Research Article

Efficient Multiple Kernel Learning Algorithms Using Low-Rank Representation

Wenjia Niu,^{1,2} Kewen Xia,^{1,2} Baokai Zu,^{1,2} and Jianchuan Bai^{1,2}

¹*School of Electronic and Information Engineering, Hebei University of Technology, Tianjin 300401, China*

²*Key Lab of Big Data Computation of Hebei Province, Tianjin 300401, China*

Correspondence should be addressed to Kewen Xia; kwxia@hebut.edu.cn

Received 23 February 2017; Revised 20 June 2017; Accepted 5 July 2017; Published 22 August 2017

Academic Editor: Cheng-Jian Lin

Copyright © 2017 Wenjia Niu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Unlike Support Vector Machine (SVM), Multiple Kernel Learning (MKL) allows datasets to be free to choose the useful kernels based on their distribution characteristics rather than a precise one. It has been shown in the literature that MKL holds superior recognition accuracy compared with SVM, however, at the expense of time consuming computations. This creates analytical and computational difficulties in solving MKL algorithms. To overcome this issue, we first develop a novel kernel approximation approach for MKL and then propose an efficient Low-Rank MKL (LR-MKL) algorithm by using the Low-Rank Representation (LRR). It is well-acknowledged that LRR can reduce dimension while retaining the data features under a global low-rank constraint. Furthermore, we redesign the binary-class MKL as the multiclass MKL based on pairwise strategy. Finally, the recognition effect and efficiency of LR-MKL are verified on the datasets Yale, ORL, LSVT, and Digit. Experimental results show that the proposed LR-MKL algorithm is an efficient kernel weights allocation method in MKL and boosts the performance of MKL largely.

1. Introduction

Support Vector Machine (SVM) is an important machine learning method [1], which trains linear learner in feature space derived by the kernel function, and utilizes generalization theory to avoid overfitting phenomenon. Recently, Multiple Kernel Learning (MKL) method has received intensive attention due to its more desirable recognition effect over the classical SVM [2, 3]. However, parameter optimization of multiple kernel introduces a high computing cost in searching the entire feature space and solving tremendous convex quadratic optimization problems. Hassan et al. [4] utilize the Genetic Algorithm (GA) to improve the search efficiency in MKL, but the availability of GA remains to be proved and its search direction is too complex to be determined. Besides, with the data volume increasing exponentially in the real world, it is intractable to solve large scale problems by using conventional optimal methods. Therefore, many approaches have been put forward to improve MKL. For example, the Sequential Minimal Optimization (SMO) algorithm [5] is a typical decomposition approach that updates one or two

Lagrange multipliers at every training step to get the iterative solutions. And some online algorithms [6, 7] refine predictors through online-to-batch conversion scheme, whereas it should be noted that the convergence rate of such decomposition approaches is unstable. Another approach is to approximate the kernel matrix such as Cholesky decomposition [8, 9] which is used to reduce the computational cost, however, at the cost of giving up recognition accuracy due to lost information.

Generally, when we set n as sample size and N as the number of kernels, the complexities of solving convex quadratic optimization problems in SVM and MKL are $O(n^3)$ and $O(Nn^{3.5})$ [10], respectively. It can be observed that the computing scale depends on the size of training set rather than the kernel space dimension [8]. In this big data era, it is imperative to find an approach that can minimize the computing scale while capturing the global data structure to perfect SVM or MKL. Low-Rank Representation (LRR) [11] recently has attracted great interest in many research fields, such as image processing [12, 13], computer vision [14], and

data mining [15]. LRR, as a compressed sensing approach, aims to find the lowest-rank linear combination of all training samples for reconstructing test samples under a global low-rank constraint. When the training samples are sufficiently complete, the process of representing data with low-rank will augment the similarities among the intraclass samples and the differences among the interclass samples. Meanwhile, if the data is corrupted, since the rank of coefficient matrix will be largely increased, the lowest-rank criterion can enforce noise correction. LRR integrates data clustering and noise correction into a unified framework, which can greatly improve the recognition accuracy and robustness in the preprocessing stage. In this sense, the recognition of SVM or MKL can be increasingly accurate and of high speed when they are combined with LRR.

In this paper, combining LRR and MKL, we will develop a novel recognition approach so as to construct a Low-Rank MKL (LR-MKL) algorithm. In the proposed algorithm, the combined Low-Rank SVM (LR-SVM) will simultaneously be utilized as the reference. We will conduct extensive experiments on public databases to show that our proposed LR-MKL algorithm can achieve better performance than original SVM and MKL.

The remainder of the paper is organized as follows: We start by a brief review on SVM in next section. In Section 3 we describe some existing MKL algorithms and their structure frames. Section 4 is devoted to introducing efficient MKL algorithms using LRR, which we present and call LR-MKL. Experiments, which demonstrate the utility of the suggested algorithm on real data, are presented in Section 5. Section 6 gives the conclusions.

2. Overview of SVM

Given input space $\mathbf{X} \subseteq \mathbb{R}^D$ and label vector $\mathbf{Y}$, $\{\mathbf{X}, \mathbf{Y}\}$ meets independent and identically distributed conditions, so the training set can be denoted as $\{\mathbf{x}_i, y_i\}_{i=1}^n$ (contains n samples). According to the theory of structural risk minimization [1], SVM can find the classification hyperplane with the maximum margin in the mapping space $\mathbb{R}^P$. Hence, the SVM training with l_1 -norm soft margin is a quadratic optimization problem:

$$\begin{aligned} \min \quad & \frac{1}{2} \langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{i=1}^n \xi_i, \\ \text{s.t.} \quad & y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \\ & \mathbf{w} \in \mathbb{R}^P, \\ & \xi_i \in \mathbb{R}_+^n, \\ & i = 1, \dots, n. \end{aligned} \quad (1)$$

Here, $\mathbf{w}$ is the weight coefficient vector, C is the penalty factor, ξ_i is the slack variable, and b is the bias term of classification hyperplane. The optimization problem can be transformed into its dual form by introducing Lagrangian multiplier α_i, α_j ,

and the data $\mathbf{X}$ can be implicitly mapped to the feature space by utilizing the kernel function K , so formula (1) changes into

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^n \alpha_i, \\ \text{s.t.} \quad & \sum_{i=1}^n \alpha_i y_i = 0, \\ & C \geq \alpha_i \geq 0, \\ & i = 1, \dots, n. \end{aligned} \quad (2)$$

Simplify the objective function of formula (2) into vector form:

$$\begin{aligned} \min \quad & \frac{1}{2} \alpha^T \mathbf{Q} \alpha + \mathbf{1}^T \alpha, \\ \text{s.t.} \quad & \mathbf{y}^T \alpha = 0, \\ & C \geq \alpha \geq 0, \end{aligned} \quad (3)$$

where $\alpha \in \mathbb{R}^n$ is the vector of Lagrangian multiplier, $\mathbf{y} \in \mathbb{Y}^n$ is the label vector, $\mathbf{1}$ is a vector of 1's ($n * 1$ -dimension), and $\mathbf{Q}_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$. If the solution of optimization problem is α_i^* , $i = 1, \dots, n$, the discriminant function can be represented as

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i^* y_i K(\mathbf{x}_i, \mathbf{x}) + b. \quad (4)$$

The kernel functions commonly used in SVM are linear kernel, polynomial kernel, radial basis function kernel, and sigmoid kernel, respectively, denoted as

$$\begin{aligned} K_{\text{LIN}}(\mathbf{x}_i, \mathbf{x}_j) &= \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \\ K_{\text{POL}}(\mathbf{x}_i, \mathbf{x}_j) &= (\gamma * \langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1)^q, \\ K_{\text{RBF}}(\mathbf{x}_i, \mathbf{x}_j) &= \exp(-\gamma * \|\mathbf{x}_i - \mathbf{x}_j\|_2^2), \\ K_{\text{SIG}}(\mathbf{x}_i, \mathbf{x}_j) &= \tanh(\gamma * \langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1). \end{aligned} \quad (5)$$

To obtain the high recognition accuracy in monokernel SVM, we need to discern what kind of kernel distribution characteristics the test data will obey. Nevertheless, it is unpractical and wasteful of resources to try different distribution characteristics one by one. In this sense, we need MKL to allocate the kernel weights based on the data structure automatically.

3. Multiple Kernel Learning (MKL) Algorithms

To improve the universal applicability of SVM algorithm, MKL is applied instead of one specific kernel function:

$$K_\mu(\mathbf{x}_i, \mathbf{x}_j) = f_\mu \left(\left\{ K_m(\mathbf{x}_i, \mathbf{x}_j) \right\}_{m=1}^M \mid \mu \right), \quad (6)$$

where K_m is the monokernel function. The multiple kernel K_μ can be obtained by function $f_\mu : \mathbb{R}^D \rightarrow \mathbb{R}^P$ combining M different K_m . And μ is the proportion parameter of kernel. There are many different methods to assign kernel weights.

Pavlidis et al. [16] propose a simple combination mode using an unweighted sum or product of heterogeneous kernels. The combining function of this Unweighted Multiple Kernel Learning (UMKL) method is

$$\begin{aligned} K_\mu(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{m=1}^M K_m(\mathbf{x}_i, \mathbf{x}_j), \\ K_\mu(\mathbf{x}_i, \mathbf{x}_j) &= \prod_{m=1}^M K_m(\mathbf{x}_i, \mathbf{x}_j). \end{aligned} \quad (7)$$

In a follow-up study, the distribution of μ in MKL becomes a vital limiting factor of availability. Chapelle and Rakotomamonjy [17] report that the optimization problem can be solved by a project gradient method in two alternative steps: first, solving a primal SVM with the given μ ; second, updating μ through the gradient function with α calculated in the first step. The kernel combining function, objective function, and gradient function of this Alternative Multiple Kernel Learning (AMKL) method are

$$\begin{aligned} K_\mu(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{m=1}^M \mu_m K_m(\mathbf{x}_i, \mathbf{x}_j), \\ J(\mu) &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \left(\sum_{m=1}^M \mu_m K_m(\mathbf{x}_i, \mathbf{x}_j) \right) \\ &\quad - \sum_{i=1}^n \alpha_i, \\ \frac{\partial J(\mu)}{\partial \mu_m} &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \frac{K_\mu(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mu_m} \\ &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j K_m(\mathbf{x}_i, \mathbf{x}_j) \quad \forall m. \end{aligned} \quad (8)$$

The Generalized Multiple Kernel Learning (GMKL) method [18] also employs the gradient tool to approach solution, but it regards kernel weights as a regularization item $r(\mu)$, which is taken as $(1/2)(\mu - 1/M)^T(\mu - 1/M)$. So the objective function and gradient function can be transformed into

$$\begin{aligned} J(\mu) &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j K_\mu(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^n \alpha_i - r(\mu), \\ \frac{\partial J(\mu)}{\partial \mu_m} &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \frac{\partial K_\mu(\mathbf{x}_i, \mathbf{x}_j)}{\partial \mu_m} - \frac{\partial r(\mu)}{\partial \mu_m} \quad \forall m. \end{aligned} \quad (9)$$

And the kernel combined function is

$$\begin{aligned} K_\mu(\mathbf{x}_i, \mathbf{x}_j) &= \prod_{m=1}^M \exp(-\mu_m (\mathbf{x}_i^m - \mathbf{x}_j^m)^2) \\ &= \exp\left(\sum_{m=1}^M -\mu_m (\mathbf{x}_i^m - \mathbf{x}_j^m)^2\right). \end{aligned} \quad (10)$$

There is another two-step alternate method using a gating model called Localized Multiple Kernel Learning (LMKL) method [19]. The formula of locally combined kernel is represented as

$$K_\mu(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^M \mu_m(\mathbf{x}_i) \langle \Phi_m(\mathbf{x}_i), \Phi_m(\mathbf{x}_j) \rangle \mu_m(\mathbf{x}_j), \quad (11)$$

where $\Phi(\mathbf{x})$ is the mapping space of feature space. To ensure nonnegativity, kernels can be composed in competitive or cooperative mode by using softmax form and sigmoid form [25], respectively:

$$\begin{aligned} \text{softmax: } \mu_m &= \frac{\exp(\langle \mathbf{v}_m, \mathbf{x} \rangle + v_{m0})}{\sum_{h=1}^M \exp(\langle \mathbf{v}_h, \mathbf{x} \rangle + v_{h0})}, \quad \forall m, \\ \text{sigmoid: } \mu_m &= \frac{1}{\exp(-\langle \mathbf{v}_m, \mathbf{x} \rangle + v_{m0})}, \quad \forall m, \end{aligned} \quad (12)$$

where $\mathbf{V} = \{\mathbf{v}_m, v_{m0}\}_{m=1}^M$ denotes the parameter of gating model. On the other hand, Qiu and Lane [20] quantify the fitness between kernel and accuracy in a Heuristic Multiple Kernel Learning (HMKL) way by exploiting the relationship between kernel matrix $\mathbf{K}$ and sample label $\mathbf{y}$. The relationship can be expressed by kernel alignment:

$$F(\mathbf{K}, \mathbf{y}\mathbf{y}^T) = \frac{\langle \mathbf{K}, \mathbf{y}\mathbf{y}^T \rangle_F}{\sqrt{\langle \mathbf{K}, \mathbf{K} \rangle_F \langle \mathbf{y}\mathbf{y}^T, \mathbf{y}\mathbf{y}^T \rangle_F}} = \frac{\langle \mathbf{K}, \mathbf{y}\mathbf{y}^T \rangle_F}{n \sqrt{\langle \mathbf{K}, \mathbf{K} \rangle_F}}, \quad (13)$$

where $\langle \mathbf{K}, \mathbf{y}\mathbf{y}^T \rangle_F = \sum_{i=1}^n \sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j) y_i y_j$, $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product. Using kernel alignment weighs the proportion of multikernels:

$$\mu_m = \frac{F(K_m, \mathbf{y}\mathbf{y}^T)}{\sum_{h=1}^M F(K_h, \mathbf{y}\mathbf{y}^T)} \quad \forall m. \quad (14)$$

Then, the concentration bound is added in kernel alignment by Cortes et al. [21] to form centering kernel:

$$[K^c]_{ij} = K_{ij} - \frac{1}{n} \sum_{i=1}^n K_{ij} - \frac{1}{n} \sum_{j=1}^n K_{ij} + \frac{1}{n^2} \sum_{i,j=1}^n K_{ij}. \quad (15)$$

Accordingly, the multikernel weights of this Centering Multiple Kernel Learning (CMKL) method are

$$\mu = \frac{\mathbf{C}^{-1} \mathbf{a}}{\|\mathbf{C}^{-1} \mathbf{a}\|_2}, \quad (16)$$

where $\mathbf{C} = \{\langle K_m^c, K_h^c \rangle_F\}_{m,h=1}^M$ and $\mathbf{a} = \{\langle K_m^c, \mathbf{y}\mathbf{y}^T \rangle_F\}_{m=1}^M$.

Later, Cortes et al. [22] studied a Polynomial Multiple Kernel Learning (PMKL) method, which utilized the polynomial combination of the base kernels with higher degree ($d \geq 1$) based on the Kernel Ridge Regression (KRR) theory:

$$K_\mu(\mathbf{x}_i, \mathbf{x}_j) = \sum \mu_{k_1 k_2 \dots k_M} K_1(\mathbf{x}_i^1, \mathbf{x}_j^1)^{k_1} \cdot K_2(\mathbf{x}_i^2, \mathbf{x}_j^2)^{k_2} \dots K_M(\mathbf{x}_i^M, \mathbf{x}_j^M)^{k_M} \quad (17)$$

$$k_m \geq 0, \sum (k_1 + k_2 + \dots + k_M) \leq d, \mu_{k_1 k_2 \dots k_M} \geq 0.$$

However, the computing complex of coefficients $\mu_{k_1 k_2 \dots k_M}$ is $O(M^d)$, which is too large to apply in practice. So $\mu_{k_1 k_2 \dots k_M}$ can be simplified as a product form by nonnegative coefficients $\mu_1^{k_1} \mu_2^{k_2} \dots \mu_M^{k_M}$, and the special case ($d = 2$) can be expressed as

$$K_\mu(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^M \sum_{h=1}^M \mu_m \mu_h K_m(\mathbf{x}_i^m, \mathbf{x}_j^m) K_h(\mathbf{x}_i^h, \mathbf{x}_j^h). \quad (18)$$

Here, the related optimization of learning K_μ can be formulated as the following min-max form:

$$\min_{\mu \in \varphi} \max_{\alpha \in \mathbb{R}^n} -\alpha^T (K_\mu + \lambda \mathbf{I}) \alpha + 2\alpha^T \mathbf{y}, \quad (19)$$

where φ is a positive, bounded, and convex set. Two bounded sets l_1 -norm and l_2 -norm are the appropriate choices to construct φ :

$$\begin{aligned} \varphi_1 &= \{\mu \mid \mu \geq 0, \|\mu - \mu_0\|_1 \leq \wedge\}, \\ \varphi_2 &= \{\mu \mid \mu \geq 0, \|\mu - \mu_0\|_2 \leq \wedge\}. \end{aligned} \quad (20)$$

Here, μ_0 and $\wedge$ are model parameters, and μ_0 is generally equal to 0 or $\mu_0/\|\mu_0\| = 1$.

Other than approaches described above, inspired by the consistency between group Lasso and MKL [26], Xu et al. [23] and Kloft et al. [24] propose an MKL iterative method in a generalized l_p -norm ($p \geq 1$) form. They are collectively called Arbitrary Norms Multiple Kernel Learning (ANMKL) method. On the basis of duality condition $\|\mathbf{w}_m\|_2^2 = \mu_m^2 \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j K_m(\mathbf{x}_i^m, \mathbf{x}_j^m)$, the updated formula of kernels weight is

$$\mu_m = \frac{\|\mathbf{w}_m\|_2^{2/(p+1)}}{\left(\sum_{h=1}^M \|\mathbf{w}_h\|_2^{2p/(p+1)}\right)^{1/p}} \quad \forall m. \quad (21)$$

It can be seen from the formulas in this section that the operation complexity of MKL is mainly decided by $\mathbf{x}_i$. So trying to simplify the feature space is an efficient way to improve the performance of MKL. Through the optimization of basis vectors, LRR can reduce dimension while retaining the data features, which is ideal for improving MKL.

4. MKL Using Low-Rank Representation

4.1. Low-Rank Representation (LRR). The theoretical advances on LRR enable us to use latent low-rank structure

in data [27, 28]. And it simultaneously obtains the representation of all samples under a global low-rank constraint. Meantime, the LRR procedure can operate in a relatively short time with guaranteed performance.

Let the input samples space $\mathbf{X}$ be represented by a linear combination in the dictionary $\mathbf{A}$:

$$\mathbf{X} = \mathbf{AZ}, \quad (22)$$

where $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n]$ is the coefficient matrix and each $\mathbf{z}_i$ is a representation coefficient vector of $\mathbf{x}_i$. When the samples are sufficient, $\mathbf{X}$ serves as the dictionary $\mathbf{A}$. By considering the noise or tainted data in practical application, LRR aims at approximating $\mathbf{X}$ into $\mathbf{AZ} + \mathbf{E}$ by the means of minimizing the rank of matrix $\mathbf{A}$ while reducing the l_0 -norm of $\mathbf{E}$, in which $\mathbf{A}$ is a low-rank matrix and $\mathbf{E}$ is the associated sparse error. It can be generally formulated as

$$\begin{aligned} \min_{\mathbf{Z}, \mathbf{E}} \quad & \text{rank}(\mathbf{A}) + \lambda \|\mathbf{E}\|_0, \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{AZ} + \mathbf{E}. \end{aligned} \quad (23)$$

Here, λ is used to balance the effect of low-rank and error term. l_0 -norm as NP-hard problem can be substituted for l_1 -norm or $l_{2,1}$ -norm. We choose $l_{2,1}$ -norm as the error term measurement here, which is defined as $\|\mathbf{E}\|_{2,1} = \sum_{j=1}^n \sqrt{\sum_{i=1}^m ([\mathbf{E}]_{ij})^2}$. Meantime, $\text{rank}(\mathbf{A})$ can relax into nuclear-norm $\|\cdot\|_*$ [29]. Consequently, the convex relaxation of formula (23) is

$$\begin{aligned} \min_{\mathbf{Z}, \mathbf{E}} \quad & \|\mathbf{Z}\|_* + \lambda \|\mathbf{E}\|_{2,1}, \\ \text{s.t.} \quad & \mathbf{X} = \mathbf{AZ} + \mathbf{E}. \end{aligned} \quad (24)$$

The optimal solution $\mathbf{Z}^*$ can be obtained via the Augmented Lagrange Multipliers (ALM) method [11].

4.2. Efficient SVM and MKL Using LRR. Kernel matrix remarkably impacts the computational efficiency and accuracy of SVM and MKL. How to find an appropriate variant of kernel matrix that contains both the initial label and the data geometry structure for recognition is a crucial task. Since LRR has been theoretically proved to be superior, in the sequel, we adopt LRR to transform the kernel for augmenting the similarities among the intraclass samples and the differences among the interclass samples. Moreover, a representation of all samples under a global low-rank constraint can be attained, which is more conducive to capturing the global data structure [30]. So LR-SVM and LR-MKL are two alternative techniques that we propose to use to improve the performance of SVM and MKL.

Firstly, based on the LRR theory, we improve the monokernel SVM as the reference item, from which the improvement brought by LRR can be displayed visually. The specific procedure of efficient LR-SVM is presented in Algorithm 1.

Algorithm 1 (efficient SVM using LRR (LR-SVM)).

Input. This includes the whole training set $\{\mathbf{X}, \mathbf{Y}\}$, the feature space of testing set $\mathbf{X}_S = [\mathbf{x}_{n+1}, \mathbf{x}_{n+2}, \dots, \mathbf{x}_{n+s}]$, the parameters t, γ, C, q of SVM, and the parameter λ of LRR.

Step 1. Normalize $\mathbf{X}, \mathbf{X}_S$.

Step 2. Perform (24) procedure on the normalized $\mathbf{X}, \mathbf{X}_S$ to project them on the coefficient feature space $\mathbf{Z}, \mathbf{Z}_S$, respectively.

Step 3. Plug $\mathbf{Z}$ and the label vector $\mathbf{Y}$ into SVM for training classification model.

Step 4. Utilize the obtained classification model to classify the coefficient feature $\mathbf{Z}_S$ of testing set $\mathbf{X}_S$, and the discriminant function is $f(\mathbf{z}) = \sum_{i=1}^n \alpha_i^* y_i K(\mathbf{z}_i, \mathbf{z}) + b$.

Output. Compare the actual label vector of test set $\mathbf{Y}_S$ and the prediction label vector $\mathbf{Y}_P$ to obtain the recognition results.

It is well known that SVM suffers from instability for the various data structures. Thus, MKL recognition becomes the development trend. Next, we combine LRR and MKL algorithms mentioned in the Section 3 and change binary-classification model into multiclassification model by pairwise (one-versus-one) strategy, through which a classifier between any two categories of samples (k is the number of categories) can be designed. Then we adopt voting method and assign sample to the category the most votes obtained. All the combined algorithms can be summarized into a frame, which is given in Algorithm 2 and we refer to it collectively as LR-MKL.

Algorithm 2 (efficient MKL using LRR (LR-MKL)).

Input. This includes the whole training set $\{\mathbf{X}, \mathbf{Y}\}$, the feature space of testing set $\mathbf{X}_S = [\mathbf{x}_{n+1}, \mathbf{x}_{n+2}, \dots, \mathbf{x}_{n+s}]$, and the parameter λ of LRR.

Step 1~Step 2. They are the same as the LR-SVM algorithm.

Step 3. Plug $\mathbf{Z}$ and the label vector $\mathbf{Y}$ into MKL to train $k(k-1)/2$ classifiers with the pairwise strategy.

Step 4. Utilize each one of the binary MKL classifiers to classify the coefficient feature $\mathbf{Z}_S$ of testing set $\mathbf{X}_S$.

Step 5. According to the prediction label vectors $\mathbf{Y}_{P_1}, \mathbf{Y}_{P_2}, \dots, \mathbf{Y}_{P_{k(k-1)/2}}$ vote for the category of each sample to get the multilabels $\mathbf{Y}_P$.

Output. Compare the actual label vector of test set $\mathbf{Y}_S$ and the prediction label vector $\mathbf{Y}_P$ to obtain the recognition results and the kernel weight vector μ .

5. Experiments and Analysis

In this section, we conduct extensive experiments to examine the efficiency of proposed LR-SVM and LR-MKL algorithms. The operating environment is based on MATLAB (R2013a) under the Intel Core i5 CPU processor, 2.53 GHz frequency parameters. The SVM toolbox used in this paper is the LIBSVM [31], which can be easily applied and is shown to be fast in large scale databases.

The simulations are performed on diverse datasets to ensure the universal recognition effect. The test datasets range

over the frequently used face databases and the standard test data of UCI repository. In the simulations, all the samples are normalized first.

- (1) Yale face database (<http://vision.ucsd.edu/content/yale-face-database>): it contains 165 grayscale images of 15 individuals with different facial expression or configuration, and each image is resized to $64 * 64$ pixels with 256 grey levels.
- (2) ORL face database (<http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.htm>): it contains 400 images of 40 distinct subjects taken at different times, varying light, facial expressions, and details. We resize them to $64 * 64$ pixels with 256 grey levels per pixel.
- (3) LSVT Voice Rehabilitation dataset (<http://archive.ics.uci.edu/ml/datasets/LSVT+Voice+Rehabilitation>) [32]: it is composed of 126 speech signals from 14 people with 309 features, divided into two categories.
- (4) Multiple Features Digit dataset (<http://archive.ics.uci.edu/ml/datasets/Multiple+Features>): it includes 2000 digitized handwritten numerals 0–9 with 649 features.

5.1. Experiments on LR-SVM. In order to demonstrate the recognition performance of SVM improved by the presented LR-SVM, we carry out numerous experiments on the Yale and ORL face database. According to the different rate of training sample (20%, 30%, 40%, 50%, 60%, 70%, and 80%), we implement seven groups of experiments on each database. To ensure stable and reliable test, each group has ten different divisions randomly, and we average them as the final results. The kernel functions are $K_{\text{LIN}}, K_{\text{POL}}, K_{\text{RBF}}, K_{\text{SIG}}$ ($q = 3$) and $\gamma = 1/g$ (g is the dimension of feature space).

The classification accuracy and run time of Yale database by using SVM and LR-SVM are shown in Figures 1 and 2, respectively. Similarly, the classification accuracy and run time of ORL database are shown in Figures 3 and 4. The solid lines depict the result of SVM with different kernels, while the patterned lines with the corresponding colour depict that of LR-SVM. As can be seen from the Figures 1 and 3, the proposed LR-SVM method consistently achieves an obvious improvement in classification accuracy compared to the original SVM method. In most cases, the classification accuracy increases with the rise in training sample rate. It is shown that the more complete the training set, the better the classification accuracy. But it is impossible for the training set to include so many samples in reality. LR-MKL has a high accuracy even under the low training sample rate, which is suitable for the real applications. Meanwhile, Figures 2 and 4 show that through LRR conversion, the run time can be reduced more than an order of magnitude, which is reasonable for the real-time requirements of data processing in the big data era.

5.2. Experiments on LR-MKL. In this section, we compare the performance of the MKL algorithms involved in Section 3

FIGURE 1: Classification accuracy of Yale by using SVM and LR-SVM.

FIGURE 2: Run time of Yale by using SVM and LR-SVM.

FIGURE 3: Classification accuracy of ORL by using SVM and LR-SVM.

FIGURE 4: Run time of ORL by using SVM and LR-SVM.

and their corresponding LR-MKL algorithms. The multi-kernel is composed of $K_{\text{LIN}}, K_{\text{POL}}, K_{\text{RBF}}, K_{\text{SIG}}$ ($q = 3$). The proportion parameter vector of kernel is $\mu = [\mu_1, \mu_2, \mu_3, \mu_4]$. The comparative algorithms are listed below:

- (i) Unweighted MKL (UMKL) [16] and LR-UMKL: (+) indicates sum form, and (*) indicates product form
- (ii) Alternative MKL (AMKL) [17] and LR-AMKL
- (iii) Generalized MKL (GMKL) [18] and LR-GMKL
- (iv) Localized MKL (LMKL) [19] and LR-LMKL: (sof) distribute μ into softmax mode, and (sig) distribute μ into sigmoid mode

- (v) Heuristic MKL (HMKL) [20] and LR-HMKL
- (vi) Centering MKL (CMKL) [21] and LR-CMKL
- (vii) Polynomial MKL (PMKL) [22] and LR-PMKL: (1) adopts the bounded set φ_1 with l_1 -norm, and (2) adopts the bounded set φ_2 with l_2 -norm
- (viii) Arbitrary Norm MKL (ANMKL) [23, 24] and LR-ANMKL: (1) iterates μ with l_1 -norm, and (2) iterates μ with l_2 -norm
- (ix) Besides the highest accuracy among the four monokernel SVM selected as the reference item, which is referred to as SVM(best)

TABLE 1: The performances of MKL algorithms and LR-MKL algorithms on the datasets Yale, ORL, LSVT, and Digit.

	Yale			ORL			LSVT			Digit		
	Acc	Time	Time	Acc	Time	Time	Acc	Time	Time	Acc	Time	Time
SVM(best)	69.3231	0.2981	1.6798	80.0002	0.2981	1.6798	76.1905	0.0142	0.9195	95.9302	0.9195	0.9195
LR-SVM(best)	79.8667	0.0109	0.1575	87.0015	0.1575	0.1575	77.6637	0.0043	0.0972	96.3304	0.0972	0.0972
UMKL(+) [16]	87.5114	2.5981	18.2352	93.5798	18.2352	18.2352	78.5714	0.0229	4.7412	97.3571	4.7412	4.7412
LR-UMKL(+)	95.8936	0.3018	1.0817	83.4813	1.0817	1.0817	73.8095	0.0152	2.0278	98.1595	2.0278	2.0278
UMKL(*) [16]	58.3248	2.7244	20.5507	77.5022	20.5507	20.5507	66.6935	0.0281	7.1661	96.0904	7.1661	7.1661
LR-UMKL(*)	93.5739	0.2636	2.1836	93.4063	2.1836	2.1836	67.0033	0.0176	3.6392	98.4618	3.6392	3.6392
AMKL [17]	85.7753	3.7236	4.6854	93.8741	4.6854	4.6854	80.9524	0.0452	11.2138	97.4725	11.2138	11.2138
LR-AMKL	94.3799	0.3869	0.4592	96.9417	0.4592	0.4592	88.0952	0.0085	6.8804	98.6952	6.8804	6.8804
GMKL [18]	86.2989	4.5330	5.0070	96.2057	5.0070	5.0070	85.7143	0.0565	8.0774	99.1499	8.0774	8.0774
LR-GMKL	95.8015	0.6253	0.7761	98.5833	0.7761	0.7761	88.9286	0.0183	4.5972	99.4673	4.5972	4.5972
LMKL(sof) [19]	87.9077	215.4055	220.3122	97.0003	220.3122	220.3122	85.0090	5.1989	166.7978	99.8898	166.7978	166.7978
LR-LMKL(sof)	97.9352	22.7204	17.3791	98.9724	17.3791	17.3791	86.7429	1.1933	98.1812	99.8898	98.1812	98.1812
LMKL(sig) [19]	88.0145	106.7552	107.0911	97.0108	107.0911	107.0911	88.7541	0.7238	48.5914	99.3750	48.5914	48.5914
LR-LMKL(sig)	98.0667	15.9711	11.7240	97.9979	11.7240	11.7240	92.6627	0.4594	24.5927	99.5625	24.5927	24.5927
HMKL [20]	63.6037	92.4410	118.2340	93.5109	118.2340	118.2340	80.5998	0.0915	10.3559	97.6258	10.3559	10.3559
LR-HMKL	91.9611	8.5972	10.9368	98.6893	10.9368	10.9368	85.1625	0.0352	6.3959	98.3479	6.3959	6.3959
CMKL [21]	86.4166	95.0618	107.6940	96.0308	107.6940	107.6940	79.9503	0.0874	10.6074	96.5014	10.6074	10.6074
LR-CMKL	94.0083	10.6380	12.9746	98.4799	12.9746	12.9746	93.9024	0.0347	6.2696	98.9113	6.2696	6.2696
PMKL(1) [22]	89.0035	6.1842	6.9065	98.4901	6.9065	6.9065	92.8571	0.1079	24.8702	99.5881	24.8702	24.8702
LR-PMKL(1)	99.1429	0.9053	0.9828	99.5712	0.9828	0.9828	95.9386	0.0573	12.9831	100.0000	12.9831	12.9831
PMKL(2) [22]	89.0261	5.3893	6.5450	98.7533	6.5450	6.5450	92.4662	0.1295	21.1679	99.5046	21.1679	21.1679
LR-PMKL(2)	98.8968	0.8494	0.8911	99.5828	0.8911	0.8911	95.5145	0.0651	13.5108	99.7941	13.5108	13.5108
ANMKL(1) [23, 24]	86.7210	6.4856	20.4564	98.4396	20.4564	20.4564	91.9827	0.1167	10.3979	98.0007	10.3979	10.3979
LR-ANMKL(1)	96.8667	0.7041	2.8519	99.4643	2.8519	2.8519	92.2247	0.0247	6.9070	99.2850	6.9070	6.9070
ANMKL(2) [23, 24]	86.6998	7.0664	21.1615	98.2204	21.1615	21.1615	93.0035	0.1194	9.7753	98.0039	9.7753	9.7753
LR-ANMKL(2)	97.2917	0.8863	3.0597	99.2857	3.0597	3.0597	92.5391	0.0224	5.0374	99.2497	5.0374	5.0374

We conduct experiments on the test datasets Yale, ORL, LSVT Voice Rehabilitation (LSVT, for short), and Multiple Features Digit (Digit, for short). The 60% samples of dataset are drawn out randomly to train classification model and the remaining samples serve as the test set. Through the optimized results of γ and penalty factor C by grid search method, we find that the classification accuracy varies not too much with γ and C ranging in a certain interval. So there is no need to search the whole parameter space which inevitably increases the computational cost. The penalty factor C can be given that trying values 0.01, 0.1, 1, 10, 100, and γ are fixed on $1/g$. Then we assign a value which has the highest average accuracy on the $5 * 2$ cross validation sets to C . Each algorithm is conducted with 10 independent runs, and we average them as the final results. The bold numbers represent the preferable recognition effect between the original algorithms and their LRR combined algorithms. The numbers in italic font denote the algorithms whose recognition precision is inferior to the SVM(best). The recognition performance of algorithms is measured by the classification accuracy and run time, illustrated by Table 1.

In most cases, our proposed LR-MKL methods consistently achieve superior results to the original MKL, which verifies the higher classification accuracy and shorter operation time. It is indicated that LRR can augment the similarities among the intraclass samples and the differences among the interclass samples while simplifying the kernel matrix. Note that UMKL(*) fails to achieve the ideal recognition effects in many cases, even less accurate than SVM(best). However, combining with LRR improves its effects to a large extent. This illustrates that simply combining kernels without according data structure is infeasible, and LRR can offset part of the consequences of irrational distribution. In general, PMKL, ANMKL, and their improved algorithms have the preferable recognition effects, especially the improved algorithms with the accuracy over 90 percent all the time. In terms of run time, it is clearly observed that the real-time performance of MKL is much worse than SVM, because MKL has a process of allocating kernel weights and the process can be very time consuming. Among them, LMKL is the worst and fails to satisfy the real-time requirement. Obviously, our combined LR-MKL can reduce the run time manifold even more than one order of magnitude, so it can speed high-precision MKL up to satisfy the real-time requirement. In brief, the proposed LR-MKL can boost the performance of MKL to a great extent.

6. Conclusion

The complexity of solving convex quadratic optimization problem in MKL is $O(Nn^{3.5})$, so it is infeasible to apply in large scale problems for its large computational cost. Our effort has been made on decreasing the dimension of training set. Note that LRR just can capture the global structure of data in relatively few dimensions. Therefore, we have given a review of several existing MKL algorithms. Based on this point, we have proposed a novel combined LR-MKL, which largely improves the performance of MKL. A large number of experiments have been carried on four real world datasets to contrast the recognition effects of various kinds of MKL and

LR-MKL algorithms. It has been shown that in most cases, the recognition effects of MKL algorithms are better than SVM(best), except UMKL(*). And our proposed LR-MKL methods have consistently achieved the superior results to the original MKL. Among them, PMKL, ANMKL, and their improved algorithms have shown possessing the preferable recognition effects.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 51208168), Hebei Province Natural Science Foundation (no. E2016202341), and Tianjin Natural Science Foundation (no. 13JCYBJC37700).

References

- [1] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.
- [2] 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.
- [3] X. Wang, X. Liu, N. Japkowicz, and S. Matwin, "Ensemble of multiple kernel SVM classifiers," *Advances in Artificial Intelligence*, vol. 8436, pp. 239–250, 2014.
- [4] E. Hassan, S. Chaudhury, N. Yadav, P. Kalra, and M. Gopal, "Off-line hand written input based identity determination using multi kernel feature combination," *Pattern Recognition Letters*, vol. 35, no. 1, pp. 113–119, 2014.
- [5] F. Cai and V. Cherkassky, "Generalized SMO algorithm for SVM-based multitask learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 6, pp. 997–1003, 2012.
- [6] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *Journal of Machine Learning Research*, vol. 7, pp. 551–585, 2006.
- [7] N. Cesa-Bianchi, A. Conconi, and C. Gentile, "On the generalization ability of on-line learning algorithms," *Institute of Electrical and Electronics Engineers. Transactions on Information Theory*, vol. 50, no. 9, pp. 2050–2057, 2004.
- [8] S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations," *Journal of Machine Learning Research*, vol. 2, no. 2, pp. 243–264, 2002.
- [9] S. Zhou, "Sparse LSSVM in primal using cholesky factorization for large-scale problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 783–795, 2016.
- [10] L. Jia, S.-Z. Liao, and L.-Z. Ding, "Learning with uncertain kernel matrix set," *Journal of Computer Science and Technology*, vol. 25, no. 4, pp. 709–727, 2010.
- [11] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, 2013.
- [12] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "RASL: robust alignment by sparse and low-rank decomposition for linearly

- correlated images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2233–2246, 2012.
- [13] B. Cheng, G. Liu, J. Wang, Z. Huang, and S. Yan, “Multi-task low-rank affinity pursuit for image segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV ’11)*, pp. 2439–2446, IEEE, Barcelona, Spain, November 2011.
  - [14] Y. Mu, J. Dong, X. Yuan, and S. Yan, “Accelerated low-rank visual recovery by random projection,” in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011*, pp. 2609–2616, Colorado Springs, Colo, USA, June 2011.
  - [15] J. Chen, J. Zhou, and J. Ye, “Integrating low-rank and group-sparse structures for robust multi-task learning,” in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD’11*, pp. 42–50, San Diego, Calif, USA, August 2011.
  - [16] P. Pavlidis, J. Cai, J. Weston, and W. N. Grundy, “Gene functional classification from heterogeneous data,” in *Proceedings of the 5th Annual International Conference on Computational Biology*, pp. 249–255, Montreal, Canada, May 2001.
  - [17] O. Chapelle and A. Rakotomamonjy, “Second order optimization of kernel parameters,” *Nips Workshop on Kernel Learning*, 2008.
  - [18] M. Varma and B. R. Babu, “More generality in efficient multiple kernel learning,” in *Proceedings of the 26th International Conference On Machine Learning, ICML 2009*, pp. 1065–1072, Montreal, Canada, June 2009.
  - [19] M. Gönen and E. Alpaydin, “Localized multiple kernel learning,” in *Proceedings of the the 25th international conference*, pp. 352–359, Helsinki, Finland, July 2008.
  - [20] S. Qiu and T. Lane, “A framework for multiple kernel support vector regression and its applications to siRNA efficacy prediction,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 6, no. 2, pp. 190–199, 2009.
  - [21] C. Cortes, M. Mohri, and A. Rostamizadeh, “Two-stage learning kernel algorithms,” in *Proceedings of the 27th International Conference on Machine Learning, ICML 2010*, pp. 239–246, Haifa, Israel, June 2010.
  - [22] C. Cortes, M. Mohri, and A. Rostamizadeh, “Learning non-linear combinations of kernels,” in *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS ’09)*, pp. 396–404, December 2009.
  - [23] Z. Xu, R. Jin, H. Yang, I. King, and M. R. Lyu, “Simple and efficient multiple kernel learning by group lasso,” in *Proceedings of the 27th International Conference on Machine Learning, ICML 2010*, pp. 1175–1182, Haifa, Israel, June 2010.
  - [24] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, “Non-sparse regularization and efficient training with multiple kernels. Arxiv Preprint abs/1003,” <https://arxiv.org/abs/1003.0079>.
  - [25] M. Gönen and E. Alpaydin, “Multiple kernel learning algorithms,” *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.
  - [26] F. R. Bach, “Consistency of the group lasso and multiple kernel learning,” *Journal of Machine Learning Research*, vol. 9, no. 2, pp. 1179–1225, 2008.
  - [27] E. J. Candes, X. Li, Y. Ma, and J. Wright, “Robust principal component analysis?” *Journal of the ACM*, vol. 58, no. 3, 2011.
  - [28] C.-F. Chen, C.-P. Wei, and Y.-C. F. Wang, “Low-rank matrix recovery with structural incoherence for robust face recognition,” in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012*, pp. 2618–2625, Providence, RI, USA, June 2012.
  - [29] J.-F. Cai, E. J. Candes, and Z. Shen, “A singular value thresholding algorithm for matrix completion,” *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.
  - [30] L. Zhuang, H. Gao, Z. Lin, Y. Ma, X. Zhang, and N. Yu, “Non-negative low rank and sparse graph for semi-supervised learning,” in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012*, pp. 2328–2335, usa, June 2012.
  - [31] C. Chang and C. Lin, “LIBSVM: a Library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, article 27, 2011.
  - [32] A. Tsanas, M. A. Little, C. Fox, and L. O. Ramig, “Objective automatic assessment of rehabilitative speech treatment in parkinson’s disease,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 1, pp. 181–190, 2014.

Hindawi

Submit your manuscripts at
<https://www.hindawi.com>