Multi-View Randomized Kernel Classification via Nonconvex Optimization

Xiaojian Ding, Fan Yang*

College of Information Engineering,
Nanjing University of Finance and Economics, Nanjing 210023, China
{wjswsl,nufe_yf}@163.com

Abstract

Multi-kernel learning (MKL) is a representative supervised multi-view learning method widely applied in multi-modal and multi-view applications. MKL aims to classify data by integrating complementary information from predefined kernels. Although existing MKL methods achieve promising performance, they fail to consider the tradeoff between diversity and classification accuracy of kernels, preventing further improvement of classification performance. In this paper, we tackle this problem by generating a number of high-quality base learning kernels and selecting a kernel subset with maximum pairwise diversity and minimum generalization errors. We first formulate this idea as a nonconvex quadratic integer programming problem. Then, we transform this nonconvex problem into a convex optimization problem and show it is equivalent to a semidefinite relaxation problem, which a semidefinite-based branch-and-bound algorithm can quickly solve. Experimental results on the real-world datasets demonstrate the superiority of the proposed method. The results also show that our method works for the support vector machine (SVM) classifier and other state-of-the-art kernel classifiers.

Introduction

Numerous methods of learning from multi-view data by considering the diversity of different views have been developed recently. These views can come from multiple sources or modalities. Multiple kernel learning (MKL) is a representative supervised multi-view learning method widely applied in multi-modal and multi-view applications (Arabacı et al. 2021; Zhang et al. 2020; Peng et al. 2019; Wu et al. 2020; Jiang et al. 2022; Shah et al. 2021; Yang et al. 2020). MKL was initially proposed to regulate the capacity of the search space for potential kernel matrices to achieve better generalization (Gönen 2013). However, it has been extensively utilized in multi-view data scenarios because MKL kernels inherently correspond to various sources or views.

For MKL analysis, learning an optimal linear or nonlinear combination of a predefined set of kernels is the foundation of using the complementary information within them and improving learning performance. Some approaches consider using linear combination methods to combine these kernels, such as fixed rule, heuristic, and optimization methods. Fixed rule methods learn the kernel as a weighted linear combination of the available kernels without training (Ben-Hur and Noble 2005). Heuristic methods find the combination parameters by some heuristic measures like conditional class probabilities calculated from the kernel matrices or the performance values trained by each kernel separately (Aiolli and Donini 2015; Fan et al. 2017; Liu et al. 2020). Optimization methods usually learn the kernel combination parameters together with the parameters of the base learner (e.g., parameter C in SVM) by solving an optimization problem (Han et al. 2018; Wang, Lu, and Zhang 2018; Chamakura and Saha 2022). Some other approaches consider designing a nonlinear combination method that uses nonlinear functions of kernels, namely, multiplication, power, and exponentiation (Gu et al. 2016; Song et al. 2018). Although the algorithms mentioned above achieve promising performance, they fail to consider the tradeoff between diversity and classification accuracy of kernels, preventing further improvement of classification performance.

Diversity and accuracy have been considered two essential characteristics in kernel combinations (Xia and Hoi 2013; Liu et al. 2016). Ko et al. (Ko, Sabourin, and de Souza Britto Jr 2009) studied the correlation between accuracy and diversity and established theoretically and empirically that the two are indeed correlated. These studies assert that to get a good combination, the combined kernels should be as diverse as possible without sacrificing accuracy. In previous studies on MKL, various approaches have been used to select base learning kernels. Some approaches (Han et al. 2018; Shen et al. 2021) studied predefined kernels from commonly used kernels (e.g., linear, Gaussian, and Polynomial) with different parameters. Chamakura and Saha (Chamakura and Saha 2022) used linear kernel, Intersection kernel, and Chi-squared kernel as base learning kernels. Ding et al. (Ding, Tang, and Guo 2020) used the Gaussian Interaction Profile of the network (GIP) kernel as a base learning kernel. Yu et al. (Yu, Gong, and Jiang 2020) generated multiple base kernels from a specific kernel set, including the Hadamard, RBF, and linear kernels. To summarize, predefined kernels can be different kernels or the same kernel with different parameters. Since a kernel learning algorithm (e.g., SVM) is sensitive to kernel parameters, few parameter candidates usually perform best. To this end, con-

*Fan Yang is the corresponding author.

sidering only the diversity between kernels cannot ensure their good performance. In addition, quantifying the diversity of kernels, however, is difficult as there is no formal definition. For this reason, few MKL approaches concentrated on the diversity and/or the accuracy of combined kernels.

The recent work in (Ding et al. 2021a,b) proposed a randomized kernel and established theoretically and empirically that SVM with the randomized kernel is not sensitive to kernel parameters. The key idea of the randomized kernel is to assign a random parameter to each input dimension (or feature). Then each input dimension is allowed to have a different parameter value. Randomizing the kernel parameters many times can obtain a pool of randomized kernels while retaining their performance. The randomized kernel was proved to satisfy Mercer's theorem. Motivated by this work, this paper proposes a new MKL learning method named RMKL that optimizes kernel weights over combinations of randomized kernels. We first introduce a kernel quality measure by considering kernels' diversity and generalization errors. By maximizing the pairwise diversity and minimizing the generalization errors of kernels, we formulate the optimization problem of the RMKL as a nonconvex quadratic integer programming problem. We transform this nonconvex problem into a convex optimization problem and show that it is equivalent to a semidefinite programming (SDP) relaxation problem. Further, wo develop a branch-and-bound branch-and-bound algorithm to solve the proposed quadratic integer programming problem. Experimental results on the real-world datasets demonstrated the superiority of the proposed method.

To sum up, our main contributions are:

- We propose a new MKL method that optimizes the kernel weights by considering both kernels' diversity and performance and formulating it as a nonconvex quadratic integer programming problem.
- We transform this nonconvex problem into a convex optimization problem and develop an efficient solution.
- We validate the effectiveness of the proposed method on real-world multi-view datasets of different types.

Preliminaries

We consider the classification learning problem from data $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^d$ belongs to some input space $\mathcal{X}$, and $\mathbf{y} = (y_1, ..., y_N)^T \in \{1, 2, ..., c\}^N$ denoting the class labels of samples $\mathbf{x}_i$.

Multiple Kernel Learning

A classifier is a function $f : \mathcal{X} \to \mathcal{Y}$ which labels each sample $\mathbf{x} \in \mathcal{X}$ with some $\mathbf{y} \in \mathcal{Y}$.

We denote by $\{k_l(\mathbf{x}, \mathbf{z}) : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}, l = 1, ..., m\}$ the set of m basis kernels to be combined. For each kernel function $k_j(\cdot, \cdot)$ denote by $K_l = [k_l(\mathbf{x}_i, \mathbf{z}_i)]_{N\times N}$ the associated kernel matrix. We denote by $\boldsymbol{\eta} = (\eta_1, ..., \eta_m)^T \in \mathbb{R}_+^m$ the vector of kernel weights used to combine the basis kernels and denote by $k(\mathbf{x}, \mathbf{z}; \boldsymbol{\eta}) = \sum_{l=1}^m \eta_l K_l(\mathbf{x}, \mathbf{z})$ and $k(\boldsymbol{\eta}) = \sum_{l=1}^m \eta_l K_l$ the combined kernel function and kernel matrix, respectively.

MKL approach can be considered as the search of the kernel $k(\mathbf{x}, \mathbf{z}; \boldsymbol{\eta})$ with reproducing kernel Hilbert space (RKHS) $\mathcal{H}_\eta$ that have good generalization properties. For a binary classification problem, we learn the optimal combination of kernels by solving the following optimization problem

$$\begin{aligned} &\mathbf{min}_{f\in\mathcal{H}_\eta, \boldsymbol{\eta}\in\varrho} \frac{1}{2}\|f\|_{\mathcal{H}_\eta}^2 + C\sum_i \xi_i \\ &\quad \mathbf{s.t.}\ y_i(f(\mathbf{x}_i) + b) \geq 1 - \xi_i, \forall i \\ &\qquad \xi_i \geq 0, \forall i \end{aligned} \tag{1}$$

where ϱ is a domain for kernel weights vector $\boldsymbol{\eta}$, $b \in \mathbb{R}$ is the bias term, and ξ_i are positive slack variables.

The main task of MKL is to learn the kernel weights $\boldsymbol{\eta}$. Different strategies differ in the way they put restrictions on $\boldsymbol{\eta}$: the linear combination ($\boldsymbol{\eta} \in \mathbb{R}^m$), the conic combination ($\boldsymbol{\eta} \in \mathbb{R}_+^m$) or the convex combination ($\boldsymbol{\eta} \in \mathbb{R}_+^m$ and $\sum_{l=1}^m \eta_l = 1$). A more general practice is to restrict $\boldsymbol{\eta}$ to a probability distribution, leading to the following definition of domain ϱ

$$\varrho = \left\{ \boldsymbol{\eta} \in \mathbb{R}_+^m : \|\boldsymbol{\eta}\|_1 = \sum_{l=1}^m |\eta_l| \leq 1 \right\}. \tag{2}$$

Using the domain ϱ defined in (2), one can obtain a sparse solution of kernel combination weights, eliminating irrelevant kernels.

Randomized Kernel

For an SVM classifier, the Gaussian kernel is the most commonly used kernel function

$$k_{\text{Gau}}(\mathbf{x}, \mathbf{z}) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{z}_i\|^2}{2\sigma^2}\right), \tag{3}$$

where σ is the kernel parameter. Standard Gaussian kernel has a single kernel width σ that controls the "spread" of the kernel. Actually, this kernel width is invariant to the region of the input space, which means all features share the same width. Different regions of the input space have different data characteristics, and more kernel parameters should be introduced to ensure each feature is associated with a kernel width value.

To this end, the work in (Ding et al. 2021b) proposed a randomized kernel, which can be expressed as

$$k_{\text{RCG}}(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{1}{2}\sum_{i=1}^d \frac{(x_i - z_i)^2}{(d\sigma_i)^2}\right), \tag{4}$$

where $\sigma_1, ..., \sigma_d$ are kernel parameters.

All kernel parameters are randomly assigned in the randomized kernel based on a uniform distribution. SVM with a randomized kernel is not sensitive to kernel parameters, and repeated runs can achieve good performance. In addition, a randomized kernel with d-dimensional parameter vector can capture the data characteristics in different regions of the input space. Therefore, different randomized kernels may provide complementary information for data distribution investigation.

The Proposed Method

Measuring the Quality of Kernels

By performing the random assignment of kernel parameters M times, a pool of M randomized kernels is obtained, denoted as

$$\mathcal{M} = \{k_1, ..., k_M\}. \tag{5}$$

In ensemble classification, accuracy–diversity tradeoff has been well studied (Aksela and Laaksonen 2006), and it is opined that accuracy should not be sacrificed for diversity. Motivated by this work, we select a subset of kernels from the set $\mathcal{M}$ while maximizing their diversity and minimizing their generalization error. Two sub-problems are involved: measuring randomized kernels' diversity and performance.

The diversity of kernels is generally accepted as a necessary condition for combining them. However, quantifying the diversity of classifiers is difficult as there is no general agreement about quantifying diversity among a set of kernels. Giacinto and Roli (Giacinto and Roli 2001) proposed a double-fault measure to measure diversity between two basis classifiers. The intuition defines this measure as two diverse classifiers performing differently on the same dataset. Similarly, the double-fault measure measures diversity between a pair of basis kernels.

Given two basis kernels k_i and k_j, let N^{ab} be the number of samples on which the predictive output of a classifier with kernels k_i and k_j is a and b, respectively. The diversity between the two basis kernels is measured by

$$D(k_i, k_j) = \frac{N^{ab} + N^{ba}}{N^{aa} + N^{ba} + N^{ab} + N^{bb}}. \tag{6}$$

We train a classifier with the basis kernels on the validation set and calculate their average k-fold cross-validation error rates to measure the basis kernels' performance. Specifically, we denote the average error rate of kernel k_i by $E(k_i)$.

Optimizing the Kernel Weights

The selected kernels should be as accurate and diverse as possible to obtain a good combination of basis kernels in the set $\mathcal{M}$. We then propose an optimization objective by selecting a subset of kernels from a pool of kernel candidates to minimize the objective function, which allows accuracy and diversity to be incorporated within a single measure. By introducing binary kernel weights, the objective of selecting a subset of kernels can be formulated by

$$\begin{aligned} &\textbf{min} \sum_{i=1}^{M}\sum_{j=1}^{M} \frac{\eta_i \eta_j}{D(k_i,k_j)} \text{and} \sum_{i=1}^{M} \eta_i E(k_i) \\ &\textbf{s.t. } \eta_i \in \{0,1\}, i = 1, ..., M, \end{aligned} \tag{7}$$

where $\eta_i = 1$ means that the kernel k_i is selected in the kernel subset. When we obtain the kernel weights $\boldsymbol{\eta}$, we consider a convex linear combination of basis kernels

$$k(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{l=1}^{M} \eta_l k_l(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{l=1}^{M} \eta_l}. \tag{8}$$

A direct way of solving (7) is to traverse the whole set $\mathcal{M}$ to find the best combination of basis kernels, which leads to the impractical computation cost.

To solve it efficiently, we employ a matrix $\mathbf{Q}$ to store the pairwise diversity terms of kernels in the set $\mathcal{M}$. Let the off-diagonal entry Q_{ij} be the inverse of the pairwise diversity given by kernels k_i and k_j, that is, $Q_{ii} = \frac{1}{D(k_i,k_j)}$, and let all diagonal entries Q_{ii} be 0. In addition, we use a vector $\mathbf{r}$ to store the accuracy terms where $r_i = E(k_i)$.

By introducing a free parameter m, we transform the problem (7) to the following quadratic-(0,1) integer programming problem

$$\begin{aligned} &\textbf{min } p(\boldsymbol{\eta}) = \boldsymbol{\eta}^T \mathbf{Q} \boldsymbol{\eta} + \mathbf{r}^T \boldsymbol{\eta} \\ &\textbf{s.t.} \sum_{i=1}^{M} \eta_i = m, \\ &\quad \eta_i \in \{0,1\}, i = 1, ..., M, \end{aligned} \tag{9}$$

with $\mathbf{Q}$ being a symmetric $M \times M$ matrix, $\mathbf{r}$ is a M real vector, and $m \in \mathbb{R}$. Here, the parameter m denotes the total number of selected kernels.

Semidefinite programming is well known to provide powerful relaxations for quadratic-(0,1) integer programming problems. As a relaxation for the boolean quadric polytope of (9), we model the dyadic product $\boldsymbol{\eta}^T \boldsymbol{\eta}$ by a matrix variable $\mathbf{X}$, and denote the diagonal of this matrix by $\mathbf{D}$.

Then, we define the SDP relaxation of the problem (9)

$$\begin{aligned} &\textbf{min } p(\boldsymbol{\eta}) = \mathbf{r}^T \boldsymbol{\eta} + \sum_{i=1}^{M}\sum_{j=1}^{M} q_{ij} X_{ij} \\ &\textbf{s.t. } X_{ij} = D_i, i = 1, ..., M \\ &\quad \sum_{j=1}^{M} X_{ij} = m D_i, i = 1, ..., M \\ &\quad \mathbf{e}^T \boldsymbol{\eta} = m \\ &\quad \mathbf{X} - \boldsymbol{\eta}\boldsymbol{\eta}^T \succeq 0, \end{aligned} \tag{10}$$

where $\mathbf{X} \in S_M$, $\mathbf{e}$ denotes the M-vector of all ones, and q_{ij} denotes the general term of matrix $\mathbf{Q}$.

Solving the Problem

It is known that the integer programming problem of (9) is NP-complete (Hartmanis 1982). Note that the objective function in (9) is not convex due to the $\mathbf{Q}$ diagonal terms. By introducing two parameters $\mathbf{u} \in \mathbb{R}^M$ and $\mathbf{v} \in \mathbb{R}^M$, we can transform this nonconvex problem into the following convex optimization problem

$$\begin{aligned} &\textbf{min } p'(\boldsymbol{\eta}) \\ &\textbf{s.t.} \sum_{i=1}^{M} \eta_i = m, \eta_i \in \{0,1\}, i = 1, ..., M, \end{aligned} \tag{11}$$

where

$$\begin{aligned} p'(\boldsymbol{\eta}) &= \boldsymbol{\eta}^T \mathbf{Q} \boldsymbol{\eta} + \mathbf{r}^T \boldsymbol{\eta} + \sum_{i=1}^{M} u_i(\eta_i^2 - \eta_i) \\ &\quad + \sum_{i=1}^{M} v_i \eta_i (\sum_{i=1}^{M} \eta_i - m) \\ &= \boldsymbol{\eta}^T \mathbf{Q}_1 \boldsymbol{\eta} + \mathbf{r}_1^T \boldsymbol{\eta} + \sum_{i=1}^{M} u_i(\eta_i^2 - \eta_i) \\ &= \boldsymbol{\eta}^T \mathbf{Q}_2 \boldsymbol{\eta} + \mathbf{r}_2^T \boldsymbol{\eta}, \end{aligned}$$

and $\mathbf{Q}_1 = \mathbf{Q} + \frac{1}{2}\left(\mathbf{u}^T\mathbf{e} + \mathbf{e}^T\mathbf{u}\right)$, $\mathbf{Q}_2 = \mathbf{Q}_1 + \mathrm{Diag}(\mathbf{v})$, $\mathbf{r}_1 = \mathbf{r} - \mathbf{u}^T m$, $\mathbf{r}_2 = \mathbf{r}_1 - \mathbf{v}$. Here, $\mathrm{Diag}(\mathbf{v})$ denotes a diagonal $M \times M$ matrix with the elements of $\mathbf{v}$ on the diagonal.

To solve (11), we can use the Branch-and-Bound (B&B) algorithm based on the relaxation of the constraints $\boldsymbol{\eta} \in [0,1]^M$, and denote $S := \left\{\boldsymbol{\eta} : \mathbf{e}^T\boldsymbol{\eta} = m, \boldsymbol{\eta} \in [0,1]^M\right\}$ by the set of feasible solutions of the continuous relaxation of (9). For all $\boldsymbol{\eta} \in S$, it is easy to see that the objective function $p'(\boldsymbol{\eta})$ equals $p(\boldsymbol{\eta})$. The main idea of the B&B algorithm is to divide the problem into smaller subproblems by branching the variables into possible values, creating a lower bound of the function in a specific domain, and finding approximate solutions that converge to the optimal solution. In the case of problem (11), the bound can be solved by finding the optimum value of the continuous relaxation. To this end, we should solve parameters $\mathbf{u}$ and $\mathbf{v}$ in $p'(\boldsymbol{\eta})$ to find a tight bound. Specifically, we need to solve the following problem

$$\begin{aligned} \max_{\mathbf{u},\mathbf{v}} \min_{\boldsymbol{\eta} \in S} &\; p'(\boldsymbol{\eta}) \\ \textbf{s.t. } &\mathbf{Q}_2 \succeq 0. \end{aligned} \tag{12}$$

Theorem 1. *(Lemaréchal and Oustry 1999) The optimal value of problem (12) is equal to the SDP relaxation (10).*

For problem (12), optimal values u_i^* are solved are by the optimal values of the dual variables associated with constraints $X_{ij} = D_i, i = 1, ..., M$ of (10), and optimal values v_i^* are solved are by the optimal values of the dual variables associated with constraints $\sum_{j=1}^{M} X_{ij} = mD_i, i = 1, ..., M$ of (10).

The Branch-and-Bound Method

We present an exact solution for solving the problem (9) by designing a B&B framework using relaxation (10) discussed above for getting lower bounds. Global optimization methods based on B&B techniques hinge on two pivotal procedures aimed at calculating upper and lower bounds for the global optimum in nonconvex problems. In the context of a minimization problem, the upper bound can be derived by selecting any point within the feasible set. Typically, a local search method is employed to refine and enhance this upper bound. Conversely, the lower bound can be determined through a convex relaxation of the original nonconvex problem.

Algorithm 1: SDP based Branch-and-Bound algorithm

Input: The minimization problem (9)
Initializaiton: Global upper-/lower-bounds GUB=$+\infty$, GLB=$-\infty$, priority queue $\mathfrak{L}$

```
1:  while $\mathfrak{L} \neq \varnothing$ do
2:      Perform Subproblem selection: $\mathfrak{O}$=sel($\mathfrak{L}$).
3:      Perform Bounding: [ UB,LB,$\boldsymbol{\eta}$, $\mathbf{X}$ ] =bound($\mathfrak{O}$).
4:      Update global bounds:
5:      GLB=min(LB, $\min_{\mathfrak{O}' \in \mathfrak{L}}$LB($\mathfrak{O}'$)).
6:      if GUB<UB then
7:          GUB=UB, $\boldsymbol{\eta}^* = \boldsymbol{\eta}$.
8:      end if
9:      Perform Pruning:
10:     $\mathfrak{L} \leftarrow \mathfrak{L} \setminus \{\mathfrak{O}' | \mathfrak{O}' \in \mathfrak{L}, \mathrm{LB}(\mathfrak{O}') \geqslant \mathrm{GUB}\}$.
11:     Perform Branching:
12:     if LB<GUB then
13:         [$\mathfrak{O}_1, \mathfrak{O}_2$]=bra($\mathfrak{O}$).
14:         $\mathfrak{L} \leftarrow \{\mathfrak{L}, \mathfrak{O}_1, \mathfrak{O}_2\}$.
15:     end if
16: end while
17: Return $\boldsymbol{\eta}^*$.
```

There are several essential components in B&B: *Bounding* (bound($\mathfrak{O}$)), *Subproblem selection* (sel($\mathfrak{L}$)), and *Branching* (bra($\mathfrak{O}$)). At each iteration of the B&B algorithm (Algorithm 1), we choose a subproblem from the priority queue $\mathfrak{L}$, and compute its upper and lower bounds LB/UB. Subsequently, the global upper bound (GUB) is updated as the minimum of the upper bounds across all branches, while the global lower bound (GLB) is updated as the minimum of the lower bounds over all leaf branches. Any subproblems with a lower bound not exceeding the global upper bound are pruned. In cases where the selected subproblem cannot be pruned, its feasible set $\mathfrak{O}$ is partitioned into at least two convex sets. The B&B algorithm converges when the global upper-bound and lower-bound closely approximate each other.

Subproblem selection The selection strategy involves choosing a subproblem from the priority queue $\mathfrak{L}$ for further exploration. At each iteration of the B&B algorithm, we select the subproblem with the lowest upper-bound estimate. This approach prioritizes exploration in the most promising regions of the solution space first.

Bounding The bounding strategy involves producing an upper bound and a lower bound. The lower bound can be obtained by solving (10) the spectral bundle method proposed by Helmberg and Rendli (Helmberg and Rendl 2000), well-established for equality-constrained semidefinite programs.

We employ a heuristic to acquire a lower bound, which is invoked at each node in the B&B tree. Our approach involves applying a variable fixation heuristic inspired by (Létocart, Plateau, and Plateau 2014). This heuristic leverages the solution of the semidefinite relaxation obtained at each node. It fixes variables under a defined threshold and subsequently applies the primal heuristic (Billionnet and

Calmels 1996) over the reduced problem. We update the solution through a fill-up and exchange procedure performed on the unreduced problem to enhance it. This iterative procedure, which increments at each iteration, continues until the reduced problem is empty.

Branching The branching strategy refers to a method for generating subproblems. The branching strategy utilizes the optimal semidefinite solution provided by the semidefinite bounding procedure, as outlined below. First, we define $\mathbf{z} = (\boldsymbol{\eta} + \mathbf{e})/2$. Next, we employ the "most-fractional" branching rule to determine which variables to branch on next: specifically, we choose a variable z_i for which z_i is closest to 0.5.

Branching on variable z_i creates two new subproblems: z_i is fixed to 0 and z_i is fixed to 1. These subproblems correspond to nodes in the B&B search tree. The branching process generates two nodes, which are subsequently added to the search tree.

Experimental Results

We conduct experiments on eight benchmark datasets to evaluate the performance of our proposed RMKL algorithm. Among these datasets, four are public multi-view datasets, and the other is gene expression microarray one-view datasets publicly available at the SchliepLAB website [1]. Their characteristics are summarized in Table 1. All dataset inputs are normalized to zero mean and unit variance range.

Dataset	Abbr	Sample	View	Class	Feature
3Sources	D1	169	3	6	10259
MSRC-v1	D2	210	6	7	2418
Caltech101-7	D3	1474	6	7	3766
NUS-WIDE	D4	30000	5	31	634
Armstrong	D5	72	1	2	12582
Chen	D6	179	1	2	22699
Chowdary	D7	104	1	2	22283
Emanuel	D8	253	1	2	15154

Table 1: Summary of the datasets

3Sources [2]: The 3Sources dataset was collected from three well-known online news sources: BBC, Reuters, and Guardian. A total of 169 articles were manually annotated with one or more of the six topical labels: business, entertainment, health, politics, sport, and technology.

MSRCv1 [3]: This is a scene recognition dataset containing 240 images in 8 categories. Following (Cai et al. 2011), we select 7 classes composed of tree, building, airplane, cow, face, car, bicycle and each class has 30 images. Six visual feature vectors are extracted from each image, which are the CM feature of dimension 48, the HOG feature of dimension 100, the GIST feature of dimension 512, the LBP feature of dimension 256, the SIFT feature of dimension 200, and the CENTRIST feature of dimension 1320.

Caltech101-7 [4]: This is an object recognition data set containing 101 categories of images. Following (Xu, Han, and Nie 2016), we select 7 classes composed of Dolla-Bill, Face, Garfield, Motorbikes, Snoopy, Stop-Sign and Windsor-Chair and each class has 1474 images. Six visual feature vectors are extracted from each image, which are the Gabor feature of dimension 48, the wavelet feature of dimension 40, the GIST feature of dimension 512, the LBP feature of dimension 928, the SIFT feature of dimension 200, and the CENTRIST feature of dimension 254.

NUS-WIDE [5]: This is a real-world web image dataset for object recognition problem. We select the front 25 from the all 31 categories in alphabetical order (bear, bird, ... ,tower), and choose the first 120 images for each class. Five low-level features are extracted to represent each image: 64 color histogram, 144 color correlogram, 73 edge direction histogram, 128 wavelet texture, and 225 block-wise color moment.

Experimental Design

Baseline Methods We compare our RMKL method with several state-of-the-art MKL methods:

GMKL (Chamakura and Saha 2022): GMKL improved localized MKL using graph modularity.

ALMKL (Liu et al. 2020): ALMKL proposed an approach for automatically learning the multiple parameters of kernel collaborative representation classification (KCRC).

TSMKL (Wang, Lu, and Zhang 2018): TSMKL introduced a fuzzy MKL model based on the Hilbert-Schmidt independence criterion (HSIC) for classification.

RPMKL (Han et al. 2018): RPMKL introduced a more general matrix regularized MKL via (r;p)-norm.

EAMKL (Aiolli and Donini 2015): The algorithm combined some weak kernels to solve a quadratic problem.

ELMKL (Guo et al. 2023): The algorithm employed ELM as the base learner, and l_1-norm constraints on the kernel weights controlled the sparsity of the linear combination of kernels.

DMKL (Strobl and Visweswaran 2013): DMKL employed the idea of deep learning to implement MKL.

PMKL (Kloft et al. 2011): PMKL extended MKL to arbitrary norms.

NMKL (Cortes, Mohri, and Rostamizadeh 2009): The algorithm introduced a nonlinear kernel combination strategy based on kernel ridge regression (KRR) and polynomial combination of kernels.

For each method, 20 repeated runs are considered, and the average performance and the standard deviation across 20 runs are reported. For each run, half of the samples (as shown in column 3 of Table 1) are randomly chosen as a training set, and the remaining samples are retained as a test set.

Parameter Setting In the proposed RMKL, two parameters need to be set. They are the number of randomized

[1]https://schlieplab.org/Static/Supplements/CompCancer/datasets.htm

[2]http://mlg.ucd.ie/datasets/3sources.html

[3]http://research.microsoft.com/en-us/projects/objectclassrecognition/

[4]http://www.vision.caltech.edu/Image_Datasets/Caltech101/

[5]https://lms.comp.nus.edu.sg/wp-content/

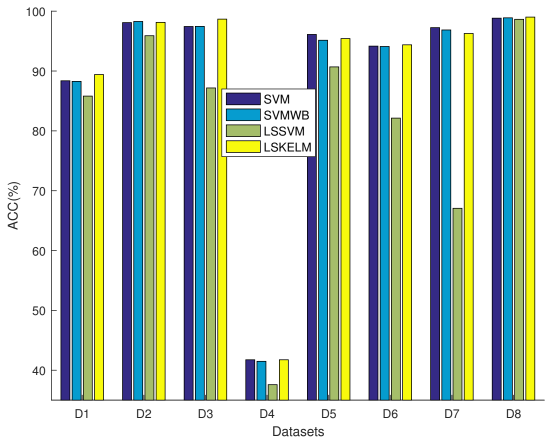

Figure 1: The performance comparison of RMKL for different classifiers.

kernels M and the number of selected kernels m. For multi-view datasets, the m value is set as the number of views in the datasets. The M value is generally 2 to 3 times the m value. This conclusion is based on a large number of experimental results. In the following experiments, we set $M = 10$ and $m = 5$ as default for one-view datasets.

Like other MKL algorithms, we employ SVM as the base learner in the proposed RMKL. In RMKL, GMKL, TSMKL, and ELMKL, the regularization parameter C is tuned on $\{10^{-2}, 0.1, 1, 10, 10^2, 10^3\}$. In ALMKL, the regularization parameter λ is experimentally set to 10^{-3}.

In EAMKL, the regularization parameter λ is tuned on $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. In PMKL, parameters pairs (p, C) are tuned on $\{1, 2, 4, 10\} \times \{10^{-2}, 0.1, 1, 10, 10^2, 10^3\}$. In RPMKL, parameters pairs (r, p, C) are tuned on $\{1, 2, 4, 10\} \times \{1, 2, 4, 10\} \times \{10^{-2}, 0.1, 1, 10, 10^2, 10^3\}$.

Evaluation Metrics We adopt three metrics average classification accuracy (ACC), rank1 times (Rank1), and win/tie/loss count to evaluate algorithm performance: ACC quantifies the average classification accuracy and standard deviation over 20 runs; Rank1 evaluates ranking information of the methods; The win/tie/loss count shows three values on a measure, i.e., the classification accuracy for which the performance of a method is significantly better/equal/worse than other methods. Here, we apply Wilcoxon's test to compare the difference statistically via pairwise comparisons (95% significance level).

Performance Comparison

Table 2 shows the performance of the compared methods in terms of three evaluation metrics. For each row, the best results are in boldface. To be more intuitive, the last but three row of Table 2 shows the average ACC (Ave) of each method among all datasets. The following observations are obtained according to the experimental results on these datasets:

- RMKL simultaneously considers the diversity and the performance of kernels. As a result, it outperforms other MKL methods consistently. In terms of Ave, the RMKL clearly obtains the best result.
- Instead of using different kinds of kernels or the same kernel but with different parameters, introducing multiple randomized parameters in kernels leads to high-quality kernels such that good performance can be obtained. This is another reason that RMKL outperforms conventional MKL methods.
- On all datasets, EAMKL achieves the second-best ACC four times, TSMKL achieves the second-best ACC two times, while PMKL, RPMKL, LMKL, and DMKL achieve the second-best ACC only one time. These results show that the baseline methods do not work for all datasets.

In the following experiment, we test the performance of RMKL on other famous kernel learning machines, such as SVM without bias (SVMWB)(Steinwart, Hush, and Scovel 2011), least square SVM (LSSVM)(Suykens and Vandewalle 1999), and LSKELM(Huang et al. 2012). The experimental results are demonstrated in Figure 1. We find four classifiers achieve similar performance on the D8 dataset, and LSSVM performs much worse than other classifiers on the remaining datasets. These results demonstrate that the proposed RMKL method works for SVM, SVMWB, and LSKELM classifiers on all datasets.

Impact of Parameters

For one-view datasets, we study the impacts of parameters of m and C on the performance of RMKL. The parameter m is used to determine the number of kernels that are selected for combination. Parameter C controls the tradeoff between achieving a low training error and minimizing the norm of the weights of a classifier. Figure 2 shows the performance of RMKL on D5 with varying parameters m and C. As can be observed, the performance of RMKL grows linearly with the number of m, although the increase is not significant. We set $m = 5$ as the default valule to save the computational cost. This result demonstrates the robustness of the proposed method again.

Conclusion

In this paper, we have studied a new MKL method based on combining high-quality kernels, which can learn the kernel weights by considering both kernels' diversity and generalization errors. The optimization problem of the proposed method can be formulated as a nonconvex quadratic integer programming problem. Then, we transform this nonconvex problem into a convex optimization problem and prove it is equivalent to an SDP relaxation problem. The experimental results on both multi-view and one-view datasets demonstrated the superiority of the proposed method. In future work, we will extend our method to more general tasks and demonstrate its effectiveness in a broader range of applications.

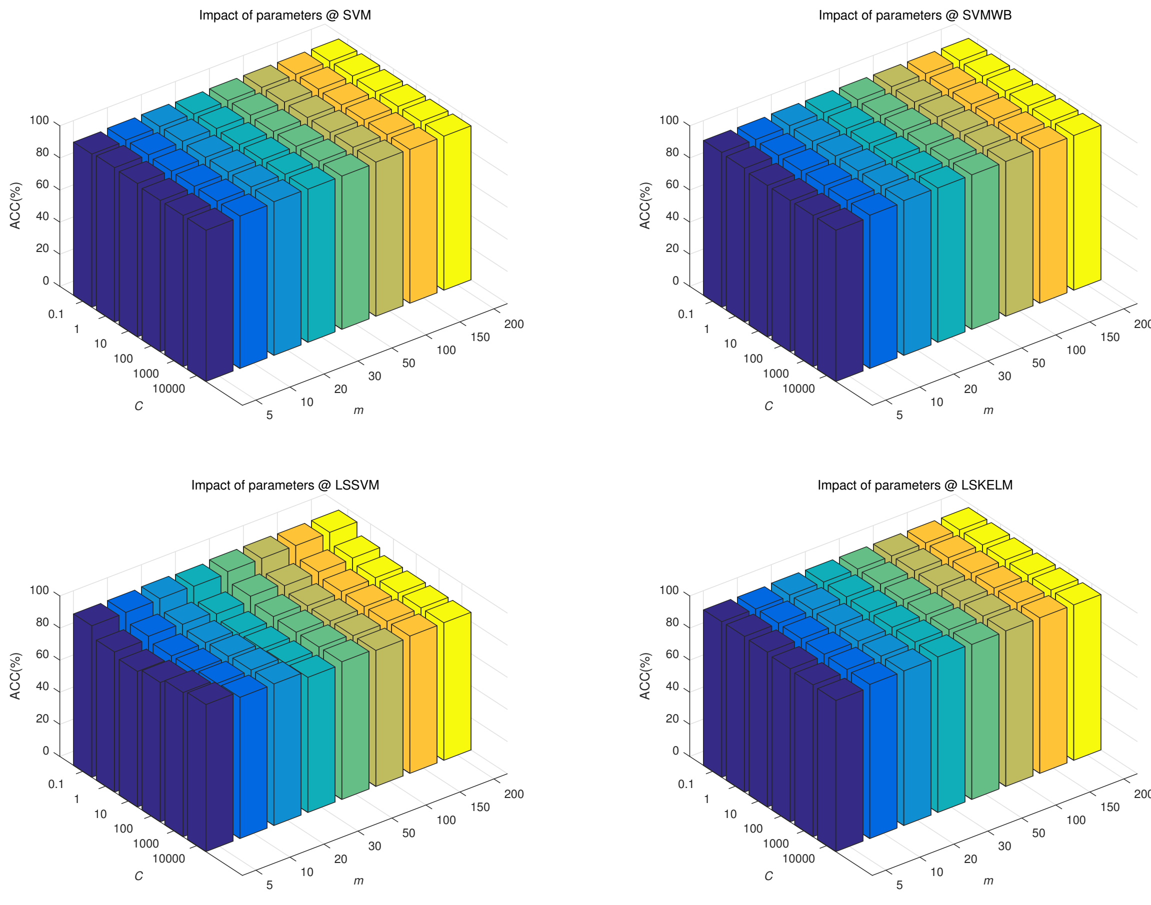

Figure 2: Parameter study on the D5 dataset in terms of ACC for a) SVM; b) SVMWB; c) LSSVM; and d) LSKELM classifiers.

Dataset	Metric	NMKL	EAMKL	ELMKL	PMKL	RPMKL	TSMKL	LMKL	ALMKL	DMKL	RMKL
D1	ACC	88.01	89.28	89.12	88.67	88.65	88.06	88.17	88.21	86.76	**89.49**
	STD	1.37	1.25	1.51	1.24	1.22	1.10	1.26	1.65	1.30	1.25
D2	ACC	97.57	96.89	97.64	98.04	97.98	98.11	97.81	97.50	97.12	**98.18**
	STD	0.80	1.14	0.74	0.71	0.82	0.71	0.84	0.66	0.72	0.73
D3	ACC	94.10	98.13	96.46	96.15	96.22	95.49	98.13	98.01	96.13	**98.58**
	STD	2.24	1.76	2.19	2.03	1.93	1.43	1.33	1.54	1.52	1.29
D4	ACC	40.88	41.93	40.82	41.41	41.27	38.37	40.94	40.63	38.39	**42.08**
	STD	1.43	1.52	1.62	1.48	1.68	2.28	1.51	1.96	1.62	1.63
D5	ACC	70.83	90.42	72.78	95.28	95.42	94.17	94.17	93.33	93.33	**96.11**
	STD	7.24	4.72	9.73	3.62	3.41	3.48	4.02	4.72	2.27	3.87
D6	ACC	92.42	88.31	93.15	93.31	93.09	90.67	93.20	93.08	93.27	**94.10**
	STD	2.39	3.19	2.84	2.35	2.61	3.74	2.29	2.13	2.54	3.46
D7	ACC	94.12	97.45	93.33	94.51	96.57	96.08	95.00	95.20	96.47	**97.65**
	STD	4.89	2.02	4.56	3.46	2.19	2.46	3.74	3.63	2.17	2.07
D8	ACC	91.48	96.72	89.69	99.01	98.89	99.32	98.64	98.33	99.01	**99.32**
	STD	2.19	2.38	5.24	1.03	1.19	0.85	0.89	1.22	1.17	0.75
	Ave	83.68	87.39	84.12	88.30	88.51	87.53	88.26	88.04	87.56	**89.44**
	Rank1	0	0	0	0	0	0	0	0	0	8
	W/T/L	7/1/0	6/2/0	7/1/0	6/2/0	6/2/0	7/1/0	7/1/0	7/1/0	7/1/0	/

Table 2: The performance comparison (%) of the compared methods on benchmark datasets.

Acknowledgments

This work is supported by Jiangsu Provincial Key Research and Development Program(BE2021001-4), National Key Research and Development Program of China(2022YFB3305504), and National Natural Science Foundation of China (62002156, 61973151).

References

Aiolli, F.; and Donini, M. 2015. EasyMKL: a scalable multiple kernel learning algorithm. *Neurocomputing*, 169: 215–224.

Aksela, M.; and Laaksonen, J. 2006. Using diversity of errors for selecting members of a committee classifier. *Pattern Recognition*, 39(4): 608–623.

Arabacı, M. A.; Özkan, F.; Surer, E.; Jančovič, P.; and Temizel, A. 2021. Multi-modal egocentric activity recognition using multi-kernel learning. *Multimedia Tools and Applications*, 80(11): 16299–16328.

Ben-Hur, A.; and Noble, W. S. 2005. Kernel methods for predicting protein–protein interactions. *Bioinformatics*, 21(suppl_1): i38–i46.

Billionnet, A.; and Calmels, F. 1996. Linear programming for the 0–1 quadratic knapsack problem. *European Journal of Operational Research*, 92(2): 310–325.

Cai, X.; Nie, F.; Huang, H.; and Kamangar, F. 2011. Heterogeneous image feature integration via multi-modal spectral clustering. In *CVPR 2011*, 1977–1984.

Chamakura, L.; and Saha, G. 2022. Localized multiple kernel learning using graph modularity. *Pattern Recognition Letters*, 155: 27–33.

Cortes, C.; Mohri, M.; and Rostamizadeh, A. 2009. Learning non-linear combinations of kernels. *Advances in neural information processing systems*, 22.

Ding, X.; Liu, J.; Yang, F.; and Cao, J. 2021a. Random compact Gaussian kernel: Application to ELM classification and regression. *Knowledge-Based Systems*, 217: 106848.

Ding, X.; Liu, J.; Yang, F.; and Cao, J. 2021b. Random radial basis function kernel-based support vector machine. *Journal of the Franklin Institute*, 358(18): 10121–10140.

Ding, Y.; Tang, J.; and Guo, F. 2020. Identification of drug–target interactions via dual laplacian regularized least squares with multiple kernel fusion. *Knowledge-Based Systems*, 204: 106254.

Fan, Q.; Wang, Z.; Zha, H.; and Gao, D. 2017. MREKLM: A fast multiple empirical kernel learning machine. *Pattern Recognition*, 61: 197–209.

Giacinto, G.; and Roli, F. 2001. An approach to the automatic design of multiple classifier systems. *Pattern recognition letters*, 22(1): 25–33.

Gu, Y.; Liu, T.; Jia, X.; Benediktsson, J. A.; and Chanussot, J. 2016. Nonlinear Multiple Kernel Learning With Multiple-Structure-Element Extended Morphological Profiles for Hyperspectral Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 54(6): 3235–3247.

Guo, X.; Zhu, C.; Hao, J.; and Zhang, S. 2023. Multi-step wind speed prediction based on an improved multi-objective seagull optimization algorithm and a multi-kernel extreme learning machine. *Applied Intelligence*, 53(13): 16445–16472.

Gönen, M. 2013. Supervised Multiple Kernel Embedding for Learning Predictive Subspaces. *IEEE Transactions on Knowledge and Data Engineering*, 25(10): 2381–2389.

Han, Y.; Yang, Y.; Li, X.; Liu, Q.; and Ma, Y. 2018. Matrix-Regularized Multiple Kernel Learning via $(r,\ p)$ Norms. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10): 4997–5007.

Hartmanis, J. 1982. Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson). *Siam Review*, 24(1): 90.

Helmberg, C.; and Rendl, F. 2000. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3): 673–696.

Huang, G.; Zhou, H.; Ding, X.; and Zhang, R. 2012. Extreme Learning Machine for Regression and Multiclass Classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2): 513–529.

Jiang, H.; Shen, D.; Ching, W.-K.; and Qiu, Y. 2022. A high-order norm-product regularized multiple kernel learning framework for kernel optimization. *Information Sciences*, 606: 72–91.

Kloft, M.; Brefeld, U.; Sonnenburg, S.; and Zien, A. 2011. Lp-norm multiple kernel learning. *The Journal of Machine Learning Research*, 12: 953–997.

Ko, A. H.-R.; Sabourin, R.; and de Souza Britto Jr, A. 2009. Compound diversity functions for ensemble selection. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04): 659–686.

Lemaréchal, C.; and Oustry, F. 1999. *Semidefinite relaxations and Lagrangian duality with application to combinatorial optimization*. Ph.D. thesis, INRIA.

Létocart, L.; Plateau, M.-C.; and Plateau, G. 2014. An efficient hybrid heuristic method for the 0-1 exact k-item quadratic knapsack problem. *Pesquisa Operacional*, 34: 49–72.

Liu, J.; Wu, Z.; Xiao, L.; and Yan, H. 2020. Learning Multiple Parameters for Kernel Collaborative Representation Classification. *IEEE Transactions on Neural Networks and Learning Systems*, 31(12): 5068–5078.

Liu, X.; Dou, Y.; Yin, J.; Wang, L.; and Zhu, E. 2016. Multiple kernel k-means clustering with matrix-induced regularization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Peng, J.; Zhu, X.; Wang, Y.; An, L.; and Shen, D. 2019. Structured sparsity regularized multiple kernel learning for Alzheimer's disease diagnosis. *Pattern Recognition*, 88: 370–382.

Shah, F.; Shah, M. S.; Akram, W.; Manzoor, A.; Mahmoud, R. O.; and Abdelminaam, D. S. 2021. Sign Language Recognition Using Multiple Kernel Learning: A Case Study of Pakistan Sign Language. *IEEE Access*, 9: 67548–67558.

Shen, X.; Lu, K.; Mehta, S.; Zhang, J.; Liu, W.; Fan, J.; and Zha, Z. 2021. MKEL: Multiple Kernel Ensemble Learning via Unified Ensemble Loss for Image Classification. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 12(4): 1–21.

Song, H.; Thiagarajan, J. J.; Sattigeri, P.; and Spanias, A. 2018. Optimizing Kernel Machines Using Deep Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 29(11): 5528–5540.

Steinwart, I.; Hush, D.; and Scovel, C. 2011. Training SVMs Without Offset. *Journal of Machine Learning Research*, 12(1).

Strobl, E. V.; and Visweswaran, S. 2013. Deep multiple kernel learning. In *2013 12th International Conference on Machine Learning and Applications*, volume 1, 414–417. IEEE.

Suykens, J. A.; and Vandewalle, J. 1999. Least squares support vector machine classifiers. *Neural processing letters*, 9(3): 293–300.

Wang, T.; Lu, J.; and Zhang, G. 2018. Two-Stage Fuzzy Multiple Kernel Learning Based on Hilbert–Schmidt Independence Criterion. *IEEE Transactions on Fuzzy Systems*, 26(6): 3703–3714.

Wu, D.; Wang, B.; Precup, D.; and Boulet, B. 2020. Multiple Kernel Learning-Based Transfer Regression for Electric Load Forecasting. *IEEE Transactions on Smart Grid*, 11(2): 1183–1192.

Xia, H.; and Hoi, S. C. 2013. MKBoost: A Framework of Multiple Kernel Boosting. *IEEE Transactions on Knowledge and Data Engineering*, 25(7): 1574–1586.

Xu, J.; Han, J.; and Nie, F. 2016. Discriminatively Embedded K-Means for Multi-View Clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yang, X.; Feng, S.; Wang, D.; and Zhang, Y. 2020. Image-text multimodal emotion classification via multi-view attentional network. *IEEE Transactions on Multimedia*, 23: 4014–4026.

Yu, X.; Gong, X.; and Jiang, H. 2020. Heterogeneous multiple kernel learning for breast cancer outcome evaluation. *BMC bioinformatics*, 21(1): 1–20.

Zhang, X.; Liu, J.; Shen, J.; Li, S.; Hou, K.; Hu, B.; Gao, J.; and Zhang, T. 2020. Emotion recognition from multimodal physiological signals using a regularized deep fusion of kernel machine. *IEEE transactions on cybernetics*, 51(9): 4386–4399.