Neural Comput & Applic (2013) 22:803–810
DOI 10.1007/s00521-011-0771-7

ORIGINAL ARTICLE

An enhanced extreme learning machine based on ridge regression for regression

Guoqiang Li · Peifeng Niu

Received: 5 September 2011 / Accepted: 22 November 2011 / Published online: 8 December 2011

Abstract The extreme learning machine (ELM) is a novel single hidden layer feedforward neural network, which has the superiority in many aspects, especially in the training speed; however, there are still some shortages that restrict the further development of ELM, such as the perturbation and multicollinearity in the linear model. To the adverse effects caused by the perturbation or the multicollinearity, this paper proposes an enhanced ELM based on ridge regression (RR-ELM) for regression, which replaces the least square method to calculate output weights. With an additional adjustment of ridge regression, all the characteristics become even better. Simulative results show that the RR-ELM, compared with ELM, has better stability and generalization performance.

Keywords Extreme learning machine · Single hidden layer feedforward neural networks · Ridge regression · Least square method

1 Introduction

In last few years, extensive research of the artificial neural networks (ANN) has already been implemented in developing theory and applications [1–6] due to their advantages: (1) ANN owns an inherent structure suitable for mapping complex nonlinear directly from the input samples and (2) ANN could provide models for a large class of natural and artificial phenomena. However, the free parameters of ANN are learnt from the given training samples by the gradient descent algorithms, which are relatively slow and have some issues related to its local minima. Owing to these shortages, it could take much more time to train networks and have a suboptimal solution.

Recently, extreme learning machine (ELM) is proposed by Huang [7], which is a novel single hidden layer feedforward neural network (SLFN) where the input weights and the bias of hidden nodes are generated randomly without tuning and the output weight is determined analytically. ELM owns an extremely fast learning algorithm and good generalization capability. In comparison with other traditional learning algorithms such as BP and RBF, the ELM algorithm has been proved a faster learning algorithm for SLFNs. In addition, the ELM overcomes most issues encountered in traditional learning methods, such as the stopping criterion, number of epochs, learning rate and local minima. So ELM is suitable for data-based modeling in complicated processes. Up to now, the ELM has been successfully applied in various areas [8–11], such as classification [12], function approximation [13], non-technical loss analysis [14], terrain reconstruction [15] and protein structure prediction [16].

In order to further enhance the performance of ELM, many modified or improved ELMs have been proposed [17–21]. There are four types of improved models of ELM: ensemble type, optimization type, incremental type and replacement type. In the ensemble type, different ELMs are trained by disjoint subsets of data but share the same hidden layer neurons. In the optimization type, various optimization methods are employed to adjust input weights and hidden layer bias of ELM and optimize the network

G. Li (✉) · P. Niu (✉)
Key Lab of Industrial Computer Control Engineering of Hebei Province, Yanshan University, Qinhuangdao 066004, China
e-mail: zhihuiyuang@163.com

P. Niu
e-mail: niupeifeng2011@163.com

G. Li · P. Niu
National Engineering Research Center for Equipment and Technology of Cold Strip Rolling, Qinhuangdao 066004, China

structure [22]. In the incremental type, the improved ELMs create new hidden layer neurons one by one according to certain criteria [23, 24]. In the replacement type, modified ELMs substitute the activation functions of ELM (sigmoid and RBF) for sine and cosine functions (or other functions), which is very helpful to improve the accuracy and the convergence rate for the problem of function approximation [25].

Although these models enhance the performance of ELM in a certain degree, there still exists great development space for the stability and the learning speed of ELM, such as the perturbation and multicollinearity in the linear model. These shortages would restrict the further development of ELM; i.e., the stability and the generalization capability of ELM could be extremely influenced when the hidden layer output matrix is singular or ill-conditioning. Therefore, it is necessary to further study the ELM.

In regression analysis, researchers often encounter the problem of multicollinearity. In the presence of multicollinearity, ordinary least squares (OLS) unbiased estimator could became very unstable due to their large variance, which leads to poor prediction [26–29]. To overcome such a problem, one alternative is ridge regression [30]. Some researches have proved that the ridge estimator is superior to the OLS estimator in the sense of mean squared error (MSE), especially for generalized ridge regression estimation.

For the shortages of the least square and advantages of generalized ridge regression, this paper proposes an enhanced ELM based on the ridge regression called RR-ELM. The proposed RR-ELM is different from ordinary ELM and other modified algorithms. It calculates the output weight matrix by generalized ridge regression rather than the least square method. The ELM method could have a better stability and generalization capability by ridge regression estimator.

The paper is organized as follows: the ELM is reviewed in Sect. 2. Section 3 describes the proposed RR-RLM. Section 4 shows the performance evaluation of RR-ELM. Finally, Sect. 5 summarizes the conclusions from this paper.

2 Extreme learning machine

In this section, we present a brief review of the ELM.

Generally speaking, we could regard an ELM as a special SLFN whose output layer is a linear combiner; thus, we may use the linear LS to calculate it analytically. Due to the universal approximation property, we could say that, if the SLFN holds sufficient hidden neurons, any function could be approximated to any desirable accuracy. Suppose there are N samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i = \left[x_1^i, x_2^i, \ldots, x_n^i\right]$ is an n-dimensional feature of the ith sample and $\mathbf{t}_i = \left[\mathbf{t}_1^i, \mathbf{t}_2^i, \ldots, \mathbf{t}_L^i\right]$ is the target vector. Let $\mathbf{W}$ be $M \times n$ input weight, $\mathbf{B}$ be $M \times 1$ bias of hidden layer neurons and $\boldsymbol{\beta}$ be $L \times M$ output weight.

The output ($\mathbf{T}$) of the ELM with M hidden neurons has the following form

$$\mathbf{t}_k^i = \sum_{j=1}^{M} \beta_{kj} g_j(\mathbf{W}, \mathbf{B}, \mathbf{X}), \quad k = 1, 2, \ldots, L \tag{1}$$

where $g_i(\cdot)$ is the activation function.

Equation 1 could be written in matrix form as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \tag{2}$$

where

$$\mathbf{H}(\mathbf{W}, \mathbf{B}, \mathbf{X}) = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g(w_M \cdot \mathbf{x}_1 + b_M) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g(\mathbf{w}_M \cdot \mathbf{x}_N + b_M) \end{bmatrix}_{N \times M},$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \vdots \\ \boldsymbol{\beta}_M \end{bmatrix}_{M \times L} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_N \end{bmatrix}_{N \times L}.$$

So the output weight $\boldsymbol{\beta}$ of (2) is estimated analytically by

$$\tilde{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \|\boldsymbol{H}\boldsymbol{\beta} - \mathbf{T}\| = \mathbf{H}^{+}\mathbf{T} \tag{3}$$

where $\mathbf{H}^{+}$ is the Moore–Penrose generalized inverse of $\mathbf{H}$. If $\mathbf{H}$ is nonsingular, Eq. (3) could be written as

$$\tilde{\boldsymbol{\beta}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{T} \tag{4}$$

ELM computes $\mathbf{H}^{+}$ in (3) by the singular value decomposition (SVD) of $\mathbf{H}$ [6]. The SVD of $\mathbf{H}$ is given by

$$\mathbf{H} = \mathbf{U}\Sigma\mathbf{V}^T = \sum_{i=1}^{n} u_i \sigma_i v_i^T \tag{5}$$

where $\mathbf{U} = (u_1, \ldots, u_n)$, $\mathbf{V} = (v_1, \ldots, v_n)$, $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)$, $\sigma_i \in \Sigma, i = 1, 2, \ldots, n$ are singular values of $\mathbf{H}$ with the order $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$.

Based on (5), the $\mathbf{H}^{+}$ in (3) could be calculated by

$$\mathbf{H}^{+} = \mathbf{V}\Sigma^{+}\mathbf{U}^T = \sum_{i=1}^{n} \frac{v_i u_i^T}{\sigma_i} \tag{6}$$

Then, the output weight $\tilde{\boldsymbol{\beta}}$ is computed by

$$\tilde{\boldsymbol{\beta}} = \mathbf{H}^{+}\mathbf{T} = \sum_{i=1}^{n} \frac{v_i u_i^T}{\sigma_i}\mathbf{T} \tag{7}$$

In most practical application, $\mathbf{T}$ contains some degree of perturbations. We could denote $\mathbf{e}$ as the perturbation. The perturbed target could be denoted as $\tilde{\mathbf{T}} = \mathbf{T} + \mathbf{e}$. So $\tilde{\boldsymbol{\beta}}$ could be computed by

$$\mathbf{H}\tilde{\boldsymbol{\beta}} = \tilde{\mathbf{T}} \tag{8}$$

$$\tilde{\boldsymbol{\beta}} = \mathbf{H}^{+}\tilde{\mathbf{T}} = \sum_{i=1}^{n} \frac{v_i u_i^T}{\sigma_i}(\mathbf{T}+\mathbf{e}) = \sum_{i=1}^{n} \frac{v_i u_i^T}{\sigma_i}\mathbf{T} + \sum_{i=1}^{n} \frac{v_i u_i^T}{\sigma_i}\mathbf{e} \tag{9}$$

When $\mathbf{H}$ is singular, it has some singular values that are very small positive or approximate to zero; then, $\boldsymbol{\beta}$ could be extremely influenced by $\mathbf{e}$ and weaken the generalization capability of ELM. The critical factor influencing the stability of ELM is the ill-conditioning of $\mathbf{H}$.

3 Proposed RR-ELM

3.1 Calculate the output weight matrix

If the hidden layer output matrix $\mathbf{H}$ holds some very small positive singular values, it would seriously influence the stability and weaken the generalization capability of the ELM. In order to address this problem, we propose a new method based on ridge regression to improve the stability of ELM.

Based on the linear model of (8), the generalized ridge regression estimator [30–32] of $\boldsymbol{\beta}$ is defined to be

$$\tilde{\boldsymbol{\beta}}(\mathbf{K}) = (\mathbf{H}^T\mathbf{H} + \mathbf{K})^{-1}\mathbf{H}^T\mathbf{T} \tag{10}$$

where $\mathbf{K}$ is a $M \times M$ diagonal matrix whose diagonal elements $k_1, k_2, \ldots, k_M$ are called ridge constants. If all the ridge constants are same,

$$\tilde{\boldsymbol{\beta}}(k) = (\mathbf{H}^T\mathbf{H} + k\mathbf{I})^{-1}\mathbf{H}^T\mathbf{T} \tag{11}$$

is called ordinary ridge regression estimator. When $k = 0$, the ridge regression estimator $\tilde{\boldsymbol{\beta}}(0) = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{T}$ is one of the least squares.

The generalized ridge regression estimator could be written as

$$\tilde{\boldsymbol{\beta}}(\mathbf{K}) = (\mathbf{H}^T\mathbf{H} + \mathbf{K})^{-1}\mathbf{H}^T\mathbf{T} = (\mathbf{H}^T\mathbf{H} + \mathbf{K})^{-1}(\mathbf{H}^T\mathbf{H})\tilde{\boldsymbol{\beta}}(0) \tag{12}$$

We could get the almost unbiased generalized ridge regression estimator $\tilde{\boldsymbol{\beta}}^*(\mathbf{K})$ [11] by

$$\tilde{\boldsymbol{\beta}}^*(\mathbf{K}) = [\mathbf{I} - (\mathbf{H}^T\mathbf{H} + \mathbf{K})^{-2}\mathbf{K}^2]\tilde{\boldsymbol{\beta}}(0) \tag{13}$$

For being the problem simple, we assume that all the ridge constants are same, and the output weight could be estimated by

$$\tilde{\boldsymbol{\beta}}^*(k) = [\mathbf{I} - k^2(\mathbf{H}^T\mathbf{H} + k\mathbf{I})^{-2}]\tilde{\boldsymbol{\beta}}(0) \tag{14}$$

If the ridge constant k is chosen not too large, the ridge regression estimator would have a smaller MSE than that of least squares estimator [29]. However, there is also a boring problem which is how to find a proper k that could make MSE minimal. In our paper, we adopt the method that was proposed by Lawless and Wang in [30], and we get a proper $\tilde{k}$ by the following forms

$$\tilde{\sigma}^2 = (\mathbf{T} - \mathbf{H}\boldsymbol{\beta}(0))^T(\mathbf{T} - \mathbf{H}\boldsymbol{\beta}(0))/(N - L) \tag{15}$$

$$\tilde{k} = \frac{L\tilde{\sigma}^2}{\tilde{\boldsymbol{\beta}}(0)^T(\mathbf{H}^T\mathbf{H})\tilde{\boldsymbol{\beta}}(0)} \tag{16}$$

where N is the number of samples.

Although the training time for RR-ELM may be slightly longer than the one for the original ELM, the proposed RR-ELM method, which replaces the least square method with the generalized ridge regression estimator, could achieve comparable generalization performance as the original ELM in the absence of multicollinearity. In addition, the RR-ELM is a better stable modeling tool than the original ELM in the presence of multicollinearity or perturbation; for example, the RR-ELM has a better stability and generalization capability than ELM.

3.2 The RR-ELM algorithm

We call the ELM algorithm based on ridge regression RR-ELM. The RR-ELM and ELM are different in solving the linear model. The following steps are the RR-ELM algorithm: Given a training set $S = \{(\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in R^n, t_i \in R^L\}$, hidden node number M and activation function $g(\cdot)$.

1. For a given number of hidden neurons M, randomly generate input weight matrix $\mathbf{W}$ and the bias matrix $\mathbf{B}$.
2. Calculate the hidden output matrix $\mathbf{H}$.
3. Calculate ridge constant k by (15) and (16).
4. Calculate the output weight matrix $\boldsymbol{\beta}$ by (14).

Table 1 Specification of benchmark data sets

Data sets	Attributes	Samples	Training data	Testing data
Servo	4	167	80	87
Abalone	8	4,177	2,000	2,177
Boston housing	13	506	250	256
Auto-MPG	7	398	200	198
Delta ailerons	5	7,129	3,000	4,129
Bank domains	8	8,192	4,500	3,692
California housing	8	20,640	8,000	12,640
House census (8L)	8	22,784	10,000	12,784
Delta elevators	6	9,517	4,000	5,517

Table 2 Performance comparison

Data sets		Algorithms	Hidden nodes	ε	Training time(s)	Training RMSE	Testing RMSE	
							Mean	Dev
Servo	Without noise	ELM	30	–	0.0001	0.0696	0.1331	0.0454
		I-ELM [34]	2,000	0.07	0.1398	0.1317	0.1440	0.0124
		EM-ELM [34]	28.6	0.07	0.0125	0.0665	0.1369	0.0220
		RR-ELM	30	–	0.0091	0.0759	0.1095	0.0092
	10% white noise	ELM	30	–	0.0013	0.0782	0.1127	0.0157
		RR-ELM	30	–	0.0109	0.0800	0.1115	0.0094
	20% white noise	ELM	30	–	0.0031	0.1080	0.1712	0.0260
		RR-ELM	30	–	0.0097	0.0866	0.1262	0.0116
Abalone	Without noise	ELM	30	–	0.0102	0.0756	0.0776	0.0190
		I-ELM [34]	1,840.6	0.07	0.4820	0.0814	0.0822	0.0030
		EM-ELM [34]	11.5	0.07	0.0117	0.0792	0.0794	0.0025
		RR-ELM	30	–	0.0328	0.0756	0.0751	0.0007
	10% white noise	ELM	30	–	0.0196	0.0820	0.0798	0.0013
		RR-ELM	30	–	0.0354	0.0826	0.0803	0.0009
	20% white noise	ELM	30	–	0.0205	0.0961	0.0931	0.0017
		RR-ELM	30	–	0.0396	0.0962	0.0944	0.0013
Boston housing	Without noise	ELM	50	–	0.0078	0.0692	0.0992	0.0111
		I-ELM [34]	2,000	0.07	0.1844	0.0922	0.1043	0.0077
		EM-ELM [34]	28.6	0.07	0.0219	0.0730	0.1052	0.0101
		RR-ELM	50	–	0.0165	0.0030	0.1159	0.0094
	10% white noise	ELM	50	–	0.0098	0.0698	0.1221	0.0086
		RR-ELM	50	–	0.0197	0.0548	0.1208	0.0129
	20% white noise	ELM	50	–	0.0121	0.0751	0.2312	0.0128
		RR-ELM	50	–	0.0176	0.0276	0.2107	0.0227
Auto-MPG	Without noise	ELM	30	–	0.0050	0.1421	0.4677	1.3758
		I-ELM [34]	2,000	0.07	0.1422	0.0789	0.0831	0.0053
		EM-ELM [34]	21.7	0.07	0.0086	0.0692	0.0803	0.0056
		RR-ELM	30	–	0.0115	0.1379	0.2675	0.0431
	10% white noise	ELM	30	–	0.0028	0.1401	0.3525	0.4924
		RR-ELM	30	–	0.0041	0.1446	0.2866	0.0492
	20% white noise	ELM	30	–	0.0020	0.1471	0.6733	0.5203
		RR-ELM	30	–	0.0018	0.1476	0.2967	0.0304
Delta ailerons	Without noise	ELM	30	–	0.0371	0.0371	0.0402	8.7×10^{-5}
		I-ELM [34]	1,779.8	0.04	0.5758	0.0409	0.0409	0.0012
		EM-ELM [34]	11.65	0.04	0.0195	0.0398	0.0398	0.0008
		RR-ELM	30	–	0.128	0.0377	0.0403	4.5×10^{-5}
	10% white noise	ELM	30	–	0.0317	0.0446	0.0497	3.8×10^{-4}
		RR-ELM	30	–	0.0532	0.0452	0.0498	3.5×10^{-4}
	20% white noise	ELM	30	–	0.0309	0.0619	0.0714	6.7×10^{-4}
		RR-ELM	30	–	0.0454	0.0624	0.0715	6.1×10^{-4}
Bank domains	Without noise	ELM	30	–	0.0698	0.0524	0.0501	0.0030
		I-ELM [34]	2,000	0.045	1.1430	0.0595	0.0601	0.0018
		EM-ELM [34]	90.9	0.045	1.0125	0.0449	0.0465	0.0011
		RR-ELM	30	–	0.0964	0.0523	0.0499	0.0031
	10% white noise	ELM	30	–	0.0544	0.0598	0.0570	0.0035
		RR-ELM	30	–	0.0787	0.0599	0.0571	0.0029
	20% white noise	ELM	30	–	0.0561	0.0777	0.0736	0.0024

Table 2 continued

Data sets		Algorithms	Hidden nodes	ε	Training time(s)	Training RMSE	Testing RMSE	
							Mean	Dev
		RR-ELM	30	–	0.0775	0.0778	0.0740	0.0026
California housing	Without noise	ELM	30	–	0.1125	0.4319	0.4796	2.9×10^{-4}
		I-ELM [34]	2,000	0.13	1.6391	0.1489	0.1328	0.0011
		EM-ELM [34]	46.65	0.13	0.5117	0.1304	0.1328	0.0014
		RR-ELM	30	–	0.1512	0.4320	0.4795	2.8×10^{-4}
	10% white noise	ELM	30	–	0.1185	0.4760	0.5243	4.3×10^{-4}
		RR-ELM	30	–	0.1339	0.4761	0.5242	4.8×10^{-4}
	20% white noise	ELM	30	–	0.1165	0.5230	0.5711	6.1×10^{-4}
		RR-ELM	30	–	0.1334	0.5228	0.5712	6.2×10^{-4}
House census (8L)	Without noise	ELM	30	–	0.1512	0.1042	0.6181	0.7363
		I-ELM [34]	2,000	0.065	2.0906	0.0827	0.0819	0.0009
		EM-ELM [34]	99.6	0.065	3.2664	0.0649	0.0681	0.0024
		RR-ELM	30	–	0.2025	0.1036	0.1462	0.0657
	10% white noise	ELM	30	–	0.1345	0.1078	0.7045	1.0319
		RR-ELM	30	–	0.1829	0.1078	0.1799	0.1056
	20% white noise	ELM	30	–	0.1343	0.1191	1.1235	2.2477
		RR-ELM	30	–	0.1914	0.1183	0.1943	0.0892
Delta elevators	Without noise	ELM	30	–	0.0592	0.3992	0.4475	0.1200
		I-ELM [34]	1,926.2	0.053	0.7781	0.0546	0.0543	0.0014
		EM-ELM [34]	28.35	0.053	0.1250	0.0530	0.538	0.0009
		RR-ELM	30	–	0.0776	0.4186	0.4201	0.1112
	10% white noise	ELM	30	–	0.0501	0.4494	0.4546	0.1195
		RR-ELM	30	–	0.0741	0.4423	0.4457	0.1329
	20% white noise	ELM	30	–	0.0511	0.5047	0.5077	0.1108
		RR-ELM	30	–	0.0737	0.4873	0.4897	0.1234

4 Performance evaluation of RR-ELM

We can evaluate the performance of RR-ELM by benchmark problems, which include nine regressions applications. The nine data sets, Servo, Abalone, Boston housing, Auto-MPG, Delta ailerons, Bank domains, California housing, House census(8L) and Delta elevators, are obtained from website of UCI. They are compared with the ELM [7], I-ELM [13] and EM-ELM [33]. For the nine regression applications, the training data and testing data that are randomly generated from the whole data set are normalized into $[-1, 1]$ before the trials of experiments. The data sets of the applications are divided into training data and testing data with the number of samples listed in Table 1.

All the experiments are carried out under Windows 2000 and Matlab 7.1 with AMD 9650, 2.31-GHz CPU and 2G RAM. We construct the algorithms of ELM and RR-ELM based on the source codes provided by Huang et al.

Every experiment is repeated 100 times. The root-mean-square error and the standard deviations of the predicted values of Table 1 are given in Table 2. In addition, the number of hidden nodes for every data set is also given in Table 2. Here, the sigmoidal activation function is used for three data sets: Servo, Abalone and Boston housing; the 'sinc' is adopted as the activation function of RR-ELM and ELM for six data sets: Auto-MPG, Delta ailerons, Bank domains, California housing, House census(8L) and Delta elevators. To further test the stability of the proposed methods, two degrees of white noise (10 and 20%) are added to the target attribute of each data set. For noise-added data sets, the parameters of employed methods are the same as the data sets without noise in order to well state the robustness of ELM and RR-ELM to perturbations (Fig. 1).

As it could be seen from the experimental results Table 2 and Figs. 2 and 3, although the training time for RR-ELM is slightly longer than the one for the original ELM, the performances of RR-ELM are better than those

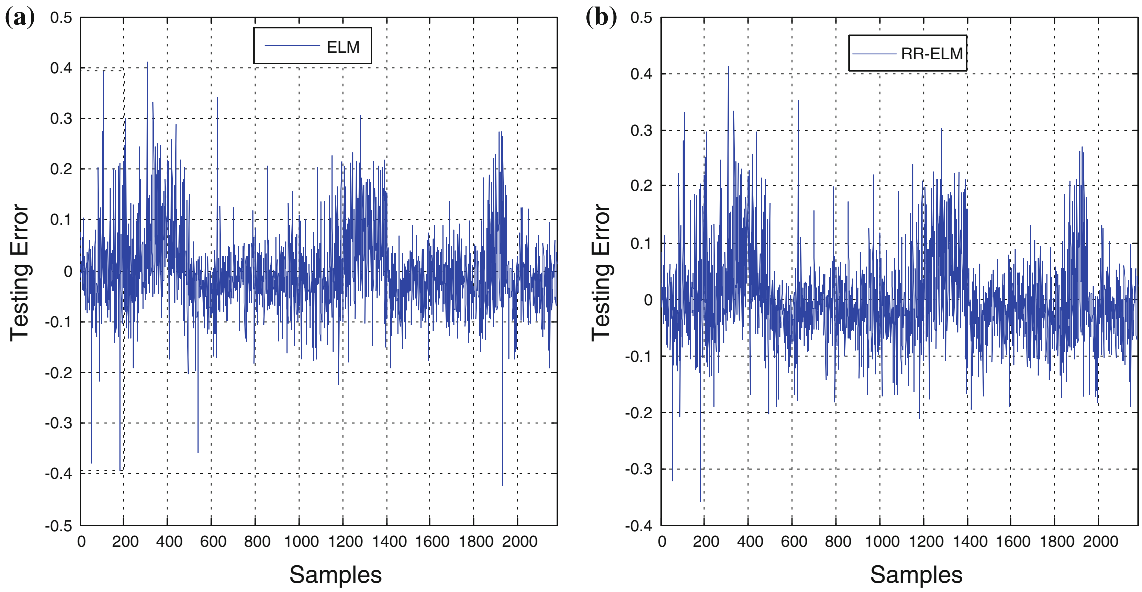

Fig. 1 The testing errors for Abalone without noise: **a** for ELM and **b** for RR-ELM

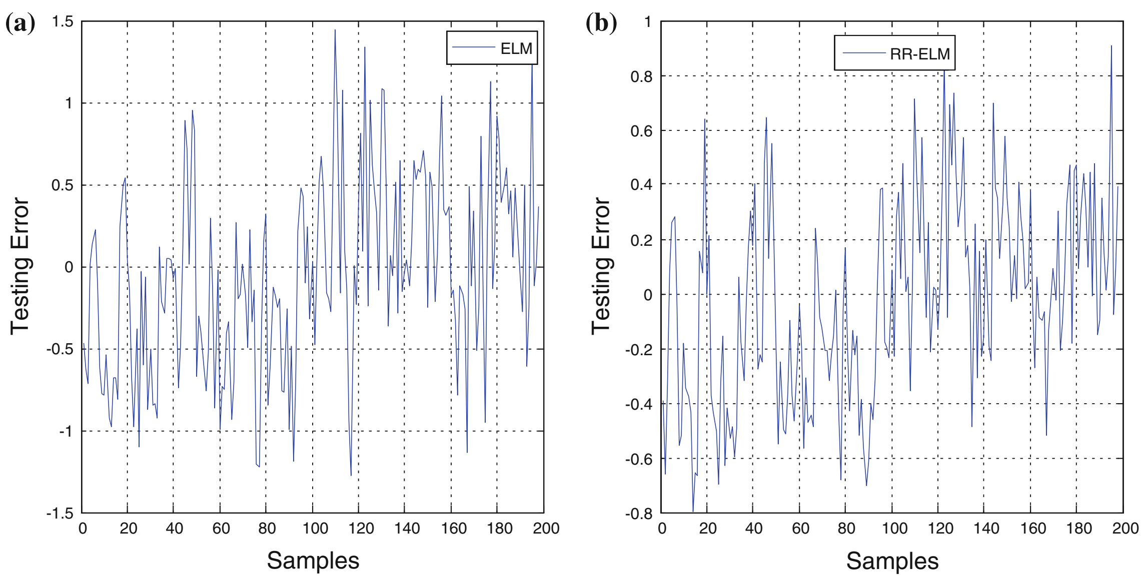

Fig. 2 The testing errors for Auto-MPG without noise: **a** for ELM and **b** for RR-ELM

of the ELM in terms of regression and prediction in most cases, especially for data sets with noise. The RR-ELM performs better generalization ability and stability than the original ELM on most applications, such as Servo, Boston housing, Auto-MPG, House census (8L) and Delta elevators, especially for Auto-MPG, House census (8L). For the remaining cases, Abalone, Delta ailerons, California housing and Bank domains, the performances of RR-ELM and the original ELM are comparable. The ELM could be thought as a better modeling tool because the training time for ELM is short than RR-ELM in the four applications.

In summation, the proposed RR-ELM achieves comparable generalization performance as the original ELM; in addition, RR-ELM is quite stable as indicated by the standard deviation of the generalization; i.e., the RR-ELM could effectively reduce the adverse effects caused by the perturbation **e** or by the very small singular values of hidden layer output matrix **H**. So the RR-ELM replaces the least square method with the ridge regression technique to

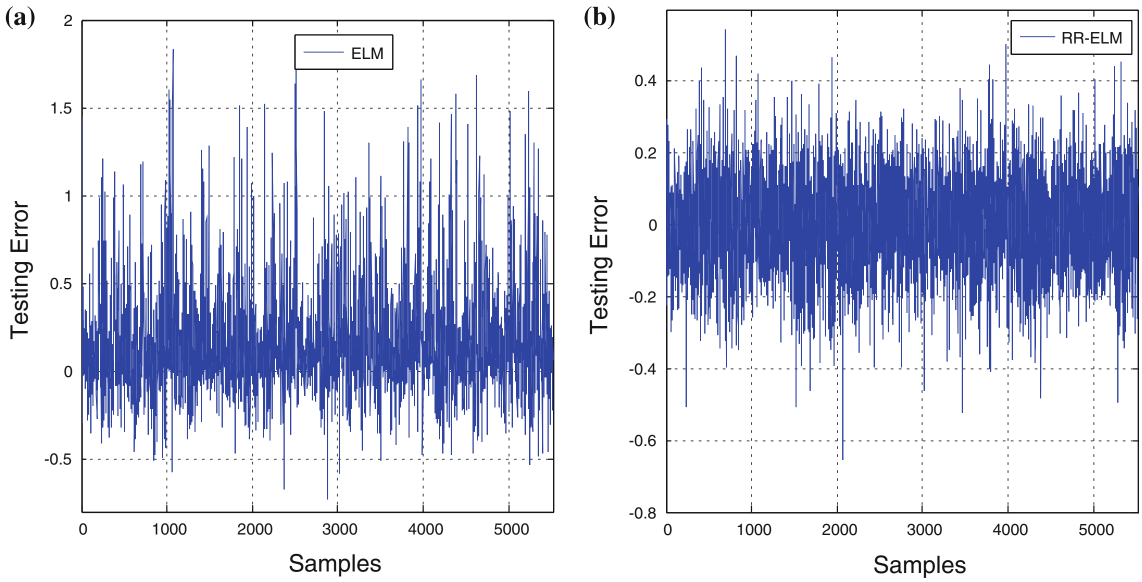

Fig. 3 The testing errors for Delta elevators without noise: **a** for ELM and **b** for RR-ELM

analytically calculate output weights of ELM in order to well solve these problems produced by the perturbation or the multicollinearity.

5 Conclusions

In this paper, the RR-ELM algorithm is proposed to improve the stability and the generalization capability of ELM. The output weight matrix is calculated analytically by the method of ridge regression estimator. The performance of comparison of RR-ELM with ELM, I-ELM and EM-ELM is evaluated on nine real-world benchmark regression applications. It shows that RR-ELM owns a good generalization performance and stability and competently reduces the adverse effects that are caused by the perturbation or the multicollinearity in the linear model. RR-ELM mainly deals with the regression problems. For the future research, we will develop new method to extend RR-ELM to other fields and study the methods to improve the training speed of RR-ELM. In addition, we will adopt PS-ABC [35], which is a high-efficiency artificial bee colony algorithm with the abilities of prediction and selection, to adjust input weights and hidden layer bias of RR-ELM and optimize the network structure in order to further improve the performance of RR-ELM.

Acknowledgments This project is supported by the National Natural Science Foundation of China (Grant No. 60774028) and Natural Science Foundation of Hebei Province, China (Grant No. F2010001318).

References

1. Han F, Ling Q-H, Huang D-S (2010) An improved approximation approach incorporating particle swarm optimization and a priori information into neural networks. Neural Comput Appl 19:255–261
2. Han F, Huang D-S (2008) A new constrained learning algorithm for function approximation by encoding a priori information into feedforward neural networks. Neural Comput Appl 17:433–439
3. Liang NY, Huang GB, Saratchandran P, Sundararajan N (2006) A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans Neural Netw 17:1411–1423
4. Samanta B (2011) Prediction of chaotic time series using computational intelligence. Expert Syst Appl 38:11406–11411
5. Tsai H-C, Lin Y-H (2011) Modular neural network programming with genetic optimization. Expert Syst Appl 38:11032–11039
6. Balamurugan G, Aravindhababu P (2011) ANN based online voltage estimation. Appl Soft Comput. doi:10.1016/j.asoc.2011.08.041
7. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501
8. Suresh S, Sarawathi S, Sundararajan N (2010) Performance enhancement of extreme learning for multi-category sparse data classification problems. Eng Appl Artif Intell 23:1149–1157
9. Minhas R, Mohammed AA, Jonathan M, Wu Q (2010) A fast recognition framework based on extreme learning machine using hybrid object information. Neurocomputing 73:1831–1839
10. Pan C, Park D, Yang Y, Yoo H (2011) Leukocyte image segmentation by visual attention and extreme learning machine. Neural Comput Appl. doi:10.1007/s00521-011-0522-9
11. Hu X-f, Zhao Z, Wang S, Wang F-l, He D-k, Wu S-k (2008) Multi-stage extreme learning machine for fault diagnosis on hydraulic tube tester. Neural Comput Appl 17:399–403
12. Zhang RX, Huang GB, Sundararajan N, Saratchandran P (2007) Multicategory classification using an extreme learning machine for microarray gene expression cancer diagnosis. IEEE ACM Trans Comput Biol Bioinform 4(3):485–495
13. Huang G-B, Chen L, Siew C-K (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892

14. Nizar AH, Dong ZY, Wang Y (2008) Power utility nontechnical loss analysis with extreme learning machine method. IEEE Trans Power Syst 23(3):946–955
15. Yeu CWT, Lim MH, Huang GB, Agarwal A, Ong YS (2006) A new machine learning paradigm for terrain reconstruction. IEEE Geosci Remote Sens Lett 3(3):382–386
16. Wang G, Zhao Y, Wang D (2008) A protein secondary structure prediction framework based on the Extreme Learning Machine. Neurocomputing 72(1–3):262–268
17. Tang X, Han M (2009) Partial Lanczos extreme learning machine for single-output regression problems. Neurocomputing 72:3066–3076
18. Feng G, Huang G-B, Lin Q, Gay R (2009) Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans Neural Netw 20(8):1352–1357
19. Lan Y, Soh YC, Huang G-B (2009) Ensemble of online sequential extreme learning machine. Neurocomputing 72:3391–3395
20. Li G, Liu M, Dong M (2010) A new online learning algorithm for structure-adjustable extreme learning machine. Comput Math Appl 60:377–389
21. Rong H-J, Ong Y-S, Tan A-H, Zhu Z-X (2008) A fast pruned-extreme learning machine for classification problem. Neurocomputing 72:359–366
22. Zhu QY, Qin AK, Suganthan PN, Huang GB (2005) Evolutionary extreme learning machine. Pattern Recogn 38(10):1759–1763
23. Huang G-B, Chen L (2008) Enhanced random search based incremental extreme learning machine. Neurocomputing 71(16–18):3460–3468
24. Huang GB, Chen L (2007) Convex incremental extreme learning machine. Neurocomputing 70(16–18):3056–3062
25. Han F, Huang DS (2006) Improved extreme learning machine for function approximation by encoding a priori information. Neurocomputing 69(16–18):2369–2373
26. Hawkinsa DM, Yin X (2002) A faster algorithm for ridge regression of reduced rank data. Comput Stat Data Anal 40:253–262
27. Foucart T (1999) Stability of the inverse correlation matrix. Partial ridge regression. J Stat Plan Inference 77:141–154
28. Turlach BA (2006) An even faster algorithm for ridge regression of reduced rank data. Comput Stat Data Anal 50:642–658
29. Tutz G, Binder H (2007) Boosting ridge regression. Comput Stat Data Anal 51:6044–6059
30. Lawless JF, Wang P (1976) A simulation study of ridge and other regression estimators. Commun Stat Theory Methods A5:307–323
31. Lipovetsky S (2010) Enhanced ridge regressions. Math Comput Model 51:338–348
32. Wan ATK (2002) On generalized ridge regression estimators under collinearity and balanced loss. Appl Math Comput 129:455–467
33. Akdeniz F, Yuksel G, Wan ATK (2004) The moments of the operational almost unbiased ridge regression estimator. Appl Math Comput 153:673–684
34. Lan Y, Soh Y-C, Huang G-B (2010) Constructive hidden nodes selection of extreme learning machine. Neurocomputing 70:1–9
35. Li G, Niu P, Xiao X (2011) Development and investigation of efficient artificial bee colony algorithm for numerical function optimization. Appl Soft Comput. doi:10.1016/j.asoc.2011.08.040