}_i)\Phi_M^T(\mathbf{x}_i)
\end{bmatrix} \tag{3.2}$$

令 $\boldsymbol{\lambda}, \gamma, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho$ 作为约束对应的对偶变量，问题 (2.6) 的拉格朗日对偶函数如下：

$$
\begin{aligned}
L(\boldsymbol{\lambda}, \gamma, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha}, \rho) = & \inf_{\mathbf{v}, \boldsymbol{\beta}, \xi, t} \left\{ \frac{1}{2} \sum_{k=1}^{M} \frac{\|\mathbf{v}_k\|^2}{\beta_k} + C\xi + \sum_{p=1}^{|\Omega|} \lambda_p \left[\frac{1}{n} \sum_{i=1}^{n} c_{pi}(1 - t_i) - \xi \right] \right. \\
& \left. + \sum_{i=1}^{n} \gamma_i (t_i^2 - \mathbf{v}^T \Psi_i \mathbf{v}) - \mu\xi - \sum_{i=1}^{n} \delta_i t_i - \sum_{k=1}^{M} \alpha_k \beta_k + \rho \left(\sum_{k=1}^{M} \beta_k - 1 \right) \right\} \\
= & \inf_{\mathbf{v}, \boldsymbol{\beta}, \xi, t} \left\{ \frac{1}{2} \sum_{k=1}^{M} \frac{\|\mathbf{v}_k\|^2}{\beta_k} - \mathbf{v}^T \sum_{i=1}^{n} \gamma_i \Psi_i \mathbf{v} + C\xi - \sum_{p=1}^{|\Omega|} \lambda_p \xi - \mu\xi \right. \\
& + \sum_{i=1}^{n} \gamma_i t_i^2 - \sum_{p=1}^{|\Omega|} \lambda_p \frac{1}{n} \sum_{i=1}^{n} c_{pit_i} - \sum_{i=1}^{n} \delta_i t_i + \sum_{p=1}^{|\Omega|} \lambda_p \frac{1}{n} \sum_{i=1}^{n} c_{pi} - \sum_{k=1}^{M} \alpha_k \beta_k \\
& \left. + \rho \left(\sum_{k=1}^{M} \beta_k - 1 \right) \right\} \\
= & \sum_{i=1}^{n} \left\{ - \frac{(\sum_{p=1}^{|\Omega|} \lambda_p c_{pi} + n\delta_i)^2}{4n^2 \gamma_i} + \frac{1}{n} \sum_{p=1}^{|\Omega|} \lambda_p c_{pi} \right\} - \rho
\end{aligned}
\tag{3.3}
$$

满足下列约束：

$$
\begin{cases}
\mathbf{E}_{\boldsymbol{\beta}} - 2 \sum_{i=1}^{n} \gamma_i \Psi_i \succeq 0, \\
\alpha_k - \rho = 0, \\
C - \sum_{p=1}^{|\Omega|} \lambda_p - \mu = 0, \\
t_i = \dfrac{1}{2n\gamma_i} \sum_{k=1}^{|\Omega|} \lambda_k c_{ki} + \dfrac{\delta_i}{2\gamma_i}, \\
\boldsymbol{\lambda}, \gamma, \mu, \boldsymbol{\delta}, \boldsymbol{\alpha} \geq 0,
\end{cases}
\tag{3.4}
$$

其中，$\mathbf{E}_{\boldsymbol{\beta}} = \operatorname{diag}\left(\frac{I_{D_1 \times D_1}}{\beta_1}, \cdots, \frac{I_{D_M \times D_M}}{\beta_M} \right)$，$\mathbf{I}_{D_k \times D_k}$ 是 $D_k \times D_k$ 单位矩阵。

割平面算法选择最违背的约束 $c^{'}$ 后，如果下列不等式成立将继续下去。

$$
\frac{1}{n} \sum_{i=1}^{n} c_i^{'}(1 - t_i^*) \geq \xi + \epsilon \tag{3.5}
$$

因为 $\xi \geq 0$，最新添加的约束满足不等式：

$$
\frac{1}{n} \sum_{i=1}^{n} c_i^{'}(1 - t_i^*) \geq \epsilon \tag{3.6}
$$

令 $L_*^{(t+1)}$ 为拉格朗日对偶函数在满足约束 $\Omega^{(t+1)} = \Omega^{(t)} \cup \{c^{'}\}$ 下的最优值，并令 $\gamma_i^{(t)}$ 为使得 $L_*^{(t)}$ 最大的 γ_i 值。向原始问题添加新约束等价于在对偶问题中添加新变量 λ_{t+1}，因此：

$$\begin{aligned} L_*^{(t+1)} &= \max_{\boldsymbol{\lambda},\gamma,\mu,\boldsymbol{\delta},\boldsymbol{\alpha},\rho} \sum_{i=1}^{n} \left\{ -\frac{(\sum_p \lambda_p c_{pi} + \lambda_{t+1} c_i^{'} + n\delta_i)^2}{4n^2\gamma_i} + \frac{1}{n}\left[\sum_{p=1}^{t} \lambda_p c_{pi} + \lambda_{t+1} c_i^{'}\right] \right\} - \rho \\ &\geq L_*^{(t)} + \max_{\lambda_{t+1}\geq 0} \sum_{i=1}^{n} \left\{ -\frac{\lambda_{t+1} c_i^{'} \sum_{p=1}^{t} \lambda_p^{(t)} c_{pi}}{2\gamma_i^{(t)} n^2} - \frac{\lambda_{t+1} c_i^{'} \delta_i^{(t)}}{2\gamma_i^{(t)} n} - \frac{(\lambda_{t+1} c_i)^2}{4\gamma_i^{(t)} n^2} + \frac{1}{n}\lambda_{t+1} c_i^{'} \right\} \end{aligned} \tag{3.7}$$

根据不等式 (3.6) 和约束 $\lambda_{t+1} \geq 0$，我们有：

$$\sum_{i=1}^{n} \left[-\frac{\lambda_{t+1} c_i^{'} \sum_{p=1}^{t} \lambda_p^{(t)} c_{pi}}{2\gamma_i^{(t)} n^2} + \frac{\lambda_{t+1} c_i^{'} \delta_i^{(t)}}{2\gamma_i^{(t)} n} \right] \leq \frac{1}{n}\lambda_{t+1} c_i^{'} - \epsilon\lambda_{t+1} \tag{3.8}$$

将上列不等式代入 (3.7)，我们得到 $L_*^{(t+1)}$ 的下界如下：

$$\begin{aligned} L_*^{(t+1)} &\geq L_*^{(t)} + \max_{\lambda_{t+1}\geq 0} \left\{ -\frac{1}{n}\sum_{i=1}^{n} \lambda_{t+1} c_i^{'} - \epsilon\lambda_{t+1} - \sum_{i=1}^{n} \frac{(\lambda_{t+1} c_i^{'})^2}{4\gamma_i^{(t)} n^2} + \sum_{i=1}^{n} \frac{1}{n}\lambda_{t+1} c_i^{'} \right\} \\ &= L_*^{(t)} + \max_{\lambda_{t+1}\geq 0} \left\{ \epsilon\lambda_{t+1} - \sum_{i=1}^{n} \frac{(\lambda_{t+1} c_i^{'})^2}{4\gamma_i^{(t)} n^2} \right\} \\ &= L_*^{(t)} + \frac{\epsilon^2}{\sum_{i=1}^{n} (c_i^{'}/r_i^{(i)} n^2)} \end{aligned} \tag{3.9}$$

通过最大化等式 (3.3) 中的拉格朗日对偶函数，$r^{(t)}$ 求解如下：

$$\begin{aligned} (\boldsymbol{\lambda}^{(t)}, \gamma^{(t)}, \mu^{(t)}, \boldsymbol{\delta}^{(t)}, \boldsymbol{\alpha}^{(t)}, \rho^{(t)}) &= \arg\max_{\boldsymbol{\lambda},\gamma,\mu,\boldsymbol{\delta},\boldsymbol{\alpha},\rho} \sum_{i=1}^{n} \left\{ -\frac{(\sum_{p=1}^{t} \lambda_p c_{pi} + n\delta_i)^2}{4n^2\gamma_i} + \frac{1}{n}\sum_{p=1}^{t} \lambda_p c_{pi} \right\} - \rho \\ &= \arg\max_{\lambda,\gamma,\mu,\boldsymbol{\delta},\boldsymbol{\alpha},\rho} \sum_{i=1}^{n} (\gamma_i - \delta_i) \end{aligned} \tag{3.10}$$

满足下面等式：

$$2n\gamma_i = \sum_{p=1}^{t} \gamma_p c_{pi} + n\delta_i \tag{3.11}$$

关于 δ_i 的约束只有 $\delta_i \geq 0$。因此，为了最大化 $\sum_{i=1}^{n}(\gamma_i - \delta_i)$，最优的 δ_i 值为 *0*。所以，下面等式成立。

$$2n\gamma_i = \sum_{p=1}^{t} \gamma_p c_{pi} \tag{3.12}$$

因此，$n\gamma_i^{(t)}$ 是一个与 n 无关的常数。此外，$\sum_{i=1}^{n}\frac{(c_i^{'})}{n}$ 计算约束向量 $c^{'}$ 中非零元素的比例，也与 n 无关。所以，$\sum_{i=1}^{n}\frac{(c_i^{'})}{\gamma_i^{(t)}n^2}$ 是一个与 n, D 无关的常数，我们用 $Q(t)$ 表示。此外，定义 $R=\max_t\{Q^{(t)}\}$ 为整个割平面算法中 $Q^{(t)}$ 的最大值。因此，在添加了最违背的约束 $c^{'}$ 后，拉格朗日对偶函数的目标值至少增加 $\frac{\epsilon^2}{R}$。此外，用 $G^{(t)}$ 表示在添加 t 个约束后满足 $\Omega^{(t)}$ 下问题 (2.6) 的目标函数值。由于弱对偶[24]，在最优解处有 $L_*^{(t)}\leq G_*^{(t)}\leq C$。因此拉格朗日函数的上界是 C，割平面算法在至多添加 $\frac{CR}{\epsilon^2}$ 个约束后终止。

重新观察引理 3.2，在我们的割平面算法中，其迭代的次数的上界是常数 $\frac{CR}{\epsilon^2}$，与 n, D 无关。此外，每次迭代花费时间为 $O(D^{3.5}+nD+D^{2.5}|\Omega|)$。所以，多核 MMC 个割平面算法的时间复杂度为 $\sum_{|\Omega|=1}^{CR/\epsilon^2}O(D^{3.5}+nD+D^{2.5}|\Omega|)=O(\frac{D^{3.5}+nD}{\epsilon^2}+\frac{D^{2.5}}{\epsilon^4})$。定理 3.1 得证。

4 多类多核聚类

在这一部分，我们将多核 MMC 拓展到多类聚类。

4.1 多类公式化描述

对于多类情况，我们首先介绍多类支持向量机的公式化描述[32]。给予点集 $\mathbf{X}=\{\mathbf{x}_1,\cdots,\mathbf{x}_n\}$ 和它们的类标记 $\mathbf{y}=\{y_1,\cdots,y_n\}\in\{1,\cdots,m\}^n$，SVM 为每个类 $p\in\{1,\cdots,m\}$ 定义了一个权值向量 $\mathbf{w}^p$，并通过 $p^*=\arg\max_{p\in\{1,\cdots,k\}}\mathbf{w}^{pT}\mathbf{x}$ 对样本 $\mathbf{x}$ 进行分类。权值向量通过下面方式获得：

$$\begin{aligned}\min_{\mathbf{w},\xi}\quad & \frac{1}{2}\sum_{p=1}^{m}\|\mathbf{w}^p\|^2+\frac{C}{n}\sum_{i=1}^{n}\xi_i\\ s.t.\quad & \forall i\in\{1,\cdots,n\}, r\in\{1,\cdots,m\}:\\ & \mathbf{w}^{y_iT}\Phi(\mathbf{x}_i)+\delta_{y_i,r}-\mathbf{w}^{rT}\Phi(\mathbf{x}_i)\geq 1-\xi_i,\\ & \forall i\in\{1,\cdots,n\}:\xi_i\geq 0.\end{aligned}\tag{4.1}$$

现在我们考虑使用 M 个特征映射的非负结合而不是单一的特征映射 Φ，多核多类 SVM 的公式化描述如下：

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{w},\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\beta_k\sum_{p=1}^{m}\|\mathbf{w}_k^p\|^2+\frac{C}{n}\sum_{i=1}^{n}\xi_i \\
s.t. \quad & \forall i\in\{1,\cdots,n\}, r\in\{1,\cdots,m\}: \\
& \sum_{k=1}^{M}\beta_k(\mathbf{w}_k^{y_i}-\mathbf{w}_k^r)^T\Phi_k(\mathbf{x}_i)+\delta_{y_i,r}\geq 1-\xi_i, \\
& \forall i\in\{1,\cdots,n\}:\xi_i\geq 0, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1,
\end{aligned} \tag{4.2}$$

其中，$\mathbf{w}_k^p$ 的上标 p 表示第 p 类，下标 k 表示第 k 个特征映射。与 SVM 不一样的是，SVM 是在有类标记的数据集上寻找最大间隔分类器，而 MMC 得目标是寻找一组类标记，使得最终能得到间隔最大的分类器。多核多类最大间隔聚类也可公式化描述如下：

$$\begin{aligned}
\min_{\mathbf{y},\boldsymbol{\beta},\mathbf{v},\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{\|\mathbf{v}_k^p\|^2}{\beta_k}+\frac{C}{n}\sum_{i=1}^{n}\xi_i \\
s.t. \quad & \forall i\in\{1,\cdots,n\}, r\in\{1,\cdots,m\}: \\
& \sum_{k=1}^{M}(\mathbf{v}_k^{y_i}-\mathbf{v}_k^r)^T\Phi_k(\mathbf{x}_i)+\delta_{y_i,r}\geq 1-\xi_i, \\
& \forall i\in\{1,\cdots,n\}:\xi_i\geq 0, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1, \\
& \forall p,q\in\{1,\cdots,m\}: \\
& -l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l,
\end{aligned} \tag{4.3}$$

其中我们对变量作了相关变化：

$$\forall i\in\{1,\cdots,n\},\forall k\in\{1,\cdots,M\}:\mathbf{v}_k^p=\beta_k\mathbf{w}_k^p \tag{4.4}$$

从而确保目标函数和最后一个约束是凸的。与二类聚类相似的是，我们需要类平衡约束来控制类别平衡。同样的，上述问题是一个整数规划问题，比多类 SVM 中的 QP 问题更加复杂。幸运的是，我们能得到与二类情况相似的定理：

定理 4.1 问题 (4.3) 等价于：

$$
\begin{aligned}
\min_{\mathbf{y},\boldsymbol{\beta},\mathbf{v},\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{\|\mathbf{v}_k^p\|^2}{\beta_k}+\frac{C}{n}\sum_{i=1}^{n}\xi_i \\
s.t. \quad & \forall i\in\{1,\cdots,n\}, r\in\{1,\cdots,m\}: \\
& \sum_{k=1}^{M}\Big(\sum_{p=1}^{m} z_{ip}\mathbf{v}_k^p-\mathbf{v}_k^r\Big)^T\Phi_k(\mathbf{x}_i)+z_{ir}\geq 1-\xi_i, \\
& \forall i\in\{1,\cdots,n\}:\xi_i\geq 0, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1, \\
& \forall p,q\in\{1,\cdots,m\}: \\
& -l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l,
\end{aligned}
\tag{4.5}
$$

其中 z_{ip} 定义如下：

$$
\begin{aligned}
& \forall i\in\{1,\cdots,n\}, \forall k\in\{1,\cdots,M\}: \\
& z_{ip}=\prod_{q=1,q\neq p}^{m} I_{[\sum_{k=1}^{M}\mathbf{v}_k^{rT}\Phi_k(\mathbf{x}_i)>\sum_{k=1}^{M}\mathbf{v}_k^{qT}\Phi_k(\mathbf{x}_i)]},
\end{aligned}
$$

这里 $I(\cdot)$ 是指示函数，样本 $\mathbf{x}_i$ 的标签为：$y_i=\arg\max_p\sum_{k=1}^{M}\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i)=\sum_{p=1}^{m}pz_{ip}$。

问题 (4.5) 中，多核聚类公式化描述的非凸约束中有 n 个松弛变量 $\{\xi_1,\cdots,\xi_n\}$，我们提出下面的定理来减少松弛变量的个数：

定理 4.2 令 $\xi^* = \frac{1}{n}\sum_{i=1}^{n}\xi_i^*$，问题 (4.5) 可等价的公式化描述如下：

$$
\begin{aligned}
\min_{\mathbf{y},\boldsymbol{\beta},\mathbf{v},\xi} \quad & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{\|\mathbf{v}_k^p\|^2}{\beta_k} + C\xi \\
s.t. \quad & \forall \mathbf{c}_i \in \{\mathbf{e}_0, \mathbf{e}_1, \cdots, \mathbf{e}_k\}, i \in \{1, \cdots, n\}: \\
& \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}(\mathbf{c}_i^T\mathbf{e}z_{ip} - c_{ip})\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i) + \frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip} \geq \frac{1}{n}\sum_{i=1}^{n}\mathbf{c}_i^T\mathbf{e} - \xi, \\
& \forall k \in \{1, \cdots, M\}: \beta_k \geq 0, \\
& \sum_{k=1}^{M}\beta_k^2 \leq 1, \xi \geq 0 \\
& \forall p, q \in \{1, \cdots, m\}: \\
& -l \leq \sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p - \mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i) \leq l,
\end{aligned}
\tag{4.6}
$$

其中，$\mathbf{e}_p$ 是 $k \times 1$ 的向量，其第 p 个元素为 1，其它元素为 0，$\mathbf{e}_0$ 是 $k \times 1$ 的零向量，$\mathbf{e}$ 为元素全 1 的向量。

经过上述的公式化描述之后，所有的非凸约束都共用一个松弛变量 ξ。针对问题 (4.6) 中指数级的约束条件数量，我们提出了割平面算法来求解这个问题，算法如下：

算法 3 多核多类最大间隔聚类的割平面算法

输入：M 个特征映射 $\Phi_1, \cdots, \Phi_M$，参数 C，l 和 ϵ，约束子集 $\Omega = \phi$

repeat

在当前工作约束集 Ω 下求解问题 (4.6)，得到 $(\mathbf{v}_k^p, \boldsymbol{\beta}), k = 1, \cdots, M, p = 1, \cdots, m$

选择最违背的约束 $\mathbf{c}$，令 $\Omega = \Omega \cup \{\mathbf{c}\}$

until

新选择的约束 $\mathbf{c}$ 违背的差值小于 ϵ

4.2 通过 CCCP 最优化

给予一个初始点 $\mathbf{v}^{(0)}$，CCCP 通过将约束 $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}\mathbf{c}_i^T\mathbf{e}z_{ip}\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i)+\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}$ 替换为其在点 $\mathbf{v}^{(t)}$ 处的一阶泰勒展开式，从而由 $\mathbf{v}^{(t)}$ 计算得到 $\mathbf{v}^{(t+1)}$。

$$
\begin{aligned}
\min_{\mathbf{y},\boldsymbol{\beta},\mathbf{v},\xi}\quad & \frac{1}{2}\sum_{k=1}^{M}\sum_{p=1}^{m}\frac{\|\mathbf{v}_k^p\|^2}{\beta_k}+C\xi \\
s.t.\quad & \forall\mathbf{c}_i\in\{\mathbf{e}_0,\mathbf{e}_1,\cdots,\mathbf{e}_k\},i\in\{1,\cdots,n\}: \\
& \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{M}\sum_{p=1}^{m}(\mathbf{c}_i^T\mathbf{e}z_{ip}^{(t)}-c_{ip})\mathbf{v}_k^{pT}\Phi_k(\mathbf{x}_i)+\frac{1}{n}\sum_{i=1}^{n}\sum_{p=1}^{m}c_{ip}z_{ip}^{(t)}\geq\frac{1}{n}\sum_{i=1}^{n}\mathbf{c}_i^T\mathbf{e}-\xi, \\
& \forall k\in\{1,\cdots,M\}:\beta_k\geq 0, \\
& \sum_{k=1}^{M}\beta_k^2\leq 1,\xi\geq 0 \\
& \forall p,q\in\{1,\cdots,m\}: \\
& -l\leq\sum_{i=1}^{n}\sum_{k=1}^{M}(\mathbf{v}_k^p-\mathbf{v}_k^q)^T\Phi_k(\mathbf{x}_i)\leq l,
\end{aligned}
\tag{4.7}
$$

与二类聚类相似，上述问题也能被公式化描述为 SOCP 问题并能有效的求解。根据 CCCP 的过程，算法 4 总结了在约束集 Ω 下求解问题 (4.6) 的过程，如下：

算法 4 满足约束子集 Ω 条件下通过 CCCP 求解问题 (4.6)

初始化 $\mathbf{v}^{(0)}$

repeat

求得 $(\mathbf{v}^{(t+1)},\boldsymbol{\beta},\xi)$ 作为二阶锥规划问题 (4.7) 的解。

令 $\mathbf{v}=\mathbf{v}^{(t+1)},t=t+1$。

until

满足停止准则。

4.3 最违背的约束

最违背的约束也就意味着其 ξ 最大，并能通过下面的定理计算得到。

定理 4.3 最违背的约束计算如下:

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

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

5 实验

在这一部分，我们将通过几个玩具数据集和真实数据集来验证多核聚类算法的精度和效率。所有的实验都是使用 MATLAB7.0，在 1.66GHZ、$IntelCore^{TM}$、Windows XP、1.5GB 内存的 PC 上运行的。接下来，我们将简单的验证 MKC 算法的性能。

5.1 数据集

我们倾向于使用属性覆盖较广的数据集：ionosphere，digits，letter 和 satellite(来自 UCI 仓库)，svmguidel-a(来自 LIBSVM 数据集)，ringnorm 和 mnist。二类数据集在创建时的设置参考文献[8]。我们也从 digits，letter 和 mnist 数据中创建一个多类数据集。我们总结所有的数据集如表 1。

5.2 聚类精度对比

我们首先研究了 MKC 算法的聚类精度。准确的说，我们使用了核矩阵 $\mathbf{K} = \sum_{k=1}^{3} \beta_k K_k$，其中 K_k 是初始的核矩阵的“估计”。我们使用线性核函数 $k_1(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T\mathbf{x}_2$ 作为 K_1，多项式核函数 $k_2(\mathbf{x}_1, \mathbf{x}_2) = (1 + \mathbf{x}_1^T\mathbf{x}_2)^d$ 作为 K_2，高斯核函数 $k_3(\mathbf{x}_1, \mathbf{x}_2) = \exp(-(\mathbf{x}_1 - \mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2)/2\sigma)$ 作为 K_3。为了便于比较，我们用 K-均值聚类 (KM) 和归切聚类 (NC) 作为基准。我们也与使用两类数据运行 MMC 的 IterSVR[8] 相比较。对于

表 1: 数据集描述

Date	Size	Dimension	Class
digits1v7	361	64	2
digits2v7	356	64	2
ionosphere	354	64	2
svmguidel-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

KM 来说，聚类中心是随机初始化的，最终的性能是对 50 次独立的运行结果求平均得到的。对于 NC 来说，高斯核的宽度是通过不断尝试 $\{0.1\sigma_0, 0.2\sigma_0, \cdots, \sigma_0\}$ 并详细的研究设置的，其中 σ_0 表示数据集中任意两个点之间的距离的范围。最后，至于 IterSVR，其初值设置是建立在随机选择初始聚类中心的 K-均值聚类的基础上，使用高斯核函数并且其宽度的设置与 NC 相同。

在所有的聚类算法中，我们将聚类的数量设置为真实的类数量 (m)。为了评估聚类精度，我们遵从文献[?]27] 中的策略：我们首先采用有类标记的数据集，去掉所有的类标记并运行聚类算法，然后我们根据原来的类标记，按照多数原则来标记结果集，并最终测量正确分类的数量。此外，我们也计算每个聚类结果的兰德指数[33]。

表格 2 总结了各种数据集的结果。我们能看到，在多数情况下 MKC 的聚类精度和兰德指数比其它聚类算法更好。

表 2: 数据集的聚类精度 (%) 和兰德指数。对于每个方法，左边是聚类精度，右边是兰德指数。符号‘-’便是相应的算法无法在合理的时间内处理数据集。此外，IterSVR 只能用于二类数据集。

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
svmguidel-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		
mnist01234	89.98	0.901	87.34	0.882	90.12	0.907	91.55	0.922	67.63	0.898	-	-		

5.3 速度

一个关于多核聚类的潜在担忧是它的速度可能比最大间隔聚类更慢。图 1 比较了 MKC 与 MMC 在方法 K_1、K_2 和 K_3 上消耗的 CPU 时间。我们可以看到 MKC 的速度与 MMC 还是可比的。实际上，MKC 在某几个数据集上甚至比 MMC 收敛的更快。然而，不像 MMC 那样需要小心定义核矩阵，MKC 的一个强大的优点是能自动选择一个最优的基核之间的结合。

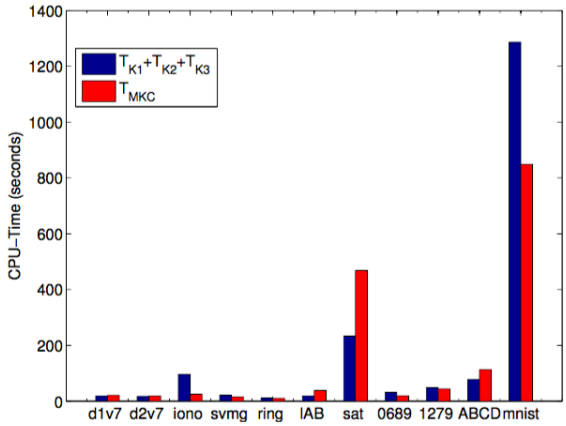

图 1: MKC 和 MMC 消耗的 CPU-时间 (秒)

5.4 MKC 的比例属性

在第三部分，我们展示了 MKC 的时间复杂度与样本数量成线性比例。图 2 刻画了经验结果的 log-log 折线图。注意到 log-log 图对应于多项式增长 $O(n^d)$，其中 d 是其斜率。我们看到，MKC 消耗的 CPU 时间大致比例为 $O(n)$，这与定理 3.1 是一致的。

此外，第三部分提到，MKC 每轮在满足约束 Ω 下求解问题 (2.6) 或者 (4.6) 至多需要 10 次迭代，图 2 再一次证明了这一定理，展示了在各种数据集上 CCCP 的迭代次数与样本大小的变化关系。

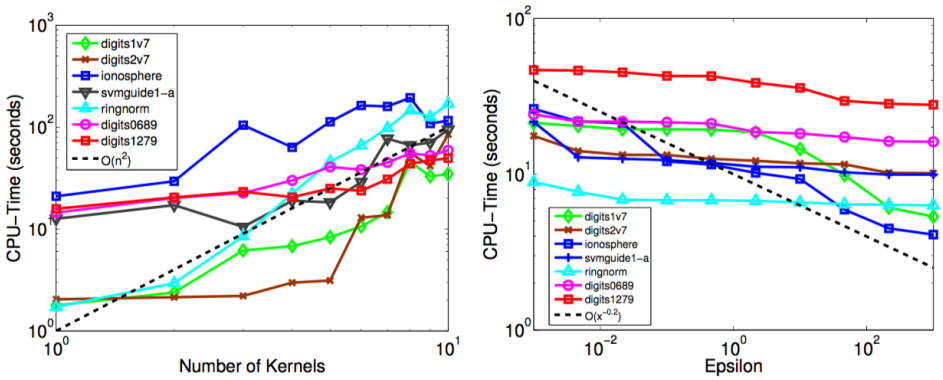

图 2: 左图：MKC 消耗的 CPU 时间与数据集大小变化关系，右图：MKC 的 CCCP 迭代平均次数与数据集大小变化关系

接下来，我们研究当基核的数量从 1 增加到 10 时，MKC 的 CPU 时间的变化情况。从图 3 我们能看到，MKC 的 CPU 时间与基核数量成二次比例关系。这比第三部分的界 $O(M^{3.5})$ 更好。

最后，第三部分声明了 MKC 总的迭代次数不超过 $\frac{CR}{\epsilon^2}$。这意味着 ϵ 越大，算法收敛越快。图 3 展示了 MKC 的 CPU 时间与 ϵ 之间的比例关系。我们能看到，经验比例大致为 $O(\frac{1}{\epsilon^{0.2}})$，相比较 $O(\frac{D^{3.5}+nD}{\epsilon^2}+\frac{D^{2.5}}{\epsilon^4})$ 更好。

5.5 MKC 的泛化能力

回想一下，最大间隔聚类采用了 SVM 的最大间隔标准。在这个实验中，我们也测试了 MKC 在未知数据上的泛化能力。我们首先在总的数据集中随机获取子集，再在获

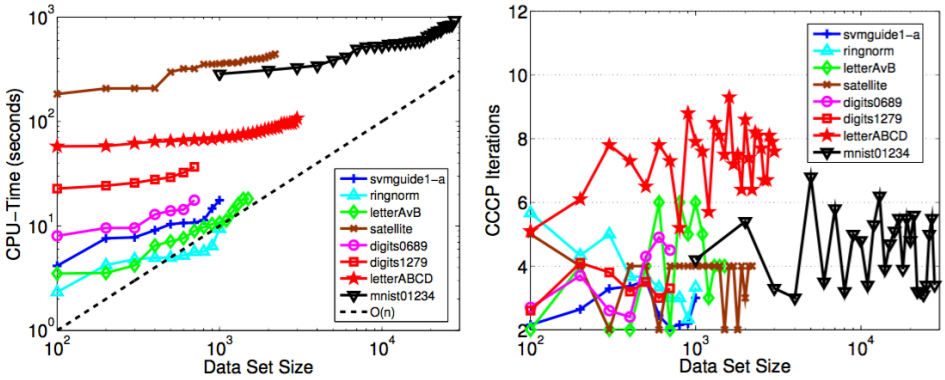

图 3: 左图：MKC 消耗的 CPU 时间与核数量的变化关系，右图：MKC 消耗的 CPU 时间与 ϵ 的变化关系

取的子集上学习得到的 MKC 模型，然后我们使用学习得到的模型在整个数据集上进行聚类。从表 3 我们能看到，从子集上学习得到的模型的聚类性能与从整个数据集上学习得到的模型的聚类性能具有可比性。这体现了多核聚类的一个重要的应用情形，即在数据集十分大时，我们能在该数据集的子集中进行聚类过程，然后将学习得到的模型对剩余的数据点进行聚类。

表 3: MKC 的泛化能力

Date	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 总结

在这篇论文中，我们将多核学习推广到无监督的学习。特别的，我们提出了多核聚类算法，能同时寻找最大间隔超平面，最适当的类标记组合和最优的核函数组合。在玩具数据集和现实数据集上的实验结果验证了 MKC 算法优越的性能。

参考文献

[1] Richard O. Duda, Peter E. Hart, and David G. Stork. Pattern classification (2nd edition). *En Broeck the Statistical Mechanics of Learning Rsity*, 2000.

[2] P. K. Chan, F. Schlag, and J. Y. Zien. Spectral k-way ratio-cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(9):1088–1096, 2006.

[3] C. H. Q. Ding. A min-max cut algorithm for graph partitioning and data clustering. *Proc. IEEE Int'l Conf. on Data Mining, 2001*, 2001.

[4] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans.pattern Anal.mach.intell*, 22(8):888–905, 2000.

[5] B. Schölkopf, J. Platt, and T. Hofmann. Generalized maximum margin clustering and unsupervised kernel learning. *Advances in Neural Information Processing Systems*, 21(5):1417 – 1424, 2007.

[6] Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. *Advances in Neural Information Processing Systems*, 17:1537–1544, 2004.

[7] Linli Xu and Dale Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. *Conference on Aaai*, pages 904–910, 2005.

[8] Kai Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. *Proceedings of the 24th international conference on Machine learning*, pages 583–596, 2007.

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

[10] Bin Zhao, Fei Wang, and Changshui Zhang. Efficient multiclass maximum margin clustering. *International Conference on Machine Learning*, pages 1248–1255, 2008.

[11] Olivier Bousquet and Daniel J. L. Herrmann. On the complexity of learning the kernel matrix. *Advances in Neural Information Processing Systems*, pages 399–406, 2002.

[12] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2001.

[13] Gert R. G. Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I. Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 5(1):323–330, 2002.

[14] Cheng Soon Ong, Alexander J. Smola, and Robert C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 6(1):1043–1071, 2005.

[15] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. *Twenty-first International Conference on Machine Learning*, 2004.

[16] Mehmet Nen and Ethem Alpaydin. Localized multiple kernel learning. *Cse.iitb.ac.in*, 7(2006):1531–1565, 2008.

[17] G. R. Lanckriet, Bie T De, N Cristianini, M. I. Jordan, and W. S. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635(10), 2004.

[18] Alain Rakotomamonjy, Francis Bach, Stéphane Canu, and Yves Grandvalet. More efficiency in multiple kernel learning. *Icml*, 7(2006):775–782, 2007.

[19] Sören Sonnenburg and Gunnar Raetsch. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7(2006):1531–1565, 2006.

[20] Ping Sun and Xin Yao. Boosting kernel models for regression. *2013 IEEE 13th International Conference on Data Mining*, pages 583–591, 2006.

[21] Jieping Ye, Shuiwang Ji, and Jianhui Chen. Learning the kernel matrix in discriminant analysis via quadratically constrained quadratic programming. *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining*, pages 854–863, 2007.

[22] Alexander Zien and Cheng Soon Ong. Multiclass multiple kernel learning. *Icml Acm*, 7(2006):2007, 2007.

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

[24] Boyd, Vandenberghe, and Faybusovich. Convex optimization. *IEEE Transactions on Automatic Control*, 51(11):1859–1859, 2006.

[25] Alex J Smola, S V N Vishwanathan, and Thomas Hofmann. Kernel methods for missing variables. *Tenth International Workshop on Artificial Intelligence & Statistics*, pages 325–332, 2005.

[26] Bernhard Schölkopf, Alexander Smola, and Klaus Robert Müller. Kernel principal component analysis. *ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING*, 27(4):555–559, 2003.

[27] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. *2004-09*, (2005), 2005.

[28] Thorsten Joachims. Training linear svms in linear time. *In SIGKDD*, pages 217–226, 2006.

[29] I͡U. E. Nesterov and Arkadiĭ Semenovich Nemirovskiĭ. Interior point polynomial algorithms in convex programming, sam. *SIAM Studies in Applied Mathematics, 13. Society for Industrial and Applied Mathematics (SIAM*, 13, 1994.

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

[31] Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear Algebra & Its Applications*, 284(98):193–228, 1998.

[32] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2(2):2001, 2002.

[33] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.