Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 4579214, 12 pages
https://doi.org/10.1155/2017/4579214

Research Article

Improving Localized Multiple Kernel Learning via Radius-Margin Bound

Xiaoming Wang,[1,2] Zengxi Huang,[1] and Yajun Du[1]

[1]*School of Computer and Software Engineering, Xihua University, Chengdu 610039, China*
[2]*Robotics Research Center, Xihua University, Chengdu 610039, China*

Correspondence should be addressed to Xiaoming Wang; wangxmwm@gmail.com

Received 28 July 2016; Accepted 21 November 2016; Published 9 January 2017

Academic Editor: Wanquan Liu

Localized multiple kernel learning (LMKL) is an effective method of multiple kernel learning (MKL). It tries to learn the optimal kernel from a set of predefined basic kernels by directly using the maximum margin principle, which is embodied in support vector machine (SVM). However, LMKL does not consider the radius of minimum enclosing ball (MEB) which actually impacts the error bound of SVM as well as the separating margin. In the paper, we propose an improved version of LMKL, which is named ILMKL. The proposed method explicitly takes into consideration both the margin and the radius and so achieves better performance over its counterpart. Moreover, the proposed method can automatically tune the regularization parameter when learning the optimal kernel. Consequently, it avoids using the time-consuming cross-validation process to choose the parameter. Comprehensive experiments are conducted and the results well demonstrate the effectiveness and efficiency of the proposed method.

1. Introduction

Over the past decade, kernel methods [1] have drawn a lot of attention of researchers in the machine learning community and have been widely applied. A kernel characterizes the similarity between two samples [2]. Actually, the performance of a kernel-based algorithm often strongly depends on the selection of the kernel. Generally, an unsuitable kernel would lead to a poor performance. Therefore, it is very critical to choose a suitable kernel for a kernel-based algorithm.

Recent researches on kernel methods have highlighted the requirement to learn a suitable kernel matrix or function from the training data. A generic technique is known as multiple kernel learning (MKL) [3]. Given a set of predefined basic kernel functions, MKL tries to find their combination by employing a criterion which maximizes a generalization performance measure or minimizes an error bound. Actually, the practical problems frequently involve multiple heterogeneous data sources [4]. Thus, MKL is in accord with this fact. Many studies [5–13] have shown that MKL can generally find the suitable combination of basic kernel functions and so can usually achieve better performance in contrast with single kernel. The idea of MKL has been applied in all sorts of kernel-based algorithms, for example, support vector machine (SVM), which is a powerful and excellent machine learning method based on Vapnik's statistical learning theory [14]. In the paper, we will only focus on the SVM-based MKL.

Localized multiple kernel learning (LMKL) [15–17], as a method of MKL, is an attractive method which combines multiple heterogeneous attributes according to their discriminative ability for each individual instance. Generally, other MKL methods try to learn a global combination in the whole input space [2, 5, 6, 18, 19], whereas LMKL believes that a sample-specific local combination should most likely better reflect the distinctive characteristics of each instance and so embodies the idea. This is the key difference between other MKL methods and LMKL. Overall, LMKL consists of an SVM learning problem and a parametric gating model. The gating model is used to assign local weights to predefined basic kernels. In LMKL, a two-step alternate optimization method is employed to train the two components. In contrast

with other MKL methods, LMKL generally provides fewer support vectors but can achieve statistically similar accuracy results. The idea of LMKL has been extended to other kernel-based methods and successfully used in some practical applications [20–23].

However, LMKL learns the kernel function (essentially the parameters of the gating model) only by maximizing the margin which is embodied in single-kernel-based SVM. A key fact is that the generalization performance of SVM depends not only on the separating margin, but also on the radius of the smallest ball that encloses the data [24–28]. In fact, it is not necessary that standard SVM (or single-kernel SVM) exploits the radius. The reason is that the radius of the minimum enclosing ball (MEB) is fixed once the kernel including its parameters is selected. However, in the context of LMKL, the radius is not fixed but is a function of the parameters of the gating model.

Actually, several attempts have recently been directed at incorporating the radius into SVM-based MKL [29]. However, most of these works direct SVM with ℓ_2 soft margin (ℓ_2-SVM) because the problem of ℓ_2-SVM can be transformed into a form of SVM with hard margin, in which the radius-margin bound holds and can be used to conduct model selection [25]. Unfortunately, for SVM with ℓ_1 soft margin (ℓ_1-SVM), the radius-margin bound does not hold, as we have no way of reducing the formulation of ℓ_1-SVM to a form of SVM with hard margin. So, one cannot directly utilize the radius-margin bound in LMKL since its formulation is rooted in ℓ_1-SVM. However, in [27], Chung et al. investigated several heuristic bounds for SVM and developed a modified radius-margin bound to conduct model selection for ℓ_1-SVM. The experimental results have indicated its effectiveness.

Inspired by the work of Chung et al. in [27] and aiming at the drawback of LMKL, in this paper, we propose an improved version of LMKL, which is named ILMKL. A noticeable characteristic of the proposed method is that it takes account of both the separating margin and the radius of MEB; that is, it integrates the information of the margin and the radius to measure the goodness of the kernel and learn the parameters of the gating model. Actually, a key insight of our work is that learning the parameters of the gating model in LMKL is similar to conducting model selection. Analogous to LMKL, the problem of the proposed method can be efficiently resolved in a coupled manner through employing the two-step alternate optimization method. Moreover, the proposed method treats the regularization parameter as an extra parameter that can be automatically learned. Consequently, we can jointly tune it with the parameters of the gating during the learning kernel function process. This improves the computational efficiency of the proposed method to some extent since it avoids the time-consuming cross-validation process. Comprehensive experiments are conducted and the experimental results well demonstrate the efficiency and effectiveness of the proposed method.

The rest of the paper is organized as follows. Section 2 reviews the related work. In Section 3, we first present the formulation of the proposed method and then detail how to solve the optimization problem. After that, we conduct some preliminary discussion on the proposed method and outline the algorithm step. Section 4 reports the experimental results, and the conclusions are drawn in Section 5.

2. Preliminaries

In the paper, we suppose a training dataset which consists of N samples and is represented by $\mathscr{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where the samples $\mathbf{x}_i \in \mathbb{R}^d$ and its corresponding labels $y_i \in \{1, -1\}$. Here, $i = 1, \ldots, N$ and d is the dimension of the sample space. Denote, for convenience, by $\mathscr{I}$ the set of all indices; that is, $\mathscr{I} = \{1, \ldots, N\}$.

2.1. Radius-Margin Bound for SVM. SVM embodies the structural risk minimization principle, which is related to the probability of incorrectly classifying an unknown sample. Geometrically, the key idea of SVM is to construct a separating hyperplane in the data space through employing the maximal margin principle among two different classes of samples [14]. In the nonlinear case, ℓ_1-SVM defines the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathscr{I}}\xi_i \\ \text{s.t.} \quad & y_i\left(\left\langle \mathbf{w}, \varphi\left(\mathbf{x}_i\right)\right\rangle + b\right) \geq 1 - \xi_i, \quad \xi_i \geq 0,\ i \in \mathscr{I}, \end{aligned} \tag{1}$$

where $\varphi(\mathbf{x}) : \mathscr{R}^d \to \mathscr{H}$, $\langle\cdot,\cdot\rangle$ is the inner product of two vectors, ξ_i represents the training error, and C is the regularization parameter that adjusts the training error and the regularization term $\|\mathbf{w}\|^2$. Problem (1) can be efficiently solved by transforming it to its corresponding dual problem [30], which is formulated as

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{J=1}^{N}\alpha_i\alpha_j y_i y_j k\left(\mathbf{x}_i, \mathbf{x}_j\right) \\ \text{s.t.} \quad & \sum_{i=1}^{N} y_i\alpha_i = 0, \quad C \geq \alpha_i \geq 0,\ i \in \mathscr{I}. \end{aligned} \tag{2}$$

Here, $k(\mathbf{x}_i, \mathbf{x}_j) = \langle\varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j)\rangle$ is called kernel function. Suppose that $\boldsymbol{\alpha}^* = [\alpha_1^*, \ldots, \alpha_N^*]^T$ solves the above optimization problem and b^* is the optimal threshold which can be computed by using the KKT condition of (1); the decision function of SVM is formulated as

$$f(\mathbf{x}) = \sum_{i\in\mathscr{I}}\alpha_i^* y_i k\left(\mathbf{x}_i, \mathbf{x}\right) + b^*. \tag{3}$$

In order to obtain a better performance in the practical applications, it is very important to choose suitable hyperparameters which include the regularization parameter C and the kernel parameter of the kernel function $k(\cdot,\cdot)$ for SVM. This is the so-called model selection [24]. Generally, one can empirically set these hyperparameters. But this is very hard work because one cannot know in advance the suitable hyperparameters when facing all kinds of practical applications. Many works have tried to find a good criterion

to automatically learn the related hyperparameters [24–26]. In [27], Chung et al. proposed the following bound for ℓ_1-SVM:

$$\left(R^2 + \frac{\Delta}{C}\right)\left(\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathcal{I}}\xi_i\right), \tag{4}$$

where Δ is a parameter and can generally be set as $\Delta = 1$ and R refers to the radius of MEB. This radius-margin bound is differentiable and is successfully used to conduct model selection for ℓ_1-SVM in [27].

2.2. Localized Multiple Kernel Learning. In the context of MKL, we assume that there exist M different mappings $\varphi_m(\mathbf{x}) : \mathbb{R}^d \to \mathcal{H}_m$ $(m = 1, \ldots, M)$, the mth mapping of which is endowed with the base kernel $k_m(\cdot,\cdot)$ of associated reproducing kernel Hilbert space (RKHS) $\mathcal{H}_m$.

As a method of MKL, LMKL is based on ℓ_1-SVM and defines its optimization problem as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\mathbf{v}_m} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & y_i\left(\left\langle\widehat{\mathbf{w}}, \widehat{\varphi}(\mathbf{x}_i)\right\rangle + b\right) \geq 1 - \xi_i, \quad \xi_i \geq 0,\ i \in \mathcal{I}, \end{aligned} \tag{5}$$

where $\widehat{\mathbf{w}} = [\mathbf{w}_1; \ldots; \mathbf{w}_m]$, $\widehat{\varphi}(\mathbf{x}_i) = [\eta_1(\mathbf{x}_i)\varphi_1(\mathbf{x}_i); \ldots; \eta_M(\mathbf{x}_i)\varphi_M(\mathbf{x}_i)]$, $b \in \mathbb{R}$, C is a regularization parameter that adjusts the training error and the regularization term $\|\widehat{\mathbf{w}}\|^2$, and ξ_i represents the training error. Here, $\mathbf{w}_m \in \mathcal{H}_m$, $\varphi_m(\mathbf{x}) \in \mathcal{H}_m$, and $\eta_m(\mathbf{x})$ is a gating function defined up to a set of parameters which need to be learned from the training data. Further, by using duality, we have the dual formulation of the primal problem in (5) as follows:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{J=1}^{N}\alpha_i\alpha_j y_i y_j \widehat{k}\left(\mathbf{x}_i, \mathbf{x}_j\right) \\ \text{s.t.} \quad & \sum_{i=1}^{N} y_i\alpha_i = 0, \quad C \geq \alpha_i \geq 0,\ i \in \mathcal{I}, \end{aligned} \tag{6}$$

where the locally combined kernel function is defined as

$$\begin{aligned} \widehat{k}\left(\mathbf{x}_i, \mathbf{x}_j\right) &= \left\langle\widehat{\varphi}\left(\mathbf{x}_i\right), \widehat{\varphi}\left(\mathbf{x}_j\right)\right\rangle \\ &= \sum_{m=1}^{M}\eta_m\left(\mathbf{x}_i\right)\eta_m\left(\mathbf{x}_j\right)k_m\left(\mathbf{x}_i, \mathbf{x}_j\right). \end{aligned} \tag{7}$$

If the used gating model $\eta_m(\mathbf{x})$ is constant (not a function of $\mathbf{x}$), LMKL finds a fixed combination over the whole input space and is similar to the original MKL formulation. The main advantage of LMKL is that it can achieve statistically similar accuracy results by storing fewer support vectors compared with the original MKL.

3. The Proposed LMKL Framework

In this section, we first present the primal optimization problem of the proposed method ILMKL and then detail how to solve it, and finally some preliminary discussion on the proposed method is given and the algorithm is outlined.

3.1. Primal Optimization Model of the Proposed Method. In the context of LMKL, it is easy to find that the radius of MEB is not fixed but is a function of the parameters of the gating model. Nevertheless, LMKL learns the parameters of the gating model only through using the separating margin. Therefore, LMKL ignores the fact that the generalization performance of SVM depends not only on the separating margin but also on the radius.

Actually, the purpose of learning the parameters of the gating model is in essence to yield an appropriate kernel matrix for good performance. In our opinion, this process is similar to model selection by which SVM chooses the appropriate parameters to achieve good performance. Therefore, following the basic idea of the work in [27], we define the primal optimization problem of ILMKL as follows:

$$\begin{aligned} \min_{\widehat{\mathbf{w}},b,\boldsymbol{\xi},\mathbf{a},R,\mathbf{v}_m,C} \quad & \left(R^2 + \frac{\Delta}{C}\right)\left(\frac{1}{2}\|\widehat{\mathbf{w}}\|^2 + C\sum_{i=1}^{N}\xi_i\right) \\ \text{s.t.} \quad & y_i\left(\left\langle\widehat{\mathbf{w}}, \widehat{\varphi}\left(\mathbf{x}_i\right)\right\rangle + b\right) \geq 1 - \xi_i, \\ & \xi_i \geq 0,\ i \in \mathcal{I} \\ & \left\|\widehat{\varphi}\left(\mathbf{x}_i\right) - \mathbf{a}\right\|^2 \leq R^2, \quad i \in \mathcal{I}. \end{aligned} \tag{8}$$

As in Section 2.2, here, $\widehat{\mathbf{w}} = [\mathbf{w}_1; \ldots; \mathbf{w}_m]$, $\widehat{\varphi}(\mathbf{x}_i) = [\eta_1(\mathbf{x}_i)\varphi_1(\mathbf{x}_i); \ldots; \eta_M(\mathbf{x}_i)\varphi_M(\mathbf{x}_i)]$, $\mathbf{w}_m \in \mathcal{H}_m$, $b \in \mathbb{R}$, $\varphi_m(\mathbf{x}) \in \mathcal{H}_m$, ξ_i represents the training error, and C is a regularization parameter that adjusts the training error and the regularization term $\|\widehat{\mathbf{w}}\|^2$. Note that $\|\widehat{\mathbf{w}}\|^2 = \sum_{m=1}^{M}\|\mathbf{w}_m\|^2$ and $\langle\widehat{\mathbf{w}}, \widehat{\varphi}(\mathbf{x}_i)\rangle = \sum_{m=1}^{M}\langle\mathbf{w}_m, \varphi_m(\mathbf{x}_i)\rangle$, and thus we can reformulate (8) into

$$\begin{aligned} \min_{\mathbf{w}_m,b,\boldsymbol{\xi},\mathbf{a},R,\mathbf{v}_m,C} \quad & \left(R^2 + \frac{\Delta}{C}\right)\left(\frac{1}{2}\sum_{m=1}^{M}\left\|\mathbf{w}_m\right\|^2 + C\sum_{i=1}^{N}\xi_i\right) \\ \text{s.t.} \quad & y_i\left(\sum_{m=1}^{M}\eta_m\left(\mathbf{x}_i\right)\left\langle\mathbf{w}_m, \varphi_m\left(\mathbf{x}_i\right)\right\rangle + b\right) \\ & \geq 1 - \xi_i, \\ & \xi_i \geq 0,\ i \in \mathcal{I} \\ & \left\|\sum_{m=1}^{M}\eta_m\left(\mathbf{x}_i\right)\left\langle\mathbf{w}_m, \varphi_m\left(\mathbf{x}_i\right)\right\rangle - \mathbf{a}\right\|^2 \leq R^2, \\ & i \in \mathcal{I}. \end{aligned} \tag{9}$$

Here, $\eta_m(\mathbf{x})$ is a gating function. As in [15], we employ the softmax gating model determining the parameters $\mathbf{v}_m$ ($m = 1, \ldots, M$) and it can be expressed as

$$\eta_m(\mathbf{x}_i) = \frac{\exp\left(v_{m0} + \sum_{d=1}^{D} v_{md} x_{id}\right)}{\sum_{p=1}^{M} \exp\left(v_{p0} + \sum_{d=1}^{D} v_{pd} x_{id}\right)} = \frac{\exp\left(\left\langle \mathbf{v}_m, \widetilde{\mathbf{x}}_i \right\rangle\right)}{\sum_{p=1}^{M} \exp\left(\left\langle \mathbf{v}_p, \widetilde{\mathbf{x}}_i \right\rangle\right)}, \tag{10}$$

where x_{id} is the dth feature of the ith sample, $\widetilde{\mathbf{x}}_i = [1, \mathbf{x}_i]$, and $\mathbf{v}_m = [v_{m0}, v_{m1}, v_{m2}, \ldots, v_{mM}] \in \mathbb{R}^{M+1}$ ($m = 1, \ldots, M$) are the parameters of the gating model associated with the mth kernel and the softmax guarantees nonnegativity. As pointed out in [17], one can use more complex gating models.

Obviously, in contrast with LMKL (5), ILMKL has two noticeable characteristics. One is that it takes into consideration the information of the radius and margin. Another characteristic of the proposed method is that it treats the regularization parameter C as a variable that can be automatically learned during the procedure of learning the parameters of the gating model. To sum up, the key insight of our method is that, in the context of LMKL, learning the parameters of the gating model is similar to conducting model selection for SVM.

3.2. Training with Alternating Optimization. Generally, it is very difficult to directly solve problem (9). In LMKL, a two-step alternate optimization method is employed to find the parameters of the gating model and the discriminant function. In our method ILMKL, we use the same strategy.

The first step is to fix $\{\mathbf{v}_m, C\}$ and solve (9) with respect to $\{\mathbf{w}_m, b, \boldsymbol{\xi}, \mathbf{a}, R\}$, and the second step is to optimize the parameters of $\{\mathbf{v}_m, C\}$ by using a gradient-descent method. The objective value obtained for a fixed $\{\mathbf{v}_m, C\}$ is an upper bound for (9) and the parameters of $\{\mathbf{v}_m, C\}$ are optimized according to the current solution. The objective value obtained at the next iteration cannot be greater than the current one due to the use of gradient-descent procedure. And as iterations progress with a proper step size selection procedure, the objective value of (9) never increases. Note that this way does not guarantee convergence to the global optimum and so the initial parameters of $\{\mathbf{v}_m, C\}$ may affect the solution quality.

In this subsection, we will discuss how to solve problem (9) when fixing $\{\mathbf{v}_m, C\}$. In the following two subsections, we will, respectively, discuss how to optimize $\mathbf{v}_m$ and C.

For a fixed $\{\mathbf{v}_m, C\}$, we have

$$\begin{aligned} \min_{\mathbf{w}_m, b, \boldsymbol{\xi}, \mathbf{a}, R, \mathbf{v}_m, C} \quad & \left(R^2 + \frac{\Delta}{C}\right)\left(\frac{1}{2}\sum_{m=1}^{M}\left\|\mathbf{w}_m\right\|^2 + C\sum_{i=1}^{N}\xi_i\right) \\ = \quad & \left(\min_{\mathbf{a}, R}\left(R^2 + \frac{\Delta}{C}\right)\right) \\ & \times \left(\min_{\mathbf{w}_m, b, \boldsymbol{\xi}}\left(\frac{1}{2}\sum_{m=1}^{M}\left\|\mathbf{w}_m\right\|^2 + C\sum_{i=1}^{N}\xi_i\right)\right). \end{aligned} \tag{11}$$

Here, we set

$$\begin{aligned} \mathfrak{I}(\mathbf{v}_m, C) &= \left(\min_{\mathbf{a}, R}\left(R^2 + \frac{\Delta}{C}\right)\right) \times \left(\min_{\mathbf{w}_m, b, \boldsymbol{\xi}}\left(\frac{1}{2}\sum_{m=1}^{M}\left\|\mathbf{w}_m\right\|^2 + C\sum_{i=1}^{N}\xi_i\right)\right) \\ J_1(\mathbf{v}_m, C) &= \min_{\mathbf{a}, R}\left(R^2 + \frac{\Delta}{C}\right) \\ J_2(\mathbf{v}_m, C) &= \min_{\mathbf{w}_m, b}\left(\frac{1}{2}\sum_{m=1}^{M}\left\|\mathbf{w}_m\right\|^2 + C\sum_{i=1}^{N}\xi_i\right). \end{aligned} \tag{12}$$

Therefore, for a fixed $\{\mathbf{v}_m, C\}$, problem (9) of the proposed method can be expressed as

$$\begin{aligned} \min_{\mathbf{w}_m, b, \boldsymbol{\xi}, \mathbf{a}, R} \quad & \mathfrak{I}(\mathbf{v}_m, C) = J_1(\mathbf{v}_m, C) \times J_2(\mathbf{v}_m, C), \\ \text{where } J_1(\mathbf{v}_m, C) = \min_{\mathbf{w}_m, b, \mathbf{a}, R} \quad & \left(R^2 + \frac{\Delta}{C}\right) \\ \text{s.t.} \quad & \left\|\sum_{m=1}^{M}\eta_m(\mathbf{x}_i)\left\langle \mathbf{w}_m, \varphi_m(\mathbf{x}_i)\right\rangle - \mathbf{a}\right\|^2 \le R^2, \quad i \in \mathcal{I}, \\ J_2(\mathbf{v}_m, C) = \min_{\mathbf{w}_m, b} \quad & \frac{1}{2}\sum_{m=1}^{M}\left\|\mathbf{w}_m\right\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & y_i\left(\sum_{m=1}^{M}\eta_m(\mathbf{x}_i)\left\langle \mathbf{w}_m, \varphi_m(\mathbf{x}_i)\right\rangle + b\right) \ge 1 - \xi_i, \quad \xi_i \ge 0, \ i \in \mathcal{I}. \end{aligned} \tag{13}$$

Note that if we fix the gating model parameters and the regularization parameter, the optimization problem (13) becomes convex. In order to find its solution, we can switch it into the dual optimization problem. By using duality, the dual problem of the primal problem in (13) can be formulated as

$$\begin{aligned}
\min_{\mathbf{w}_m,b,\boldsymbol{\xi},\mathbf{a},R} \quad & \mathfrak{I}\left(\mathbf{v}_m, C\right) \\
& = J_1\left(\mathbf{v}_m, C\right) \times J_2\left(\mathbf{v}_m, C\right), \\
\text{where } J_1\left(\mathbf{v}_m, C\right) = \frac{\Delta}{C} + \max_{\boldsymbol{\beta}} \quad & \sum_{i=1}^{N} \beta_i k\left(\mathbf{x}_i, \mathbf{x}_i\right) \\
& - \sum_{i=1}^{N}\sum_{J=1}^{N} \beta_i \beta_j \widehat{k}\left(\mathbf{x}_i, \mathbf{x}_j\right) \\
\text{s.t.} \quad & \sum_{i=1}^{N} \beta_i = 1, \quad \beta_i \geq 0, \ i \in \mathcal{F}, \\
J_2\left(\mathbf{v}_m, C\right) = \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \alpha_i \\
& - \frac{1}{2}\sum_{i=1}^{N}\sum_{J=1}^{N} \alpha_i \alpha_j y_i y_j \widehat{k}\left(\mathbf{x}_i, \mathbf{x}_j\right) \\
\text{s.t.} \quad & \sum_{i=1}^{N} y_i \alpha_i = 0, \\
& C \geq \alpha_i \geq 0, \ i \in \mathcal{F},
\end{aligned} \tag{14}$$

where the locally combined kernel function $\widehat{k}(\mathbf{x}_i, \mathbf{x}_j)$ is defined as (7). Obviously, this formulation corresponds to, respectively, solving a canonical SVM dual problem and a canonical support vector domain description (SVDD) [31] dual problem with the kernel matrix $\widehat{K}(\eta) = (\widehat{k}(\mathbf{x}_i, \mathbf{x}_j))_{N\times N}$, which should be positive semidefinite.

Finally, once the final gating model $\eta_m(\mathbf{x})$ has been learned and problem (14) is solved, the resulting discriminant function of the proposed method ILMKL can be expressed as follows:

$$f\left(\mathbf{x}\right) = \operatorname{sign}\left(\sum_{i=1}^{N}\sum_{m=1}^{M} \alpha_i \eta_m\left(\mathbf{x}_i\right) \eta_m\left(\mathbf{x}\right) k_m\left(\mathbf{x}_i, \mathbf{x}\right) + b\right). \tag{15}$$

3.3. Optimizing the Parameters of the Gating Model. In order to optimize the parameters $\mathbf{v}_m$ of the gating model $\eta_m(\mathbf{x})$ by using a gradient-descent method, one needs to calculate the derivatives of the primal objective with respect to the parameters $\mathbf{v}_m$. Next, we will discuss how to calculate the derivatives of the parameters.

First, note that

$$\begin{aligned}
\frac{\partial\left(\exp\left(\left\langle \mathbf{v}_k, \widetilde{\mathbf{x}}\right\rangle\right)\right)}{\partial \mathbf{v}_m} &= \begin{cases} 0, & k \neq m \\ \exp\left(\left\langle \mathbf{v}_k, \widetilde{\mathbf{x}}\right\rangle\right)\widetilde{\mathbf{x}}, & k = m \end{cases} \\
&= \delta_m^k \exp\left(\left\langle \mathbf{v}_k, \widetilde{\mathbf{x}}\right\rangle\right)\widetilde{\mathbf{x}}.
\end{aligned} \tag{16}$$

So, we have

$$\frac{\partial \eta_k\left(\mathbf{x}\right)}{\partial \mathbf{v}_m} = \delta_m^k \eta_k\left(\mathbf{x}\right)\widetilde{\mathbf{x}} - \eta_k\left(\mathbf{x}\right)\eta_m\left(\mathbf{x}\right)\widetilde{\mathbf{x}}. \tag{17}$$

Thus, we can calculate the derivatives of $k(\mathbf{x}_i, \mathbf{x}_j)$ with respect to the gating model parameters $\mathbf{v}_m$ as follows:

$$\begin{aligned}
\frac{\partial\left(k\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)}{\partial \mathbf{v}_m} = \sum_{k=1}^{M} k_k\left(\mathbf{x}_i, \mathbf{x}_j\right) \Bigg(& \eta_k\left(\mathbf{x}_j\right) \frac{\partial\left(\eta_k\left(\mathbf{x}_i\right)\right)}{\partial \mathbf{v}_m} \\
& + \eta_k\left(\mathbf{x}_i\right) \frac{\partial\left(\eta_k\left(\mathbf{x}_j\right)\right)}{\partial \mathbf{v}_m} \Bigg).
\end{aligned} \tag{18}$$

Further, according to the above formula, the following can be obtained:

$$\begin{aligned}
\frac{\partial\left(J_1\left(\mathbf{v}_m, C\right)\right)}{\partial \mathbf{v}_m} &= \sum_{i=1}^{N} \beta_i \frac{\partial\left(k\left(\mathbf{x}_i, \mathbf{x}_i\right)\right)}{\partial \mathbf{v}_m} \\
&\quad - \sum_{i=1}^{N}\sum_{J=1}^{N} \beta_i \beta_j \frac{\partial\left(k\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)}{\partial \mathbf{v}_m} \\
\frac{\partial\left(J_2\left(\mathbf{v}_m, C\right)\right)}{\partial \mathbf{v}_m} &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{J=1}^{N} \alpha_i \alpha_j y_i y_j \frac{\partial\left(k\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)}{\partial \mathbf{v}_m}.
\end{aligned} \tag{19}$$

Finally, the derivatives of $\mathfrak{I}(\mathbf{v}_m, C)$ with respect to the parameters $\mathbf{v}_m$ of the gating model $\eta_m(\mathbf{x})$ can be formulated as

$$\begin{aligned}
\frac{\partial\left(\mathfrak{I}\left(\mathbf{v}_m, C\right)\right)}{\partial \mathbf{v}_m} &= J_2\left(\mathbf{v}_m, C\right) \frac{\partial\left(J_1\left(\mathbf{v}_m, C\right)\right)}{\partial \mathbf{v}_m} \\
&\quad + J_1\left(\mathbf{v}_m, C\right) \frac{\partial\left(J_2\left(\mathbf{v}_m, C\right)\right)}{\partial \mathbf{v}_m}.
\end{aligned} \tag{20}$$

3.4. Optimizing the Regularization Parameter. In our method, the regularization parameter C is treated as a variable that can be learned when learning the gating model. Similar to the process of optimizing the parameter of the gating model, we employ a gradient-descent method to optimize the regularization parameter C and so the derivative of (14) with respect to C is needed. In the following, we will discuss how to compute the derivative.

Actually, the derivative of (14) $\mathfrak{I}(\mathbf{v}_m, C)$ with respect to C can be expressed as

$$\begin{aligned}
\frac{\partial\left(\mathfrak{I}\left(\mathbf{v}_m, C\right)\right)}{\partial C} &= J_2\left(\mathbf{v}_m, C\right) \frac{\partial\left(J_1\left(\mathbf{v}_m, C\right)\right)}{\partial C} \\
&\quad + J_1\left(\mathbf{v}_m, C\right) \frac{\partial\left(J_2\left(\mathbf{v}_m, C\right)\right)}{\partial C}.
\end{aligned} \tag{21}$$

(1) Initialize $\ln(C_{\text{init}})$ and initialize $\mathbf{v}_m$ to small random numbers for $m = 1, \ldots, M$;
(2) **while** stopping criterion not met **do**
(3) Compute C with $C = \exp(\ln(C))$;
(4) Calculate $\widehat{k}(\mathbf{x}_i, \mathbf{x}_j)$ with the gating model according to (7) when fixing $\mathbf{v}_m$;
(5) Compute $\mathfrak{J}(\mathbf{v}_m, C)$ by using an canonical SVM solver and an canonical SVDD solver with $\widehat{k}(\mathbf{x}_i, \mathbf{x}_j)$ according to (14);
(6) Compute $\partial(\mathfrak{J}(\mathbf{v}_m, C))/\partial \mathbf{v}_m$ for $m = 1, \ldots, M$ with (20);
(7) Compute $\partial \mathfrak{J}(\mathbf{v}_m, C)/\partial(\ln(C))$ with (23);
(8) Update $\mathbf{v}_m$ and $\ln(C)$ by the gradient-descent method;
(9) **end while**

Algorithm 1: ILMKL.

Obviously, one obtains in advance $\partial(J_1(\mathbf{v}_m, C))/\partial C$ and $\partial(J_2(\mathbf{v}_m, C))/\partial C$, which are, respectively, computed as

$$\begin{aligned}
&\frac{\partial J_1(\mathbf{v}_m, C)}{\partial C} = \frac{\partial\left(R^2 + \Delta/C\right)}{\partial C} = \frac{\partial(\Delta/C)}{\partial C} = -\frac{\Delta}{C^2}, \\
&\frac{\partial\left(J_2(\mathbf{v}_m, C)\right)}{\partial C} \\
&\quad = \frac{1}{C}\left(\sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j k\left(\mathbf{x}_i, \mathbf{x}_j\right)\right) \\
&\quad = \frac{1}{C} J_2(\mathbf{v}_m, C).
\end{aligned} \tag{22}$$

3.5. Discussion. It should be noted that $C > 0$ must hold in the whole procedure. However, actually in the iterations, this condition may be broken. In order to deal with this problem, following [27], we can use $\ln(C)$ instead of C in the solving procedure. The reason is that $\ln(C)$ can be any real number when $C > 0$. Thus, the positivity of C is dodged. Here, we need to rewrite the above partial derivatives. According to the chain rules, we can modify (21) as the following:

$$\frac{\partial \mathfrak{J}(\mathbf{v}_m, C)}{\partial(\ln(C))} = J_2(\mathbf{v}_m, C)\frac{\partial J_1(\mathbf{v}_m, C)}{\partial(\ln(C))} + J_1\frac{\partial J_2(\mathbf{v}_m, C)}{\partial(\ln(C))}, \tag{23}$$

where

$$\begin{aligned}
\frac{\partial J_1(\mathbf{v}_m, C)}{\partial(\ln(C))} &= C\frac{\partial J_1(\mathbf{v}_m, C)}{\partial C}, \\
\frac{\partial J_2(\mathbf{v}_m, C)}{\partial(\ln(C))} &= C\frac{\partial J_2(\mathbf{v}_m, C)}{\partial C}.
\end{aligned} \tag{24}$$

Finally, according to the above discussion, we outline the complete algorithm of ILMKL in Algorithm 1.

4. Experiments

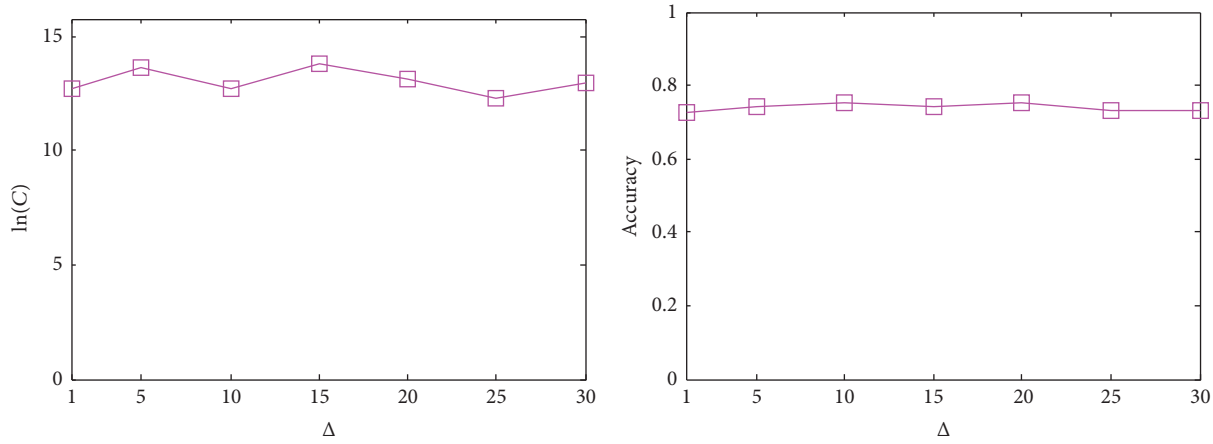
In this section, the experimental results will be reported. In the first experiment, we investigate the influence of parameters on ILMKL performance. In the second experiment, we further explore the possibility of learning the regularization parameter C under different initial value on a synthetic dataset. In the third experiment, we conduct the experiments on several UCI datasets and compare the proposed method with traditional MKL methods.

4.1. Parameter Influence on Performance of the Proposed Method. In the training procedure of LMKL, the regularization parameter C must be predefined. In our method ILMKL, the parameter C can be automatically tuned during the learning process of the parameters of the gating model. However, we need to first set the parameter Δ and the initial value (denoted by C_{init}) of C. In [27], the authors suggested $\Delta = 1$ and $\ln(C_{\text{init}}) = 0$. In this subsection, we will investigate the influence of these parameters on the final learned regularization parameter C and the classification performance of the proposed method.

We used the Sonar dataset, which was selected from the UCI repository [32]. In the experiment, 50% of the datasets were randomly selected for training and the rest for testing. The data were preprocessed in the following way: first, the mean and the standard deviation of each feature were computed according to training data; then, training examples were normalized to have mean 0 by subtracting the mean and unit variance; finally, testing examples were correspondingly preprocessed using the mean and the standard deviation. The base kernels include one linear kernel and one polynomial kernel with degree of two. All kernel matrices are calculated and normalized to unit trace before training.

Figure 1 shows the experimental results under different Δ. From Figure 1(a), we can find that the value of Δ actually influences the final value of learned C. The reason, in our opinion, is that the algorithm may fall into a local minimum since we adopt the gradient-descent method which cannot guarantee finding the global minimum. Actually, the final values of learned C under different Δ are close to each other on the whole. Moreover, the classification accuracies under different Δ have almost no difference according to Figure 1(b). Figure 2 shows the experimental results under different initial value of C. From Figure 2(a), similar to the case which is shown in Figure 1(a), we can find that the initial value C_{init} of C also impacts the final learned C. However, according to Figure 2(b), the initial value scarcely influences the classification accuracy.

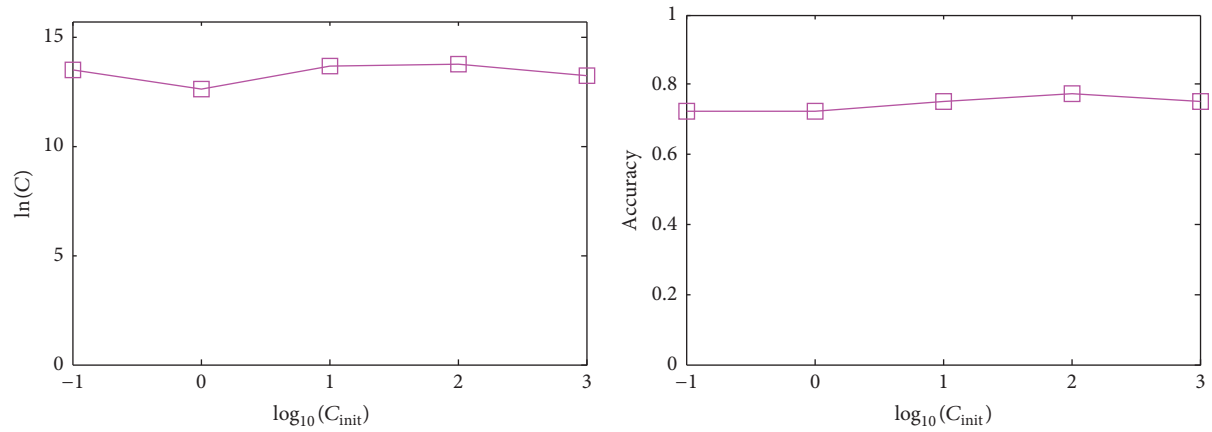
Therefore, it can be concluded that the proposed method is effective to learn a suitable C in the SVM-based LMKL scenario under different Δ and initial value C_{init} of C.



(a) The final learned C of ILMKL under different Δ

(b) The classification accuracy of ILMKL under different Δ

Figure 1: The experimental results of ILMKL under different Δ on the Sonar dataset.



(a) The final learned C of ILMKL under different initial value C_{init}

(b) The classification accuracy of ILMKL under different initial value C_{init}

Figure 2: The experimental results of ILMKL under the initial value C_{init} on the Sonar dataset.

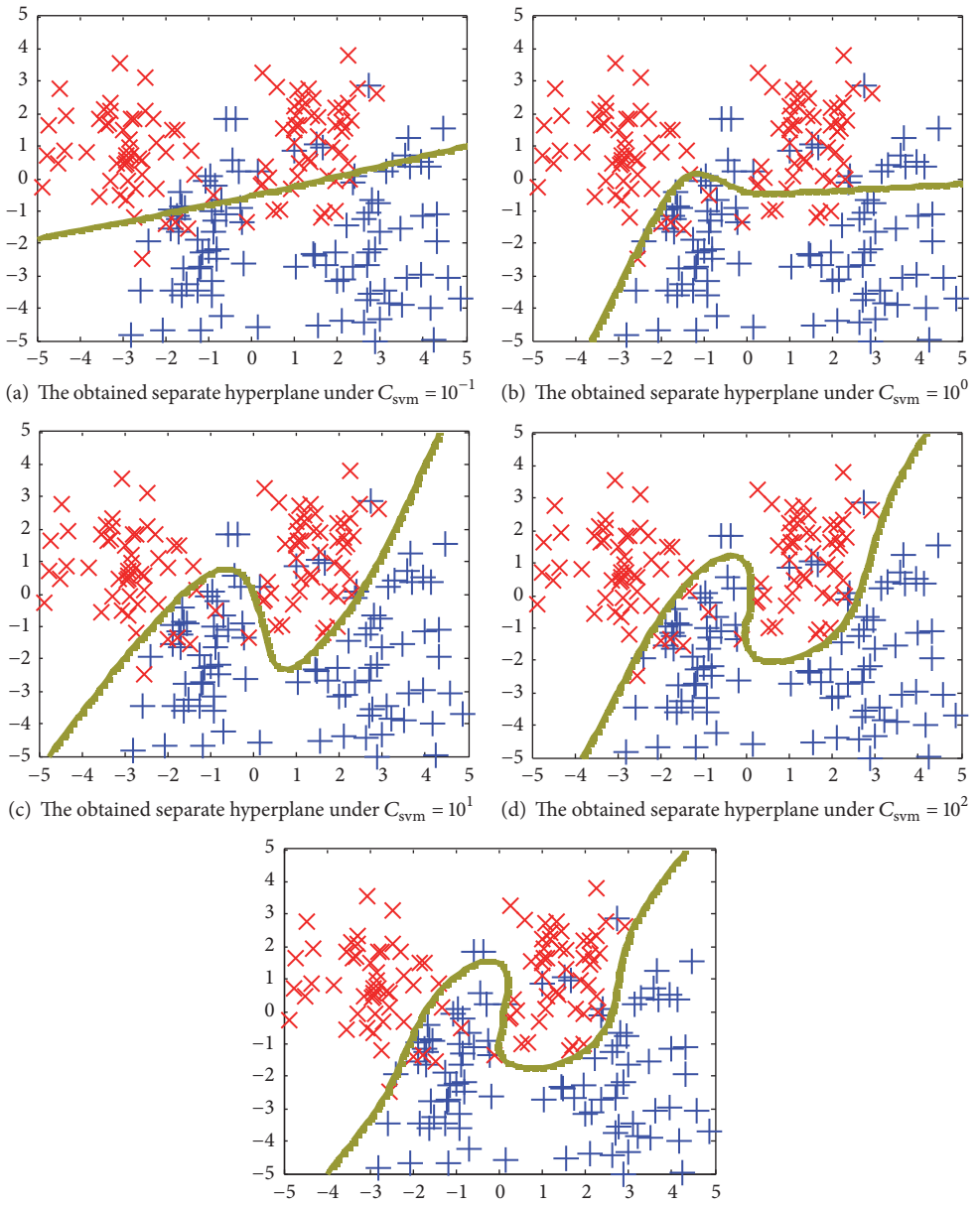
4.2. Experimental Results on Synthetic Dataset. In the above experiments, the final regularization parameter learned C is always much larger than the initial value C_{init} of C and so C almost always increases in the learning progress. In these experiments, we will show that the final learned C actually can be larger or smaller than the initial value C_{init} of C.

Following [15], we create a synthetic dataset, which consists of two classes, and each class contains 200 samples. The samples come from four Gaussian components (two for each class), and each component, respectively, has the following mean vector and covariance matrix:

$$\mu_1 = [-3.0, +1.0],$$
$$\mu_2 = [+1.0, +1.0],$$
$$\mu_3 = [-1.0, -2.2],$$
$$\mu_4 = [+3.0, -2.2]$$
$$\Sigma_1 = \begin{bmatrix} 0.8 & 0.0 \\ 0.0 & 2.0 \end{bmatrix},$$
$$\Sigma_2 = \begin{bmatrix} 0.8 & 0.0 \\ 0.0 & 2.0 \end{bmatrix},$$
$$\Sigma_3 = \begin{bmatrix} 0.8 & 0.0 \\ 0.0 & 4.0 \end{bmatrix},$$
$$\Sigma_4 = \begin{bmatrix} 0.8 & 0.0 \\ 0.0 & 4.0 \end{bmatrix}, \tag{25}$$

where the samples in class 1 are from the first two components (denoted by red ×) and others belong to class 2 (denoted by blue +). Here, we adopt the same base kernels as in Section 4.1, that is, linear kernels and one polynomial kernel with degree of two. Before training, all kernel matrices are computed and preprocessed to unit trace in advance.

Figure 3 illustrates the experimental results of LMKL under different regularization parameter C. It can be easily found that the values of the regularization parameter C



(a) The obtained separate hyperplane under $C_{\text{svm}} = 10^{-1}$

(b) The obtained separate hyperplane under $C_{\text{svm}} = 10^{0}$

(c) The obtained separate hyperplane under $C_{\text{svm}} = 10^{1}$

(d) The obtained separate hyperplane under $C_{\text{svm}} = 10^{2}$

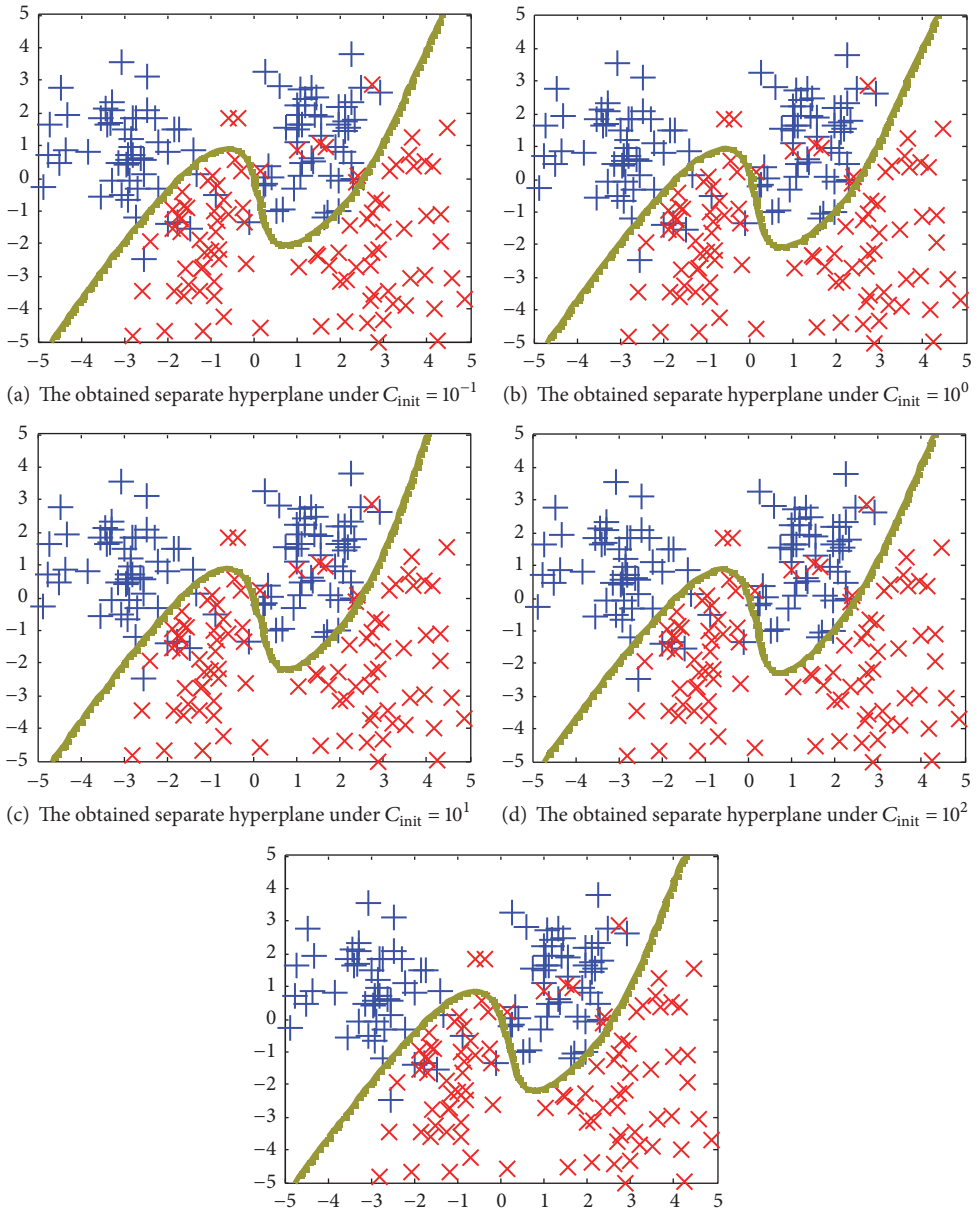
(e) The obtained separate hyperplane under $C_{\text{svm}} = 10^{3}$

FIGURE 3: The experimental results of LMKL on the synthetic dataset.

impact severely the result of LMKL. Obviously, the experimental result illustrated in Figure 3(c) is better. Here, the regularization parameter C is set as $C = 10$. The experimental results of the proposed method are illustrated in Figure 4. It can be found that we almost obtain the same result under different initial value C_{init} of C. Moreover, the final learned C is sometimes smaller and sometimes larger than the initial value. Note that the final values of learned C are different under the different initial value C_{init} of C. The reason, as pointed out in Section 4.1, is that the gradient-descent method is employed to optimize the regularization parameter C. However, the final learned regularization parameters are close to each other. The experiments further validate the fact that our method can effectively automatically learn the regularization parameter C. This is an advantage over traditional LMKL.

4.3. Experimental Results on UCI Datasets. In this subsection, we report the performance comparison about SimpleMKL [2], LMKL [15], and the proposed method on several UCI datasets [32].

In the experiment, we use 50% of each dataset as a training set and the rest as the test set. As in Section 4.1, the data were normalized (i.e., 0 mean and 1 standard deviation).



(a) The obtained separate hyperplane under $C_{\text{init}} = 10^{-1}$

(b) The obtained separate hyperplane under $C_{\text{init}} = 10^{0}$

(c) The obtained separate hyperplane under $C_{\text{init}} = 10^{1}$

(d) The obtained separate hyperplane under $C_{\text{init}} = 10^{2}$

(e) The obtained separate hyperplane under $C_{\text{init}} = 10^{3}$

FIGURE 4: The experimental results of ILMKL on the synthetic dataset.

The base kernels include seven Gaussian kernels with the widths of $[3^{-3}, 3^{-2}, 3^{-1}, 3^{0}, 3^{1}, 3^{2}, 3^{3}]$ and four polynomial kernels with degrees of one to four. Before training, all kernel matrices are calculated in advance and preprocessed to unit trace. Each experiment is repeated 50 times, and the mean accuracy and standard deviation were computed. In the experiments, SimpleMKL and LMKL employ the cross-validation technique to choose the regularization parameter C from the set $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}\}$. For our method ILMKL, it is not necessary to use cross-validation to select the parameter C because it can learn an appropriate value.

Table 1 reports the classification accuracies of several SVM-based MKL methods on the selected datasets. From Table 1, it can be found that LMKL has comparable performance to SimpleMKL. However, on the whole, it can be found that ILMKL has a clear improvement in the classification performance in contrast with SimpleMKL and LMKL. These experimental results indicate that the generalized performance in SVM-based MKL can be improved when the information of radius of MEB is considered. The proposed method ILMKL embodies the idea.

For a rigorous comparison, simultaneously, we further conducted the paired two-tailed t-tests [33] on these methods. In t-test, the p value depicts the probability that two sets generate from distributions with equal means. If the p value is smaller, then the difference of the two mean values is more significant. Generally, 0.05 is viewed as a typical

Table 1: Classification accuracy (mean ± standard derivation) on the selected datasets.

Dataset	SimpleMKL	LMKL	ILMKL
Ionosphere	92.07 ± 2.02	91.64 ± 2.77	**93.89** ± 1.21
Heart	84.01 ± 1.38	83.92 ± 1.65	**85.91** ± 2.84
Wdbc	96.19 ± 0.44	95.54 ± 2.35	**96.97** ± 1.96
Wpbc	76.59 ± 1.02	77.40 ± 2.21	**78.42** ± 1.14
Liver	64.45 ± 3.64	62.13 ± 2.93	**64.87** ± 3.46
Musk1	89.79 ± 2.02	86.44 ± 3.37	**90.27** ± 4.68
Sonar	**83.42** ± 3.30	79.95 ± 3.15	82.98 ± 5.02
Fourclass	99.95 ± 0.06	99.93 ± 0.14	**99.97** ± 0.09
Coloncancer	74.98 ± 8.18	74.83 ± 3.93	**76.11** ± 1.16
Germannumber	72.74 ± 1.55	73.32 ± 1.78	**75.24** ± 1.15
Pima	74.37 ± 2.05	75.23 ± 3.12	**77.01** ± 1.52

Table 2: p value of t-test on the selected datasets.

Dataset	LMKL/SimpleMKL	ILMKL/SimpleMKL	ILMKL/LMKL
Ionosphere	**0.0862**	0.0296	0.0451
Heart	**0.0639**	0.0371	0.0339
Wdbc	0.0418	**0.0641**	0.0207
Wpbc	0.0325	0.0153	0.4759
Liver	0.0393	0.0426	**0.0931**
Musk1	0.0244	**0.0915**	0.0185
Sonar	0.0167	**0.0733**	0.0239
Fourclass	**0.8514**	0.0435	**0.1384**
Coloncancer	0.7426	0.0158	0.0256
Germannumber	0.0462	0.0192	0.0374
Pima	0.0437	0.0382	0.0427

Table 3: Rate of support vectors comparison on the selected datasets.

Dataset	SimpleMKL	LMKL	ILMKL
Ionosphere	0.6982 ± 0.0926	0.4862 ± 0.0504	0.4565 ± 0.0674
Heart	0.7681 ± 0.0671	0.5948 ± 0.1486	0.4333 ± 0.1102
Wdbc	0.3640 ± 0.0213	0.2191 ± 0.0182	0.2010 ± 0.0218
Wpbc	0.9948 ± 0.0153	0.7928 ± 0.0952	0.7857 ± 0.0579
Liver	0.9982 ± 0.0026	0.9203 ± 0.0550	0.9302 ± 0.0111
Musk1	0.9316 ± 0.0309	0.6793 ± 0.0544	0.6185 ± 0.1155
Sonar	0.9126 ± 0.0330	0.6456 ± 0.0367	0.5398 ± 0.1079
Fourclass	0.3790 ± 0.0177	0.1804 ± 0.0514	0.1388 ± 0.0840
Coloncancer	1.0000 ± 0.0000	0.9274 ± 0.0265	0.9258 ± 0.0816
Germannumber	0.8972 ± 0.0373	0.7006 ± 0.0356	0.7378 ± 0.0407
Pima	0.7937 ± 0.0872	0.5893 ± 0.0876	0.5759 ± 0.0209

threshold of the p value; that is, it is considered statistically significant when the p value is smaller than 0.05. Table 2 reports the experimental results of the t-tests. For example, the p value of the t-test when comparing LMKL and SimpleMKL on the Ionosphere dataset is 0.0862 (>0.05), meaning that SimpleMKL does not perform significantly better than LMKL on this dataset at the 0.05 significant level though SimpleMKL has better classification accuracy according to Table 1. However, ILMKL performs significantly better than SimpleMKL since the p value of the t-test is 0.0296 (<0.05) on this dataset at the 0.05 significant level. From Table 2, ILMKL has on the whole significant improvement in the generalized performance in contrast with SimpleMKL and LMKL.

Finally, we investigated the support vector percentages of several methods on the selected datasets, which are reported in Table 3. Generally, fewer support vectors mean less test time. From Table 3, LMKL tends to have more support vectors in contrast with SimpleMKL. The proposed method

ILMKL has on the whole similar support vector percentages to LMKL. So, our method inherits the advantage of LMKL that it stores fewer support vectors but can achieve statistically similar accuracy results compared with other MKL methods.

5. Conclusions

In this paper, by following the work in [27], we presented a novel LMKL method. Different from traditional LMKL, our method takes into consideration the information of both the radius and the margin when learning the parameters of the gating model. As a result, our method can achieve better accuracy. Simultaneously, our method can automatically tune the regularization parameter C during the process of learning the parameters of the gating model. Therefore, this can improve the computational efficiency of our method by avoiding using the time-consuming cross-validation technique to find a suitable regularization parameter. Comprehensive experiments are conducted on several toy and benchmark datasets and the results well demonstrate the efficiency and effectiveness of the proposed method.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

Acknowledgments

This work is supported in part by the Scientific Research Project "Chunhui Plan" of the Ministry of Education of China (Grant no. Z2015102), the Key Scientific Research Foundation of Sichuan Provincial Department of Education (no. 11ZA004), and the National Natural Science Foundation of China (Grants nos. 61103168, 61532009, and 61602390).

References

[1] B. Schölkopf and A. Smola, *Learning with Kernels*, MIT Press, Cambridge, Mass, USA, 2002.

[2] A. Rakotomamonjy, F. Bach, Y. Grandvalet, and S. Canu, "SimpleMKL," *Journal of Machine Learning Research*, vol. 9, pp. 2491–2521, 2008.

[3] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research*, vol. 5, pp. 27–72, 2004.

[4] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proceedings of the 21st International Conference on Machine Learning*, pp. 41–48, Banff, Canada, July 2004.

[5] S. Sonnenburg, G. Ratsch, and C. Schafer, "A general and efficient multiple kernel learning algorithm," in *Advances in Neural Information Processing Systems*, pp. 1273–1280, 2005.

[6] S. Sonnenburg, G. Ratsch, C. Schafer, and B. Scholkopf, "Large scale multiple kernel learning," *Journal of Machine Learning Research*, vol. 7, pp. 1531–1565, 2006.

[7] F. R. Bach, "Consistency of the group lasso and multiple kernel learning," *Journal of Machine Learning Research*, vol. 9, pp. 1179–1225, 2008.

[8] M. Gönen and E. Alpaydin, "Cost-conscious multiple kernel learning," *Pattern Recognition Letters*, vol. 31, no. 9, pp. 959–965, 2010.

[9] M. Gönen and E. Alpaydn, "Regularizing multiple kernel learning using response surface methodology," *Pattern Recognition*, vol. 44, no. 1, pp. 159–171, 2011.

[10] A. Nazarpour and P. Adibi, "Two-stage multiple kernel learning for supervised dimensionality reduction," *Pattern Recognition*, vol. 48, no. 5, pp. 1854–1862, 2015.

[11] Z. Wang, Q. Fan, S. Ke, and D. Gao, "Structural multiple empirical kernel learning," *Information Sciences*, vol. 301, no. 20, pp. 124–140, 2015.

[12] W. Luo, J. Yang, W. Xu, J. Li, and J. Zhang, "Higher-level feature combination via multiple kernel learning for image classification," *Neurocomputing*, vol. 167, no. 1, pp. 209–217, 2015.

[13] X. Zhang and M. H. Mahoor, "Task-dependent multi-task multiple kernel learning for facial action unit detection," *Pattern Recognition*, vol. 51, no. 3, pp. 187–196, 2016.

[14] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.

[15] M. Gönen and E. Alpaydin, "Localized multiple kernel learning," in *Proceedings of the International Conference on Machine Learning*, pp. 352–359, 2008.

[16] M. Gönen and E. Alpaydin, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.

[17] M. Gönen and E. Alpaydın, "Localized algorithms for multiple kernel learning," *Pattern Recognition*, vol. 46, no. 3, pp. 795–807, 2013.

[18] M. Hu, Y. Chen, and J. T.-Y. Kwok, "Building sparse multiple-kernel SVM classifiers," *IEEE Transactions on Neural Networks*, vol. 20, no. 5, pp. 827–839, 2009.

[19] X. Xu, I. W. Tsang, and D. Xu, "Soft margin multiple kernel learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 5, pp. 749–761, 2013.

[20] Y. Song, Y.-T. Zheng, S. Tang et al., "Localized multiple kernel learning for realistic human action recognition in videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 9, pp. 1193–1202, 2011.

[21] Y. Han and G. Liu, "Probability-confidence-kernel-based localized multiple kernel learning with l_p norm," *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, vol. 42, no. 3, pp. 827–837, 2012.

[22] Y. Han, K. Yang, Y. Ma, and G. Liu, "Localized multiple kernel learning via sample-wise alternating optimization," *IEEE Transactions on Cybernetics*, vol. 44, no. 1, pp. 137–148, 2014.

[23] C. Jose, P. Goyal, P. Aggrwal, and M. Varma, "Local deep kernel learning for efficient non-linear SVM prediction," in *Proceedings of the 30th International Conference on Machine Learning (ICML '13)*, pp. 486–494, Atlanta, Ga, USA, June 2013.

[24] V. Vapnik and O. Chapelle, "Bounds on error expectation for support vector machines," *Neural Computation*, vol. 12, no. 9, pp. 2013–2036, 2000.

[25] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1–3, pp. 131–159, 2002.

[26] S. S. Keerthi, "Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1225–1229, 2002.

[27] K.-M. Chung, W.-C. Kao, C.-L. Sun, L.-L. Wang, and C.-J. Lin, "Radius margin bounds for support vector machines with the RBF kernel," *Neural Computation*, vol. 15, no. 11, pp. 2643–2681, 2003.

[28] K. Duan, S. S. Keerthi, and A. N. Poo, "Evaluation of simple performance measures for tuning SVM hyperparameters," *Neurocomputing*, vol. 51, pp. 41–59, 2003.

[29] H. Do, A. Kalousis, A. Woznica, and M. Hilario, "Margin and radius based multiple kernel learning," in *Proceedings of the European Conference on Machine Learning (ECML '09)*, pp. 330–343, 2009.

[30] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1987.

[31] D. M. J. Tax and R. P. W. Duin, "Support vector domain description," *Pattern Recognition Letters*, vol. 20, no. 11–13, pp. 1191–1199, 1999.

[32] M. Lichman, *UCI Machine Learning Repository*, University of California, School of Information and Computer Science, Irvine, Calif, USA, 2007, http://archive.ics.uci.edu/ml/.

[33] E. Alpaydin, *Introduction to Machine Learning*, The MIT Press, Cambridge, Mass, USA, 2004.