Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 372748, 9 pages
http://dx.doi.org/10.1155/2015/372748

Research Article

Distance Based Multiple Kernel ELM: A Fast Multiple Kernel Learning Approach

Chengzhang Zhu,[1] Xinwang Liu,[1] Qiang Liu,[1] Yuewei Ming,[1] and Jianping Yin[2]

[1] *College of Computer, National University of Defense Technology, Changsha 410073, China*
[2] *State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha 410073, China*

Correspondence should be addressed to Chengzhang Zhu; kevin.zhu.china@gmail.com

Received 20 August 2014; Revised 7 November 2014; Accepted 10 November 2014

Academic Editor: Tao Chen

We propose a distance based multiple kernel extreme learning machine (DBMK-ELM), which provides a two-stage multiple kernel learning approach with high efficiency. Specifically, DBMK-ELM first projects multiple kernels into a new space, in which new instances are reconstructed based on the distance of different sample labels. Subsequently, an ℓ_2-norm regularization least square, in which the normal vector corresponds to the kernel weights of a new kernel, is trained based on these new instances. After that, the new kernel is utilized to train and test extreme learning machine (ELM). Extensive experimental results demonstrate the superior performance of the proposed DBMK-ELM in terms of the accuracy and the computational cost.

1. Introduction

Currently, classification and regression are two major problems targeted by most of machine learning, pattern recognition, and data mining methods. For example, one may use a classifier in fingerprint identification system [1] or introduce regression to predict stock price [2], and so forth. As a kind of structure that can process classification and regression problems, single hidden layer feedforward networks (SLFNs) have been extensively studied. In previous work, many methods have been proposed to train SLFNs, such as back propagation algorithm [3] and SVM [4]. However, the above training methods may have two main drawbacks, that is, local extremum and long training time.

Recently, Huang et al. have proposed extreme learning machine (ELM) to train SLFNs in an extremely fast fashion [5, 6]. It has been proved that ELM can overcome the main drawbacks of previous SLFNs training methods. Specifically, ELM can learn a global optimal solution of the SLFNs' parameters, which can contribute to a high performance in both classification and regression problems, in an extremely short time because it only needs to train the output weights between hidden layer and output layer via the least square method [7, 8]. The reason is that ELM only needs to train the output weights between the hidden layer and the output layer via the least square method. Another attractive feature of ELM is that it establishes a unified model for solving both classification and regression problems [8]. Considering the outstanding advantages of ELM, numerous valuable applications based on the ELM have been proposed, such as [9, 10]. Meanwhile, many researchers have promoted the evolution of ELM recently, including proposed online sequential ELM [11], voting based ELM [12], weighted ELM [13], sparse ELM [14], and kernel based ELM [8]. As it can get rid of the impact of the number of hidden nodes, one of their work, the kernel based ELM, is applicable to a wide range and has perfect performance. So far, most studies that towards kernel based ELM focus on using a single kernel. However, since description ability of single kernel is weaker than multiple kernels in most cases, it may get better results to use multiple kernels in kernel based ELM. Moreover, using multiple kernels can handle multiple source information fusing problem, which can further improve the performance of classification and regression in some cases.

Multiple kernel learning (MKL) is a kind of machine learning method that can enable classifier and regressor to utilize multiple kernel information. Given a set of base kernels, the goal of MKL is to construct a new kernel,

which can be more suitable to address the problem at hand, through learning an appropriate combination of base kernels. Typically, MKL can be sorted into two categories, one-stage approach and two-stage approach, by its approach. The one-stage approach learns the combination coefficients of the base kernels and the parameters of the classifier jointly by solving a joint optimization objective function. After [15] pioneered this kind of approach, which got great attention, a lot of work following it has been proposed, including [16–19], to name just a few. On the contrary, the two-stage approach, such as [20, 21], constructs a new kernel firstly by finding a suitable combination of base kernels and then it uses this combination in classifier or regressor.

In previous work, some researchers tried to introduce multiple kernels into ELM, such as [22, 23], and got satisfactory results. Although these efforts made pioneering achievements, all of them fail to achieve an extreme learning speed or cannot make sense in both classification and regression cases. In this paper, we propose a novel multiple kernel based ELM, named distance based multiple kernel extreme learning machine (DBMK-ELM), which is a fast two-stage multiple kernel learning approach and can be adapted to both classification and regression. In the first stage, DBMK-ELM finds the combination coefficients of pregenerated base kernels based on training samples. It first projects original base kernels into a new space and reconstructs new instances based on the distance of training samples. Then, it transfers the multiple kernel learning problem to a binary classification problem or a regression problem and solves it using the least square method. Finally, it constructs the new kernel from base kernels based on the learned combination coefficients. In the second stage, DBMK-ELM adopts the new kernel in kernel based ELM. Experimental results demonstrate the following advantages of our proposed DBMK-ELM: (1) the training time of DBMK-ELM is extremely short compared with traditional MKL methods; (2) DBMK-ELM can fully use multiple source information and outperform previous MKL methods in terms of the classification and regression accuracy; (3) DBMK-ELM can improve the robustness and the accuracy of basic kernel based ELM in both classification and regression cases.

The rest of paper is organized as follows. Section 2 briefly introduces ELM. Then, Section 3 presents the proposed DBMK-ELM. Meanwhile, Section 4 evaluates the performance of DBMK-ELM via extensive experiments. Finally, Section 5 concludes the paper.

2. Related Work

Our proposed method is extended from ELM, specifically, kernel based ELM. In this section, we briefly introduce ELM and kernel based ELM.

2.1. Extreme Learning Machine. Extreme learning machine is a perfect training method of single hidden layer feedforward networks (SLFNs). Since it was proposed by Huang et al. [6], ELM has been widely used in numerous areas. The main advantages include but are not limited to (1) the extreme training speed; (2) the fascinating generalization performance. These advantages are attributed to the fact that ELM randomly generates the weights between input layer and hidden layer and uses a least squares method to learn the other weights. For instance, if we consider a SLFNs as Figure 1 illustrating, which has L hidden layer nodes and one output layer node, the output function of it is as follows:

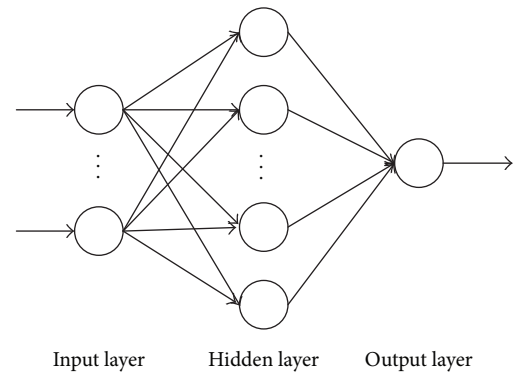

Figure 1: Single hidden layer feedforward networks.

$$f(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta}, \tag{1}$$

where $\boldsymbol{\beta}$ is the weights between hidden layer and output layers and $\mathbf{h}(\mathbf{x}) = [G(\mathbf{a}_1, b_1, \mathbf{x}), \ldots, G(\mathbf{a}_L, b_L, \mathbf{x})]$ is the value vector of hidden layer that maps original features into a new feature space. Different from other SLFNs training methods, ELM can use any $\mathbf{h}(\mathbf{x})$, in which $G(\mathbf{a}_i, b, \mathbf{x})$ is a nonlinear piecewise continuous function, such as sigmoid function and Gaussian function. Moreover, $\mathbf{a}$, the weights between input layer and hidden layer, and b, the bias of hidden nodes, can be randomly chosen based on any continuous probability distribution in ELM. Thus, the only parameter that ELM must learn is $\boldsymbol{\beta}$. ELM learns $\boldsymbol{\beta}$ through solving the following optimization problem [8]:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C\sum_{i=1}^{N}\xi_i^2 \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} = y_i - \xi_i, \quad i = 1, \ldots, N, \end{aligned} \tag{2}$$

where $\{\mathbf{x}_i, y_i\}$ is training samples. According to KKT theorem, $\boldsymbol{\beta}$ can be calculated from

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^\top\mathbf{H}\right)^{-1}\mathbf{H}^\top\mathbf{Y} \tag{3}$$

or

$$\boldsymbol{\beta} = \mathbf{H}^\top\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^\top\right)^{-1}\mathbf{Y}, \tag{4}$$

where $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1), \ldots, \mathbf{h}(\mathbf{x}_N)]^\top$ and $\mathbf{Y} = [y_1, \ldots, y_N]^\top$. One can use (3) for the case where the number of training samples is huge, that is, $N \gg L$, and use (4) on the contrary, that is, $N \ll L$.

2.2. Kernel Based Extreme Learning Machine. Kernel method can be easily introduced into ELM. Specifically, ELM kernel K can be derived from ELM feature mapping function $\mathbf{h}(\mathbf{x})$ as follows:

$$\mathbf{K} = \mathbf{H}\mathbf{H}^{\top} : K_{i,j} = k\left(\mathbf{x}_i, \mathbf{x}_j\right) = h\left(\mathbf{x}_i\right) \cdot h\left(\mathbf{x}_j\right). \quad (5)$$

Therefore, ELM output function can be written as follows:

$$f\left(\mathbf{x}\right) = \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^{\top} \left(\frac{\mathbf{I}}{C} + \mathbf{K}\right)^{-1} \mathbf{Y}. \quad (6)$$

In this case, users do not have to know the $\mathbf{h}(\mathbf{x})$ or set the number of hidden nodes L, that is, the dimension of ELM feature space.

3. Distance Based Multiple Kernel ELM

Since the goal of multiple kernel learning is to construct a new kernel that more suitable for problem processing, the nature of "good" kernel must be considered. Generally, kernel can be seen as a measure of similarity. Each entry in a kernel matrix represents a similarity of two corresponding samples. From this point of view, a "good" kernel can display the true similarity of sample pairs. In other words, if two sample pairs have similar similarity, their corresponding value in the "good" kernel will also be similar. To this end, we propose distance based multiple kernel ELM. It measures similarities of sample pairs by their "label distance" that will be defined in the following part and uses this information to construct a "good" kernel. DBMK-ELM is a kind of two-stage multiple kernel learning method. In the first stage, it learns a new kernel. In the second stage, it uses the new kernel in kernel based ELM.

3.1. Label Distance. We first define the "label distance". As a significant part of training samples, the label contains class information in the classification case and dependent variable information in the regression case, which can be used to measure true similarity between samples. Considering different label meaning between classification and regression, we discuss "label distance" in these two cases separately.

In the classification case, the label means the class that a sample belongs to and samples can be seen as similar when they are in the same class. In other words, if two samples have the same label, they can be seen as similar. On the contrary, if two samples have different labels, they can be seen as different. However, in this case, it is difficult to discriminate how different two samples are, because the difference between classes is not clear. Therefore, label distance is defined to 0 if two samples have the same label and defined to 1 if two samples have different labels. Formally, we define label distance $g(y, y')$ as follows:

$$g\left(y, y'\right) = \begin{cases} 0 & y = y' \\ 1 & y \neq y'. \end{cases} \quad (7)$$

In the regression case, the label means the value of the dependent variable. Typically, the similarity of samples is directly represented in the difference of their values of dependent variable in regression cases, for example, pollution prediction, housing number prediction, and stock price prediction, to name just a few. Thus, label distance can be defined as distance between two values of the dependent variable. Admittedly, various measurements can be used to measure this distance, but Euclidean distance is used in this paper. We, in the regression case, formally define the label distance $g(y, y')$ as follows:

$$g\left(y, y'\right) = \left|y - y'\right|. \quad (8)$$

3.2. Multiple Kernel Learning Based on Distance. Since label distance has been defined above, the distance information can be used to guide the new kernel learning. In this subsection, we show how does DBMK-ELM perform multiple kernel learning based on distance. As we discussed, the new optimal kernel can be seen as a linear combination of base kernels. Therefore, the goal of distance based multiple kernel learning (DBMK) is to learn the combination coefficients of base kernels from which a new kernel that each entry value is coincident to the label distance of the corresponding sample pair can be generated. Considering values of the same entry in each base kernels, corresponding to the same sample pair, if they are seen as input features and the label distance of the same sample pair is seen as output value, the DBMK can be transformed to the regression problem, in which parameters of the regressor are the combination coefficients that need to be learned. To this end, DBMK learns the combination coefficients following the next two steps. Firstly, it reconstructs a new sample space, named K-space, in which each sample corresponds to a sample pair in the original sample space, based on multiple base kernels and label distance of sample pairs. Secondly, it solves a regression problem in this new space to find combination coefficients of kernels. After this two steps, DBMK-ELM can obtain the new optimal kernel through combining the base kernels using learned combination coefficients.

For machine learning problems, including classification and regression, if training samples $(\mathbf{x}, y)$ are drawn from a distribution P over $\mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^n \times \mathbb{R}$ and p base kernels, which must satisfy the positive semi-definite condition, are generated by a set of kernel functions $\{k_1(\cdot,\cdot), \ldots, k_p(\cdot,\cdot)\}$ and denoted as $\mathbf{K}_1, \ldots, \mathbf{K}_p$ with $\mathbf{K}_i$: $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$, DMKL-ELM reconstructs the K-space as $\{\mathbf{z}_{\mathbf{x}\mathbf{x}'}, t_{yy'} \mid ((\mathbf{x}, y), (\mathbf{x}', y') \sim P \times P) \subset \mathbb{R}^p \times \mathbb{R}\}$ where

$$\begin{aligned} \mathbf{z}_{\mathbf{x}\mathbf{x}'} &= \left(k_1\left(\mathbf{x}, \mathbf{x}'\right), \ldots, k_p\left(\mathbf{x}, \mathbf{x}'\right)\right), \\ t_{yy'} &= g\left(y, y'\right). \end{aligned} \quad (9)$$

In the K-space, DBMK-ELM learns the combination coefficients through solving a regression problem. In this problem, the training samples are $(\mathbf{z}, t)$, the whole samples in K-space, in which input feature vector is $\mathbf{z}$ and the output label is t. Though numbers of methods can be used to solve the regression problem, DBMK-ELM applies an ℓ_2-norm regularization least squares linear regression, which is similar to ELM output weight learning, to solve it, considering a fast

training speed. This method aims to minimize the training error and the norm of combination coefficients at the same time. If K-space has N samples, the optimization objective function of it can be formalized as follows:

$$\begin{aligned} \min_{\boldsymbol{\mu}} \quad & \frac{1}{2}\left\|\boldsymbol{\mu}\right\|^2 + \frac{1}{2}C\sum_{i=1}^{N}\xi_i^2 \\ \text{s.t.} \quad & \boldsymbol{z}_i\boldsymbol{\mu} = t_i - \xi_i, \quad i = 1, \ldots, N, \end{aligned} \tag{10}$$

where $\boldsymbol{\mu}$ is the combination coefficients that DBMK-ELM needs to learn, $(\mathbf{z}_i, t_i)$ corresponds to the ith sample in K-space, ξ_i represents the training error generated by ith sample, and C is a trade-off parameter.

Using KKT theorem, solving (10) is equivalent to solving its dual optimization problem:

$$L = \frac{1}{2}\left\|\boldsymbol{\mu}\right\|^2 + \frac{1}{2}C\sum_{i=1}^{N}\xi_i^2 - \sum_{i=1}^{N}\alpha_i\left(\mathbf{z}_i\boldsymbol{\mu} - t_i + \xi_i\right), \tag{11}$$

where α_i is the Lagrange multiplier corresponding to the ith sample. In this case, KKT optimality conditions can be written as follows:

$$\begin{aligned} \frac{\partial L}{\partial \boldsymbol{\mu}} = 0 &\longrightarrow \boldsymbol{\mu} = \sum_{i=1}^{N}\alpha_i\mathbf{z}_i^\top = \mathbf{Z}^\top\boldsymbol{\alpha} \\ \frac{\partial L}{\partial \xi_i} = 0 &\longrightarrow \alpha_i = C\xi_i, \quad i = 1, \ldots, N \\ \frac{\partial L}{\partial \alpha_i} = 0 &\longrightarrow \mathbf{z}_i\boldsymbol{\mu} - t_i + \xi_i = 0, \quad i = 1, \ldots, N, \end{aligned} \tag{12}$$

where $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_N]^\top$ and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^\top$. According to (12), we have

$$\boldsymbol{\mu} = \mathbf{Z}^\top\left(\frac{\mathbf{I}}{C} + \mathbf{Z}\mathbf{Z}^\top\right)^{-1}\mathbf{T} \tag{13}$$

and equally

$$\boldsymbol{\mu} = \left(\frac{\mathbf{I}}{C} + \mathbf{Z}^\top\mathbf{Z}\right)^{-1}\mathbf{Z}^\top\mathbf{T}, \tag{14}$$

where $\mathbf{T} = [t_1, \ldots, t_N]^\top$ in both equations above. We can use (13) to calculate $\boldsymbol{\mu}$ when the number of training samples in K-space is not huge and use (14) in the opposite case to speed up the computation.

Finally, DBMK-ELM obtains the new optimal kernel by combining base kernels according to $\boldsymbol{\mu}$ as follows:

$$K_{\text{new}} = \mu_1 K_1 + \cdots + \mu_p K_p. \tag{15}$$

Similarily, for each two sample pair (x_i, x_j), their new optimal kernel function can be written as follows:

$$k_{\text{new}}\left(x_i, x_j\right) = \mu_1 k_1\left(x_i, x_j\right) + \cdots + \mu_p k_p\left(x_i, x_j\right). \tag{16}$$

3.3. Multiple Kernel Extreme Learning Machine. In the second stage, DBMK-ELM uses the learned new kernel, in which multiple kernel information is included, in the kernel based ELM in both the training and testing case. Therefore, the output function of DBMK-ELM can be written as follows:

$$f(\mathbf{x}) = \begin{bmatrix} k_{\text{new}}(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k_{\text{new}}(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^\top \left(\frac{\mathbf{I}}{C} + \mathbf{K}_{\text{new}}\right)^{-1}\mathbf{Y}, \tag{17}$$

where $\mathbf{Y} = [y_1, \ldots, y_N]^\top$. At this point, DBMKL-ELM can successfully deal with the classification and regression problem benefitting from multiple kernel at a fast speed.

For a classification or regression problem, DBMK-ELM first learns the combination coefficients of pregenerated base kernels using (13) or (14). Then, it calculates a new optimal kernel by (15). Finally, the result of the problem can be obtained by (17). The algorithm of DBMK-ELM can be illustrated as Algorithm 1.

3.4. Connected to TS-MKL. A previous successful multiple kernel learning approach TS-MKL (two-stage multiple kernel learning) [21] also follows the idea that uses the label information to learn kernel. It denotes +1 for the same label and −1 for different labels to construct its target labels. This is very similar to DBMK-ELM label distance in the classification case. The experimental results show that this method can find a really good kernel that achieves the state-of-the-art classification performance. However, TS-MKL can only solve the classification problem. DBMK-ELM does not just consider the difference between classes but uses label distance to measure the similarity of a sample pair. In this way, DBMK-ELM not only adapts to the classification problem but also adapts to the regression problem.

4. Experiments

In this section, we first compare DBMK-ELM to several methods in both classification and regression benchmarks using pregenerated kernels. In order to verify DBMK-ELM performance on multiple kernel learning, SimpleMKL [16] and unweighted sum of kernel methods (UW) have been compared. Meanwhile, we also compare DBMK-ELM with basic kernel based ELM [8] in which the best kernel in base kernels has been used. Then, we compare the classification accuracy between DBMK-ELM and ELM in multiple kernel classification benchmarks, in which different kernels are generated from different channels, in order to demonstrate the ability for multisource data fusion of DBMK-ELM. In addition, we compare DBMK-ELM with the state-of-the-art multiple kernel extreme learning machine, namely, the ℓ_1-MK-ELM and the radius-incorporated MK-ELM (R-MK-ELM) [23], in classification benchmark mentioned above, since these two methods can only suit classification cases. Finally, we conduct parameter sensitivity test for DBMK-ELM.

4.1. Benchmark Data Sets. We choose 12 classification benchmark data sets including *ionosphere*, *sonar*, *wdbc*, *wpbc*,

Input:
The set of training samples, $(\mathbf{X}_{\text{trn}}, \mathbf{Y}_{\text{trn}})$;
The set of testing samples, $(\mathbf{X}_{\text{tst}}, \mathbf{Y}_{\text{tst}})$;
Output:
the output vector of DBMK-ELM, $\mathbf{o}$;
(1) Pre-generate several different positive semi-definite base kernels $K_1, \ldots, K_p$ for training samples $(\mathbf{X}_{\text{trn}}, \mathbf{Y}_{\text{trn}})$;
(2) Reconstruct new sample space for $\mathbf{K}_1, \ldots, \mathbf{K}_p$ using (9) and get new samples $(\mathbf{Z}, t)$;
(3) Learn base kernel combination coefficients $\boldsymbol{\mu}$ by (13) or (14);
(4) Construct the new optimal kernel by (15) and (16);
(5) Calculate the output $\mathbf{o}$ using (17) for each sample $(\mathbf{x}_i, y_i)$ in testing set $(\mathbf{X}_{\text{tst}}, \mathbf{Y}_{\text{tst}})$;
(6) **return o**;

Algorithm 1: Our proposed DBMK-ELM.

Table 1: Summary of the classification problems data sets.

Datasets	Number of training set	Number of testing set	Number of features	Number of classes
Ionosphere	234	117	34	2
Sonar	138	70	60	2
wdbc	379	190	30	2
wpbc	129	65	33	2
Breast	70	36	9	6
Glass	142	72	9	6
Wine	118	60	13	3

Table 2: Summary of the regression problems data sets.

Datasets	Number of training set	Number of testing set	Number of features
PM10	333	167	7
Bodyfat	168	84	14
Housing	337	169	13
Pollution	40	50	15
Spacega	666	333	6
Servo	111	56	4
Yacht	205	103	6

breast, *glass*, and *wine* from UCI Machine Learning Repository [24]. Table 1 shows the number of training samples, testing samples, features, and classes in these data sets.

The regression benchmark data sets used in this experiment are taken from UCI Machine Learning Repository [24] and Statlib [25]. These sets include *PM10* [25], *bodyfat* [25], *housing* [24], *pollution* [25], *spacega* [25], *servo* [24], and *yacht* [24]. We display the information, including the number of training samples, testing samples, and features of these sets in Table 2.

Three multiple kernel classification benchmarks from bioinformatics data sets are selected in our experiment. The first of them is the original *plant* data set of TargetP [26]. The others are *PsortPos* and *PsortNeg* [27] that both for bacterial protein locations problem. We show the number of training samples, testing samples, kernels, and classes in these data sets on Table 3.

For each data set, we randomly select two-thirds of the data samples as training data and the rest as testing data. We repeat this procedure 20 times for each data set and obtain 20 partitions of original data. All algorithms in the experiment are evaluated on each partition and the averaged results are reported for each benchmark.

4.2. Parameters Setting and Evaluation Criteria. For both classification and regression benchmark data sets, we generate 23 kernels on full feature vector, including 20 Gaussian kernels $(e^{-\gamma\|\mathbf{x}_i-\mathbf{x}_j\|^2})$ with $\gamma = \{2^{-10}, 2^{-9}, \ldots, 2^9\}$, 3 polynomial

Table 3: Summary of the multiple kernel classification benchmark data sets.

Datasets	Number of training set	Number of testing set	Number of kernels	Number of classes
Plant	627	313	69	4
PsortPos	361	180	69	4
PsortNeg	963	481	69	5

kernels of degrees 1, 2, and 3. For the kernel based ELM [8], we test all the 23 kernels generated above and display the best result of them in our experiments according to the testing accuracy. For all algorithms, the regulation parameter C is selected from $\{10^{-1}, 10^0, \ldots, 10^3\}$ via 3-fold cross validation on training data.

We select accuracy and computational efficiency as the performance evaluation criteria. The accuracy means the classification accuracy rate in testing data for classification problems or the mean square error (MSE) in testing data for regression problems. In addition, for the regression problem, sample labels have been normalized to $[-1, 1]$. For all cases, the computational efficiency is evaluated by the training time.

The reported results for each benchmark include the mean value and the standard deviation of criteria in 20 partitions. In order to measure the statistical significance for the accuracy improvement, we further use the *paired student's t-test*, in which P value means the probability that

Table 4: Classification case: classification accuracy (%). Boldface means no statistical difference from the best one (P val ≥ 0.05).

Data	DBMK-ELM	SimpleMKL [16]	ℓ_1-MK-ELM [23]	R-MK-ELM [23]	ELM [8]	UW
Ionosphere	**94.23 ± 1.44**	**94.23 ± 1.96**	**94.62 ± 1.78**	**94.32 ± 1.82**	90.38 ± 2.88	93.33 ± 1.93
	(**0.25**)	(**0.22**)	(**1.00**)	(**0.23**)	(0.00)	(0.00)
Sonar	**84.71 ± 5.35**	**84.36 ± 5.51**	**84.64 ± 5.75**	82.57 ± 5.22	**84.21 ± 4.71**	81.57 ± 5.16
	(**1.00**)	(**0.62**)	(**0.90**)	(0.00)	**(0.56)**	(0.00)
wdbc	**97.13 ± 1.32**	**97.05 ± 1.11**	**97.16 ± 1.19**	**97.08 ± 1.45**	**97.21 ± 1.26**	**97.11 ± 1.41**
	(**0.70**)	(**0.44**)	(**0.74**)	(**0.49**)	(**1.00**)	(**0.62**)
wpbc	**77.38 ± 3.57**	75.31 ± 3.55	75.23 ± 3.56	75.46 ± 3.91	**76.23 ± 3.72**	**76.31 ± 4.16**
	(**1.00**)	(0.01)	(0.00)	(0.02)	(**0.16**)	(**0.18**)
Breast	**69.03 ± 5.65**	65.56 ± 7.45	65.83 ± 8.26	67.08 ± 7.34	65.83 ± 7.55	66.94 ± 6.98
	(**1.00**)	(0.01)	(0.02)	(0.04)	(0.03)	(0.00)
Glass	**71.04 ± 4.31**	54.86 ± 4.22	68.61 ± 5.71	**70.14 ± 4.33**	67.57 ± 5.18	**69.31 ± 5.46**
	(**1.00**)	(0.00)	(0.01)	(**0.15**)	(0.01)	(**0.08**)
Wine	**97.92 ± 1.94**	**98.33 ± 1.71**	**98.17 ± 2.22**	**98.00 ± 2.45**	97.58 ± 2.13	**98.00 ± 2.27**
	(**0.20**)	(**1.00**)	(**0.61**)	(**0.41**)	(0.02)	**0.43**
AVG	**84.49**	81.39	83.47	83.52	82.72	83.22

Table 5: Classification case: classification training time (s).

Data	DBMK-ELM	SimpleMKL [16]	ℓ_1-MK-ELM [23]	R-MK-ELM [23]	ELM [8]	UW
Ionosphere	0.0210 ± 0.0016	0.8216 ± 0.4888	0.4115 ± 0.0558	1.9217 ± 0.0419	0.0009 ± 0.0001	0.0009 ± 0.0001
Sonar	0.0063 ± 0.0006	0.2919 ± 0.1673	0.2498 ± 0.0396	0.7698 ± 0.0242	0.0003 ± 0.0001	0.0003 ± 0.0000
wdbc	0.0580 ± 0.0051	4.2746 ± 4.2881	0.9743 ± 0.2876	6.4733 ± 0.2250	0.0023 ± 0.0004	0.0024 ± 0.0004
wpbc	0.0061 ± 0.0006	0.3777 ± 0.3230	0.0685 ± 0.0303	0.6542 ± 0.0202	0.0004 ± 0.0001	0.0004 ± 0.0001
Breast	0.0016 ± 0.0004	13.5007 ± 16.8849	0.1701 ± 0.0405	0.3024 ± 0.0214	0.0003 ± 0.0002	0.0002 ± 0.0001
Glass	0.0083 ± 0.0016	6.5990 ± 16.2960	0.2948 ± 0.1141	0.7010 ± 0.0631	0.0005 ± 0.0001	0.0004 ± 0.0001
Wine	0.0063 ± 0.0014	1.2676 ± 0.7903	0.2299 ± 0.0387	0.6000 ± 0.0168	0.0004 ± 0.0002	0.0004 ± 0.0001
AVG	0.0153	3.8762	0.3427	1.632	0.0007	0.0007

two compared sets come from distributions with an equal mean. Typically, if the P value less than 0.05, the compared sets are considered having statistically significant difference.

4.3. Classification Performance. The classification accuracy of different methods is shown in Table 4. The content in Table 4 has following meanings, the first part is the mean ± standard deviation and the second part is the P value calculated by the paired Student's t-test. The bold value in each cell of Table 4 represents the highest accuracy and those having no significant difference compared with the highest one. We also show the classification training time in Table 5, which presents as the mean ± standard deviation.

As we can see from Table 4, DBMK-ELM achieves the highest correct classification rate or has no significant different compared with the best one. Meanwhile, the results in Table 5 prove that the time cost of this approach is significantly lower than SimpleMKL, ℓ_1-MK-ELM, and R-MK-ELM.

4.4. Regression Performance. The regression accuracy of different methods is shown in Table 6 with the same representation of Table 4. And the regression training time is shown in Table 7.

In this case, we can see DBMK-ELM has the significant highest regression accuracy compared with other methods. From the time cost point of view, this situation is similar to the classification problem; that is, DBMK-ELM dramatically improved training time compared to SimpleMKL.

4.5. Multiple Kernel Classification Benchmark Performance. The classification accuracy for multiple kernel classification benchmarks of DBMK-ELM and other methods is shown in Table 8. And the multiple kernel classification training time is shown in Table 9. From the results, we can see DBMK-ELM significantly better than ELM. That means DBMK-ELM has the ability to perform multisource data fusion, thereby improving the performance of ELM. The DBMK-ELM is better than the state-of-the-art multiple kernel extreme learning machine in this case regarding the classification accuracy and the training time.

4.6. Parameter Sensitivity Test. In our proposed DBMK-ELM, there are two regularization parameters need to be set. In order to describe more clearly, we use $C1$ and $C2$ represents the regularization parameter in ELM training and multiple kernel learning, respectively. We choose classification data set *ionosphere* and regression data set *yacht* to test parameter sensitivity. For each data set, we set a wide range of $C1$ and

Table 6: Regression case: regression accuracy (%). Boldface means no statistical difference from the best one (P val ≥ 0.05).

Data	DBMK-ELM	SimpleMKL [16]	ELM [8]	UW
PM10	**0.3242 ± 0.0171 (1.00)**	0.3347 ± 0.0152 (0.00)	0.3305 ± 0.0172 (0.01)	**0.3256 ± 0.0167 (0.46)**
Bodyfat	**0.0559 ± 0.0227 (1.00)**	0.0898 ± 0.0264 (0.00)	0.0755 ± 0.0254 (0.00)	0.1321 ± 0.0246 (0.00)
Housing	**0.1618 ± 0.0256 (1.00)**	0.2006 ± 0.0430 (0.00)	0.1760 ± 0.0264 (0.00)	0.2149 ± 0.0323 (0.00)
Pollution	**0.2738 ± 0.0469 (1.00)**	0.3167 ± 0.0660 (0.00)	**0.2767 ± 0.0438 (0.61)**	0.3110 ± 0.0606 (0.00)
Servo	**0.1926 ± 0.0408 (1.00)**	0.2121 ± 0.0528 (0.04)	0.2046 ± 0.0396 (0.01)	0.2567 ± 0.0379 (0.00)
Spacega	**0.1415 ± 0.0073 (1.00)**	0.1735 ± 0.0136 (0.00)	0.1655 ± 0.0104 (0.00)	0.1633 ± 0.0106 (0.00)
Yacht	**0.1083 ± 0.0192 (1.00)**	0.1947 ± 0.0323 (0.00)	0.1839 ± 0.0228 (0.00)	0.2420 ± 0.0342 (0.00)
AVG	**0.1797**	0.2174	0.2018	0.2351

Table 7: Regression case: regression training time (s).

Data	DBMK-ELM	SimpleMKL [16]	ELM [8]	UW
PM10	0.0430 ± 0.0021	7.3150 ± 4.2744	0.0010 ± 0.0002	0.0010 ± 0.0002
Bodyfat	0.0088 ± 0.0009	5.4048 ± 3.9174	0.0004 ± 0.0001	0.0004 ± 0.0001
Housing	0.0451 ± 0.0028	18.3088 ± 11.7929	0.0011 ± 0.0002	0.0010 ± 0.0002
Pollution	0.0004 ± 0.0000	0.0829 ± 0.0384	0.0001 ± 0.0000	0.0001 ± 0.0000
Servo	0.0036 ± 0.0004	1.6819 ± 1.0180	0.0003 ± 0.0003	0.0002 ± 0.0000
Spacega	0.1954 ± 0.0040	42.0023 ± 54.0406	0.0058 ± 0.0008	0.0058 ± 0.0008
Yacht	0.0152 ± 0.0012	4.8831 ± 2.3245	0.0005 ± 0.0001	0.0005 ± 0.0001
AVG	0.0445	11.3827	0.0013	0.0013

Table 8: Multiple kernel classification case: classification accuracy (%). Boldface means no statistical difference from the best one (P val ≥ 0.05).

Data	DBMK-ELM	SimpleMKL [16]	ℓ_1-MK-ELM [23]	R-MK-ELM [23]	ELM [8]	UW
Plant	**91.82 ± 1.43 (1.00)**	67.38 ± 3.43 (0.00)	58.85 ± 2.96 (0.00)	85.32 ± 2.56 (0.00)	78.51 ± 2.23 (0.00)	74.79 ± 2.55 (0.00)
PsortPos	**87.92 ± 2.03 (1.00)**	80.22 ± 2.91 (0.00)	70.31 ± 3.35 (0.00)	84.14 ± 2.12 (0.00)	80.83 ± 2.45 (0.00)	81.03 ± 2.69 (0.00)
PsortNeg	**91.52 ± 0.86 (1.00)**	84.80 ± 1.74 (0.00)	73.78 ± 1.82 (0.00)	89.75 ± 1.23 (0.00)	85.87 ± 1.81 (0.00)	87.31 ± 1.42 (0.00)
AVG	**90.42**	77.47	67.64	86.40	81.74	81.04

$C2$. Specifically, we have used 10 different values of $C1$ and 10 different values of $C2$ from $\{10^{-1}, 10^0, \ldots, 10^7, 10^8\}$. For each $(C1, C2)$ pair, we repeat 20 times on each data set to get the average accuracy. The result of classification case and regression case is shown in Figures 2 and 3, respectively. As can be seen from the results, the performance of DBMK-ELM is not sensitivity while $C1$ and $C2$ vary within a wide range.

4.7. Discussion. The experimental results have illustrated that DBMK-ELM can achieve a high accuracy with a fast learning speed. However, two issues need to discussed.

(1) *Why is the learning speed of DBMK-ELM much slower than basic ELM in some cases?* The main reason is that there are substantial samples to learn in the K-space, which is constructed in multiple kernel learning step. Specifically, if there are n original samples, there will be n^2 corresponding new samples. Therefore, the training time difference between DBMK-ELM and basic ELM will be magnified with the training samples increasing. It may be possible to reduce the training time gap between DBMK-ELM and basic ELM if we use sampling techiques in the K-space.

(2) *In which cases should we use DBMK-ELM?* DBMK-ELM can obtain more accurate results and a faster learning speed compared with traditional multiple kernel learning method, SimpleMKL. Despite the fact that it is much better than other multiple kernel learning methods, DBMK-ELM has more time cost compared with basic ELM method. In this way, a trade-off between accuracy and time cost is needed. The experimental results show that DBMK-ELM significantly improves testing accuracy compared with basic ELM in regression and multisource data fusion problems in most cases. But in the classification case, where kernels are generated from one data source, DBMK-ELM has no significant difference compared with basic ELM in testing accuracy. Therefore, a preferable choice is to apply DBMK-ELM in regression and multisource data fusion problems and use basic ELM in single data source generated kernel classification problems.

Table 9: Multiple kernel classification case: classification training time (s).

Data	DBMK-ELM	SimpleMKL [16]	ℓ_1-MK-ELM [23]	R-MK-ELM [23]	ELM [8]	UW
Plant	0.6545 ± 0.0272	9.1479 ± 0.6296	4.5658 ± 0.5378	7.8680 ± 0.6125	0.0064 ± 0.0006	0.0065 ± 0.0006
PsortPos	0.2062 ± 0.0124	2.3172 ± 0.2031	0.8097 ± 0.0893	2.4892 ± 0.3435	0.0019 ± 0.0005	0.0017 ± 0.0003
PsortNeg	1.5343 ± 0.1060	25.5086 ± 1.1290	11.9782 ± 0.7766	24.7445 ± 1.0424	0.0187 ± 0.0009	0.0187 ± 0.0011
AVG	0.7983	12.3246	5.7846	11.7006	0.0090	0.0090

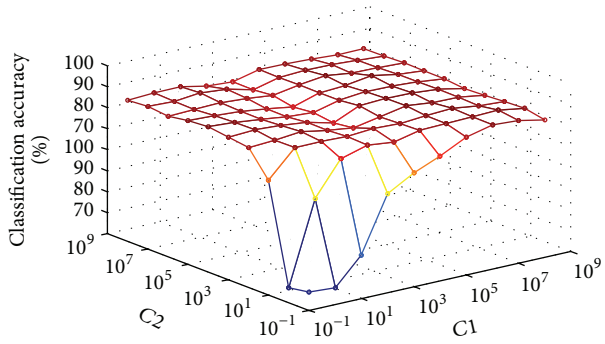

Figure 2: *Classification case:* performances of DBMK-ELM with different parameters on the *ionosphere* data set.

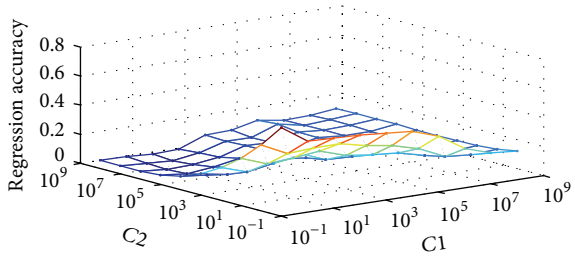

Figure 3: *Regression case:* performances of DBMK-ELM with different parameters on the *yacht* data set.

5. Conclusion

In this paper, we have proposed DBMK-ELM, a new multiple kernel based ELM, to extend the basic kernel based ELM. The proposed multiple kernel learning method can unify classification and regression problems. Moreover, DBMK-ELM is able to learn from multiple kernels at an extremely fast speed. Experimental results show that DBMK-ELM achieves a significant performance enhancement, in terms of the accuracy and the time cost in both classification and regression problems. In future, we will consider how to define a better distance among different classes and how to extend DBMK-ELM to the semisupervised learning problem.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Project nos. 60970034, 61170287, and 61232016), the National Basic Research Program of China (973) under Grant no. 2014CB340303, and the Hunan Provincial Science and Technology Planning Project of China (Project no. 2012FJ4269).

References

[1] J. Yin, E. Zhu, X. Yang, G. Zhang, and C. Hu, "Two steps for fingerprint segmentation," *Image and Vision Computing*, vol. 25, no. 9, pp. 1391–1403, 2007.

[2] C. Zhu, J. Yin, and Q. Li, "A stock decision support system based on DBNs," *Journal of Computational Information Systems*, vol. 10, no. 2, pp. 883–893, 2014.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[4] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 2000.

[5] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 2, pp. 985–990, July 2004.

[6] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.

[7] Y. Wang, F. Cao, and Y. Yuan, "A study on effectiveness of extreme learning machine," *Neurocomputing*, vol. 74, no. 16, pp. 2483–2490, 2011.

[8] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[9] Y. Yuan, Y. Wang, and F. Cao, "Optimization approximation solution for regression problem based on extreme learning machine," *Neurocomputing*, vol. 74, no. 16, pp. 2475–2482, 2011.

[10] J. Lin, J. Yin, Z. Cai, Q. Liu, K. Li, and V. Leung, "A secure and practical mechanism of outsourcing extreme learning machine in cloud computing," *IEEE Intelligent Systems*, vol. 28, no. 6, pp. 35–38, 2013.

[11] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[12] J. Cao, Z. Lin, G.-B. Huang, and N. Liu, "Voting based extreme learning machine," *Information Sciences*, vol. 185, no. 1, pp. 66–77, 2012.

[13] W. Zong, G.-B. Huang, and Y. Chen, "Weighted extreme learning machine for imbalance learning," *Neurocomputing*, vol. 101, pp. 229–242, 2013.

[14] Z. Bai, G. B. Huang, D. Wang, H. Wang, and M. B. Westover, "Sparse extreme learning machine for classification," *IEEE Transactions on Cybernetics*, vol. 44, no. 10, pp. 1858–1870, 2014.

[15] 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.

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

[17] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, "L_p-norm multiple kernel learning," *The Journal of Machine Learning Research*, vol. 12, pp. 953–997, 2011.

[18] X. Liu, L. Wang, J. Yin, and L. Liu, "Incorporation of radius-info can be simple with SimpleMKL," *Neurocomputing*, vol. 89, pp. 30–38, 2012.

[19] X. Liu, L. Wang, J. Zhang, and J. Yin, "Sample-adaptive multiple kernel learning," AAAI, 2014.

[20] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola, "On kernel-target alignment," in *Advances in Neural Information Processing Systems*, vol. 14, pp. 367–373, MIT Press, 2002, http://papers.nips.cc/paper/1946-on-kernel-target-alignment.pdf.

[21] A. Kumar, A. Niculescu-Mizil, K. Kavukcoglu, and H. Daumé, "A binary classification framework for two-stage multiple kernel learning," in *Proceedings of the 29th International Conference on Machine Learning (ICML '12)*, pp. 1295–1302, July 2012.

[22] L.-J. Su and M. Yao, "Extreme learning machine with multiple kernels," in *Proceedings of the 10th IEEE International Conference on Control and Automation (ICCA '13)*, pp. 424–429, June 2013.

[23] X. Liu, L. Wang, G. B. Huang, J. Zhang, and J. Yin, "Multiple kernel extreme learning machine," *Neurocomputing*, vol. 149, part A, pp. 253–264, 2015.

[24] K. Bache and M. Lichman, UCI machine learning repository, 2013, http://archive.ics.uci.edu/ml.

[25] http://lib.stat.cmu.edu/datasets/.

[26] O. Emanuelsson, H. Nielsen, S. Brunak, and G. von Heijne, "Predicting subcellular localization of proteins based on their N-terminal amino acid sequence," *Journal of Molecular Biology*, vol. 300, no. 4, pp. 1005–1016, 2000.

[27] J. L. Gardy, M. R. Laird, F. Chen et al., "PSORTb v.2.0: expanded prediction of bacterial protein subcellular localization and insights gained from comparative proteome analysis," *Bioinformatics*, vol. 21, no. 5, pp. 617–623, 2005.