Incorporation of radius-info can be simple with SimpleMKL

Xinwang Liu a,b,*, Lei Wang b, Jianping Yin a, Lingqiao Liu c

a School of Computer Science, National University of Defense Technology, Changsha, Hunan 410073, China
b School of Computer Science and Software Engineering, University of Wollongong, NSW 2522, Australia
c Research School of Information Science and Engineering (RSISE), The Australian National University, ACT 0200, Australia

ARTICLE INFO

Article history:
Received 27 April 2011
Accepted 3 January 2012
Communicated by J.S. Shawe-Taylor
Available online 19 March 2012

Keywords:
Multiple kernel learning
Minimal enclosing ball
Kernel methods
Support vector machines
Radius margin bound

ABSTRACT

Recent research has shown the benefit of incorporating the radius of the Minimal Enclosing Ball (MEB) of training data into Multiple Kernel Learning (MKL). However, straightforwardly incorporating this radius leads to complex learning structure and considerably increased computation. Moreover, the notorious sensitivity of this radius to outliers can adversely affect MKL. In this paper, instead of directly incorporating the radius of MEB, we incorporate its close relative, the trace of data scattering matrix, to avoid the above problems. By analyzing the characteristics of the resulting optimization, we show that the benefit of incorporating the radius of MEB can be fully retained. More importantly, our algorithm can be effortlessly realized within the existing MKL framework such as SimpleMKL. The mere difference is the way to normalize the basic kernels. Although this kernel normalization is not our invention, our theoretic derivation uncovers why this normalization can achieve better classification performance, which has not appeared in the literature before. As experimentally demonstrated, our method achieves the overall best learning performance in various settings. In another perspective, our work improves SimpleMKL to utilize the information of the radius of MEB in an efficient and practical way.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Efficiently learning an optimal kernel for a given task plays a pivotal role in kernel-based learning approaches [12,10]. The past few years have witnessed Multiple Kernel Learning (MKL) as an appealing way to achieve this goal. By representing a kernel with a linear [1,10] or nonlinear [2,7] combination of a set of basic kernels, MKL jointly learns the weights of basic kernels and the structural parameters of SVMs. In the seminal work [10], MKL is formulated as a Semi-Definite Programming (SDP) problem. Later on, more efficient algorithms, such as Semi-Infinity Linear Programming (SILP) [14], Subgradient Descent (SD) method [13], level method [18] and a closed form solution [19,8], have been proposed, making MKL a powerful tool for practical applications. Following the large-margin principle, most of the above MKL algorithms optimize the basic kernels’ weights by maximizing the margin only.

Recent research shows that it is beneficial to consider the radius of the Minimal Enclosing Ball (MEB) of training data in a kernel-induced feature space. As indicated in Gai et al. [6], this radius is not only an integral part of the generalization error bound of SVMs, but also a critical quantity to handle the scaling issue in existing MKL framework. Methods have recently been proposed to incorporate this radius into MKL. In [5], the authors minimize the ratio of the radius to the margin and approximate it by a convex optimization problem. However, their formulation cannot solve the scaling and initialization issues. In [6], a tri-level optimization method is developed to incorporate this radius, adding an extra level of quadratic optimization on top of existing MKL framework. A rigorous and elegant theoretical analysis is conducted for the resulting multilevel optimization. However, their method has the following shortcomings: (i) more complex learning structure than existing MKL; (ii) considerably increased computation; and (iii) the learning performance of MKL can be hurt by the notorious sensitivity of the radius of MEB to outliers. Altogether, these shortcomings could significantly affect the efficiency and robustness of existing MKL framework.

The above shortcomings are mainly due to a direct incorporation of the radius of MEB as it is. In this paper, instead of incorporating this radius, we incorporate the trace of data scattering matrix by showing its close relationship to the radius of MEB. In doing so, our method avoids the extra level of quadratic optimization needed to compute the radius value and improves the robustness to outliers. We analyze the characteristics of the resulting optimization to display that the scaling issue can be well addressed. More importantly, we theoretically show that the optimization problem in our method can be written...
in the common form taken by existing MKL algorithms. Hence, our algorithm can be realized effortlessly with existing packages, such as the widely used SimpleMKL. The mere (but critical) difference is the way to normalize basic kernels. In our algorithm, each basis kernel is normalized by the trace of data scattering matrix in the feature space induced by it. We show that the common way of normalizing each basis kernel with the trace of the kernel matrix is a special case of our normalization. Geometrically, the common way of normalizing makes existing MKL algorithms implicitly consider the scattering radius of data with respect to the centroid of the data cloud, which is much more efficient in approximating the radius of MEB. In this sense, existing MKL algorithms like SimpleMKL have considered the radius of MEB but in a relatively crude form. On the other hand, our analysis also explains why this normalization, which has been used in recent works [3,9], can help improving the classification performance, which does not appear in literature. As demonstrated by experimental study, our method achieves better learning performance than existing SimpleMKL and the method proposed in Gai et al. [6], especially when outlier exists. In another perspective, our work improves SimpleMKL and the alike to utilize the information of the radius of MEB and adequately preserves its efficiency for practical applications.

The rest of this paper is organized as follows. We review the related work in Section 2. Then, we firstly give the formulation of MKL with \( \text{tr}S_r \), and prove it can arrive in the same form of SimpleMKL and give an efficient algorithm. Section 4 shows our experimental results and the last section is conclusion.

### 2. Related work

Let \( X = (x_1, \ldots, x_n) \in \mathbb{R}^{n \times d} \) be a set of training samples, where \( n \) and \( d \) are the number of training samples and the dimensionality of samples, respectively. Let \( y = (y_1, \ldots, y_n) \in \{ \pm 1 \} \) be the class label of \( x_i \). Let \( \phi_p : X \rightarrow H_p \) be the \( p \)-th feature mapping, inducing a kernel \( k_p \) in the Hilbert space \( H_p \), where \( p = 1, \ldots, m \). \( K_p \) is the kernel matrix computed with \( k_p \) on the training set \( X \). In MKL framework, each sample \( x \) is mapped to \( m \) feature spaces by \( \phi(x; \gamma) = [\sqrt{\gamma_1} \phi_1(x), \ldots, \sqrt{\gamma_m} \phi_m(x)]^T \), where \( \gamma_i \) is the weight of the \( p \)-th basic kernel. Correspondingly, there is \( k(\gamma) = \sum_{p=1}^m \gamma_p k_p \) and \( K(\gamma) = \sum_{p=1}^m \gamma_p K_p \). Most existing MKL algorithms [14,13,18] find the optimal basic kernel weights by maximizing the margin,

\[
\min_{\gamma, \alpha, b, \xi} \frac{1}{2} \| \omega \|^2 + \frac{C}{n} \sum_{i=1}^n \xi_i \\
\text{s.t.} \quad y_i (\omega^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i,
\]

where \( \omega \) is the normal of the separating hyperplane, \( b \) the bias, and \( \xi = [\xi_1, \ldots, \xi_n]^T \) the slack variables.

It is known in the literature [12] that the generalization error bound of SVMs depends not only on the margin but also on the radius of the MEB of training data. Also, the work in Gai et al. [6] is proposed to incorporate the radius of MEB to solve the scaling problem in the existing MKL algorithms. As pointed out in Gai et al. [6], a larger margin can be arbitrarily obtained by scaling \( \gamma \) to \( \gamma(\tau > 1) \), which will affect the convergence of optimization. Although putting an extra norm-constraint on \( \gamma \) can remove this issue, how to identify a suitable norm-constraint for a given kernel learning task is a problem itself in the literature. Moreover, even if a norm-constraint is applied, a better kernel could be misjudged as a worse one by a simple down-scaling, as detailed in Gai et al. [6]. These motivate the incorporation of radius information into MKL.

Optimizing both radius and margin has recently been addressed in Do et al. [5], Gai et al. [6]. Both take the following formulation,

\[
\min_{\gamma, \alpha, b, \xi} \frac{1}{2} \| \omega \|^2 + R^2 + C \sum_{i=1}^n \xi_i \\
\text{s.t.} \quad y_i (\omega^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i,
\]

where \( R^2 \) is the radius of the MEB and \( q = 1 \) or 2. Note that \( R^2 \) is a function of \( \gamma \). The work in Do et al. [5] focuses on approximating the above optimization with a convex one instead of solving the scaling problem. Comparatively, the work in Gai et al. [6] is closer to ours. It directly solves the optimization in (2) and removes the scaling issue. To achieve this, they reformulate (2) to a tri-level optimization problem:

\[
\min_{\gamma, b} \frac{1}{2} \| \omega \|^2 + R^2 + C \sum_{i=1}^n \xi_i \\
\text{s.t.} \quad y_i (\omega^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i,
\]

where

\[
J(\gamma) = \left\{ \max_x \frac{1}{2R^2} (x \cdot y)^T K(\gamma)(x \cdot y) \right\} \\
\text{s.t.} \quad x^T y = 0, \quad 0 \leq x \leq C, \quad \forall i
\]

and

\[
R^2 = \left\{ \max_{\beta} \text{diag}(K(\gamma)) - \beta^T K(\gamma) \beta \right\} \\
\text{s.t.} \quad \beta^T \mathbf{1} = 1, \quad 0 \leq \beta, \quad \forall i
\]

After proving the differentiability of \( J(\gamma) \), they solve the optimization problem (3) as follows: given a \( \gamma \), \( R^2 \) is computed by solving the Quadratic Programming (QP) in (5). Take \( R^2 \) into (4) and solve another QP to obtain \( J(\gamma) \). Then, update the kernel weight \( \gamma \) by gradient-based optimization algorithms. The above procedure is repeated until convergence. Because an extra QP needs to be solved in each iteration, their method can considerably increase the computation of MKL, especially when the number of training samples is large. Moreover, since the optimization in (5) is very sensitive to outliers, the obtained \( R^2 \) can be noisy and the noise will in turn affect the optimization of kernel weights via the tri-level optimization structure.

A straightforward remedy is to introduce slack variables to (5) to relax the MEB formulation as

\[
R^2 = \left\{ \max_{\beta} \text{diag}(K(\gamma)) - \beta^T K(\gamma) \beta \right\} \\
\text{s.t.} \quad \beta^T \mathbf{1} = 1, \quad 0 \leq \beta, \quad \forall i
\]

where \( D \) is a regularization parameter.

The formulation (6) would be robust to noise. However, using it not only leads to the tri-level learning structure in Gai et al. [6], but also introduces another hyper-parameter, \( D \), to tune. This is not in our goal—incorporating radius information into existing MKL framework without incurring extra computation. In this paper, we address these issues to maintain the computational efficiency and the robustness as of the existing MKL framework.

### 3. Proposed method

#### 3.1. Relationship between \( \text{tr}(S_r) \) and \( R^2 \)

Let \( m \) be the mean of all samples in the primal space. The total scatter matrix is defined [16] as \( S_r = \sum_{i=1}^n (x_i - m)(x_i - m)^T \). It is straightforward to derive its trace in a kernel-induced feature
space as
\[
\text{tr}(\mathbf{S}_T) = \text{tr}(\mathbf{K}(\gamma))-\frac{1}{n} \mathbf{1}^T \mathbf{K}(\gamma) \mathbf{1} = \sum_{p=1}^{m} \gamma_p a_p,
\] (7)
where \( a_p = \text{tr}(\mathbf{K}_p) / (-1/n) \mathbf{1}^T \mathbf{K}_p \mathbf{1} \). Note that \( \text{tr}(\mathbf{S}_T) \) is a linear function of \( \gamma \).

There is a close relationship between \( \text{tr}(\mathbf{S}_T) \) and \( R^2 \), as shown in Wang [17]. For the problem in (5), if we simply set its solution as \( \beta = 1/n \), it can be obtained that
\[
\frac{1}{n} \text{tr}(\mathbf{S}_T) = \beta^T \text{diag}(\mathbf{K}(\gamma)) - \beta^T \mathbf{K}(\gamma) \beta.
\] (8)
Note that \( \beta \) satisfies all the constraints in (5). Hence, \( \text{tr}(\mathbf{S}_T)/n \) can be seen as an approximation of \( R^2 \). Moreover, all training samples are treated equally (assigned the same \( \beta_i = 1/n \)) in computing \( \text{tr}(\mathbf{S}_T) \), rather than being distinguished with different \( \beta_i \) by optimizing (5) to compute \( R^2 \). When outliers exist, considering all samples equally makes \( \text{tr}(\mathbf{S}_T) \) more robust than \( R^2 \). Similar mechanism has been observed in Kernel FDA, which often achieves higher classification performance than SVMs when data contains noise.

To better illustrate the relationship, we plot \( \text{tr}(\mathbf{S}_T)/n \) and \( R^2 \) when different kernels are employed. Fig. 1(a) corresponds to a polynomial kernel with different degrees, while Fig. 1(b) is for a Gaussian kernel with different widths. The toy data set in our experiment is used, and both cases of with and without outlier are considered. We can see that: (i) \( \text{tr}(\mathbf{S}_T) \) and \( R^2 \) change in a similar way with respect to different kernels; (ii) when an outlier is added, the variation of \( R^2 \) is much larger than that of \( \text{tr}(\mathbf{S}_T) \), which indicates that \( R^2 \) is more sensitive to outliers. In the following, by utilizing the close relationship, we substitute \( R^2 \) with \( \text{tr}(\mathbf{S}_T) \) to incorporate the radius information.

Remark 1. By uniformly weighting each sample, the \( \text{tr}(\mathbf{S}_T)/n \) is more robust than the radius of the MEB in characterizing the scattering of training points. In computing the radius of MEB, a point further away from the center of data is often assigned larger weight, making the radius sensitive to sampling noise. In contrast, by uniformly weighting points, \( \text{tr}(\mathbf{S}_T)/n \) computes an average radius of data scattering, which is more stable with respect to sampling noise. This has been used to explain the advantage of Kernel Fisher Discriminant Analysis (KFDA) [11] over SVMs in some situations (see [11] for details).

3.2. Incorporating \( \text{tr}(\mathbf{S}_T) \) into MKL

By substituting \( R^2 \) with \( \text{tr}(\mathbf{S}_T) \) in (2), we obtain our \( \text{tr}(\mathbf{S}_T) \)-margin as follows:
\[
\min J_1(\gamma; \mathbf{k}) \quad \text{s.t.} \quad \gamma_p \geq 0, \quad \forall p,
\] (9)
where
\[
J_1(\gamma; \mathbf{k}) = \left\{ \min_{\omega, b, \xi} \frac{1}{2} \text{tr}(\mathbf{S}_T)(\|\omega\|^2 + C \sum_{i=1}^{n} \xi_i) \right\},
\] (10)
where \( J_1(\gamma; \mathbf{k}) \) indicates that we search for a linear combination on the basic kernel set \( \mathbf{k} \).

Remark 2. Following Remark 1, using the “\( \text{tr}(\mathbf{S}_T) \)-margin” rather than fully reflecting the “\( R^2 \)-margin” can give a more robust criterion for kernel parameter learning. Note that our formulation (9), i.e. “\( \text{tr}(\mathbf{S}_T) \)-margin”, may NOT be a generalization bound anymore. However, note that a good kernel parameter learning criterion does not need to be a bound of generalization error. The more critical point is that if the criterion correlates well with the generalization performance. Our experiments demonstrate the stronger correlation of “\( \text{tr}(\mathbf{S}_T) \)-margin” with the generalization performance.

We prove the following Propositions 1–3 to show that our formulation in (9) is free of the scaling problem and able to maintain the same SVM decision function no matter how the basic kernels are scaled.

Let \( \mathbf{k} = (k_1, k_2, \ldots, k_m) \) be a group of basic kernels. Also, we define another two groups of basic kernels \( \mathbf{k}_1 = (\tau k_1, \tau k_2, \ldots, \tau k_m) \) and \( \mathbf{k}_2 = (d_1 k_1, d_2 k_2, \ldots, d_m k_m) \) by scaling \( \mathbf{k} \), where \( \tau \) and \( d_1, \ldots, d_m \) are any positive scalars.

Proposition 1. \( J_1(\gamma; \mathbf{k}_1) = J_1(\gamma; \mathbf{k}_1) \).

Proof. Due to \( \mathbf{k}_1 = \tau \mathbf{k} \), the mapping functions corresponding to \( \mathbf{k}_1 \) and \( \mathbf{k} \) satisfy \( \phi_1(\mathbf{x}; \gamma) = \sqrt{\tau} \phi(\mathbf{x}; \gamma) \).
\[
J_1(\gamma; \mathbf{k}_1) = \left\{ \min_{\omega, b, \xi} \frac{1}{2} \text{tr}(\mathbf{S}_T)(\|\omega\|^2 + C \sum_{i=1}^{n} \xi_i) \right\},
\] (11)
where \( \mathbf{S}_T = \sqrt{\tau} \mathbf{S} \), the (11) can be rewritten as:
\[
J_1(\gamma; \mathbf{k}_1) = \left\{ \min_{\omega, b, \xi} \frac{1}{2} \text{tr}(\mathbf{S})(\|\omega\|^2 + C \sum_{i=1}^{n} \xi_i) \right\}.
\]

Fig. 1. The close relationship between \( \text{tr}(\mathbf{S}_T) \) and \( R^2 \), the square radius of MEB (the value is shown in logarithm). (a) Polynomial kernel. (b) Gaussian kernel.
Proof. For conclusion (i), since $y_1$ is an optimal solution of (P1), there exists $\delta > 0$ such that for any $\gamma_1(\gamma \geq 0)$ that satisfies $|\gamma - y_1| / \gamma_1 \leq \delta$, we have $J_1(y, \gamma_1) \leq J_1(y, \gamma)$. Then, for any $\gamma_1(\gamma > 0)$ that satisfies $|\gamma_1 - y_i| / \gamma_1 \leq \delta$, we have $J_1(y, \gamma) \geq J_1(y, \gamma_1)$. Hence, we obtain the following conclusion: for any $\gamma_1(\gamma > 0)$ that satisfies $|\gamma_1 - y_i| / \gamma_1 \leq \delta$, we have $J_1(y, \gamma) \geq J_1(y, \gamma_1)$. Due to also $\gamma_1 / \gamma_1 \leq \delta$, the constraint of (P2), $\gamma_1 / \gamma_1 \leq \delta$, is an optimal solution of (P2).

For conclusion (ii), as $y_2$ is an optimal solution of (P2), there exists $\delta > 0$ that for any $\gamma_2(\gamma > 0)$, $1)$ that satisfies $|\gamma_2 - y_2| / \gamma_2 \leq \delta$, we have $J_1(y_2, \gamma_2) \leq J_1(y_2, \gamma)$. For any $\gamma_2(\gamma > 0)$, $1)$ that satisfies $|\gamma_2 - y_2| / \gamma_2 \leq \delta$, we have $J_1(y_2, \gamma) \geq J_1(y_2, \gamma_2)$. Due to also $\gamma_2 / \gamma_2 \leq \delta$, the constraint of (P1), $\gamma_2 / \gamma_2 \leq \delta$, is an optimal solution of (P1).

This proposition shows that our formulation (9) is invariant to norm constraints. This leads us to Proposition 3.

Proposition 3. Let (P1): $\min_{y_1, y_2} J_1(y, \gamma_1, \gamma_2)$ s.t. $y_1 \geq 0, y_2 \geq 0$ and $\|\gamma_1\|_1 = 1$. Then (i) Problems (P1) and (P2) have the same optimal function value. (ii) For any optimal solution of (P2), denoted by $\gamma_2$, $d \otimes \gamma_2 / d \otimes \gamma_2$, is also the optimal solution of (P1), where $d = \{d_1, \ldots, d_m\}$ and $\otimes$ denotes componentwise product.

Proof. By Proposition 2, the $y_2$ is equivalent to the one without any norm constraint: $\min_{y_1, y_2} J_1(y, \gamma_1, \gamma_2)$ s.t. $y_1 \geq 0, y_2 \geq 0$, and $\|\gamma_1\|_1 = 1$. Then, the problems (P3) and (P4) have one-to-one correspondence due to the linear transform $\gamma = d \otimes \gamma$. Then, by Proposition 2, problem (P4) is equivalent to the one with a norm constraint, which is indeed the problem (P1). For conclusion (ii), suppose $y_2$ is an optimal solution of (P2), $y_2$ is also the optimal solution of (P3) by Proposition 2. Due to the one-to-one correspondence between the solution of (P3) and (P4), $d \otimes y_2$ is the optimal solution of (P4). By Proposition 2, $d \otimes y_2 / d \otimes y_2$, is also the optimal solution of (P1). It completes the proof.

Proposition 3 indicates that with any initial scaling of the basic kernels, our formulation (9) can still end up the same kernel combinations up to the scaling $d \otimes \gamma_2$. In view of Proposition 1, we know that the SVM decision function will keep unchanged.

Based on the above three propositions, we display that our formulation (9) will achieve the same optimal decision function no matter how the basic kernels are scaled. This indicates that our formulation can solve the scaling problem pointed out in Gai et al. [6].

3.3. Solving our optimization problem

A straightforward way to solve our optimization problem (9) is to follow the method in Gai et al. [6]. Nevertheless, we show that our problem can be converted to the commonly used form of MKL and can be solved effortlessly with existing packages such as SimpleMKL.

Theorem 1. The optimal solution of optimization problem (9), denoted as $y^*$, can be denoted as $y^* = \text{tr}(S)\eta^*$, where $\eta^*$ is the optimal solution of the following optimization problem (13):

$$\begin{align*}
\min_{\eta} & \quad J_1(\eta, \gamma) = \left\{ \begin{array}{ll}
\min_{\eta} & \quad \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} \xi_i \\
t & \quad y_i(\mathcal{D}^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \forall i,
\end{array} \right. \\
\text{s.t.} & \quad \sum_{\eta_p=1}^{m} a_{\eta_p} \eta_p = 1, \quad \eta_p \geq 0, \forall \eta_p,
\end{align*}$$

where $J_1(\eta, \gamma) = \left\{ \begin{array}{ll}
\min_{\eta} & \quad \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} \xi_i \\
t & \quad y_i(\mathcal{D}^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \forall i,
\end{array} \right.$

Proof. Let $\eta_p = \sqrt{\text{tr}(S)}\eta_p$, the (10) is transformed as (15).

$$\begin{align*}
\min_{\eta} & \quad J_1(\eta) = \left\{ \begin{array}{ll}
\min_{\eta} & \quad \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} \xi_i \\
t & \quad y_i(\mathcal{D}^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \forall i,
\end{array} \right. \\
\text{s.t.} & \quad \sum_{\eta_p=1}^{m} a_{\eta_p} \eta_p = 1, \quad \eta_p \geq 0, \forall \eta_p.
\end{align*}$$

which is exactly the optimization problem (13). After obtaining the optimal solution of (13), denoted as $\eta^*$, we can get the optimal solution of (9) by $y^* = \text{tr}(S)\eta^*$. By Theorem 1, our formulation (9) can be converted to (13). It has a similar form of SimpleMKL [13], with the only difference of using a weighted equality constraint. The following Corollary 1 further converts the problem (13) to a standard SimpleMKL problem.

Corollary 1. Let $k = [k_1, k_2, \ldots, k_m]$. If $\eta^*$ is an optimal solution of the following problem:

$$\begin{align*}
\min_{\eta} & \quad J_1(\eta, k) = \left\{ \begin{array}{ll}
\min_{\eta} & \quad \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{n} \xi_i \\
t & \quad y_i(\mathcal{D}^T \phi(x_i; \gamma) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \forall i,
\end{array} \right. \\
\text{s.t.} & \quad \sum_{\eta_p=1}^{m} a_{\eta_p} \eta_p = 1, \quad \eta_p \geq 0, \forall \eta_p,
\end{align*}$$

Then $\eta^* \otimes z/ \mathcal{D} \eta^* / \mathcal{D} z$ is an optimal solution of optimization problem (13), where $z = [z_1, \ldots, z_m]^T$.

Proof. For optimization problem (13), if we let $\eta = a \otimes \eta$, there is $\eta^* = \eta / a$. Then, the optimization problem (13) is transformed as $\min_{\eta} J_1(\eta/a)$. s.t. $\eta_p = 1, \eta_p \geq 0, \forall \eta_p$. By Proposition 3,
if \( \eta^* \) is an optimal solution of (17), \( \eta^* \odot 1/\| \eta^* \odot 1 \| \) is an optimal solution of optimization problem (13). □

By Corollary 1, we can solve the problem (13) by using the optimal solution of SimpleMKL. The only thing that we need to modify is to normalize each basic kernel matrix with its tr(\( K_p \)), i.e., \( a_p^{-1} \). After obtaining the optimal solution of SimpleMKL, we have been able to obtain the optimal SVM decision function. Hence, if the purpose is just classification, we do not even need to retrieve the solutions to problem (13) and (9), although this can be done via a simple scaling. Our algorithm is summarized in Algorithm 1.

Algorithm 1. Our proposed MKL with tr(S_p).

- **Input**: a group of basic kernel \( \{ K_p \}_{p=1}^m \).
- **Initialize**: \( \eta_0^p = 1/m \).
- **Normalize**: basic kernel as \( K_p^* = a_p^{-1} K_p, \forall p \), where \( a_p = \text{tr}(K_p)^{-1} \frac{1}{n} K_p 1 \).
- **Run SimpleMKL** on \( \{ K_p^* \}_{p=1}^m \) and obtain its optimal solution \( \eta^* \) and \( z^* \).
- **Obtain the optimal decision function** by \( \eta^* \) and \( z^* \).

3.4. Revisiting SimpleMKL in our framework

Recalling (7), we know tr(\( K(\gamma) \)) is just the first term of tr(\( S_p \)). If we substitute tr(\( S_p \)) in our problem (9) with tr(\( K \)), we obtain a variant of our algorithm:

\[
\min_{\gamma} J_1(\gamma; K) \quad \text{s.t.} \quad g \geq 0, \quad g p,
\]

where

\[
J_1(\gamma; K) = \left\{ \frac{1}{2} \text{tr}(K(\gamma))/\| \gamma \|^2 + C \sum_{i=1}^n \zeta_i \right\}
\]

\[
\text{s.t.} \quad y_i (\gamma^T \phi(x_i; y) + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, \quad \forall i.
\]

It is easy to verify that for this variant, all Propositions 1–3, Theorem 1, and Corollary 1 are still valid if simply changing all terms related to tr(\( S_p \)) to become related to tr(\( K \)), respectively. Moreover, according to Theorem 1, this variant can be converted to the same MKL formulation in Lanckriet et al. [10], where tr(\( K \)) = c is put as a constraint. Furthermore, by Corollary 1, this variant can end up the exact form of the existing SimpleMKL with each basic kernel normalized by tr(\( K_p \)). Such an interesting connection implies that the SimpleMKL actually has considered radius information to some extent. It essentially estimates and minimizes the radius of the MEB whose center is fixed at the origin of a kernel-induced feature space. And this is jointly realized by normalizing each basis kernel with its tr(\( K_p \)) and applying an \( \ell_1 \)-norm constraint. Minimizing such a radius can be related to recent theoretical result in MKL [15,20]. However, compared with tr(\( S_p \)) used in our algorithm, tr(\( K \)) is a less efficient estimate of \( \mathcal{R}^2 \), the radius of the MEB obtained by solving (5). An illustration which demonstrates their relation is given in Fig. 2(a). The advantage of our algorithm will be validated in our experiments.

4. Experimental result

We compare our algorithm with two state-of-the-art MKL algorithms: SimpleMKL [13] and MKL incorporating the radius of MEB [6]. We implement our algorithm based on the SimpleMKL package and the method of [6] by following their paper. The three algorithms are tested on a toy data set and 11 UCI data sets used in Gai et al. [6]. We will verify that our algorithm is able to achieve higher classification performance than the other two, especially when outliers exist. Also, our algorithm maintains the computational efficiency of SimpleMKL and needs less computation than the method in Gai et al. [6]. For short, we call our algorithm TrStMKL and the method in Gai et al. [6] RadiusMKL, respectively.

4.1. Experiments on the toy data set

To demonstrate the robustness of our algorithm, we compare the three algorithms in the presence of outliers. As plotted in Fig. 2(b), the two-dimensional toy data set contains one positive (in blue) and one negative (in red) classes. They distribute as two slightly overlapped rings. We randomly generate 500 positive samples and 500 negative samples. Then, 50 positive samples and 50 negative samples are randomly taken to form a training set, while the remaining 900 samples are used for test. In particular, we add one outlier into the training set, as shown in the top-right corner of Fig. 2(b). The position of the outlier will be set as (5, 5), (10, 10), (15, 15), (20, 20), (25, 25) and (30, 30), respectively, to test the robustness of the compared algorithms. To make the...
classification accuracy with respect to the number of training samples. To accumulate statistics, each experiment is repeated with different number of training samples. FIG. 3 (b) plots performance remains quite stable with the variation of outliers. In contrast, our algorithm, TrStMKL, attains the best classification performance in the whole course, and the difference of the algorithms easy to observe, we use three polynomial kernels (with degree 1–3) as the basic kernels. To ensure each algorithm to show its best performance, we choose the regularization parameter C from a sufficiently large range \( \{2^{-5}, 2^{-3}, \ldots, 2^{15}\} \) by applying four-fold cross validation on each training set. To accumulate statistics, each experiment is repeated for 30 times and the mean accuracy and standard variation are reported.

We first test the algorithms with the outlier at different positions. FIG. 3(a) plots the classification accuracy with the variation of outliers. It can be seen that the accuracy of RadiusMKL decreases with the increase of the magnitude of outliers. This is because the radius of MEB in the objective function of RadiusMKL is affected by the outlier and deviates from the true situation, and this in turn hurts the optimization of basic kernel weight. SimpleMKL also degrades quickly with increasing magnitude of the outlier and shows even worse performance than RadiusMKL. As aforementioned, SimpleMKL considers the scattering radius of training data with respect to the origin of a feature space, which is not an efficient estimate of the radius of MEB. In contrast, our algorithm, TrStMKL, attains the best classification performance in the whole course, and the performance remains quite stable with the variation of outliers.

Then, fixing the outlier at (30,30), we compare the algorithms with different number of training samples. FIG. 3(b) plots the classification accuracy with respect to the number of training samples. As seen, with the decreasing number of training samples, our algorithm shows slower performance degradation than the other two. It can maintain almost the classification accuracy v.s. outlier position. 

### Table 3

Significance test for the difference of classification accuracy between TrStMKL and RadiusMKL [6]. McNemar test with \( \alpha = 0.05 \) is used. The leftmost column is the result compared on original data, while the rest four columns are for the data with one outlier added with the magnitude of 5, 10, 15 and 20, respectively. See the text for details.

<table>
<thead>
<tr>
<th>Data</th>
<th>0</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLD</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>FOUR</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>HEART</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>INDO</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>LIVER</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>SONAR</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>WPBC</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MUSK1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>GEML</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>SPLI</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

---

1 A Gaussian kernel (or more general, stationary kernels) maps all data, including outliers, to a fixed-radius hypersphere, making the radius of MEB less sensitive to outliers. In contrast, a polynomial kernel (or more general, non-stationary kernels) is more vulnerable to outliers.
accuracy of 80% even after the number has been reduced to 40 only. This advantage can again be attributed to the use of \(\text{tr}(\mathbf{S})\). It considers all the training samples equally and is thus more robust to sampling noise which becomes pronounced in the case of small sample. These results preliminarily verify the superior performance of our algorithm in the presence of outliers.

\[\text{Fig. 4. Classification accuracy with variation of outliers. (a) Average on all data. (b) Coloncancer. (c) Fourclass. (d) Germannum. (e) Heart. (f) Ionosphere. (g) Liver. (h) Musk1. (i) Sonar. (j) Splice. (k) Wdbc. (l) Wpbc.}\]
4.2. Experiments on UCI datasets

Firstly, we compare the three algorithms on the 11 UCI data sets used in Gai et al. [6]. All the experiments use the following setting: For each data set, 40% data are used for training and the rest for test. The regularization parameter $C$ is chosen from $\{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$ by four-fold cross validation on each training set. All data sets have been normalized to have zero mean and unit variation on each feature. The basic kernels are the same as those in Rakotomamonjy et al. [13] and Gai et al. [6], which include 10 Gaussian kernels with width of [0.5, 1, 2, 5, 7, 10, 12, 15, 17, 20] and 10 polynomial kernels with degree 1–10. Each experiment is repeated 30 times and the mean accuracy and standard variation are reported. The results are listed in Table 1. The highest accuracy achieved by these algorithms is shown in bold for each data set. Among the 11 data sets, our algorithm consistently gives the highest accuracy on eight of them. To obtain a more precise comparison, we conduct statistical test between our algorithm and the other one. With the presence of an outlier, our algorithm still consistently gives the best classification performance. In order to display more details of the results, we plot the classification accuracy on the 11 data sets in Fig. 4(b)–(l), respectively. As shown, our algorithm achieves statistically significant improvement over the other two on Heart, Ionosphere, Sonar, Splice and Wdbc, while does not produce inferior performance on the remaining data sets. We conduct McNemar test again to check the statistical difference between these algorithms when the outlier with the magnitude $5, 10, 15, 20$ is applied. The results are listed in the rightmost four columns of Tables 2 and 3, respectively. As shown, our algorithm achieves statistically significant improvement over the other two on multiple data sets. The improvement is much more significant than that in the case of no outlier. This again demonstrates the robustness of our algorithm.

The training time spent by different algorithms is compared in Fig. 5. As observed, our proposed algorithm can maintain the training time of SimpleMKL and is faster than RadiusMKL by avoiding solving one extra quadratic programming problem. For the convenience of experimentation, we implement TrStMKL by ourselves and this makes its training time a bit different from SimpleMKL.

From the above experiments, we can see that our algorithm (i) achieves comparable or even slightly better performance than that of SimpleMKL and RadiusMKL in the standard setting, (ii) shows significant improvement over them in the presence of outliers, (iii) maintains the computation of SimpleMKL and more computationally efficient that RadiusMKL. In terms of the ratio of classification performance to training time, the proposed algorithm is the most efficient one.

5. Conclusion

In this paper, we incorporate the trace of data scattering matrix, instead of the radius of the minimal enclosing ball, into existing MKL formulation. Theoretical analysis shows that our
algorithm is not only able to retain all the benefit from the incorporation of radius information, but also more robust and computational efficient. Our algorithm is easily implemented with the existing SimpleMKL package by slightly changing the way of basic kernel normalization. In addition, we show that the existing SimpleMKL actually has implicitly considered radius information to some extent, although it is not as efficient as our algorithm. Experimental study verifies the effectiveness and efficiency of our algorithms, especially in the presence of outliers.

Acknowledgments

The authors thank the anonymous reviewers for numerous suggestions that significantly improved the quality of the paper. This work is in part supported by the National Natural Science Foundation of China (Grant nos. 60970034, 61170287, and 60603015), the Foundation for the Author of National Excellent Doctoral Dissertation (Grant no. 2007B4), and the Foundation for the Author of Hunan Provincial Excellent Doctoral Dissertation.

References


