Selecting the Optimal Hidden Layer of Extreme Learning Machine Using Multiple Kernel Learning

Wentao Zhao, Pan Li*, Qiang Liu, Dan Liu and Xinwang Liu
College of Computer, National University of Defense Technology
Changsha, Hunan, China 410073
[e-mail: lipan16@nudt.edu.cn]
*Corresponding authors: Pan Li

Received March 28, 2018; revised May 29, 2018; accepted July 20, 2018; published December 31, 2018

Abstract

Extreme learning machine (ELM) is emerging as a powerful machine learning method in a variety of application scenarios due to its promising advantages of high accuracy, fast learning speed and easy of implementation. However, how to select the optimal hidden layer of ELM is still an open question in the ELM community. Basically, the number of hidden layer nodes is a sensitive hyperparameter that significantly affects the performance of ELM. To address this challenging problem, we propose to adopt multiple kernel learning (MKL) to design a multi-hidden-layer-kernel ELM (MHLK-ELM). Specifically, we first integrate kernel functions with random feature mapping of ELM to design a hidden-layer-kernel ELM (HLK-ELM), which serves as the base of MHLK-ELM. Then, we utilize the MKL method to propose two versions of MHLK-ELMs, called sparse and non-sparse MHLK-ELMs. Both two types of MHLK-ELMs can effectively find out the optimal linear combination of multiple HLK-ELMs for different classification and regression problems. Experimental results on seven data sets, among which three data sets are relevant to classification and four ones are relevant to regression, demonstrate that the proposed MHLK-ELM achieves superior performance compared with conventional ELM and basic HLK-ELM.

Keywords: Extreme learning machine, multiple kernel learning, hidden layer kernel, optimization

The work is supported by National Natural Science Foundation of China under Grant No. 61702539 and Hunan Provincial Natural Science Foundation of China under Grant No. 2018JJ3611.

1. Introduction

Extreme learning machine(ELM), which is a fast learning method to train single hidden layer feedback neural networks(SLFNs)[1], has become a prevailing research topic in the past decades[2]. Considering that the conventional back propagation during training general neural networks suffers from slow training speed and local minima, ELM addresses these two disadvantages by randomizing weights and bias of hidden layer nodes. Then, ELM can quickly obtain the optimal weights by analytically solving optimization problems rather than iterative methods[3]. Besides the featured characteristic of fast training speed, ELM is also easy to implementation in both classification and regression tasks[4]. With above advantages, ELM has been successfully applied in many application scenarios[4,5,6].

However, how to select an optimal hidden layer in ELM is still an open question in the ELM community. Different numbers of hidden layer nodes induce different architectures of neural networks, resulting in varied capability of random feature representing. Hence, the determination of the number of hidden layer nodes significantly affects the overall performance of ELM. To address such technical challenge, there are two main ways in the state-of-the-art works. The first way is to find the concrete optimal number of hidden layer nodes. For example, [7] proposed evolutionary ELM(E-ELM) to find the optimal number of hidden layer nodes with evolutionary algorithms. Based on E-ELM, [8] further proposed self-adaptive evolutionary ELM(SaE-ELM) to obtain the optimal number of hidden layer nodes. Besides, incremental learning[9], self-adaptive[10] and swarm optimization[11] were used to solve this problem. Another way is to ensemble multiple approaches, which can both boost the performance and stability of machine learning[12]. [13] proposed a kind of adaptive ensemble model of ELMs to determine the optimal combination of various ELMs. Besides, multiple kernel learning was also applied to ELM, such as multiple kernel ELM (MK-ELM)[14] and multiple-kernel-learning-based ELM[15]. However, both these two multiple kernel learning methods focused on the selection of different types of kernels rather than the determination of hidden layer.

Inspired by MK-ELM[14], we adopt multiple kernel learning (MKL) to design a multi-hidden-layer-kernel ELM (MHLK-ELM) in this paper. MHLK-ELM can effectively find the optimal linear combination of base hidden-layer kernels (HLKs) with different numbers of hidden layer nodes. Compared to the previous MK-ELM, MHLK-ELM makes two significant improvements. First, the base hidden-layer kernels of MHLK-ELM can effectively correlate different hidden layer nodes information. Second, the proposed method aims to obtain the optimal combination of different numbers of hidden layer nodes specified in multiple base hidden-layer kernels rather than the optimal combination of different kernels. Then, we utilize the MKL method to propose two versions of MHLK-ELMs, called sparse and non-sparse MHLK-ELM. Both two types of MHLK-ELMs can effectively find out the optimal linear combination of multiple HLK-ELMs for different classification and regression problems. Experimental results on seven classification and regression data sets demonstrate that the proposed MHLK-ELM achieves superior performance compared with MK-ELM, conventional ELM and basic HLK-ELM in terms of accuracy and stability. The contributions of this paper are detailed as follow:

(1) We propose a novel multiple kernel framework of ELM. Given a specific classification or regression task, the proposed framework can effectively obtain the optimal linear combination of base hidden-layer kernels.

(2) According to the diverse norm constrict on combination coefficients γ, we present sparse and non-sparse MHLK-ELMs, which adopt l_1 -norm and l_q -norm $(q>1)$ constraints on γ, respectively.

The remaining of this paper is organized as follows. Section 2 reviews the background of extreme learning machine, extreme learning machine with kernel and multiple kernel learning. Then, Section 3 introduces the proposed MHLK-ELMs including sparse MHLK-ELM and non-sparse MHLK-ELM. After that, Section 4 presents the comparative experimental results to validate the effectiveness of the proposed method. Finally, Section 5 concludes this paper.

2. Related Work

In this section, we introduce the related knowledge about extreme learning machine, extreme learning machine with kernel and multiple kernel learning.

2.1 Extreme learning machine

Extreme learning machine(ELM) was first proposed by G. Huang[16]. Similar to the framework of SLFN, ELM is also a simple neural network with single hidden layer. The difference between them is that the hidden layer weights and bias of ELM are randomized and need not to tune. The unified formulation of ELM is as follow.

$$f_{ELM}(x) = h(x) * \beta \tag{1}$$

where $h(x) = [h_1(x), h_2(x), ..., h_L(x)]$ is the hidden layer output respect to input sample $x \epsilon \Re^d$. Specifically, $h(x)$ represents random feature mapping from input sample space $\Re^d$ to randomized feature space $\Re^L$. Besides, $\beta \in \Re^{|\phi(\cdot)| \times T}$ is the optimal output weights corresponding to randomized hidden layer weights. And the formulation of $h_i(x)$ is:

$$h_i(x) = g(\omega_i x + b_i) \tag{2}$$

where $g(\cdot)$ represents the active function, ω_i and b_i are the weight and bias of i th hidden layer output, respectively.

For N training samples, T is a vector which represents their corresponding ground truth labels. The hidden layer output H can be represented as follow:

$$H = \begin{pmatrix} h^1(x_1) & \cdots & h^L(x_1) \\ \vdots & \ddots & \vdots \\ h^1(x_N) & \cdots & h^L(x_N) \end{pmatrix} \tag{3}$$

Besides, the optimal object of ELM is to minimize the training errors and the norm of output weights, which can improve the training accuracy and generalization performance of ELM, respectively. Here we mainly introduce the theory of ELM for m-class classification, and its objective function can be formulated as follow:

$$min_{\beta,\xi} \quad \frac{1}{2} \| \beta \|_F^2 + \frac{C}{2} \sum_{i=1}^{N} | | \xi_i | |^2 \qquad s.t. \quad h(x_i)\beta = t_i^T - \xi_i^T, \quad \forall i \tag{4}$$

where $h(\cdot)$ is the randomized hidden layer function, β refers to the output weights. Besides, t_i represents the class label of x_i, which has some difference between classification and regression problem. In classification problem with $m(m>1)$ classes, t_i is a $m-D$ vector

generated by m output neurons, which represents m class labels. But in regression problem, t_i is the predict value outputted by a single output neuron towards input sample x_i.

Based on the randomized hidden layer weights and the object function, the optimal solution can be obtained by solving the quadratic programming(QP) problem. The optimal output weights $\hat{\beta}$ is as follow:

$$\hat{\beta} = H^T(\frac{I}{C} + HH^T)^{-1}T \tag{5}$$

Finally, the output function of ELM is:

$$f_{ELM}(x) = h(x)\hat{\beta} = h(x)H^T(\frac{I}{C} + HH^T)^{-1}T \tag{6}$$

2.2 Extreme learning machine with kernel

Inspired by the kernel method in support vector machine(SVM), [17,18] first bring the kernel method to ELM. The kernel function for ELM is as follow:

$$\Omega_{ELM} = HH^T \tag{7}$$

Specifically,

$$\Omega_{ELM(i,j)} = h(x_i)\cdot h(x_j) = K(x_i, x_j) \tag{8}$$

where, kernel function $K(u,v)$ is given, while feature mapping $h(\cdot)$ need not to be known to users. So the decision function of kernel ELM is constructed:

$$f_{K-ELM}(x) = h(x)\beta = h(x)H^T(\frac{I}{C} + \Omega_{ELM})^{-1}T = \begin{pmatrix} K(x, x_1) \\ \vdots \\ K(x, x_N) \end{pmatrix}(\frac{I}{C} + \Omega_{ELM})^{-1}T \tag{9}$$

where $x_1 \cdots x_N$ are the training samples, and x refers to the testing sample.

2.3 Multiple kernel learning

Multiple kernel learning(MKL)[19] is one of the most popular method to find the optimal linear combination of different kernels, and it has been applied to many machine learning methods, such as SVM[20], fisher discriminant analysis(FDA)[21] and ELM[14,15]. In MKL[22], the optimal kernel is assumed to be the linear combination of a group of base kernels, so the optimal linear combination $k(\cdot,\cdot,\gamma)$ is

$$k(\cdot,\cdot,\gamma) = \sum_{p=1}^{k}\gamma_p k_p(\cdot,\cdot), \qquad \sum_{p=1}^{k}\gamma_p = 1 \tag{10}$$

where $k_p(\cdot,\cdot)(p = 1,\cdots,k)$ is the pth base kernel, and $\gamma_p(p = 1,\cdots,k)$ are their corresponding combination coefficients. And E.q(10) can be equivalently formulated as E.q(11):

$$\phi(\cdot;\gamma) = [\sqrt{\gamma_1}\phi_1(\cdot), \sqrt{\gamma_2}\phi_2(\cdot), \cdots, \sqrt{\gamma_k}\phi_k(\cdot)] \tag{11}$$

where $\phi_p(\cdot)(p = 1,...,k)$ is the pth feature mapping corresponding to kernel $k_p(\cdot,\cdot)$, and $\phi(\cdot,\gamma)$ represents the feature mapping respecting to $k(\cdot,\cdot,\gamma)$.

According to the different constrict on γ, there are two different linear combination results. E.q(10) described one situation that γ is constricted with l_1 norm, which causes sparse combination of base kernels. Besides, we use the constrict on γ with $l_q(q > 1)$ norm, which

can cause the non-sparse combination of base kernels. In Section 3, we will detail the concrete sparse and non-sparse multiple kernel learning algorithms for ELM.

3. The Proposed Multiple Hidden-Layer-Kernel ELM

In this section, we will present the details of the proposed MHLK-ELM. First, we propose a hidden-layer-kernel ELM(HLK-ELM), which integrate kernel functions[23] with random feature mapping of ELM. Then, we introduce two types of linear combinations of base HLK-ELMs, named sparse MHLK-ELM and non-sparse MHLK-ELM, based on different constricts towards combination coefficients γ.

3.1 Hidden-Layer-Kernel ELM

In this part, we propose a new learning method called HLK-ELM, which combines the advantages of kernel functions and random feature mapping. Typically, HLK-ELM consists of two steps as follows: First, according to E.q(3), we generate random feature mapping $H(x)\epsilon\Re^L$ regarding to different numbers of hidden layer nodes. Then, we use different kernel types $K(H(x_i),H(x_j))$ to get a kernel matrix Ω_H.

$$\Omega_H=\begin{pmatrix} K(H(x_1),H(x_1)) & \cdots & K(H(x_1),H(x_N)) \\ \vdots & \ddots & \vdots \\ K(H(x_N),H(x_1)) & \cdots & K(H(x_N),H(x_N)) \end{pmatrix} \tag{12}$$

where $x_i(i=1,...N)$ represents the i th training sample, and we term Ω_H as a hidden-layer kernel(HLK). Similar to E.q(10) in the kernel ELM, we can solve the optimal $\hat{\beta}_H$ corresponding to Ω_H. Accordingly, we name an ELM with a HLK as HLK-ELM.

By combining random feature mapping and kernel functions, HLK-ELM can obtain more abundant expression of initial samples. As for the specific kernel type, e.g, Gaussian kernel, the random feature mapping provides a simple but fast way to generate a suitable feature space according to the different numbers of hidden layer nodes. Hence, we propose a multiple kernel framework to find the optimal linear combination of base hidden-layer kernels.

3.2 Sparse MHLK-ELM

To obtain an efficient HLK-ELM with different HLKs, we extend MK-ELM proposed in [14] and propose a new method called MHLK-ELM, which can effective obtain the optimal linear combination of HLK-ELMs with various numbers of hidden layer nodes. The objective function of MHLK-ELM is formulated as:

$$\begin{aligned} &\min_{\gamma}\min_{\beta,\xi} \quad \frac{1}{2}\|\beta\|_F^2+\frac{C}{2}\sum_{i=1}^{N}||\xi_i\|^2 \\ &s.t. \quad \beta^T\phi(x_i;\gamma)=t_i^T-\xi_i^T, \quad \forall i, \quad \sum_{p=1}^{k}\gamma_p=1,\gamma_p\geq 0 \end{aligned} \tag{13}$$

Where $\beta=[\beta_1,...,\beta_p]\epsilon\Re^{(|\phi_1(\cdot)|+...+|\phi_k(\cdot)|)\times T}$, and $\beta_p\epsilon\Re^{|\phi_P(\cdot)|\times T}$ is the p th optimal solution corresponding to the p th base HLK $\phi(\cdot)$. Similar to E.q(4), t_i and ξ_i represent the groundtruth label and training error respecting to x_i, respectively.

As shown in E.q(13), the object function is to optimize the parameter of β and kernel combination coefficients γ jointly. In order to solve this object function, we first substitute $\phi(x_i;\gamma)$ in E.q(13) using E.q(11). Thus, E.q(13) can be transformed to E.q(14) equivalently.

$$\begin{aligned} &\min_{\gamma}\min_{\beta,\xi} \quad \frac{1}{2}\|\beta\|_F^2 + \frac{C}{2}\sum_{i=1}^{N}\|\xi_i\|^2 \\ &s.t. \quad \sum_{p=1}^{k}\sqrt{\gamma_p}\beta_p^T\phi_p(x_i) = t_i^T - \xi_i^T, \quad \forall i, \quad \sum_{p=1}^{k}\gamma_p = 1, \gamma_p \geq 0 \end{aligned} \tag{14}$$

Then, we define $\hat{\beta}$ as follow:

$$\hat{\beta} = [\sqrt{\gamma_1}\beta_1, \sqrt{\gamma_2}\beta_2, ..., \sqrt{\gamma_k}\beta_k] \tag{15}$$

According to E.q(15), the objective function can be equivalently rewritten as:

$$\begin{aligned} &\min_{\gamma}\min_{\beta,\xi} \quad \frac{1}{2}\sum_{p=1}^{k}\frac{\|\hat{\beta}_p\|_F^2}{\gamma_p} + \frac{C}{2}\sum_{i=1}^{N}\|\xi_i\|^2 \\ &s.t. \quad \sum_{p=1}^{k}\hat{\beta}_{\mathrm{p}}{}^T\phi_p(x_i) = t_i^T - \xi_i^T, \quad \forall i, \quad \sum_{p=1}^{k}\gamma_p = 1, \gamma_p \geq 0 \end{aligned} \tag{16}$$

Note that E.q (16) is a jointly-convex optimization problem. Hence, we can solve this optimization problem through following steps:

(1) Rewritting E.q(16) to a dual optimization problem: based on the KKT theorem, E.q(16) can be equivalently rewritten as:

$$L(\hat{\beta},\xi,\gamma) = \frac{1}{2}\sum_{p=1}^{k}\frac{\left\|\hat{\beta}_p\right\|^2}{\gamma} + \frac{C}{2}\sum_{i=1}^{N}\left\|\xi_i\right\|^2 - \sum_{t=1}^{T}\sum_{i=1}^{N}\alpha_i t(\sum_{p=1}^{k}\hat{\beta}_{\mathrm{p}}{}^T\phi_p(x_i) - t_{\mathrm{i}}^T + \xi_i^T) + \tau(\sum_{p=1}^{k}\gamma_p - 1) \tag{17}$$

where $\alpha_i (i = 1,...,N)$ and τ are Lagrangian multiples. Besides, we omit the non-negative constrict towards γ_p in E.q(17) because γ_p keeps non-negative in the iterating progress.

(2) Solving the (KKT) optimality conditions of E.q(17): according to the KKT conditions, we can get the following formulations by taking the derivatives of Eq.(17) respecting to β, γ and ξ.

$$\frac{\partial L(\hat{\beta},\xi,\gamma)}{\partial \hat{\beta}_p} = 0 \rightarrow \hat{\beta}_p = \gamma_p\sum_{t=1}^{T}\sum_{i=1}^{N}\alpha_{it}\phi_p(x_i), \quad \forall p \tag{18}$$

$$\frac{\partial L(\hat{\beta},\xi,\gamma)}{\partial \xi_{ti}} = 0 \rightarrow \xi_{ti} = \frac{\alpha_{ti}}{C}, \quad \forall t \quad \forall i \tag{19}$$

$$\frac{\partial L(\hat{\beta},\xi,\gamma)}{\partial \gamma_p} = 0 \rightarrow -\frac{1}{2}\frac{\left\|\hat{\beta}_P\right\|_F^2}{\gamma_p^2} + \tau = 0, \quad p = 1,...,k \tag{20}$$

Then, letting two Lagrangian multiple items equal to zero, we can obtain E.q(21) and E.q(22) where α_i and τ are non-zero.

$$\sum_{p=1}^{k}\hat{\beta}_{\mathrm{p}}{}^T\phi_p(x_i) = t_i^T - \xi_i^T, \quad \forall i \tag{21}$$

$$\sum_{p=1}^{k}\gamma_p = 1 \tag{22}$$

(3) Calculating the formulation of parameters γ and α. By incorporating E.q(17)-(22), we can obtain the following formulation:

$$(K(\cdot,\cdot;\gamma)+\frac{I}{C})\alpha = Y^T \tag{23}$$

where $K(\cdot,\cdot;\gamma) = \phi(x_i,\gamma)^T\phi(x_j,\gamma) = \sum_{p=1}^{k}\gamma_p K_p(x_i,x_j)$. $Y=[y_1,...,y_N]\epsilon\Re^{T\times N}$ is the corresponding label matrix. Besides, α in E.q(23) is the structural parameter of ELM, which can be determined by:

$$\alpha = (K(\cdot,\cdot;\gamma)+\frac{I}{C})^{-1}Y^T \tag{24}$$

Similarly, according to E.q(20), we can obtain:

$$\gamma_p = \frac{\| \hat{\beta}_p \|_F}{\sqrt{\tau}}, \quad p=1,...,k \tag{25}$$

Combining E.q(25) and $\sum_{p=1}^{k}\gamma_p = 1$, we can know $\sqrt{\tau} = \sum_{p=1}^{k}\| \hat{\beta}_p \|$. After that, we can get the final formulation of γ_p:

$$\gamma_p = \frac{\| \hat{\beta}_p \|_F}{\sum_{p=1}^{k}\| \hat{\beta}_p \|_F} \tag{26}$$

E.q(26) is the formulation applied in updating progress, where the initial $\gamma_p(p=1...k)=1/k$, and $\| \hat{\beta}_p \|_F$ can be calculated as follow:

$$\| \hat{\beta}_p \|_F = \gamma_p\sqrt{\sum_{s,t=1}^{T}\sum_{i,j=1}^{N}\alpha_{it}\alpha_{js}K_p(x_i,x_j)} \tag{27}$$

From E.q(26) and E.q(27), it is obvious that γ_p is keeping non-negative in updating progress because $\| \hat{\beta}_p \|_F$ and initial γ_p are both non-negative.

Based on the above formulations, the target optimization problem can be solved by alternatively optimizing α and γ_p. Moreover, the l_1 norm constrict on γ_p induces sparse combination coefficients. Accordingly, we name this method as the sparse MHLK-ELM. The pseudo code of the sparse MHLK-ELM is given in Algorithm 1.

In Algorithm 1, N_L and K_{Type} refer to the number of hidden layer nodes and the kernel type of base HLKs, respectively.

3.3 Non-Sparse MHLK-ELM

In some literature[24,25], non-sparse MKL algorithms generally performed better than sparse ones. Compared to sparse MKL algorithms, non-sparse MKL algorithms[26] can keep the information of every base kernel with non-sparse constricts on γ_p. Until now, non-sparse MKL algorithms have been successfully applied to many machine learning methods[21,27].

Algorithm 1 Sparse MHLK-ELM

Input: N_L, K_{Type}, C, and Y

Output: α and γ

1. Generate $K_{HLK}(.;.;\gamma)$ based on N_L and K_{Type}
2. Initialize the $\gamma = \gamma_0$ and $t = t_0$
3. **repeat**
4. Calculating $K(.;.;\gamma) = \sum_{p=1}^{m} \gamma_p^t K_p$
5. Updating α^{t+1} by E.q(24)
6. Updating γ^{t+1} according to E.q(26)
7. $t = t+1$
8. **until** $\max |\gamma^t - \gamma^{t-1}| \leq 1e^{-4}$
9. $\alpha = \alpha^t$, $\gamma = \gamma^t$

Typically, the parameter γ_p in the non-sparse MHLK-ELM is constricted by $l_q(q>1)$ norms[22,28,29]. Hence, the objective function of the non-sparse MHLK-ELM can be formulated as

$$\begin{aligned} &\min_{\gamma} \min_{\beta,\xi} \quad \frac{1}{2}\sum_{p=1}^{k} \frac{\| \hat{\beta}_p \|_F^2}{\gamma_p} + \frac{C}{2}\sum_{i=1}^{N} | | \xi_i \|^2 \\ &s.t. \quad \sum_{p=1}^{k} \hat{\beta}_{\mathrm{p}}{}^T \phi_p(x_i) = t_i^T - \xi_i^T, \quad \forall i, \quad \sum_{p=1}^{k} \gamma_p^q = 1, \gamma_p \geq 0 \end{aligned} \tag{28}$$

According to E.q(25), we have

$$-\frac{1}{2}\frac{\left\|\hat{\beta}_P\right\|_F^2}{\gamma_p^2} + q\gamma_p^{q-1}\tau = 0, \quad p = 1,...,k \tag{29}$$

where $\sum_{p=1}^{k} \gamma_p^q = 1$. Thus, γ_p can be calculated by

$$\gamma_p = \frac{\| \hat{\beta}_p \|_F^{2/q+1}}{(2q\tau)^{1/(q+1)}} \tag{30}$$

Then, we can obtain

$$(2q\tau)^{1/(q+1)} = (\sum_{p=1}^{k} | | \hat{\beta}_p \|_F^{2q/q+1})^{1/q} \tag{31}$$

Based on equations (30) and (31), γ_p is determined by

$$\gamma_p = \frac{\| \hat{\beta}_p \|_F^{2/q+1}}{(\sum_{p=1}^{k} | | \hat{\beta}_p \|_F^{2q/q+1})^{1/q}} \tag{32}$$

Algorithm 2 Non-sparse MHLK-ELM

Input: N_L, K_{Type}, q, C, and Y

Output: α and γ

Generate $K_{HLK}(.;.;\gamma)$ based on N_L and K_{Type}

Initialize the $\gamma = \gamma_0$ and $t = t_0$

repeat

  Calculating $K(.;.;\gamma) = \sum_{p=1}^{m} \gamma_p^t K_p$

  Updating α^{t+1} by E.q(24)

  Updating γ^{t+1} according to E.q(32)

  $t = t + 1$

until $\max | \gamma^t - \gamma^{t-1} | \leq 1e^{-4}$

$\alpha = \alpha^t$, $\gamma = \gamma^t$

4. Experiment Results and Analysis

4.1 Experimental Setup

To demonstrate the advantages of the proposed method, we carry out two groups of comparative experiments as follows: (1) Since the goal of the proposed method is to find the optimal linear combination of base hidden-layer kernels regarding to different numbers of hidden layer nodes, we compare the proposed method with base ELMs and HLK-ELMs. Specifically, we select the best performance of base ELMs and HLK-ELMs, which named as best ELM and best MHLK-ELM, respectively. Besides, we show mean ELM and mean HLK-ELM by computing the mean performance of base ELMs and HLK-ELMs. (2) To further show the superior performance of the proposed method, we compare it with MK-ELM, which is the typical method of finding the optimal linear combination of base kernel ELMs.

Furthermore, we experiment in both classification and regression problems to verify the scalability of the proposed method. Specifically, the regularization parameter C is selected from $[10^{-3}, 10^{-2}, \cdots, 10^{4}]$ by 5-fold cross validation on the training data sets for five times. Regarding kernel matrices used in the base HLK-ELM method, we use four different kernels as follows:

(1) Gaussian kernel

$$k(x_i, x_j) = exp(- \| x_i - x_j \|^2 / \sigma), \tag{33}$$

(2) Laplacian kernel

$$k(x_i, x_j) = exp(- \| x_i - x_j \| / \sqrt{\sigma}), \tag{34}$$

(3) Inverse square distance kernel

$$k(x_i, x_j) = 1 / ((\| x_i - x_j \|^2 / \sigma) + 1), \tag{35}$$

(4) Inverse distance kernel

$$k(x_i, x_j) = 1 / ((\| x_i - x_j \| / \sqrt{\sigma}) + 1), \tag{36}$$

where σ is kernel parameter. For each base HLK-ELM of MHLK-ELM and base kernel ELM of MK-ELM, We set nine kernel parameters $2^t \sigma_0 (t \in \{-4, -3, -2, -1, 0, 1, 2, 3, 4\})$, where

σ_0 is set to be the average pairwise Euclidean distance among samples. Moreover, the numbers of hidden layer nodes used in the base ELMs and HLK-ELMs are selected from 200 to 800 with a step size 50 in classification problems. Similarly, the numbers of hidden layer nodes used in the base ELMs and HLK-ELMs are selected from 10 to 200 with a step size 10 in regression problems.

4.2 Data Sets

To compare the performance of different methods, we carry out comparative experiments using a variety of data sets including classification and regression data sets. Furthermore, we ran 10 times of independent experiments on each data set to have the average results with respect to different methods. In each experiment trail, 60% of a data set is selected as the training data set, otherwise as the testing data set.

(1) Classification data sets

We select three classification data sets from UCI[30] with different numbers of class labels, feature dimensions and data sizes. The detailed information of these data sets are showed in **Table 1**.

Table 1. Summary of classification data sets

Data sets	# Training	#Testing	#Feature dimension	#Classes
Glass[1]	128	86	9	6
Ecoli[2]	202	134	7	8
Vowel[3]	528	462	10	11

(2) Regression data sets

For evaluating the performance of the proposed method addressing regression problems, we adopt four data sets from UCI[30] and LIACC[31] for experiments. Similar to the classification data sets, these data sets are also different in both data size and feature dimension. **Table 2** shows the detailed information about the four data sets.

Table 2. Summary of regression data sets

Data sets	# Training	#Testing	#Feature dimension
Auto price[4]	95	64	16
Machine Cpu[4]	125	84	10
Housing[5]	304	202	14
Stock[4]	569	380	9

Note that the output of each data record in regression data sets is normalized to a range of [-1,1].

[1] http://archive.ics.uci.edu/ml/datasets/Glass+Identification
[2] http://archive.ics.uci.edu/ml/datasets/Ecoli
[3] http://archive.ics. uci.edu/ml/datasets/Connectionist+Bench+%28Vowel+Recognition+-+Deterding+Data%29
[4] http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html
[5] http://mlearn.ics.uci.edu/databases/housing/

4.3 Experimental Results with Sparse MHLK-ELM

4.3.1 Performance Comparison on Classification Problems

Table 3. Performance comparison of different methods on Glass, Ecoli and Vowel data sets. The two rows of each cell represent ACC ± standard derivation and mAP ± standard derivation, respectively. The ACC and mAP on Vowel is equal.

Data sets	Kernel type		The proposed MHLK-ELM	MK-ELM	Best HLK-ELM	Best ELM	Mean HLK-ELM	Mean ELM
Glass	Gaussian	ACC	69.07 ± 0.18	67.67 ± 0.07	68.14 ± 0.12	68.49 ± 0.09	67.53	66.97
		mAP	57.89 ± 0.27	51.37 ± 0.19	56.96 ± 0.22	57.00 ± 0.27	55.91	53.66
	Laplace	ACC	70.70 ± 0.15	60.00 ± 0.24	71.40 ± 0.30	68.02 ± 0.26	70.23	66.78
		mAP	58.38 ± 0.39	42.32 ± 0.07	59.85 ± 0.31	55.89 ± 0.33	57.91	53.65
	Inverse Square	ACC	69.07 ± 0.33	71.16 ± 0.07	68.37 ± 0.29	67.91± 0.12	67.57	66.90
		mAP	59.33 ± 0.52	59.60 ± 0.18	57.44 ± 0.16	55.49 ± 0.17	56.74	53.61
	Inverse	ACC	72.09 ± 0.11	69.42 ± 0.10	71.74 ± 0.11	68.49 ± 0.13	71.01	67.04
		mAP	59.43 ± 0.28	51.04 ± 0.31	59.82 ± 0.21	55.38 ± 0.31	58.90	53.69
Ecoli	Gaussian	ACC	86.81± 0.06	86.00 ± 0.04	86.59 ± 0.07	85.41± 0.09	86.35	85.13
		mAP	69.04 ± 0.63	68.79 ± 0.48	69.15 ± 0.59	65.97 ± 0.53	68.85	65.64
	Laplace	ACC	86.00 ± 0.05	85.33 ± 0.09	84.74 ± 0.07	85.41± 0.09	84.04	85.23
		mAP	67.11± 0.50	67.16 ± 0.54	67.15 ± 0.50	66.19 ± 0.55	66.04	65.83
	Inverse Square	ACC	86.96 ± 0.06	86.15 ± 0.04	86.30 ± 0.07	85.41± 0.10	85.89	85.15
		mAP	68.93 ± 0.52	68.60 ± 0.47	68.89 ± 0.57	66.33 ± 0.51	68.52	65.71
	Inverse	ACC	86.52 ± 0.07	86.15 ± 0.05	85.48 ± 0.04	85.48 ± 0.10	85.15	85.21
		mAP	68.12 ± 0.51	65.58 ± 0.55	67.75 ± 0.53	66.29 ± 0.54	67.52	65.72
Vowel	Gaussian	ACC/mAP	62.12 ± 0.01	64.07 ± 0.03	60.41± 0.07	52.01± 0.03	59.72	50.33
	Laplace	ACC/mAP	62.58 ± 0.02	64.29 ± 0.02	60.97 ± 0.08	52.60 ± 0.04	60.42	50.49
	Inverse Square	ACC/mAP	61.88 ± 0.03	64.07 ± 0.03	60.52 ± 0.08	51.36 ± 0.02	59.77	50.34
	Inverse	ACC/mAP	62.60 ± 0.02	64.29 ± 0.02	61.28 ± 0.04	51.84 ± 0.01	60.39	50.11

For classification problem, the main performance evaluation are accuracy(ACC) and mean average precision(mAP), both of the two metrics are used in this paper. And the concrete meaning of them are as follows:

$$ACC = \frac{TP + TN}{TP + TN + FN + FP}, \tag{37}$$

$$mAP = \frac{1}{m}\sum_{i=1}^{m}\frac{TP}{TP + FP}. \tag{38}$$

Besides, we compare the robustness by calculating the standard derivation of ACC and mAP.

The experiment results of 10 times random experiments are shown in **Table 3**, from which we can clearly see the performance on classification tasks with the proposed method. Especially, because the number of every class in the Vowel data set is same , the experiment results of ACC and mAP on Vowel are equal.

4.3.2 Performance Comparison on Regression Problems

Different from using ACC and mAP as performance evaluation, the regression problem concerns the root mean square error(RMSE) between the desired output and the prediction output. And the concrete meaning of RMSE is as follow:

$$RMSE=\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[t(x_i)-t_d(x_i)\right]^2}, \tag{39}$$

Table 4. Performance comparison of different method on Auto price, Housing, Machine Cpu and Stock data sets. Each cell represent RMSE ± standard derivation corresponding to specific kernel type.

Data sets	Kernel type	The proposed MHLK-ELM	MK-ELM	Best HLK-ELM	Best ELM	Mean HLK-ELM	Mean ELM
Auto price	Gaussian	0.2258 ± 0.0076	0.2177 ± 0.0068	0.2540 ± 0.0056	0.2461 ± 0.0021	0.2883	0.2787
	Laplace	0.2209 ± 0.0041	0.2916 ± 0.0049	0.2427 ± 0.0025	0.2441 ± 0.0028	0.2946	0.2959
	Inverse Square	0.2228 ± 0.0060	0.2085 ± 0.0060	0.2504 ± 0.0037	0.2345 ± 0.0017	0.2848	0.2794
	Inverse	0.2236 ± 0.0073	0.3066 ± 0.0053	0.2547 ± 0.0056	0.2417 ± 0.0026	0.2880	0.2817
Housing	Gaussian	0.2541 ± 0.0003	0.2628 ± 0.0006	0.3021 ± 0.0005	0.3112 ± 0.0010	0.3268	0.3304
	Laplace	0.2380 ± 0.0002	0.2623 ± 0.0006	0.2761 ± 0.0010	0.2970 ± 0.0007	0.3172	0.3320
	Inverse Square	0.2491 ± 0.0003	0.2634 ± 0.0005	0.2885 ± 0.0010	0.3122 ± 0.0016	0.3170	0.3304
	Inverse	0.2448 ± 0.0003	0.2734 ± 0.0006	0.2779 ± 0.0003	0.30528 ± 0.0003	0.3131	0.3302
Machine Cpu	Gaussian	0.2258 ± 0.0132	0.2320 ± 0.0198	0.2910 ± 0.0204	0.2570 ± 0.0151	0.3727	0.2793
	Laplace	0.2244 ± 0.0137	0.3103 ± 0.0177	0.2860 ± 0.0166	0.2653 ± 0.0184	0.3103	0.2840
	Inverse Square	0.2221 ± 0.0139	0.2113 ± 0.0182	0.2811 ± 0.0191	0.2631 ± 0.0168	0.2988	0.2794
	Inverse	0.2078 ± 0.0140	0.6944 ± 0.0022	0.2561 ± 0.0171	0.2596 ± 0.0166	0.2763	0.2788
Stock	Gaussian	0.0602 ± 0.0000	0.0552 ± 0.0000	0.0704 ± 0.0000	0.1238 ± 0.0001	0.1498	0.2247
	Laplace	0.0592 ± 0.0000	0.2080 ± 0.0030	0.0721 ± 0.0001	0.1402 ± 0.0017	0.1919	0.2312
	Inverse Square	0.0588 ± 0.0000	0.0542 ± 0.0000	0.0676 ± 0.0001	0.1215 ± 0.0003	0.1470	0.2273
	Inverse	0.0610 ± 0.0000	0.1283 ± 0.0001	0.0704 ± 0.0000	0.1202 ± 0.0002	0.1340	0.2313

where $t(x_i)$ and $t_d(x_i)$ represent the predict target value and the real target value of *i*th sample, respectively.

The performance comparative results on four real world data sets about regression problems are shown in **Table 4**. Besides, we also compare the stability by calculating the standard derivation of 10 times random experiments, which are also described in **Table 4**.

4.3.3 Result Analysis

From the experiments results in classification and regression data sets, we can clearly compare the performance of the proposed method and conclude as follow:

(1) For classification tasks, sparse MHLK-ELM can obtain better performance than MK-ELM both on Glass and Ecoli data sets, while MK-ELM perform slightly better than sparse MHLK-ELM on Vowel data set. Compared with base HLK-ELMs and base ELMs, MHLK-ELM can obtain comparable even slightly better performance than single best ELM and single best HLK-ELM. Besides, the classification performance of MHLK-ELM is much better than mean ELM and mean HLK-ELM.

(2) For regression tasks, from **Table 4** we can see that the proposed method can obtain the best performance on four data sets among almost all the methods. Although MK-ELM with base Gaussian and Inverse Square kernels perform slightly better than MHLK-ELM, MHLK-ELM can obtain more stable effect than MK-ELM. **Table 3** and **Table 4** show that the proposed method may perform better on regression tasks than classification tasks.

(3) For different data sets, ELM and HLK-ELM exist performance difference. Specifically, for the seven data sets, ELM perform slightly better than HLK-ELM on Auto price and Machine cpu data set, while HLK-ELM is better than ELM on other data sets, especially on the Vowel data set. But for all these data sets, MHLK-ELM can obtain the comparable results to the better one between single best ELM and single best HLK-ELM, which further verify the effectiveness and stability of the proposed method.

(4) For different kernel types, two multiple kernel methods perform various superiority. MK-ELM can obtain better performance with Gaussian and Inverse Square kernels, while MHLK-ELM perform better with Laplace and Inverse kernels. Besides, there does not exist absolute best kernel type for all data sets. But for specific data sets, there exist better kernel type. Indeed, trying to find a linear combination of different kernels will be another choice for selecting the optimal kernel type corresponding to specific data set.

4.4 Experimental Results with Non-Sparse MHLK-ELM

Sparse MHLK-ELM can effectively select the kernels which contribute more to the task, but it may disable part of effective kernels. So sufficiently using all base kernels information is necessary in some situations. Non-sparse MHLK-ELM, which ensures combination coefficients $\gamma_i > 0 (i = 1, ..., k)$ by controlling the $l_q (q > 1)$ norm constrict on γ_i, can obtain multiple kernel result with complete base kernels information.

In order to study the impact of various q value, we experiment with various q values on three classification data sets. According to the setting in [21], the parameter q which controls the sparsity of base kernel combination is taken from $\{32/31; 16/15; 8/7; 4/3; 2; 4; 8; 16\}$. Besides, we add $q = 1$ in this paper to better compare the performance between sparse MHLK-ELM and non-sparse MHLK-ELM. And the other setting are same to the previous experiments.

The results of experiments are shown in **Fig. 1**, from which we can conclude as follows:

(1) For sparse and non-sparse MHLK-ELM, there are not very huge performance difference on three classification data sets between the two methods. On the whole, the non-sparse MHLK-ELM can obtain slightly better performance than sparse MHLK-ELM.

(2) For non-sparse MHLK-ELM, the value of q will impact the classification performance. But there are not obvious tendency indicating the optimal value, which may be another research point in the future work.

5. Conclusion

In this paper, we have proposed a multi-hidden-layer-kernel ELM called MHLK-ELM based on multiple kernel learning (MKL) to select an optimal linear combination of base hidden-layer kernels. Specifically, we first integrate kernel functions with random feature mapping of ELM to design a hidden-layer-kernel ELM (HLK-ELM), which serves as the base of MHLK-ELM. Then, we utilize the MKL method to propose two versions of MHLK-ELMs, called sparse and non-sparse MHLK-ELMs. Experimental results demonstrate that both two types of MHLK-ELMs can effectively find out the optimal linear combination of multiple

HLK-ELMs for different classification and regression problems. Besides, we analyze the impacts of different types of norms and different kernel types, which are two interesting research topics in our future work.

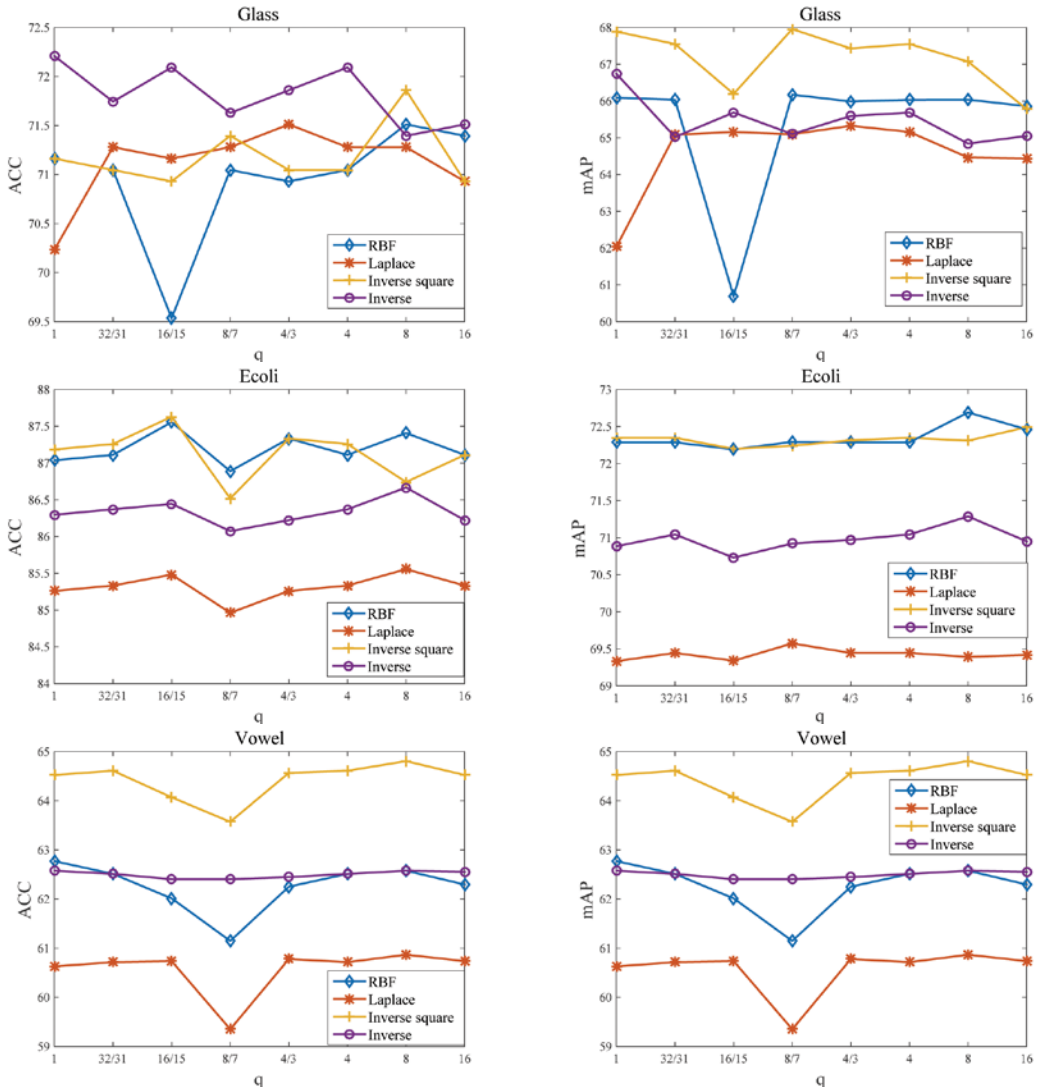

Fig. 1. Illustration of classification performance of the proposed method with different norms and kernel types on Glass, Ecoli and Vowel data sets

Acknowledgement

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions.

References

[1] G. B.Huang and C. K. Siew, "Extreme learning machine: RBF network case," in *Proc. of Int. Conf. on Control, Automation, Robotics and Vision*, vol. 2, pp.1029-1036, 2012. Article (CrossRef Link)

[2] G. B. Huang, H. Zhou, X. Ding and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems Man & Cybernetics Part B*, vol. 42, no. 2, pp. 513-529, 2012. Article (CrossRef Link)

[3] Lendasse Amaury, Q. He, Miche Yoan and G. B. Huang, "Advances in Extreme Learning Machines," *Neurocomputing*, vol. 149, no. PA, pp.158–159, 2015. Article (CrossRef Link)

[4] Huang G B, Wang D H and Lan Y, "Extreme learning machines: a survey," *International Journal of Machine Learning & Cybernetics*, vol. 2, no 2, pp.107-122, 2011. Article (CrossRef Link)

[5] E. Cambria, G. B. Huang, L. L. C. Kasun, "Extreme learning machines," *IEEE Intelligent Systems*, vol. 28, no. 6, pp. 30-59, 2013. Article (CrossRef Link)

[6] Q. Liu, S. Zhou, C. Zhu, X. Liu, J. Yin, "MI-ELM: Highly Efficient Multi-Instance Learning Based on Hierarchical Extreme Learning Machine," *Neurocomputing*, 2016, 173: 1044-1053. Article (CrossRef Link)

[7] Q. Y. Zhu, A. K. Qin, P. N. Suganthan, "Evolutionary extreme learning machine," *Pattern Recognition*, vol. 38, no.10, pp. 1759-1763, 2015. Article (CrossRef Link)

[8] J. W. Cao, Z. Lin, and G. B. Huang, "Self-Adaptive Evolutionary Extreme Learning Machine," *Neural Processing Letters*, vol. 36, no. 3, pp. 285-305, 2012. Article (CrossRef Link)

[9] G. B. Feng, G. B. Huang, Q. P. Lin, and Gay Robert, "Error Minimized Extreme Learning Machine With Growth of Hidden Nodes and Incremental Learning," *IEEE Trans Neural Netw*, vol. 20, no.8, pp.1352-7, 2009. Article (CrossRef Link)

[10] Wang G. G, Lu M, Dong Y. Q, and Zhao X. J, "Self-adaptive extreme learning machine," *Neural Computing & Applications*, vol. 27, no.2, pp. 291-303, 2016. Article (CrossRef Link)

[11] F. Han, H. F. Yao, and Q. H. Ling, "An Improved Extreme Learning Machine Based on Particle Swarm Optimization," in *Proc. of International Conference on Intelligent Computing (ICIC 2011)*, pp. 699-704, 2011. Article (CrossRef Link)

[12] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, V. C. M. Leung. "A Survey on Security Threats and Defensive Techniques of Machine Learning: A Data Driven View," IEEE Access, 6: 12103-12117, 2018. Article (CrossRef Link)

[13] Heeswijk Mark Van, Miche Yoan and Lindh-Knuutila, "Adaptive Ensemble Models of Extreme Learning Machines for Time Series Prediction," in *Proc. of 19th Int. Conf. on Artificial Neural Networks – ICANN 2009*, pp. 305-314, 2009. Article (CrossRef Link)

[14] Liu X. W, Wang L, Huang G. B. and Zhang J, "Multiple kernel extreme learning machine," *Neurocomputing*, vol. 149, no. PA, pp. 253-264, 2015. Article (CrossRef Link)

[15] X. D. Li, W. Mao, and W. Jiang, "Multiple-kernel-learning-based extreme learning machine for classification design," *Neural Computing & Applications*, vol. 27, no. 1, pp. 175-184, 2016. Article (CrossRef Link)

[16] G. B. Huang, Q. Y. Zhu and C. K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489-501, 2006. Article (CrossRef Link)

[17] B. Frénay and M. Verleysen, "Parameter-insensitive kernel in extreme learning for non-linear support vector regression," *Neurocomputing*, vol. 74, no. 16, pp. 2526-2531, 2011. Article (CrossRef Link)

[18] G. B. Huang and C. K. Siew, "Extreme Learning Machine with Randomly Assigned RBF Kernels," *International Journal of Information Technology*, vol. 11, no. 1, pp. 16-24, 2005. Article (CrossRef Link)

[19] Gönen Mehmet and E. Alpaydın, "Multiple Kernel Learning Algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211-2268, 2011. Article (CrossRef Link)

[20] A. Rakotomamonjy, F. R. Bach, S. Canu and Y. Grandvalet, "Simplemkl," *Journal of Machine Learning Research*, vol. 9, no.3, pp. 2491-2521, 2008. Article (CrossRef Link)

[21] F. Yan, J. Kittler, K. Mikolajczyk, A. Tahir, "Non-sparse multiple kernel fisher discriminant analysis," *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 607–642, 2012. Article (CrossRef Link)
[22] Y. Shi, T. Falck, A. Daemen, L. C. Tranchevent, J. A. Suykens, B. D. Moor and Y. Moreau, "L2-norm multiple kernel learning and its application to biomedical data fusion," *Bmc Bioinformatics*, vol. 11, no. 1, pp. 1-24, 2010. Article (CrossRef Link)
[23] G. B. Huang, "An insight into extreme learning machines: Random neurons, random features and kernels," *Cognitive Computation*, vol. 6, no. 3, pp.376–390, 2014. Article (CrossRef Link)
[24] Z. Xu, R. Jin, H. Yang, I. King and M. R. Lyu, "Simple and efficient multiple kernel learning by group lasso," in *Proc. of International Conference on Machine Learning*, pp. 1175–1182, 2010. Article (CrossRef Link)
[25] H. Yang, Z. Xu, J. Ye, I. King and M. R. Lyu, "Efficient sparse generalized multiple kernel learning," *IEEE Transactions on Neural Networks*, vol. 22, no. 3, pp. 433–446, 2011. Article (CrossRef Link)
[26] M. Kloft, U. Brefeld, P. Laskov and S. Sonnenburg, "Non-sparse multiple kernel learning," in *Proc. of NIPS Workshop on Kernel Learning Automatic Selection of Optimal Kernels*, pp. 775–782, 2008. Article (CrossRef Link)
[27] W. Samek, A. Binder and M. Kawanabe, "Multi-task learning via non-sparse multiple kernel learning," in *Proc. of Int. Conf. on Computer Analysis of Images and Patterns*, pp. 335–342, 2011. Article (CrossRef Link)
[28] M. Kloft, U. Brefeld and A. Zien, "l*p* -norm multiple kernel learning," *Journal of Machine Learning Research*, vol. 12, no. 2, pp. 953–997, 2011. Article (CrossRef Link)
[29] M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K. R. Mller and A. Zien, "Efficient and accurate lp-norm multiple kernel learning," in *Proc. of Int. Conf. on Neural Information Processing Systems*, pp. 997–1005, December7-10, 2009. Article (CrossRef Link)
[30] Uci irvine machine learning repository, Article (CrossRef Link).
[31] LIACC Regression DataSets, Article (CrossRef Link).

Wentao Zhao received his Ph.D. degree from National University of Defense Technology (NUDT) in 2009. He is now a Professor at NUDT. His research interests include network performance optimization, information processing and machine learning. Since 2011, Dr. Zhao has been serving as a member of Council Committee of Postgraduate Entrance Examination of Computer Science and Technology, NUDT. He has edited one book entitled ``Database Principle and Technology'' and several technical papers such as Communications of the CCF, WCNC'17, ICANN'17, WF-IoT, MDAI, FAW.

Pan Li received his B.Eng. degree from University of Science &Technology Beijing (USTB) in 2016. He is now a M.S. candidate at National University of Defense Technology (NUDT). His research interests include machine learning and cyber security. He has published several papers such as ICC'18, WASA'18, IEEE ACCESS.

Qiang Liu received his Ph.D. degree from the National University of Defense Technology (NUDT) in 2014. Now, he is an Assistant Professor at NUDT. He has contributed several archived journal and conference papers, such as NETW, TKDE, TWC, Trans on Cybernetics, ACCESS, PR, Neurocomputing, ICL, NCA, ICC'18, EDBT'17, WCNC'17, etc. He was a co-chair of SmartMM'18 and ICCCS'18 Workshop. His research interests include 5G, Internet of Things, wireless network security and machine learning.

Dan Liu received her M.S. degree from National University of Defense Technology (NUDT) in 2007. Her research interests include computer application and information processing.

Xinwang Liu received his PhD degree from National University of Defense Technology (NUDT), China. He is now Assistant Researcher of School of Computer, NUDT. His current research interests include kernel learning and unsupervised feature learning. Dr. Liu has published 40+ peer-reviewed papers, including those in highly regarded journals and conferences such as IEEE T-IP, IEEE T-NNLS, IEEE T-IFS, ICCV, AAAI, IJCAI, etc. He served on the Technical Program Committees of IJCAI 2016-2018 and AAAI 2016-2018.