Neurocomputing 198 (2016) 20–26

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Preliminary study on Wilcoxon-norm-based robust extreme learning machine

Xiao-Liang Xie*, Gui-Bin Bian, Zeng-Guang Hou, Zhen-Qiu Feng, Jian-Long Hao

State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Article history:
Received 1 April 2015
Received in revised form
3 November 2015
Accepted 6 December 2015
Available online 8 March 2016

Keywords:
Extreme learning machine
Wilcoxon neural network
Wilcoxon-norm based robust extreme learning machine

ABSTRACT

The fact that the linear estimators using the rank-based Wilcoxon approach in linear regression problems are usually insensitive to outliers is known in statistics. Outliers are the data points that differ greatly from the pattern set by the bulk of the data. Inspired by this fact, Hsieh et al. introduced the Wilcoxon approach into the area of machine learning. They investigated four new learning machines, such as Wilcoxon neural network (WNN), and developed four gradient descent based backpropagation algorithms to train these learning machines. The performances of these machines are better than ordinary nonrobust neural networks in outliers exist tasks. However, it is hard to balance the learning speed and the stability of these algorithms which is inherently the drawback of gradient descent based algorithms. In this paper, a new algorithm is used to train the output weights of single-layer feedforward neural networks (SLFN) with input weights and biases being randomly chosen. This algorithm is called Wilcoxon-norm based robust extreme learning machine or WRELM for short.

1. Introduction

It is reported that the modern age of neural network began with the work of McCulloch and Pitts in 1943 [1]. Since then, some popular and powerful artificial neural networks (ANN) have been proposed, such as self organizing maps (SOM) [2], radial basis function neural networks (RBF) [3], and support vector machines (SVM) [4]. Several learning algorithms have been proposed in the literature for training the aforementioned learning machines [2–6]. Among these machines, one simple structure is multilayer perceptron artificial neural networks (MLP). Some off-line algorithms have been introduced to learn the weights and biases of MLP. One well-known gradient descent based batch learning algorithm is back-propagation (BP) [5]. In order to improve the convergence speed of BP algorithm, several improvements were made in [6,7]. One problem associated with MLP is how to decide the stop criterion of training process, and another problem is how to decide the number of hidden layers and the number of neurons in each layer. It has been proved that a single-hidden layer feedforward neural network with additive hidden nodes and with a nonpolynomial activation function can approximate any continuous function in a compact set [8]. Huang et al. rigorously proved that SLFNs with randomly assigned input weights and hidden neurons' biases and with almost any nonzero activation functions can universally approximate any continuous function on any compact input sets [9,10]. Based on this concept, the extreme learning machine (ELM) algorithm was proposed for batch learning [9,11], which has attracted tremendous attention from various fields for recent years [12–17]. ELM was also extended to semi-supervised/unsupervised tasks [18] and online sequential learning applications (OS-ELM)[19]. Most of these algorithms are based on the principle of least square error minimization, so the performances of these algorithms are easily affected by outliers. In other words, these algorithms are not robust. Inspired by different mechanisms, two robust algorithms were proposed, namely least trimmed squares (LTS) [20–23] and rank-based Wilcoxon neural networks (WNN) [24–26]. LTS and WNN have good generalization capability in outliers existing tasks, but some vital parameters, like learning rate, have to be decided by try and error. In this paper, a new learning machine based on Wilcoxon norm is proposed, then the generalization capability and training speed of both robust and nonrobust algorithms will be compared.

This paper is organized as follows. Section 2 reviews the Wilcoxon neural network proposed by Hsieh [20] and discusses some related problems. Section 3 illustrates the basic background of ELM and discusses the proposed WRELM in detail. The experimental results are conducted in Section 4. Finally, some conclusions are included in Section 5.

---

* Corresponding author.

E-mail addresses: xiaoliang.xie@ia.ac.cn (X.-L. Xie), guibin.bian@ia.ac.cn (G.-B. Bian), hou@compsys.ia.ac.cn (Z.-G. Hou), zhenqiu.feng@ia.ac.cn (Z.-Q. Feng), jianlong.hao@ia.ac.cn (J.-L. Hao).

http://dx.doi.org/10.1016/j.neucom.2015.12.113
0925-2312/

2. Wilcoxon SLFN

2.1. Wilcoxon norm

The Wilcoxon norm of a vector will be used as the objective function for Wilcoxon learning machines. In order to define the Wilcoxon norm of a vector, a score function is introduced. The score function is a nondecreasing function $\phi : [0,1] \to \mathbb{R}^1$ which satisfies $\int_0^1 \phi(u)\,du = 0$ and $\int_0^1 \phi^2(u)\,du = 1$.

The score $a_\phi(\cdot)$ associated with the score function ϕ is defined by

$$a_\phi(i) = \phi\left(\frac{i}{N+1}\right), \quad i = 1, 2, \ldots, N \tag{1}$$

where N is a fixed positive integer. Hence $a_\phi(1) \le a_\phi(2) \le \ldots \le a_\phi(N)$. It can be shown that the following function is a pseudonorm (seminorm) on $\mathbb{R}^N$:

$$\| e \|_W = \sum_{i=1}^{N} a(R(e_i))e_i = \sum_{i=1}^{N} a(i)e_{(i)} \tag{2}$$

where $e = [e_1, \ldots, e_N]^T \in \mathbb{R}^N$, $R(e_i)$ denotes the rank of e_i among $e_1, \ldots, e_N$, $e_{(1)} \le \ldots \le e_{(N)}$ are the ordered values of $e_1, \ldots, e_N$, $a(i) = \phi[i/(N+1)]$, and $\phi(u) = \sqrt{12}(u - 0.5)$. We call $|e|_W$ defined in Eq. (2) the Wilcoxon norm of the vector e.

It is easy to show that the proposed Wilcoxon norm above satisfies the following properties for a pseudonorm:

(a) $\| e \|_W \ge 0$ for all $e \in \mathbb{R}^N$, if and only if $e_1 = \cdots = e_N$, $|e|_W = 0$.
(b) $\| \alpha e \|_W = |\alpha| \, \| e \|_W$ for all $\alpha \in \mathbb{R}^1$ and $e \in \mathbb{R}^N$.
(c) $|e_1 + e_2|_W \le |e_1|_W + |e_2|_W$ for all $e_1, e_2 \in \mathbb{R}^N$.

2.2. Wilcoxon neural network

In this part, just the core concept of WNN will be illustrated, more details on WNN can refer to [24]. Consider the single-hidden layer Wilcoxon neural network with $n+1$ nodes in its input layer, m nodes in its hidden layer, and p nodes in its output layer.

Let the input vector be $x = [x_1, x_2, \ldots, x_n, 1]^T \in \mathbb{R}^{n+1}$, and let v_{ij} denote the connection from the ith input node to the jth hidden node. The input u_j and output r_j of the jth hidden node are respectively given by

$$u_j = \sum_{i=1}^{n+1} v_{ji} x_i, \quad r_j = f(u_j), \quad \text{for } j = 1, 2, \ldots, m \tag{3}$$

where f is the activation function of hidden nodes.

Let w_{kj} denote the connection weight from the output of the jth hidden node to the kth output node. Then, the output of kth output node t_k and final output y_k are respectively given by

$$t_k = \sum_{j=1}^{m} w_{kj} r_j, \quad y_k = t_k + b_k, \quad \text{for } k = 1, 2, \ldots, p \tag{4}$$

where b_k is the bias of the kth output node.

Assume that the training data set is $\{(\boldsymbol{x}_i, d_i)\}_1^N$ with $\boldsymbol{x}_i \in \mathbb{R}^{n+1}$ and $\boldsymbol{d}_i \in \mathbb{R}^p$, where N is the number of training data, $\boldsymbol{x}_i = [x_{1i}, \ldots, x_{ni}, 1]^T$ is the ith input vector, and $\boldsymbol{d}_i$ is the desired output for the input $\boldsymbol{x}_i$. In the WNN, the approach is to choose network weights ($\boldsymbol{v}$ and $\boldsymbol{w}$) that minimize the Wilcoxon norm of the total residuals of training data

$$D(\boldsymbol{v}, \boldsymbol{w}) = \sum_{k=1}^{p} \sum_{i=1}^{N} a(R(e_{i,k}))e_{i,k} = \sum_{k=1}^{p} \sum_{i=1}^{N} a(i)e_{(i),k} \tag{5}$$

where $e_{i,k} = d_{i,k} - t_{i,k}$, $R(e_{i,k})$ denotes the rank of the residual $e_{i,k}$ among $e_{1,k}, \ldots, e_{N,k}$ and $e_{(1),k} \le \ldots \le e_{(N),k}$ are the ordered values of $e_{1,k}, \ldots, e_{N,k}$.

The neural network used above is the same as the one used in the traditional artificial neural network, except the bias terms at the output node. The main reason is that the Wilcoxon norm is a pseudonorm rather than the usual norm. $\| e \|_W = 0$ implies that $e_1 = \cdots = e_N$, not implies that $e_1 = \cdots = e_N = 0$. Therefore, without the bias terms, the resulting predictive function with small Wilcoxon norm of total residuals may deviate from the desired function by constant offsets. The bias term b_k is estimated by the median of the residuals at the kth output node, i.e., $b_k = \text{med}_{1 \le i \le N}\{d_{ki} - t_{ki}\}$.

The proposed gradient descent based algorithm in [24] can train WNN effectively, however, there is one practical issue involved in real application. The speed of convergence depends highly on the magnitude of the learning rate parameter which is highly task dependant. To guarantee the network convergence, and avoid oscillations during training, the learning rate parameter must be set to a relatively small value, which clearly affects the speed of the algorithm [1]. In this paper, we use an algorithm in linear regression to train WNN, and it will be discussed in the following section.

3. Wilcoxon-norm-based robust extreme learning machine

In this section, a brief description of the ELM algorithm developed by Huang et al. in [9] is given first. Then the WRELM algorithm is introduced.

3.1. ELM algorithm

In supervised batch learning applications, learning algorithms use a finite number of input-output samples for learning networks' parameters. For N arbitrary distinct samples $(x_i, y_i) \in \mathbb{R}^n \times \mathbb{R}^p$, standard SLFNs with m hidden neurons and activation function (or radial basis function) $g(x)$ are modeled as

$$\sum_{j=1}^{m} \boldsymbol{w}_j G(\boldsymbol{a}_j, b_j, \boldsymbol{x}_i) = \boldsymbol{y}_i, \quad \text{for } i = 1, \ldots, N \tag{6}$$

where $\boldsymbol{a}_j$ and b_j are the learning parameters of hidden neurons and $\boldsymbol{w}_j$ is the weight connecting the jth hidden node to output neurons. For additive hidden neuron with the activation function $g(x)$ (e.g., sigmoid or threshold), $G(\boldsymbol{a}_j, b_j, \boldsymbol{x})$ is given in [19]

$$G(\boldsymbol{a}_j, b_j, \boldsymbol{x}) = g(\boldsymbol{a}_j \cdot \boldsymbol{x} + b_j), \quad b_j \in \mathbb{R}. \tag{7}$$

For RBF hidden neuron with Gaussian activation function $g(x)$, $G(a_i, b_i, x)$ is given by $G(\boldsymbol{a}_j, b_j, \boldsymbol{x}) = g\left(\frac{\| \boldsymbol{x} - \boldsymbol{a}_j \|}{2b_j^2}\right)$, $b_j \in \mathbb{R}$.

Eq. (6) can be written compactly as

$$H \cdot W = Y \tag{8}$$

where

$$H = \begin{bmatrix} G(\boldsymbol{a}_1, b_1, \boldsymbol{x}_1) & \cdots & G(\boldsymbol{a}_m, b_m, \boldsymbol{x}_1) \\ \vdots & \cdots & \vdots \\ G(\boldsymbol{a}_1, b_1, \boldsymbol{x}_N) & \cdots & G(\boldsymbol{a}_m, b_m, \boldsymbol{x}_N) \end{bmatrix}, W = \begin{bmatrix} \boldsymbol{w}_1^T \\ \vdots \\ \boldsymbol{w}_m^T \end{bmatrix}_{m \times p}$$

$$\text{and} \quad Y = \begin{bmatrix} \boldsymbol{y}_1^T \\ \vdots \\ \boldsymbol{y}_N^T \end{bmatrix}_{N \times p}.$$

H is called the hidden layer output matrix of the network [9]. The ith column of H is the ith hidden node's output vector with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$.

By minimizing the objective function $\| H \cdot W - Y \|_2^2$, the estimation of output weights of hidden layer can be calculated by

$$W = \arg \min_{\mathbf{w}_k} \| H \cdot W - Y \|_2^2 = H^+ Y \tag{9}$$

where H^+ is the Penrose–Moore pseudo inverse of H. More details on ELM can refer to [27,28].

3.2. Description of the proposed WRELM

Like ELM algorithm, if the weights and biases of the input layer of WNN are randomly chosen, the dimension of the parameters to be learned in WNN could be greatly reduced. Based on this principle, the WRELM algorithm is proposed.

After the input weights and the hidden layer biases are randomly chosen (independent of the training data), single-layer Wilcoxon neural network can be simply considered as a linear system

$$y_{i,k} = b_k + H_i \cdot \boldsymbol{w}_k + e_{i,k}, \quad \text{for } i = 1, \dots, N, \; k = 1, \dots, p \tag{10}$$

where H_i is the ith row of hidden layer output matrix H, and $\mathbf{w}_k \in \Re^{m \times 1}$ to be learned is the weight connecting the hidden neurons to the kth output neuron, and $e_{i,k}$ is a random variable with density f_k and distribution function F_k. In its general form, Jaeckel's rank dispersion function can be stated as

$$D_R(e_k) = \sum_{i=1}^{n} e_{i,k} \, a[R(e_{i,k})] \tag{11}$$

where $a(1) \le a(2) \le \dots \le a(N)$ is a set of scores generated by $a(i) = \varphi(i/(n+1))$ and $e_{i,k} = y_{i,k} - H_i \cdot \mathbf{w}_k$. One usually used score function is $\phi(u) = \sqrt{12}(u - 0.5)$. Some other forms of score functions can been found in [29–31]. It is easy to prove that $D_R(e)$ is an even $(D_R(e) = D_R(-e))$ and location free $(D_R(e) = D_R(e - \gamma I))$ dispersion function. Jaeckel shows that $D_R(e)$ is a nonnegative continuous, and convex function of $W = [\boldsymbol{w}_1, \dots, \boldsymbol{w}_p]$ which attains its minimum with bounded W if X has full rank [30].

We denote the rank based estimator of $\boldsymbol{w}_k$ by $\tilde{\boldsymbol{w}}_k$, which is

$$\tilde{\boldsymbol{w}}_k = \arg \min_{\boldsymbol{w}_k} D_R(Y_k - H \cdot \boldsymbol{w}_k) \tag{12}$$

$$\tilde{\boldsymbol{w}}_k = \arg \min_{\boldsymbol{w}_k} \| Y_k - H \cdot \boldsymbol{w}_k \|_W \tag{13}$$

where $\| \cdot \|_W$ is the pseudo-norm defined in (2).

In order to minimize $D_R(Y_k - H \cdot \boldsymbol{w}_k)$, we need to compute its partial derivative with respect to $\boldsymbol{w}_k$ which exists almost everywhere [32,33]

$$\nabla D_R = \frac{\partial D_R}{\partial \boldsymbol{w}_k} = -S(Y_k - H \cdot \boldsymbol{w}_k) = -H^T a(R(Y_k - H \cdot \boldsymbol{w}_k)). \tag{14}$$

Thus $\tilde{\boldsymbol{w}}_k$ is the solution to the following R-normal equations

$$H^T a(R(Y_k - H \cdot \boldsymbol{w}_k)) = 0_N \tag{15}$$

Let $\boldsymbol{w}_{k0}$ denote the true parameters which satisfy R-normal equations and the scale factor

$$\tau_k = \left(\sqrt{12} \int_{-\infty}^{+\infty} f_k^2(x) \, dx \right)^{-1}, \quad k = 1, \dots, p \tag{16}$$

where f_k is the probability density function of the noise e_k. If the following requirements are satisfied, the dispersion function $D_R(\cdot)$ can be approximated by a quadratic function $Q(\cdot)$ [34]

$$Q(Y_k - H \cdot \boldsymbol{w}_k) = \frac{1}{2\tau_k} (\boldsymbol{w}_k - \boldsymbol{w}_{k0})^T H^T H (\boldsymbol{w}_k - \boldsymbol{w}_{k0}) - (\boldsymbol{w}_k - \boldsymbol{w}_{k0})^T S(Y_k - H \cdot \boldsymbol{w}_{k0}) + D(Y_k - H \cdot \boldsymbol{w}_{k0}). \tag{17}$$

(a) The density f_k is absolutely continuous and its Fisher information $I(f_k) = \int_{-\infty}^{+\infty} [f_k'(x)]^2 / f_k(x) \, dx < \infty$.
(b) $\lim_{N \to \infty} N^{-1} X^T X = \Sigma$, where X is an $N \times m$ design matrix and Σ is a $m \times m$ positive definite matrix.
(c) $\lim_{N \to \infty} \max_{1 \le i \le N} x_{iq}^2 / \sum_{j=1}^{N} x_{jq}^2 \to 0$ for all $q = 1, \dots, m$.

The following estimate minimizes $Q(\cdot)$ in Eq. (17) [35]

$$\tilde{\boldsymbol{w}}_k(t+1) = \tilde{\boldsymbol{w}}_k(t) + \tau_k(t) (H^T H)^{-1} H^T a(R(Y_k - H \cdot \tilde{\boldsymbol{w}}_k(t))). \tag{18}$$

The scale factor $\tau_k(t)$ in (18) needs to be estimated. One estimate of $\int_{-\infty}^{+\infty} f_k^2(x) \, dx$ is by Schuster who first obtained a kernel type of estimate of $f_k(x)$ [36]

$$\tilde{f}_k(x) = \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{x - e_{i,k}}{h} \right)$$

where h is the kernel bandwidth and $K(\cdot)$ is a uniform kernel function

$$K(x) = \begin{cases} 1, & x \in [-1/2, 1/2] \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

Then $\delta_k = \int_{-\infty}^{+\infty} f_k^2(x) \, \mathrm{d}x$ can be estimated by

$$\hat{\delta}_k = 1/N^2 h \sum_{i=1}^{N} \sum_{j=1}^{N} I(|e_{i,k} - e_{j,k}| < h/2). \tag{20}$$

A modified version of the above estimate, $\hat{\delta}_{k,c}$ is proposed to ease the computation [32,37]

$$\hat{\delta}_{k,c} = \frac{1}{Nc} + \frac{1}{N(N-1)h} \sum_{i=1}^{N} \sum_{j \ne i}^{N} K\left(\frac{e_{i,k} - e_{j,k}}{h} \right) \tag{21}$$

where c is a fixed constant. When $h = c/\sqrt{N}$, the modified $\hat{\delta}_{k,c}$ is consistent of δ_k [32]. Finally, the proposed WRELM algorithm is described in the following.

Step (1) Set $t = 1$ and stop criterion (the maximum epoch number T_{max}). Randomly assign the input weights V, and output weights $W_{m \times p}(t)$.
Step (2) Compute the hidden layer output matrix of the network H by using Eq. (8) and calculate its Penrose–Moore pseudo inverse H^+.
Step (3) Obtain residuals $e_{N \times p}(t)$ and $\hat{\delta}_k(t)$ by

$$\hat{\delta}_k(t) = \frac{1}{Nc} + \frac{2}{\sqrt{N}(N-1)c} \sum_{i=2}^{N} \sum_{j<i}^{N} I\left(|e_{i,k} - e_{j,k}| < \frac{c}{2\sqrt{N}} \right). \tag{22}$$

Step (4) Compute the scale factor $\tau_k(t)$ by

$$\tau_k(t) = \frac{1}{\sqrt{12} \hat{\delta}_k(t)}. \tag{23}$$

Step (5) Update the kth column of output weights $W(t+1)$ by

$$\mathbf{w}_k(t+1) = \mathbf{w}_k(t) + \tau_k(t) H^+ a(R(e_k)). \tag{24}$$

Step (6) If $t > T_{max}$, then stop; otherwise go to Step (3).
Step (7) Compute the bias of the kth output neuron by $b_k = med_{1 \le i \le N}\{e_{k,i}\}$.

4. Illustrative examples

In this section, we compare the performances of five neural networks for both artificial regression problem and some other real world benchmark nonlinear regression examples. In those examples, in order to test the generalization capability of these learned machines, the machines are tested by another set of testing data without noise or outliers. Five learning machines compared here include two nonrobust neural networks, namely standard ANN and ELM in [9], three robust neural networks, namely LTS in [20], original Wilcoxon neural network in [24] and WRELM introduced in this paper. The root mean square error (RMSE) is used to quantify the performance of the learned machines for both the training errors and the generalization errors.

For a fair comparison, each neural network's hidden layer nodes have the same number of hidden nodes, and the same activation functions.

Before demonstrating those simulation results, the outlier models used here are introduced. In the simulation, two kinds of outlier models are considered. One is called the gross error model, and another is the artificial outlier model. In a gross error model, outliers are actually generated form another distribution with a longer tail or with larger variance. Usually, a gross error model can be written as [38]

$$D_\varepsilon = \{D \,|\, D = (1-\varepsilon)G + \varepsilon H, 0 \le \varepsilon \le 1\} \tag{25}$$

where ε is the probability of occurrence of an outlier, G is a usual distribution (noise), H is a symmetric long tailed distribution (outliers).

4.1. Artificial problem

In this simulation, the true function is given by the Hermite function [24]

$$y = 1.1 \cdot (1 - x + 2x^2) \cdot e^{-x^2/2}, \quad x \in [-5, 5]. \tag{26}$$

A training data set (x_i, y_i) with 100 data is generated, where x_i's are uniformly randomly distributed in the interval $[-5, 5]$. The gross error model used for modeling outliers is $D_\varepsilon = 0.85 \cdot G + 0.15 \cdot H$, where $G \backsim N(0, 0.1)$ and $H \backsim N(0, 1)$. For all the machines concerned, the number of hidden layer nodes is 20 and the activation functions of the hidden nodes are sigmoid functions.

The learning rate η used in BP training algorithm of ANN is 0.008, in LTS algorithm is 0.003, and in BP algorithm of WNN is 0.001. The number of training epochs for ANN is 8000, and for LTS is 80,000, and for WNN is 8000. The trainings of the above three algorithms are time consuming, while the WRELM is trained only 5 rounds in neglectable time.

The simulation results are shown in Fig. 1. For highly corrupted data as shown in Fig. 1(a), LTS, WNN and WRELM are robust to outliers, that means they are not affected by outliers. While the performances of least square based ANN and ELM as shown in Fig. 1(b) are severely affected by outliers. In this example, the performance of WRELM is almost as good as other two robust algorithms, but WRELM converges pretty faster and it is easier to apply from a practical point of view.

4.2. Real world benchmark regression problems

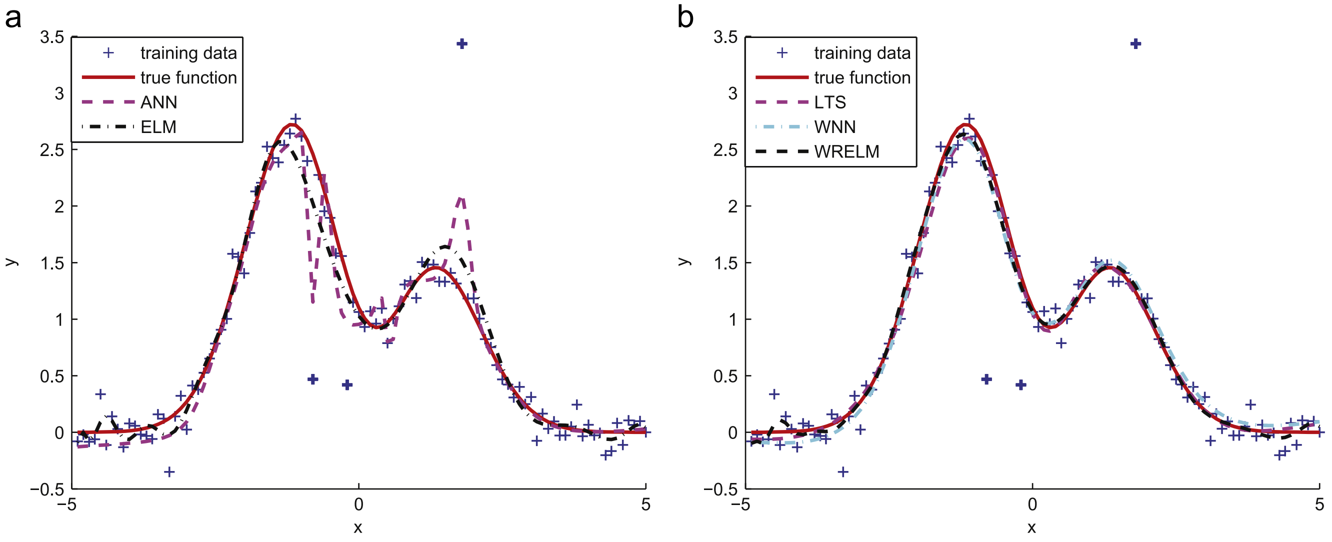
Example (1) *Fuel Consumption Prediction of Automobiles*: In this example [39], a regression benchmark problem is studied, namely, auto-mpg. This problem is to predict city-cycle fuel consumption of different models of car by 3 multivalued discrete and 4 continuous input attributes and one continuous output attributes. The dataset contains 392 data. In our simulation, about 3/4 of the total data are randomly chosen to form the training data set and the remaining data to form the testing data set. The corrupted training data set is formed by keep the normalized input attributes unchanged but with 5% randomly chosen output attribute values replaced by random values from a uniform distribution defined on $[-100, 100]$. The testing data set remains unchanged. For simplicity, the eight input attributes are normalized to the range $[-1, 1]$.

In this simulation, the learning rate which is chosen by trial and error, in BP algorithm of WNN is 0.01, in BP algorithm of ANN is 0.001, and in LTS algorithm is 0.001.

Fig. 2 verifies that compared with ELM and ANN, the three robust algorithms LTS, WNN and WRELM achieve good generalization performance when there exist outliers.

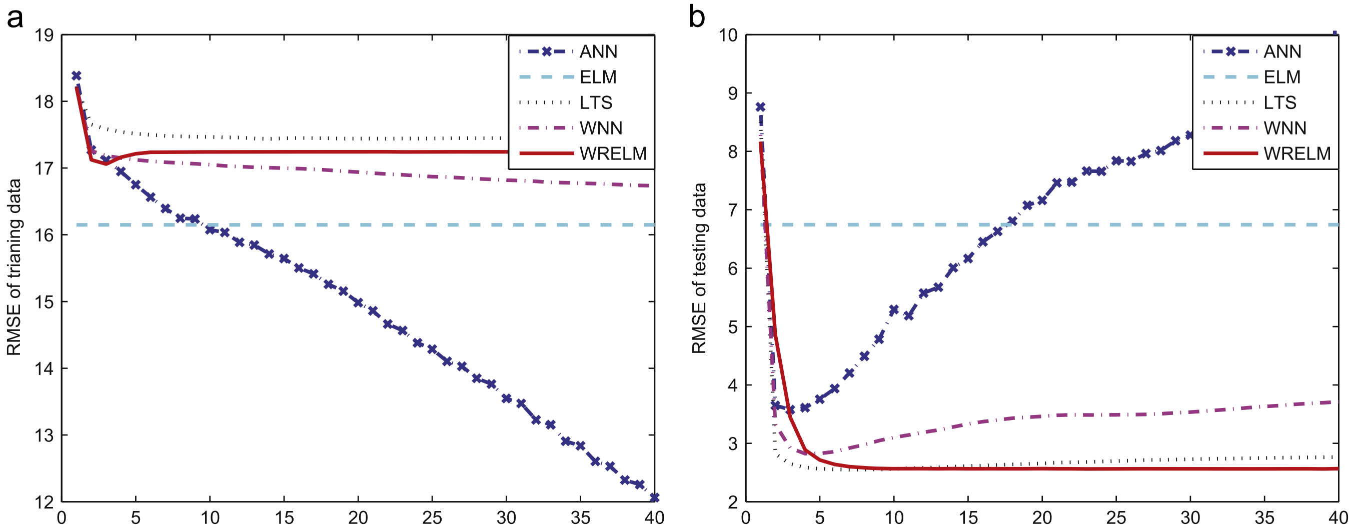
From Fig. 2, we can see that RMSEs of training data of traditional ANN and WNN trained by backpropagation algorithm decrease as training epochs increasing, but the RMSE curves of testing data of the two machines form a “V” shape. At the beginning stage of training process, the RMSEs of testing data decrease as training epochs increasing, after some epochs of training, they increase as training epochs increasing. So it is hard to determine the proper number of training epochs which makes application of these algorithms difficult. Compared with ANN and WNN, the RMSEs of training/testing data of LTS algorithm and WRELM algorithm converge as the number of training epochs increases. It can be further seen from Fig. 2(b) that WRELM algorithm archives least RMSE for the testing data in just a few training epochs in this application.

Example (2) *Abalone Age Prediction*: This problem has 4177 cases predicting the age of abalone from physical measurements [39]. The age of abalone is determined by cutting the shell through the cone, staining it, and counting the number of rings through a microscope which is a boring and time-consuming task. Other 8 measurements, which are easier to obtain, are used to predict



nonrobust algorithms robust algorithms

Fig. 1. Simulation results of Example 1.



RMSE of training data VS training epochs

RMSE of testing data VS training epochs

Fig. 2. Performance comparison of the concerned algorithms in Example 2 (the training epochs for ANN and LTS should be multiplied by 25, and for WNN should be multiplied by 50).

Table 1
Performances of both nonrobust and robust algorithms in abalone age prediction.

Algorithms	Time(Seconds)		Training		Testing		η	Epochs	#Nodes
	Mean	Dev	Mean	Dev	Mean	Dev			
ANN	28.9195	0.4107	6.8668	0.0002	2.5747	0.0734	0.0002	600	20
ELM [9]	**0.0142**	0.0001	**6.8065**	0.0566	2.5983	0.0114	–	1	20
LTS [20]	283.9649	55.9718	7.2171	0.0037	2.2730	0.0207	0.0002	5000	20
WNN [24]	24.9874	0.6430	7.0637	0.0699	2.0355	0.0151	0.001	500	20
WRELM	1.3656	0.0001	7.0506	0.0674	**2.0028**	0.0149	–	5	20
ANN	36.8601	1.0196	6.4117	0.0020	2.4861	0.1030	0.0002	600	30
ELM [9]	**0.0237**	0.0001	**6.3159**	0.0001	2.4635	0.0010	–	1	30
LTS [20]	356.6765	145.3539	6.6625	0.0023	2.1135	0.0025	0.0002	5000	30
WNN [24]	32.2266	1.1770	6.5274	0.0002	1.9847	0.0002	0.001	500	30
WRELM	1.3780	0.0001	6.5198	0.0002	**1.9727**	0.0004	–	5	30

the age. The 8 measurements are sex, length, diameter, height, whole weight, viscera weight, shell weight and rings. For simplicity, the eight input attributes are normalized to the range $[-1, 1]$. In this regression problem, about 75% of the total data are randomly chosen to form the training data set with 10% of the total training data are corrupted by keep the input attributes unchanged but output values are replaced by random values from a uniform distribution defined on [0, 50].

Table 1 summarizes the results for this benchmark regression problem in terms of training time, training RMSE and testing RMSE for each network with different number of nodes. We run each of the concerned five algorithms 10 times. The learning rates of BP algorithm of ANN, LTS and WNN are chosen by trial and error in consideration of converge speed and stability. The input weights and biases of hidden layer nodes of ELM and WRELM are of the same at each simulation, and they are chosen randomly in range $[-1, 1]$.

From Table 1, we can see that although the training time of ELM is neglectable and the RMSE of training uncorrupted data set of ELM is the smallest among the five algorithms, however, RMSE of testing data of this machine is pretty large, in other words, ELM has bad generalization performance when outliers exist. WRELM algorithm has fastest convergence speed with smallest RMSE of testing data among the other four algorithms.

Example (3) *Electrical Energy Output Prediction of Combined Cycle Power Plant*: A combined cycle power plant (CCPP) is composed of gas turbines, steam turbines and heat recovery steam generators. In a CCPP, the electricity is generated by gas and steam turbines, which are combined in one cycle, and is transferred from one turbine to another. The net hourly electrical energy output of the plant can be predicted by four hourly average ambient variables, namely, temperature, ambient pressure, relative humidity and exhaust vacuum [39].

In this example, the data set contains 9568 data points collected from a combined cycle power plant over 6 years (2006–2011), when the power plant was set to work with full load. For simplicity, both four input features and the output are normalized to the range [0, 1]. In this regression problem, 75% of the total data are randomly chosen to form the training data set with about 10% of the total training data are corrupted by keep the input features unchanged but corresponding outputs are replaced by random values between 0 and 1.

The performances of all concerned algorithms for this benchmark regression problem are listed in Table 2. The parameters of each algorithm are configured using the same way as it is used in Example (2). From Table 2, we can see the algorithm proposed in this paper achieves smallest prediction error and costs less time compared with other robust algorithms.

5. Conclusion

In this paper, a robust ELM-like learning machine was proposed, which called Wilcoxon-norm based robust extreme learning

Table 2
Performances of both nonrobust and robust algorithms in power plant's electrical energy output prediction.

Algorithms	Time(Seconds)		Training		Testing		η	Epochs	#Nodes
	Mean	Dev	Mean	Dev	Mean	Dev			
ANN	6.88	0.05	0.4406	0.0265	0.4251	0.0349	2.00e−4	600	30
ELM [9]	**0.03**	0.001	**0.1258**	1.53e−8	0.0600	2.00e−08	–	1	30
LTS [20]	63.13	4.00	0.2092	0.0258	0.1541	0.0351	2.00e−4	5000	30
WNN [24]	5.81	0.04	0.1282	4.08e−06	0.0586	8.07e−7	0.001	500	30
WRELM	4.37	0.01	0.1274	1.22e−08	**0.0557**	1.71e−08	–	5	30

machine or WRELM for short. Like ELM algorithm, after the input weights and the hidden layer biases are chosen randomly, single-layer WNN can be simply considered as a linear system, so the output weights can be tuned by robust linear regression methods. Based on this principle, the new robust algorithm called WRELM was introduced. Performance of WRELM was compared with ANN, ELM, LTS, and WNN on both artificial regression problem and some real world benchmark regression problems. The results indicate that WRELM algorithm, like WNN algorithm and LTS algorithm, is robust to outliers, but with no additional vital parameters, such as learning rate in gradient descent based algorithms, to been decided. The WRELM algorithm can converge fast (usually about 5 epochs), and is stable with good generalization capability.

Acknowledgments

This research is supported in part by the National High Technology Research and Development Program ("863 Program") of China (Grant #2015AA042306, and #2013AA013803), National Natural Science Foundation of China (Grant #61203318, #61203342, and #61333016), and Early Career Development Award of SKLMCCS.

References

[1] F.M. Ham, I. Kostanic, Principles of Neurocomputing for Science and Engineering, McGraw-Hill Higher Education, New York, 2000.
[2] T. Kohonen, Self-organizing formation of topologically correct feature maps, Biol. Cybern. 43 (1) (1982) 59–69.
[3] M. Powell, Radial basis function for multivariable interpolation: a review, Algorithm Approx. (1987) 143–167.
[4] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.
[5] P. Werbos, Beyond regression: new tools for prediction and analysis in the behavioral sciences (Ph.D. thesis), Harvard University, 1974.
[6] J. Gibb, A. Ieee, C. Lau, Back Propagation Family Album, 1996.
[7] M. Riedmiller, H. Braun, A direct adaptive method for faster backpropagation learning: the RPROP algorithm, in: IEEE International Conference on Neural Networks, 1993, pp. 586–591.
[8] M. Leshno, V.Y. Lin, A. Pinkus, S. Schocken, Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, Neural Netw. 6 (6) (1993) 861–867.
[9] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: a new learning scheme of feedforward neural networks, in: IEEE International Joint Conference on Neural Networks, vol. 2, 2004, pp. 985–990.
[10] G.-B. Huang, L. Chen, C.-K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, IEEE Trans. Neural Netw. 17 (4) (2006) 879–892.
[11] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1) (2006) 489–501.
[12] A. Mohammed, R. Minhas, Q.J. Wu, M. Sid-Ahmed, Human face recognition based on multidimensional PCA and extreme learning machine, Pattern Recognit. 44 (10) (2011) 2588–2597.
[13] B. Chacko, V. Vimal Krishnan, G. Raju, P. Babu Anto, Handwritten character recognition using wavelet energy and extreme learning machine, Int. J. Mach. Learn. Cybern. 3 (2) (2012) 149–161.
[14] J. Cao, Z. Lin, G.-B. Huang, N. Liu, Voting based extreme learning machine, Inf. Sci. 185 (1) (2012) 66–77.
[15] Z. Qi, Y. Tian, Y. Shi, Robust twin support vector machine for pattern classification, Pattern Recognit. 46 (1) (2013) 305–316.
[16] Z. Bai, G.-B. Huang, D. Wang, H. Wang, M.B. Westover, Sparse extreme learning machine for classification, IEEE Trans. Cybern. 44 (10) (2014) 1858–1870.
[17] G. Huang, G.-B. Huang, S. Song, K. You, Trends in extreme learning machines: a review, Neural Netw. 61 (2015) 32–48.
[18] G. Huang, S. Song, J.N. Gupta, C. Wu, Semi-supervised and unsupervised extreme learning machines, IEEE Trans. Cybern. 44 (12) (2014) 2405–2417.
[19] N.-Y. Liang, G.-B. Huang, P. Saratchandran, N. Sundararajan, A fast and accurate online sequential learning algorithm for feedforward network, IEEE Trans. Neural Netw. 17 (6) (2006) 1411–1423.
[20] A. Rusiecki, Robust LTS backpropagation learning algorithm, in: Computational and Ambient Intelligence, Springer Berlin Heidelberg, 2007, pp. 102–109.
[21] T.-D. Nguyen, R. Welsch, Outlier detection and least trimmed squares approximation using semi-definite programming, Comput. Stat. Data Anal. 54 (12) (2010) 3212–3226.
[22] X. Li, D. Coyle, L. Maguire, T.M. McGinnity, A least trimmed square regression method for second level fMRI effective connectivity analysis, Neuroinformatics 11 (1) (2013) 105–118.
[23] Y.-L. Lin, J.-G. Hsieh, J.-H. Jeng, W.-C. Cheng, On least trimmed squares neural networks, Neurocomputing 161 (2015) 107–112.
[24] J.-G. Hsieh, Y.-L. Lin, J.-H. Jeng, Preliminary study on Wilcoxon learning machines, IEEE Trans. Neural Netw. 19 (2) (2008) 201–211.
[25] U.K. Sahoo, G. Panda, B. Mulgrew, B. Majhi, Development of robust distributed learning strategies for wireless sensor networks using rank based norms, Signal Process. 101 (2014) 218–228.
[26] M. Shateri, S. Ghorbani, A. Hemmati-Sarapardeh, A. H. Mohammadi, Application of Wilcoxon generalized radial basis function network for prediction of natural gas compressibility factor, J. Taiwan Inst. Chem. Eng. 50 (2015) 131-141.
[27] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 42 (2) (2012) 513–529.
[28] G.-B. Huang, An insight into extreme learning machines: random neurons, random features and kernels, Cogn. Comput. 6 (3) (2014) 376–390.
[29] J. Jureckova, Asymptotic linearity of a rank statistic in regression parameter, Ann. Math. Stat. 40 (6) (1969) 1889–1900.
[30] L.A. Jaeckel, Estimation regression coefficients by minimizing the dispersion of the residuals, Ann. Math. Stat. 43 (5) (1972) 1449–1458.
[31] Y. Choi, O. Ozturk, A new class of score generating function for regression models, Stat. Probab. Lett. 57 (2) (2002) 205–214.
[32] T.P. Hettmansperger, J.W. McKean, Statistical Inference Based on Ranks, Krieger Malabar, FL, 1991.
[33] T. Hettmansperger, J. McKean, Statistical inference based on ranks, Psychometrika 43 (1) (1978) 69–79, http://dx.doi.org/10.1007/BF02294090, ISSN 0033–3123.
[34] T. Hettmansperger, Robust Non-Parametric Statistics, Wiley, New York, 1998.
[35] T.P. Hettmansperger, J.W. McKean, Robust Nonparametric Statistical Methods, CRC Press, Boca Raton, FL, 2010.
[36] E.P. Schuster, On the rate of convergence of an estimate of a functional of a probability density, Scand. Actuar. J. 1974 (2) (1974) 103–107.
[37] C. Qing, A.P., X. Biao, Rank regression in stability analysis, J. Biopharm. Stat. 13 (3) (2003) 463–479.
[38] K. Liano, Robust error measure for supervised neural network learning with outliers, IEEE Trans. Neural Netw. 7 (1) (1996) 246–250.
[39] A. Asuncion, D. Newman, UCI Mach. Learn. Repos. (2007), URL ⟨http://www.ics.uci.edu/~mlearn/MLRepository.html⟩.

Xiao-Liang Xie was born in April of 1983. He received the B.S. degree in Electrical Engineering and Automation from the Civil Aviation University of China, in 2006, and the Ph.D. degree in Control Theory and Control Engineering from the Chinese Academy of Sciences, in 2011 respectively. He is an Associate Professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include machine learning, robotics, and control theory.

Gui-Bin Bian is an Associate Professor with State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. He received B.S. degree in Mechanical Engineering from the North China University of Technology, in 2004, M.S. degree in Control Theory and Engineering and Ph.D. degree in Mechanical Engineering, both from Beijing Institute of Technology, in 2007 and 2010 respectively. His current research interests include medical robotics, mechanical design, path planning, dynamics modeling and control.

Zeng-Guang Hou received the B.S. degree and M.E. degree in Electrical Engineering from the Yanshan University (formerly North-East Heavy Machinery Institute), China, in 1991 and 1993, respectively, and the Ph.D. degree in Electrical Engineering from the Beijing Institute of Technology in 1997. He is now a Professor in the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include robotics, neural networks, optimization algorithms, and intelligent control systems.

Zhen-Qiu Feng received the B.S. degree in Automation from the Central South University, China, in July 2010. He is currently working toward the Ph.D. degree in Control Theory and Control Engineering at the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. He is also with University of Chinese Academy of Sciences, Beijing. His current research interests include surgical robot control and system, medical image segmentation and reconstruction.

Jian-Long Hao received the B.S. degree in Automation from the Northwestern Polytechnical University, China, in July 2012. He is currently working toward the Ph.D. degree in Control Theory and Control Engineering at the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. He is also with University of Chinese Academy of Sciences, Beijing. His current research interests include surgical simulation, human-robot interaction and haptics.