Neurocomputing 146 (2014) 75–82

Contents lists available at ScienceDirect

Neurocomputing

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

# A study on residence error of training an extreme learning machine and its application to evolutionary algorithms

Ai-Min Fu [a], Xi-Zhao Wang [b], Yu-Lin He [b,*], Lai-Sheng Wang [a,**]

[a] College of Science, China Agricultural University, Beijing 100083, China
[b] Key Laboratory of Machine Learning and Computational Intelligence in Hebei Province, School of Mathematics and Computer Science, Hebei University, Baoding 071002, China

ARTICLE INFO

*Article history:*
Received 23 September 2013
Received in revised form
9 April 2014
Accepted 14 April 2014
Available online 24 July 2014

*Keywords:*
Extreme learning machine
Genetic algorithm
Rank of matrix
Residence error
Solution stability

ABSTRACT

This paper delivers a study on the change of rank of input matrix in Extreme Learning Machine (ELM) and the relationship between the rank of input matrix and the residence error of training an ELM. From the viewpoint of data analysis, the study reveals why ELM has a decreasing residence error with the increase of number of nodes in hidden layer and what role the Sigmoid function plays in increasing the rank of input matrix. Furthermore the relationship between the stability of solutions and the rank of output matrix is also discussed. An application of residence error to genetic algorithms of minimizing $L_1$-norm ELM is given.



## 1. Introduction

Extreme learning machines (ELMs) proposed in [14,15] are a type of single hidden layer feed-forward neural networks (SLFNs) in which the weights between input layer and hidden layer are chosen randomly while the weights between hidden layer and output layer are obtained by solving a system of linear matrix equations. ELMs adopt the sum of squared losses on the training error as the objective function, and then turn training of output weights into a regularized least square problem. It has been shown that SLFNs only with randomly generated input weights and tunable output weights can maintain their universal approximation ability [10,11,29]. In comparison with gradient-descent based algorithms, ELMs have much more efficient training and usually lead to better generalization performance [21,25,27].

One can find considerable references [2,12,13,18,24–29] in recent decade regarding the ELM study. We are now interested in ELM's training residence error and approximation capability, and a very brief literature review is given as follows. Hornik in [6] proved that, if the activation function is continuous, bounded, and non-constant, then continuous mappings can be approximated by SLFNs with additive hidden nodes over compact input sets. Leshno et al. in [16] improved the result of [6] by proving that SLFNs with additive hidden nodes and with a non-polynomial activation function can approximate any continuous target functions [16]. Huang and Babri [7] show that an SLFN with at most $N$ hidden nodes and with almost any nonlinear activation function can learn $N$ distinct observations with zero error, where $N$ is the number of training samples. Furthermore, Huang et al. [8–10] recently proposed a series of learning algorithms referred to as incremental extreme learning machines (I-ELMs) where the number of hidden layer nodes are gradually added and showed that such I-ELMs can converge to any continuous function as long as the hidden activation functions are nonlinear piecewise continuous. Following [9], Feng et al. [4] proposed an error minimized extreme learning machine (EM-ELM), which can add random hidden nodes to SLFNs one by one or group by group (with varying group size). During the growth of the networks, the output weights are updated incrementally. The convergence of this approach is proved.

Based on the result of [1] in which the authors pointed out that for the feed-forward neural networks, the smaller the norm of weights and training error is, the better generalization performance the networks tend to have, all ELM algorithms tend to find the minimum norm least square solution such that a smaller training error can be achieved. This study focuses on the change of rank of input matrix in ELM and the relationship between the rank of input matrix and the residence error of training an ELM. It theoretically confirms that the increase of input dimension given by random weights and the increase of rank of middle matrix

* Corresponding author. Tel.: +86 18531315747.
** Corresponding author.
*E-mail addresses:* yulinhe@ieee.org (Y.-L. He), wanglaish@126.com (L.-S. Wang).

http://dx.doi.org/10.1016/j.neucom.2014.04.067
0925-2312/

induced by Sigmoid transformation play the crucial role in the entire process of training an ELM.

All existing references indicate that ELM is a useful and efficient technique for supervised learning. But there is no article yet to clearly explain why the ELM can effectively work well with a simple structure and fast training. This paper makes an attempt to give an explanation from the angle of relationship between ELM's training residence error and the rank of input matrix.

The rest of this paper is organized as follows. Section 2 lists a brief review on the approximation ability and error analysis of ELMs. Section 3 investigates the increase of dimension for input matrix and the relationship of rank between input matrix and middle matrix, and Section 4 studies the impact of Sigmoid transformation on the increase of rank of output matrix. Section 5 studies the estimation of residence error and stability of solution. Then, an application of residence error to genetic algorithms of minimizing $L_1$-norm ELM is given in Section 6. Section 7 of this paper presents our conclusion.

## 2. Extreme learning machine

Usually an ELM means a three layer neural network in which the weights between input layer and hidden layer are randomly selected and the weights between hidden layer and output layer are determined by solving a generalized system of linear equations (i.e., by computing the pseudo inverse of a matrix). Fig. 1 depicts the basic structure of an ELM in which we suppose that the input layer has $n$ nodes, the hidden layer has $m$ nodes, and the output layer have has only one node.

We now analyze the training process of an ELM. The training task is to determine the connection weights $r_{ij}$ and $\beta_j$ $(i=1,2,\cdots,n;\ j=1,2,\cdots,m)$. Since the weights $r_{ij}$ $(i=1,2,\cdots,n;\ j=1,2,\cdots,m)$ are randomly selected, the training task is reduced to determine $\beta_j$ $(j=1,2,\cdots,m)$ only. Suppose that the set of training data contains $N$ examples which can be expressed as an input matrix $A$ (with $N$ rows and $n$ columns) and a $N$-dimensional output vector $b$, respectively denoted by

$$A_{N\times n}=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{N_1} & a_{N_2} & \cdots & a_{N_n} \end{pmatrix} \text{ and } b_{N\times 1}=\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{pmatrix}.$$

The weights between the input layer and hidden layer are expressed as a matrix with $n$ rows and $m$ columns, i.e., $R=(r_{ij})_{n\times m}$, and the weights between the hidden layer and output layer are denoted as an $m$-dimensional vector, i.e., $\beta=(\beta_1,\beta_2,\cdots,\beta_m)^T$. Let

$$S_{N\times m}\triangleq \underset{N\times n}{A}\ \underset{n\times m}{R}=(s_{ij})_{N\times m} \text{ and } \underset{N\times m}{H}\triangleq (f(S_{ij}))_{N\times m}=(h_{ij})_{N\times m},$$

where $f(x)=(1/1+e^{-x})$ denotes the Sigmoid function. Then the training task of the ELM is transferred to solve the following system of linear equations $H\beta=b$, which is equivalent to the following optimization problem:

$$\min_{\beta\in R^m}\| \underset{N\times 1}{b}-\underset{N\times m}{H}\ \underset{m\times 1}{\beta}\|^2. \tag{1}$$

Obviously the solution of Eq. (1) is not unique in a general case. From the viewpoint of regularization, Huang et al. [10,13,15] suggested to use the minimum-norm minimum least square solution as the final one:

$$\min\left(\min_{\beta\in R^m}\| \underset{N\times 1}{b}-\underset{N\times m}{H}\ \underset{m\times 1}{\beta}\|^2\right). \tag{2}$$

It is easy to see that the solutions of Eqs. (1) and (2) can be respectively expressed as

$$\beta=H^- b \text{ and } \beta=H^+ b,$$

where $H^-$ denotes any generalized inverse matrix $H$ while $H^+$ denotes the plus-generalized inverse this is unique for an matrix $H$.

We divided the above-mentioned training process of an ELM as three steps: (1) dimension increase for input matrix; (2) rank increase for output matrix; and (3) solving a system of linear equations with full rank matrix of coefficients. The three steps are depicted in Fig. 2.

Step 1 shows a process of increasing dimension of input matrix $A$ since in almost every case of ELM applications the number of hidden nodes is much bigger than the number of input nodes. Ref. [24] discussed the impact of increasing dimension of input matrix and pointed out that without the dimension increase the ELM will not obtain a good generalization and approximation ability. In fact, the central supporting theorem of EML algorithms, given by Zhang et al. in [29], stated a process of approximation with the increase of number of hidden nodes.

Step 2 gives a process of increasing rank of input matrix. Although in step 1 the input matrix $A$ (with $N$ rows and $n$ columns) becomes $S$ (with $N$ rows and $m$ columns) through the multiplication to a random weight matrix $R$ and $m$ is bigger than $n$, the rank of matrix $S$ is less than or equal to the rank of matrix $A$. It is because the step 1 is only a linear transformation (the simple proof remains in next section). The essence of step 2 is a nonlinear transformation. ELM uses a Sigmoid function which usually plays a role of transforming from a waning rank matrix $S$ to a full rank matrix $H$.

Step 3 means to solve a system of linear equations. If the output layer has more than one node then it is a system of linear matrix equations. It is well known that the criterion of least square is frequently used to solve the system. It is evaluated by the approximation error (i.e., the residence error). Here we are mainly interested in the relationship between the approximation error and the rank of coefficient matrix.

To be convenient for our following discussions, we summarize the used symbols or notations as follows:

$A$—input matrix;
$R$—random weight matrix;
$\beta$—weight vector to be solved;
$S$—middle matrix;
$H$—output matrix;
$H^+$—solution matrix;
$b$—expected output vector.

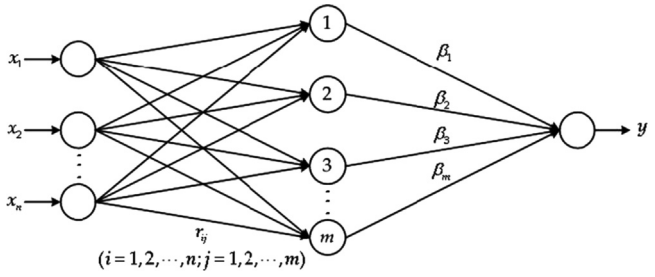



**Fig. 1.** A simple ELM structure.

## 3. Increase of dimension for input matrix

This section has two aims. One is to make clear the impact of dimension increase (from $A$ to $S$) on the solution of $H\beta=b$, the

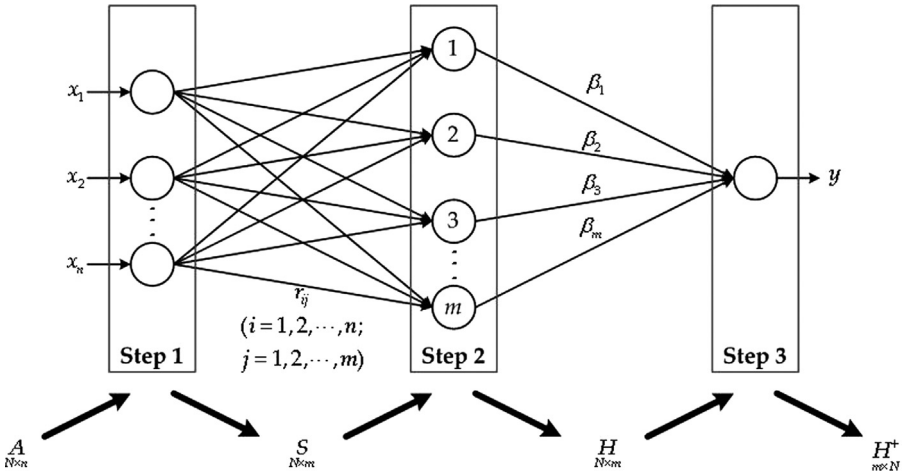



**Fig. 2.** Three steps for training an ELM.

other is to find the change tendency of the residence error $||H\beta-b||$ when the matrix is becoming matrix $S$.

Suppose that the input matrix $A$ becomes a middle matrix $S$ via the following transformation $S=AR$ where $A$ has $N$ rows and $n$ columns, $R$ is the random weights matrix with $n$ rows and $m$ columns, and $S$ is the middle matrix with $N$ rows and $m$ columns. We generally consider that $m$ is bigger than $n$, and therefore, this is a process of increasing dimension of input matrix. We have the following:

**Proposition 1.** *Rank(S) ≤ Rank(A).*

*One can find the proof of Proposition 1 from a fundamental linear algebra textbook.*

**Proposition 2.** *Suppose that R is a full rank matrix with n rows and m columns. Let S = AR be the middle matrix with N rows and m columns. Then*

$$\min_{\alpha\in R^n}||A\alpha-b||=\min_{\beta\in R^m}||S\beta-b||, \tag{3}$$

where $b$ is the expected output vector and $R$ is a full-rank matrix.

**Proof.** Let $b\in R^N$, $L\subset R^N$.Then, the distance between $b$ and $L$ can be defined as

$$D(b,L)=\min_{x\in L}||x-b||,$$

where $||x-b||=\sqrt{(x-b)^T(x-b)}$ denotes the distance between two $N$-dimensional vectors $x$ and $b$. Rewrite $A=\{a_1,a_2,\cdots a_n\}$ as a set of $N$-dimensional vectors and let $Span(A)$ denote the linear space spanned by $\{a_1,a_2,\cdots,a_n\}$, i.e.,

$$Span(A)=\{k_1a_1+k_2a_2+\cdots+k_na_n|k_j\in R^{'},\ j=1,2\cdots n\}.$$

Then,

$$D(b,Span(A))=\min_{\alpha\in R^N}||A\alpha-b||. \tag{4}$$

Similarly,

$$\min_{\beta\in R^m}||S\beta-b||=D(b,Span(S))=D(b,Span(AR)). \tag{5}$$

Noting that $R$ is a full-rank matrix, we have $Span(A)=Span(AR)$ which implies Eq. (4) is identical to Eq. (5). The proof is completed. □

The right side of Eq. (3) denotes the residence error in solving the system $S\beta=b$ while the left denotes the residence error in solving the original problem $A\alpha=b$. It is easy to see from Fig. 1 that, if we delete the step 2, then $S$ will be identical to $H$, and the solution of $S\beta=b$ will be the final solution $H^+b$. Proposition 2 tells us such a fact that the dimension-increase of input matrix has no impact on the reduction of residence error of the system of linear equations if we do not have the Sigmoid transformation in step 2.

**Proposition 3.** *Let $H_{N\times m}=\{h_1,h_2,\ldots,h_m\}$ and $H^*_{N(m+1)}=\{h_1,h_2,\ldots,h_m,h_{m+1}\}$ denote two sets of vectors, $h_j$ is an N-dimensional vector $(1\le j\le m+1)$,*

$$\beta_{m\times1}=(\beta_1,\beta_2,\ldots,\beta_m)^T,\beta^*_{(m+1)1}=(\beta_1,\beta_2,\ldots,\beta_m\beta_{m+1})^T.$$

*Then,*

$$\min_{\beta^*\in R^{m+1}}||b-H^*\beta^*||\le\min_{\beta\in R^m}||b-H\beta||.$$

**Proof.** Noting $H\subset H^*$, we have $Span(H)\subset Span(H^*)$, which implies

$$\min_{\beta\in R^{m+1}}||b-H^*\beta^*||=D(b,Span(H^*))\le D(b,Span(H))=\min_{\varphi\in R^m}||b-H\beta||.$$

The proof is completed. □

Since the dimension-increase of input matrix is equivalent to add nodes of hidden layer, Proposition 3 tells us a conclusion that, when we incrementally train an ELM by gradually adding nodes in hidden layer, the training error is monotonically decreasing. It is worth pointing out that the conclusion has been included in the Lemma 3.1. [4]. In comparison with the Lemma 3.1 in [4], the current proof is more strict and understanding.

## 4. Sigmoid transformation leading to an increase of rank of output matrix

We now focus on step 2, i.e., the transformation from matrix $S$ to matrix $H$. From the definition of step 2, we know

$$H_{N\times m}=(h_{ij})_{N\times m}=(Sigm(s_{ij}))_{N\times m},$$

where $S_{N\times m}=(s_{ij})_{N\times m}$, $Sigm(x)=(1/1+e^{-x})$, $x\in R'$.

We have two aims in this section. One is to make clear the change of rank from $S$ to $H$ while the other is to establish relationship between the rank of $H$ and the residence error $||H\beta-b||$.

**Example 1.** Let $S=\begin{pmatrix}2&1\\1&a\end{pmatrix}$ where $a=\ln\frac{e+e^{-1}}{2}$. It is easy to check that $S$ is a full-rank matrix. After conducting the Sigmoid

transformation, matrix $S$ is transferred to

$$H=\begin{pmatrix} \frac{1}{1+e^{-2}} & \frac{1}{1+e^{-1}} \\ \frac{1}{1+e^{-1}} & \frac{1}{1+e^{-a}} \end{pmatrix}.$$

Noting that $|H|=0$ (where $|H|$ denotes the determinant of matrix $H$), we get that Matrix $H$ is not full rank.

**Example 1.** shows that it is possible that the Sigmoid transformation transfers a full rank matrix to a waning rank matrix.

**Proposition 4.** *Suppose that* $S=\{s_1,s_2,\ldots,s_N\}$ *denotes a set of n-dimensional vectors,* $s_i=(s_{i1},s_{i2},\cdots,s_{in})$, $i=1,2,\ldots,N$, $s_i\neq s_j(i\neq j)$. $1\leq Rank(S)<n$. *Then, with probability 1, the Sigmoid transformation will transfer S into a set of vectors of full rank, i.e., with probability 1* $Rank(H)=n$ *where* $H=\{h_1,h_2,\ldots,h_N\}$, $h_i=(h_{i1},h_{i2},\ldots,h_{in})$, $h_{ij}=Sigm(s_{ij})=(1+e^{-s_{ij}})^{-1}$, $i=1,2,\ldots,N$, $j=1,2,\ldots,n$.

**Proof.** For simplicity we prove the proposition with $n=3$. For general case we can similarly complete the proof. It can be divided as two cases: $Rank(S)=1$ and $Rank(S)=2$. The first case indicates that all vectors in $S$ are located in a line. After Sigmoid transformation, the line will be transferred a curve in 3-dimensional space. We prove that with probability 1 the curve is a spatial curve (i.e., the probability with that the curve located in a plane is zero).

Consider the Sigmoid transformation

$$(x_1,x_2,x_3)\mapsto\left(\frac{1}{1+e^{-x_1}},\frac{1}{1+e^{-x_2}},\frac{1}{1+e^{-x_3}}\right)\in(0,1]^3.$$

Without losing generality, let $x=\lambda v$ denote a line passing through the original, where $v\in R^3$, $\lambda$ is a parameter. Define

$$V(\lambda)=\left(\frac{1}{1+e^{-\lambda v_1}},\frac{1}{1+e^{-\lambda v_2}},\frac{1}{1+e^{-\lambda v_3}}\right).$$

From [22] we know that $V(\lambda)$ is a spatial curve if and only if the torsion of $V(\lambda)$ is not zero where the torsion is defined as $\tau=-(\mathrm{d}\gamma/d\lambda)\beta$ where

$$\gamma=\frac{V'(\lambda)V''(\lambda)}{|V''(\lambda)|}\ \text{and}\ \beta=\frac{V''(\lambda)}{|V''(\lambda)|}.$$

Noting that Sigmoid $f(y)=(1/1+e^{-y})$ has the properties $f'=f(1-f)$ and $f''=f(1-2f)$, we can directly evaluate the torsion and get that the torsion $\tau=0$ if and only if two components of $v$ are zero (i.e., each of the three axes of coordinates), which implies that with probability 1 the curve $V(\lambda)$ is a spatial curve. In this way we have

$$\begin{aligned}&\text{Probability}\ (H\ \text{is a waning rank matrix})\\ &=\text{Probability}\ (V(\lambda)\text{is anon}-\text{spatial curve})=0.\end{aligned}\quad\square$$

Fig. 3 shows an example of the transformed spatial curve. For the case of Rank($S$)=2 which indicates that all vectors of $S$ are located in a plane, it is easy to check that the Sigmoid transformation will change the plane to a special curved surface $\Omega$ in $(0,1]^3$. Then

$$\begin{aligned}&\text{Probability}\ (\text{H is a waning rank matrix})\\ &\leq\text{Probability}\ (\{v|v\in S,\ Sigm(v)\subset\Omega\cap P,\ P\ is\ any\ plane\ or\ line\})\\ &\leq\text{Probability}\ (\{v\in R^3|v\ dropping\ in\ a\ plane\ area\})=0.\end{aligned}$$

Fig. 4 shows an example of intersection of a plane and its Sigmoid transformation. When $n>3$, the proof is still available while some concepts of differential geometry are used, but the expression is rather complicated. Here we do not show the part of $n>3$. The proof is completed.

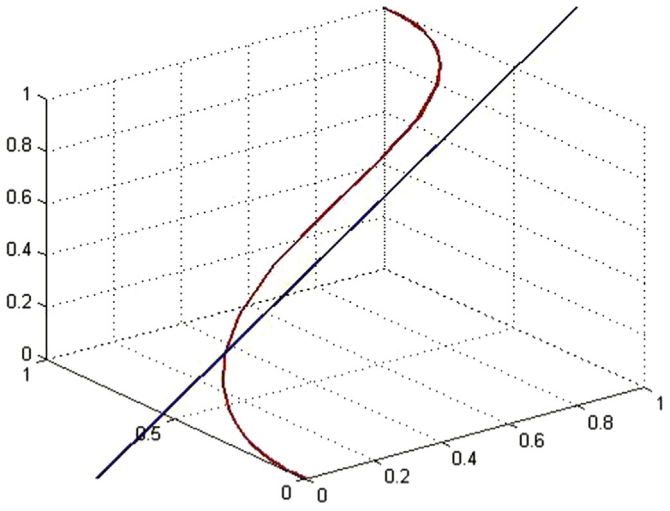



**Fig. 3.** An example of spatial curve transformed by Sigmoid.

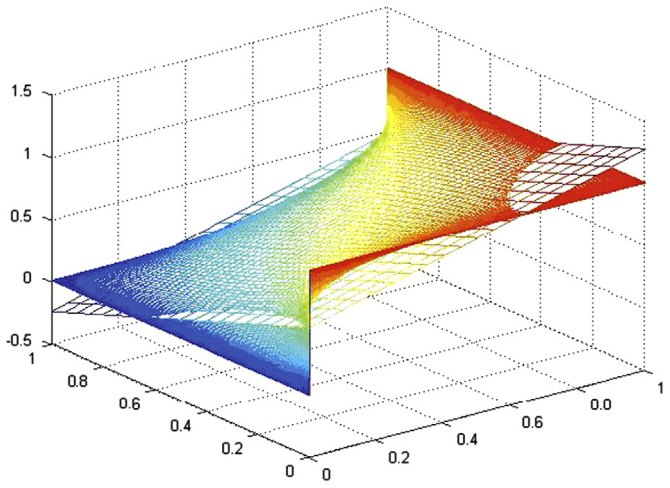



**Fig. 4.** An example of intersection of a plane and its Sigmoid transformation.

**Proposition 4.** *indicates that the Sigmoid transformation will transfer a matrix S before step 2 to a full-rank matrix H after step 3 with probability 1. Noting that S is coming from step 1 via a linear transformation with dimension increase and is generally of waning rank, we can clearly see the effect of rank increase of Sigmoid transformation. There is an essential difference between the full and waning rank matrices for the residence error computation and the stability of least square solution based on pseudo inverse of matrix. Section 5 shows that difference.*

Lots of numerical experiments show that a same conclusion can also be obtained for the infinitely differential activation function. However, we do not have a strictly mathematical proof on this speculation. The major difficulty involves some specific properties of the function. We are trying to extract the common conditions of the set of functions and then to derive a proof.

Besides the Sigmoid activation function discussed in this paper, other sigmoidal functions, e.g., radial basis, sine, cosine, exponential, and many nonsingular functions shown in [7] are considered to have the similar impacts and conclusions. But there is not a rigorous proof.

## 5. Estimation of residence error and stability of solution

Focusing on step 3 in Fig. 2., we discuss the difference for solving linear system of equations $H\beta=b$ between full rank $H$ and

waning rank $H$, and the related issues of solution stability. Since the solution can be expressed as $\beta = H^+ b$, we consider the issue of continuity for generalized inverse $H^+$. $H^+$ is referred to as continuous if and only if

$\lim_{||\delta H||\to 0}(H+\delta H)^+ = H^+$,

where $\delta H$ denotes a perturbation of matrix $H$.

**Proposition 5.** *The generalized inverse $H^+$ is continuous if $H$ is a full-rank matrix.*

**Proof.** Without loosing generality we suppose that the rank of $H$ is $n$, which implies that $H^T H$ is a $n \times n$ non-singular matrix. In fact it is a symmetric and positive matrix and $H^+ = (H^T H)^{-1} H^T$. Denote the perturbation matrix by $\delta H$, then we have

$(H+\delta H)^T (H+\delta H) = H^T H + (H+\delta H)^T \delta H + (\delta H)^T H.$

According to Banach theorem we know that $(H+\delta H)^T (H+\delta H)$ is a non-singular matrix if $||(H^T H)^{-1}[(H+\delta H)^T(\delta H)+(\delta H)^T H]|| < 1$. It is easy to see that we always take the $||\delta H||$ small enough such that the equality holds well. In other words, there exists a small positive number $\eta$ such that the inequality holds well if $||\delta H|| \le \eta$. In this way, the generalized inverse matrix can be expressed as

$(H+\delta H)^+ = [(H+\delta H)^T (H+\delta H)]^{-1} (H+\delta H)^T.$

Let $||\delta H|| \to 0$, we have

$\lim_{\|\delta H\|\to 0}[(H+\delta H)^T(H+\delta H)]^{-1} = (H^T H)^{-1}$ and $\lim_{||\delta H||\to 0}(H+\delta H)^T = H^T$,

which implies $\lim_{||\delta H||\to 0}(H+\delta H)^+ = (H^T H)^{-1} H^T = H^+$. It just is the conclusion of this proposition. □

The continuity of generalized inverse $H^+$ plays an essential role for getting a stable solution. Since the input matrix of ELM is transferred to a middle matrix by multiplying a group of random weights and then the weight perturbation is unavoidable, the discussion about the solution stability is particularly meaningful. Full rank of $H$ results in the continuity of $H^+$. The following example shows that for a matrix $H$ with waning rank, its generalized inverse $H^+$ is generally discontinuous.

**Example 2.** Let $H = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$, than $rank(H) = 1$. $H$ is no of full rank. It is easy to calculate that $H^+ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$. Suppose that $\delta H = \begin{bmatrix} 0 & 0 \\ 0 & \varepsilon \\ 0 & 0 \end{bmatrix}$, $\varepsilon \neq 0$, then $H+\delta H = \begin{bmatrix} 1 & 0 \\ 0 & \varepsilon \\ 0 & 0 \end{bmatrix}$. Noting that the matrix $H+\delta H$ is of full rank, i.e., $rank(H+\delta H) = 2 > rank(H)$, we get $(H+\delta H)^+ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \varepsilon^{-1} & 0 \end{bmatrix}$.

It is easy to see that the limit of $(H+\delta H)^+$ does not exist when $||\delta H|| \to 0$.

For a matrix of full rank, the generalized inverse can be expressed as $H^+ = (H^T H)^{-1} H^T$. Based on this expression, we can write the residence error as

$||b - H\beta||^2 = b^T b - b^T (H(H^T H)^{-1} H^T) b.$

We continue to discuss $\beta = H^+ b$, the solution of $H\beta = b$. Suppose that there are two perturbations for coefficient matrix $H$ and right vector $b$, $\delta H$ and $\delta b$, respectively. The corresponding perturbation equations and their solutions are denoted as

$(H+\delta H)(\beta+\delta\beta) = b + \delta b$

and

$(\beta+\delta\beta) = (H+\delta H)^+ (b+\delta b),$

respectively. Furthermore supposing that the perturbation $\delta H$ is small enough such that the rank of $H$ has not change, we estimate $\delta\beta$, i.e., the impact of perturbation on the solution.

**Proposition 6.** *Suppose that (1) $rank(H+\delta H) = rank(H)$ and (2) $||H^+||||\delta H|| < 1$. Then we have*

$$||\delta\vec{x}|| \le \frac{||H^+||}{1-\Delta}\left(2||\delta H||\vec{x}|| + ||\delta\vec{b}|| + \frac{\Delta||\vec{r}||}{1-\Delta}\right),$$

*where $\Delta = ||H^+||||\delta H||$ and $\vec{r} = \vec{b} - H\vec{x}$.*

**Proof.** From $\vec{x} + \delta\vec{x} = (H+\delta H)^+(\vec{b} + \delta\vec{b})$ and $\vec{x} = H^+\vec{b}$ we obtain that

$\delta\vec{x} = \{(H+\delta H)^+ - H^+\}\vec{b} + (H+\delta H)^+ \delta\vec{b}.$

By a direct derivation we can rewrite $\delta\vec{x}$ as

$$\delta\vec{x} = -B^+(\delta H)\vec{x} + B^+ B^{+T}(\delta H)^T\vec{r} + (I - B^+ B)(\delta H)^T H^{+T}\vec{x} + B^+(\delta\vec{b}), \quad (6)$$

where $B = H+\delta H$. The condition (2) $||H^+||||\delta H|| < 1$ implies that $||B^+||$ is upper bounded:

$$||B^+|| \le \frac{||H^+||}{1 - ||H^+||||\delta H||}.$$

Taking norm for both sides of Eq. (6) we have

$$\begin{aligned}
||\delta\vec{x}|| &\le ||B^+||||\delta H||||\vec{x}|| + ||B^+||^2||\delta H||||\vec{r}|| + ||\delta H||||H^+||||\vec{x}|| + ||B^+||||\delta\vec{b}|| \\
&= ||B^+|(||\delta H||||\vec{x}|| + ||\delta\vec{b}||) + ||B^+||^2||\delta H||||\vec{r}|| + ||\delta H||||H^+||||\vec{x}|| \\
&\le \frac{||H^+||}{1-\Delta}(||\delta H||||\vec{x}|| + ||\delta\vec{b}||) + ||H^+||||\delta H||||\vec{x}|| + \frac{||H^+||^2}{(1-\Delta)^2}||\delta H||||\vec{r}|| \\
&= \frac{\Delta||\vec{x}|| + ||H^+||||\delta\vec{b}||}{1-\Delta} + \frac{\Delta||H^+||||\vec{r}||}{(1-\Delta)^2} + \Delta||\vec{x}|| \\
&= \frac{\Delta||\vec{x}|| + ||H^+||||\delta\vec{b}|| + \Delta||\vec{x}|| - \Delta^2||\vec{x}||}{1-\Delta} + \frac{\Delta||H^+||||\vec{r}||}{(1-\Delta)^2} \\
&= \frac{1}{1-\Delta}\left(2\Delta||\vec{x}|| + ||H^+||||\delta\vec{b}|| - \Delta^2||\vec{x}|| + \frac{\Delta||H^+||||\vec{r}||}{1-\Delta}\right). \quad (7)
\end{aligned}$$

Replacing $\Delta$ in numerator of Eq. (7) with $\|H^+\|\|\delta H\|$, we get the estimation

$$\begin{aligned}
||\delta x|| &\le \frac{||H^+||}{1-\Delta}\left(2||\delta H||||\vec{x}|| - ||H^+||||\delta H||^2||\vec{x}|| + ||\delta\vec{b}|| + \frac{\Delta||\vec{r}||}{1-\Delta}\right) \\
&\le \frac{||H^+||}{1-\Delta}\left(2||\delta H||||\vec{x}|| + ||\delta\vec{b}|| + \frac{\Delta||\vec{r}||}{1-\Delta}\right),
\end{aligned}$$

which completes the proof. □

This proposition tells us such a result that, if the residence error $||\vec{r}|| = ||\vec{b} - H\vec{x}||$ is big, then the perturbation has a much impact on the solution.

To numerically verify the relationship between the residence error $||b - H\beta||^2$ and solution stability, i.e., to observe the changes of residence error and solution stability with the increase of rank of $H$, we conduct a numerical experiment. Assume that $H$ is a matrix with 150 rows and 100 columns, thus the full rank

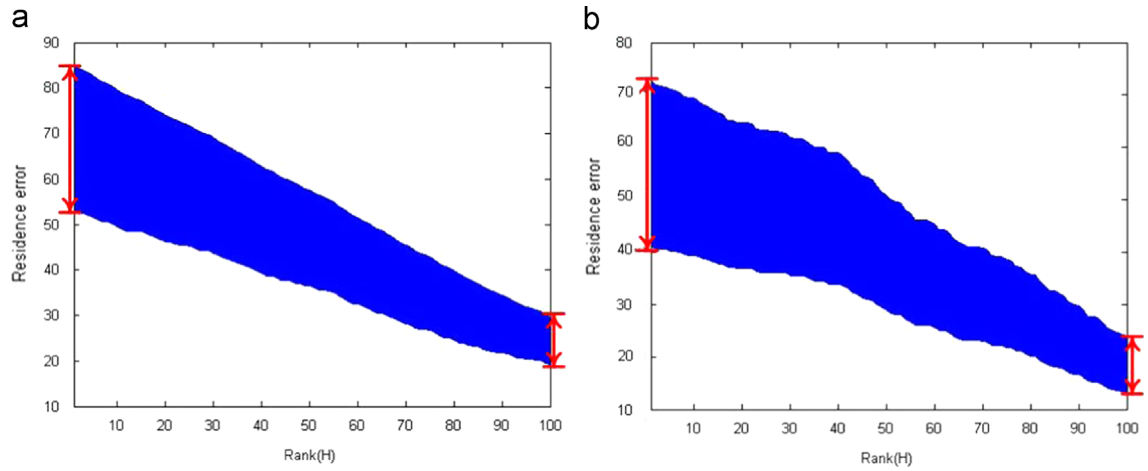



**Fig. 5.** The varieties of residence error and solution stability. (a) Classification with $b_1$ and (b) regression with $b_2$.

of $H$ is 100. Without loss of generality, let $H=(h_{ij})_{150\times100}, h_{ij}=\begin{cases}0.9+\delta & , \text{ if } i=j\le r\\ 0 & , \text{ otherwise}\end{cases}$ for $rank(H)=r$, $r=1,2,\cdots,100$. And for each $r$, let $\delta$ range from $-0.1$ to 0.1 with a step of 0.0001. Here, the parameter $\delta$ plays the role of perturbation. We consider two different learning tasks, i.e., $b$ is a matrix for classification ($b_1$) and $b$ is a vector for regression ($b_2$): $b_1=(b_{ij}^{(1)})_{150\times3}$ meets $\sum_{i=1}^{150}\sum_{j=1}^{3}b_{ij}^{(1)}=150$ and $\sum_{j=1}^{3}b_{ij}^{(1)}=1$ and $b_2=(b_i^{(2)})_{150\times1}$, where $b_i^{(2)}$ is the random number in interval [0, 1], $i=1,2,\cdots,150$. The detailed results are presented in Fig. 5.

From Fig. 5 we can obviously find that (1) with the increase of rank of $H$, the residence errors for both classification and regression decrease gradually, and (2) with the increase of rank of $H$, the solutions of $H\beta=b$ become more and more stable (because these blue bands become more and more narrow). This indicates that the variances of solutions reduce gradually. In summary, this experiment demonstrates the theoretical analysis mentioned above, i.e., in comparison with $H$ having waning rank, the full-rank matrix can obtain the smaller residence error. And, the full-rank matrix $H$ is more insensitive to the perturbation and can get the more stable solution for $H\beta=b$.

More numerical experiments have been conducted for different types of $H$ with $rank(H)=r$, $r=1,2,\cdots,100$. The experimental results are basically similar to that shown in Fig. 5.

## 6. An application to genetic algorithms of minimizing $L_1$-norm ELM

The residence error is deceasing with the increase of nodes of hidden layer, and depends strongly on the rank of output matrix. This conclusion can give some guidelines for choosing hidden nodes and training ELM as follows:

(1) The maximum number $m$ of hidden nodes is equal to the number $N$ of distinct training samples. (It is based on the conclusion in [15]: when $m=N$, the matrix $H$ is square and invertible if the input weights $r_{ij}$ and the hidden biases $b_i$ are randomly chosen, and ELM can approximate these training samples with zero error.
(2) Due to the decrease of residence error, we can determine the number of hidden nodes by gradually adding nodes till an acceptable accuracy.
(3) Since the matrix $H$ will be of full-rank with probability 1, we basically do not need to worry about the singular case caused by adding hidden nodes.

We know that training an ELM is to minimize a square error function, which can be finally transferred to a problem of solving a pseudo inverse of matrix. This transformation can be conducted since the square error function has quite good analytic properties. Motivated by reducing the effect of noisy data, the objective function to be minimized, i.e., the square error function is often replaced with an absolute error function. It means that we need to minimize a function including absolute operations rather than to minimize a square error function. Some related research can be found from references [3,19,23]. Mathematically it is a problem of optimization in $L_1$-space.

Since the absolute function is continuous but not differentiable, we cannot derive a simplified form similar to the case of square error function. Similar to Eq. (1), the mathematical model can be formulated as $\min_{\beta\in R^m}|b_{N\times1}-H_{N\times m}\beta_{m\times1}|$ where symbols $b$, $H$, and $\beta$ have the meaning as same as in Section 2, and for a given vector $x=(x_1,x_2,\ldots,x_m)^T$ the $L1$-norm is defined as $|x|=|x_1|+|x_2|+\ldots+|x_m|$. In comparison with minimizing Eq. (1), the absolute function optimization is much more difficult. It cannot be transferred to a problem of solving a system of linear equations although the input matrix $H$ is of full-rank. From references one can find some existing study on the absolute function optimization [5,17,20], the genetic algorithm is one of the most feasible and effective methodologies. When the number of hidden layer nodes of an ELM is given, a genetic algorithm for minimizing the absolute error function is described as follows:

A genetic algorithm for training $L_1$-ELM:

Objective function to be minimized: $F(\beta)=|b_{N\times1}-H_{N\times m}\beta_{m\times1}|$ where $b$ and $H$ are defined as in Section 2;

Parameters to be determined: $\beta=(\beta_1,\beta_2,\ldots,\beta_m)^T$;

Step 1: Population initialization. Usually there are two ways to initialize the population of parameters. One is to randomly choose from given intervals while the other is to use some heuristic information which can help speed up the convergence and reduce the iteration number. Let $M$ be the population size;
Step 2: Coding. A coding mechanism is used for coding each chromosome of the population. We use the bit-coding in our approach, i.e., each chromosome is represented as a string of either 0 or 1 with fixed length;
Step 3: Crossover. For each chromosome, a mate-chromosome is randomly selected from the unassigned chromosomes. Then the population is considered as a set of chromosome-pairs. For each pair in the population, $N$ tail-genes are exchanged, where $N$ is a given number. Then a new pair of chromosomes is generated. Usually we set up a crossover probability;

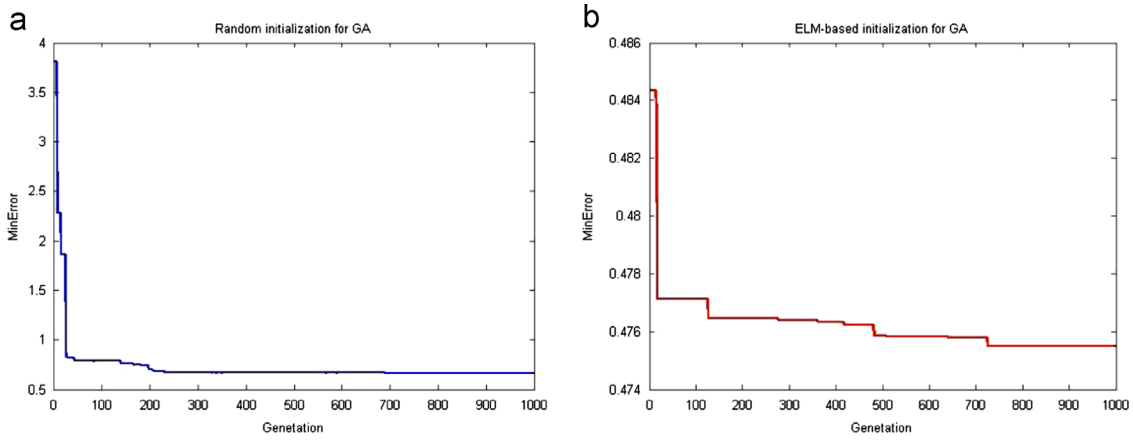



**Fig. 6.** Error change for two mechanisms of population initialization. (a) Random population and (b) ELM-based population.

Step 4: Mutation. With a mutation probability, randomly select a number of genes from all new chromosomes, and change these gene-values from 0 to 1 or from 1 to 0;
Step 5: According to the objective function, a fitness for each chromosome is evaluated. Find the best chromosome which has the minimum fitness value;
Step 6: Repeat steps 3–5 until the iteration times reach a predefined number. Output the best chromosome and its corresponding fitness value.

A key step for designing a genetic algorithm is the population initialization, which has a direct impact on the convergence performance of the algorithm. A better initialization can significantly increase the convergence but a worse initialization may lead the iteration divergence. We try to apply the result of ELM residence error to the population initialization. We know that the optimization problem $\min_{\beta \in R^m} |b_{N\times k} - H_{N\times m}\beta_{m\times k}|$ is not equivalent to $\min_{\beta \in R^m} \|b_{N\times k} - H_{N\times m}\beta_{m\times k}\|^2$. The vector which makes the square error function attain minimum is generally not the vector which makes the absolute error function attain minimum. However, from the angle of manifold regularization, one may think of there exists an implicit relation between both vectors. The manifold regularization framework has a fundamental smoothness assumption, that is, if two points $x_1$ and $x_2$ are close to each other, then the conditional probabilities $(y|x_1)$ and $P(y|x_2)$ should be similar as well. In this way, if we consider optimization problem for a set of functions: $\{F(\beta;\lambda) = ||b - H\beta||^{\lambda} | \lambda > 0\}$, we consider there are some relations among the minimum-points for different values of $\lambda$ although it is difficult to find the exact relation. Since the case of $\lambda = 2$ is easy to handle, we choose its solutions for the fixed random weights and for different numbers of hidden layer nodes as seeds of initial chromosomes in the population. The rest chromosomes of the population are randomly generated.

**Example 3.** We try to train an ELM with absolute error minimization for $N = 150$, $n = 3$, and $m = 4$ where $N$, $n$, and $m$ denote the number of raining samples, the number of input nodes, and the number of hidden layer nodes, respectively. The typical Iris data set is selected. Suppose that the population size $M$ is 50 and (1) all initial chromosomes are randomly generated from $[-50, 50]$ or (2) the population contains some seed chromosomes given by solving the regular ELMs for m ranging from 4 to 8 and the perturbations of these seeds. The maximum number of generation, the probability of crossover, and the probability of mutation are 1000, 0.9 and 0.1, respectively. The error changes with the iteration increasing as shown in Fig. 6.

From Fig. 6(a) and (b), one can see that, regarding the genetic algorithm of solving an absolute ELM problem, the population initialization based on our ELM-residence and manifold regularization plays a real role of speeding-up convergence.

## 7. Conclusions

This paper presents a theoretical study on the residence error of training an ELM. After dividing the ELM training process into three steps and analyzing the change from input matrix to output matrix, we get the following conclusions:

(1) The random weight plays a role in increasing dimension of inputs but they do not increase the rank of input matrix;
(2) Sigmoid transformation transfers the middle matrix to an output matrix of full rank with probability 1;
(3) regarding solutions of linear systems resulting from output layer and the stability of the solutions, there is an essential difference between full-rank and waning-rank matrices; and
(4) an application to the genetic algorithm for solving absolute ELM problems is shown.

## Acknowledgment

The authors thank the editors and anonymous reviewers. Their valuable and constructive comments and suggestions helped us in significantly improving this paper. This research is supported by the National Natural Science Foundation of China (Grant nos. 11271367, 71371063, and 61170040).

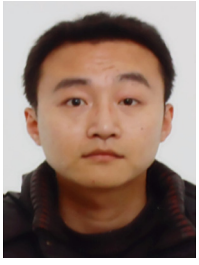
## References

[1] P.L. Bartlett, The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network, IEEE Trans. Inf. Theory 44 (2) (1998) 525–536.
[2] B.P. Chacko, V.R.V. Krishnan, G. Raju, P.B. Anto, Handwritten character recognition using wavelet energy and extreme learning machine, Int. J. Mach. Learn. Cybern. 3 (2) (2012) 149–161.
[3] T. Falas, A.G. Stafylopatis, The impact of the error function selection in neural network-based classifiers, in: Proceedings of the 1999 IEEE International Joint Conference on Neural Networks (IJCNN'99), 1999, pp. 1799–1804.
[4] G.R. Feng, G.B. Huang, Q.P. Lin, R. Gay, Error minimized extreme learning machine with growth of hidden nodes and incremental learning, IEEE Trans. Neural Netw. 20 (8) (2009) 1352–1357.
[5] M. Gen, R. Cheng, Genetic Algorithms and Engineering Optimization, John Wiley & Sons, New Jersey, 2000.
[6] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Netw. 4 (1991) 251–257.

[7] G.B. Huang, H.A. Babri, Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions, IEEE Trans. Neural Netw. 9 (1) (1998) 224–229.
[8] G.B. Huang, L. Chen, Convex incremental extreme learning machine, Neurocomputing 70 (16–18) (2007) 3056–3062.
[9] G.B. Huang, L. Chen, Enhanced random search based incremental extreme learning machine, Neurocomputing 71 (16–18) (2008) 3460–3468.
[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, Y.Q. Chen, H.A. Babri, Classification ability of single hidden layer feedforward neural networks, IEEE Trans. Neural Netw. 11 (3) (2000) 799–801.
[12] G.B. Huang, D.H. Wang, Y. Lan, Extreme learning machines: a survey, Int. J. Mach. Learn. Cybern. 2 (2) (2011) 107–122.
[13] 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.
[14] G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: a new learning scheme of feed-forward neural networks, in: Proceedings of the 2004 IEEE International Joint Conference on Neural Networks (IJCNN'04), 2004, pp. 985–990.
[15] G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1) (2006) 489–501.
[16] M. Leshno, V.Y. Lin, A. Pinkus, S. Schocken, Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, Neural Netw. 6 (1993) 861–867.
[17] J.P. Li, M.E. Balazs, G.T. Parks, P.J. Clarkson, A species conserving genetic algorithm for multimodal function optimization, Evol. Comput. 10 (3) (2002) 207–234.
[18] J. Luo, C.M. Vong, P.K. Wong, Sparse Bayesian extreme learning machine for multi-classification, in: Proceedings of the IEEE Transactions on Neural Networks and Learning Systems, 2014, http://dx.doi.org/10.1109/TNNLS.2013.2281839 (in press).
[19] J.C. Lv, Z. Yi, An improved backpropagation algorithm using absolute error function, Lect. Notes Comput. Sci. 3496 (2005) 585–590.
[20] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[21] L.C. Shi, B.L. Lu, Eeg-based vigilance estimation using extreme learning machines, Neurocomputing 102 (2013) 135–143.
[22] M. Spivak, A Comprehensive Introduction to Differential Geometry, Publish or Perish, Inc., Boston, 2005.
[23] K. Taji, T. Miyake, H. Tamura, On error backpropagation algorithm using absolute error function, in: Proceedings of the 1999 IEEE International Conference on Systems, Man, and Cybernetics (SMC'99), 1999, pp. 401–406.
[24] R. Wang, S. Kwong, X.Z. Wang, A study on random weights between input and hidden layers in extreme learning machine, Soft Comput. 16 (9) (2012) 1465–1475.
[25] X.Z. Wang, A.X. Chen, H.M. Feng, Upper integral network with extreme learning mechanism, Neurocomputing 74 (16) (2011) 2520–2525.
[26] X.Z. Wang, Q.Y. Shao, Q. Miao, J.H. Zhai, Architecture selection for networks trained with extreme learning machine using localized generalization error model, Neurocomputing 102 (2013) 3–9.
[27] Y.G. Wang, F.L. Cao, Y.B. Yuan, A study on effectiveness of extreme learning machine, Neurocomputing 74 (16) (2011) 2483–2490.
[28] J. Wu, S.T. Wang, F.L. Chung, Positive and negative fuzzy rule system, extreme learning machine and image classification, Int. J. Mach. Learn. Cybern. 2 (4) (2011) 261–271.
[29] R. Zhang, Y. Lan, G.B. Huang, Z.B. Xu, Universal approximation of extreme learning machine with adaptive growth of hidden nodes, IEEE Trans. Neural Netw. Learn. Syst. 23 (2) (2012) 365–371.

**Ai-Min Fu** received the Master's degree in College of Mathematics and Computer Science from Hebei University, China, in 2000. Now, he is currently working toward the Ph.D. degree in China Agricultural University. His research interests include neural networks, extreme learning machine, and the recent machine learning with big data.

**Xi-Zhao Wang** received the Ph.D. degree in computer science from Harbin Institute of Technology in 1998. He is presently the dean and professor at the School of Mathematics and Computer Science in Hebei University. His main research interests include learning from examples with neural network and extreme learning machine, multi-classifier fusion, and the recent machine learning with big data. He is an IEEE Fellow.



**Yu-Lin He** received the Master's degree in computer science from Hebei University, China, in 2009, where he is currently working toward the Ph.D. degree in the College of Mathematics and Computer Science, Hebei University. His research interests include neural networks, extreme learning machine, decision trees and approximate reasoning.

**Lai-Sheng Wang** received the B. S. degree from Jilin University, China in 1982, and M. S. degree from Jilin University, China, in 1989. He received the Ph.D. degree from China Agricultural University, Beijing, China, in 2001. Currently, he is a professor at the School of Science, China Agricultural University, Beijing, China. His research interests include data mining, image processing and pattern recognition.