Int. J. Mach. Learn. & Cyber. (2016) 7:765–772
DOI 10.1007/s13042-014-0292-7

ORIGINAL ARTICLE

Two-stage extreme learning machine for high-dimensional data

Peng Liu · Yihua Huang · Lei Meng · Siyuan Gong · Guopeng Zhang

Received: 6 February 2014 / Accepted: 6 August 2014 / Published online: 14 August 2014

Abstract Extreme learning machine (ELM) has been proposed for solving fast supervised learning problems by applying random computational nodes in the hidden layer. Similar to support vector machine, ELM cannot handle high-dimensional data effectively. Its generalization performance tends to become bad when it deals with high-dimensional data. In order to exploit high-dimensional data effectively, a two-stage extreme learning machine model is established. In the first stage, we incorporate ELM into the spectral regression algorithm to implement dimensionality reduction of high-dimensional data and compute the output weights. In the second stage, the decision function of standard ELM model is computed based on the low-dimensional data and the obtained output weights. This is due to the fact that two stages are all based on ELM. Thus, output weights in the second stage can be approximately replaced by those in the first stage. Consequently, the proposed method can be applicable to high-dimensional data at a fast learning speed. Experimental results show that the proposed two-stage ELM scheme tends to have better scalability and achieves outstanding generalization performance at a faster learning speed than ELM.

Keywords Spectral regression (SR) · Extreme learning machine (ELM) · High-dimensional data · Dimensionality reduction (DR)

1 Introduction

Extreme learning machine (ELM) has been attracting many researchers in recent years for its outstanding learning performance [1, 2]. It was originally developed for the single-hidden-layer feed-forward neural networks (SLFNs) [3–10], which was extended to the "generalized", i.e., may not be neuron alike [11, 12]. The hidden layer of ELM does not need be tuned. Thus, it has less computational complexity. On the other hand, ELM aims to reach not only the smallest training error but also the smallest norm of output weights. Consequently, it tends to achieve good generalization performance at a faster learning speed than traditional support vector machine (SVM). In addition, ELM provides a unified solution to regression and classification problems.

Supervised learning is a class of learning tasks and techniques that only make use of labeled data for training. ELM was originally proposed for solving fast supervised learning problems, while practical algorithms in supervised machine learning degrade in performance (prediction accuracy) when faced with many features that are not necessary for predicting the desired output. An important question in the fields of machine learning, knowledge discovery, computer vision and pattern recognition is how

P. Liu (✉) · L. Meng · G. Zhang
Internet of Things Perception Mine Research Center, China University of Mining and Technology, Xuzhou 221008, Jiangsu, China
e-mail: liupeng@cumt.edu.cn

P. Liu · L. Meng · G. Zhang
National and Local Joint Engineering Laboratory of Internet Application Technology on Mine, Xuzhou 221008, Jiangsu, China

Y. Huang
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, Jiangsu, China

S. Gong
National Key Laboratory for Coal Resources and Safe Mining, China University of Mining and Technology, Xuzhou 221116, Jiangsu, China

to extract a small number of good features. Similar to SVM, ELM is also sensitive to high-dimensional data, which not only has a bad influence on the performance of ELM, but dramatically impact the learning speed of ELM [13]. One way to address the problem of high-dimensional data is to utilize dimensionality reduction (DR) techniques. Spectral regression (SR), which is fundamentally based on regression and spectral graph analysis [13–15], can avoid Eigen-decomposition of dense matrices in traditional DR methods. Consequently, it can be carried out faster than many classical DR methods, such as principal component analysis (PCA), linear discriminant analysis (LDA) and locally linear embedding (LLE), etc. Moreover, it can be performed either in supervised, unsupervised or semi-supervised situation. Since SR and ELM are all essentially based on the regression model and can be performed in supervised scenarios, motivated by the similarity of their models, we intend to incorporate ELM into SR to handle high-dimensional data classification tasks. An improved SR is proposed in the first stage, in which an embedding function in ELM feature space can be solved effectively by the ELM algorithm. In the second stage, the decision function of the standard ELM model is computed based on low-dimensional data. It is worth noting that the output weights of ELM obtained in the first stage can also be applied to compute the final decision function of the second stage. Consequently, the proposed method not only overcomes the influence of high-dimensional data, but maintains the advantage of fast learning speed of ELM. Experimental results on classification demonstrate the effectiveness and efficiency of the proposed method.

In particular, the following contributions have been made in this paper.

(1) We incorporate ELM into the spectral regression algorithm to implement dimensionality reduction of high-dimensional data, which can further speed up the kernel spectral regression algorithm.
(2) A two-stage extreme learning machine model is presented based on the modified spectral regression algorithm and the proposed method can be applicable to high-dimensional data at a fast learning speed. Experimental results on real-world data sets verify the effectiveness and efficiency of this method.

The paper is structured as follows. In Sect. 2 and Sect. 3, we briefly introduce the extreme learning machine model and SR, respectively. The proposed two-stage ELM is introduced in Sect. 4. The experimental results are presented in Sect. 5. Finally, we give the related conclusions in Sect. 6. In order to avoid confusion, we give a list of the main notations used in this paper in Table 1.

Table 1 Notations of the paper

Notations	Descriptions
$\mathbb{R}^{\mathrm{d}}$	The input d-dimensional Euclidean space
n	The number of total training data points
m	The number of classes that the samples belong to
$\boldsymbol{X}$	$\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$ is the training data matrix
$\boldsymbol{y}$	$\boldsymbol{y} = (y_1, \ldots, y_n) \in \mathbb{R}^n$ is the 0-1 label vector. $\boldsymbol{y}_i \in \mathbb{R}$ is the lable of $\boldsymbol{x}_i$
$G(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{x})$	The hidden node function of ELM
$\boldsymbol{H}$	The hidden layer matrix $\boldsymbol{H} \in \mathbb{R}^{L \times n}$
$\|\quad\|_{\mathbb{H}}$	The norm in the random mapping space $\mathbb{H}$

2 Extreme learning machine

The output function of ELM for generalized SLFNs in the case of one output node is [16]

$$f_L(\boldsymbol{x}) = \sum_{i=1}^{L} \beta_i h_i(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta} \tag{1}$$

where $\boldsymbol{h}(\boldsymbol{x}) = [h_1(\boldsymbol{x}), \ldots, h_L(\boldsymbol{x})]$ is the output (row) vector of the hidden layer with respect to the input $\boldsymbol{x}$ and $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_L]^T$is the vector of the output weights between the hidden layer of L nodes and the output node. ELM is to minimize the training error as well as the norm of the output weights. The optimal model is as follows [16]:

$$\begin{aligned} &\text{Minimize} : L_{\mathrm{ELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{\mathrm{C}}{2}\sum\nolimits_{i=1}^{N} \varepsilon_i^2 \\ &\text{Subject to} : \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta} = t_i - \varepsilon_i, i = 1, \ldots, n. \end{aligned} \tag{2}$$

where t_i is the expected output value of the single output node and $_i$ is the training error value of the single output node with respect to the training sample $\boldsymbol{x_i}$. If ELM has multi-output nodes, an m-class classifier is corresponding to m output nodes. The classification problem for ELM with multi-output nodes is

$$\begin{aligned} &\text{Minimize} : L_{\mathrm{ELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{\mathrm{C}}{2}\sum\nolimits_{i=1}^{N} \|\varepsilon_i\|^2 \\ &\text{Subject to} : \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta} = \boldsymbol{t}_i^{\boldsymbol{T}} - \varepsilon_i^{\boldsymbol{T}}, i = 1, \ldots, n. \end{aligned} \tag{3}$$

where $\boldsymbol{t_i} = [t_{i,1}, \ldots, t_{i,m}]^{\boldsymbol{T}}$is the expected output vector of the m output nodes and $\varepsilon_{\boldsymbol{i}} = [\varepsilon_{i1}, \ldots, \varepsilon_{im}]^{\boldsymbol{T}}$ is the training error vector of the m output nodes with respect to the training sample $\boldsymbol{x_i}$. In this case, the output function of ELM is usually represented by the form of matrices as follows:

$$f_L(\boldsymbol{x}) = \boldsymbol{H}\boldsymbol{\beta}$$

where $\boldsymbol{H}$ is the hidden-layer output matrix denoted by

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}_1) \\ \boldsymbol{h}(\boldsymbol{x}_2) \\ \vdots \\ \boldsymbol{h}(\boldsymbol{x}_n) \end{bmatrix} = \begin{bmatrix} h_1(\boldsymbol{x}_1)\ldots h_L(\boldsymbol{x}_1) \\ h_1(\boldsymbol{x}_2)\ldots h_L(\boldsymbol{x}_2) \\ \vdots\vdots\vdots \\ h_1(\boldsymbol{x}_n)\ldots h_L(\boldsymbol{x}_n) \end{bmatrix}. \quad (4)$$

3 Spectral regression algorithm

Given a training set with l labeled samples $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_l$ and $(n - l)$ unlabeled samples $\boldsymbol{x}_{l+1}, \boldsymbol{x}_{l+2}, \ldots, \boldsymbol{x}_n$, where the sample $\boldsymbol{x}_i \in \boldsymbol{R}^d$ belongs to one of m classes, and let l_k be the number of labeled samples in the k-th class (the sum of l_k is equal to l). The SR is summarized as follows [17]:

Step 1 Constructing the adjacency graph G: Let $\boldsymbol{X}$ be the training set and G denote a graph with n nodes, where the i-th node corresponds to the sample $\boldsymbol{x}_i$. In order to model the local structure as well as the label information, the graph G will be constructed through p-nearest neighbor method.

Step 2 Constructing the weight matrix W: Let W be the sparse symmetric $n \times n$ matrix, where W_{ij} represents the weight of the edge joining vertices i and j.

(1) If there is no any edge between nodes i and j, then $W_{ij} = 0$;

(2) Otherwise, if both $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the k-th class, then $W_{ij} = 1/l_k$, *else* $W_{ij} = \delta \cdot s(i,j)$, where $\delta(0 < \delta \leq 1)$ is a given parameter to adjust the weight between supervised and unsupervised neighbor information. Therein, $s(i, j)$ is a similarity evaluation function between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. We have two variations that the first one is Simple-minded function $s(i, j) = 1$ and the second one is Heat kernel function:

$$s(i,j) = exp\left(-\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 / 2\sigma^2\right)$$

where $\sigma \in \mathbb{R}$.

Step 3 Eigen-decomposing: Let D be the $n \times n$ diagonal matrix, whose (i, i)-th element is the sum of the i-th column (or row) of W. Find $\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{m-1}$, which are the generalized eigenvectors corresponding to the largest m of the eigen-problem:

$$W\boldsymbol{y} = \lambda D\boldsymbol{y}, \quad (5)$$

where the first eigenvector y_0 is a vector of all ones with eigenvalue 1.

Step 4 Regularized least squares: Calculate m-1 vectors $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{m-1} \in \mathbb{R}^L$. $\boldsymbol{\alpha}_k (k = 1, \ldots, m-1)$ is the solution of regularized least square problem:

$$\boldsymbol{\alpha}_k = \arg\min_{\boldsymbol{\alpha}} \left(\sum_{i=1}^{n} \left(\boldsymbol{\alpha}^T \boldsymbol{x}_i - y_i^k\right)^2 + \gamma \|\boldsymbol{\alpha}\|^2 \right), \quad (6)$$

where y_i^k is the i-th element of y^k. In order to obtain $\boldsymbol{\alpha}_k$, the following linear equations system can be used to solve through the classic Gaussian elimination method.

$$\left(\boldsymbol{X}\boldsymbol{X}^T + \gamma \boldsymbol{I}\right)\boldsymbol{\alpha}_k = \boldsymbol{X}\boldsymbol{y}_k$$

where $\boldsymbol{I}$ is a $n \times n$ identity matrix.

Step5: SR Embedding: Let $\boldsymbol{A}$ be an $n \times (m-1)$ obtained transformation matrix through the previously mentioned processes, where $\boldsymbol{A} = [\alpha_1, \ldots, \alpha_{m-1}]$. The testing samples can be embedded into $m - 1$ dimensional subspace by

$$\boldsymbol{x} \to \boldsymbol{z} = \boldsymbol{A}^T \boldsymbol{x}. \quad (7)$$

If we choose a nonlinear function in Reproducing Kernel Hilbert Spaces (RKHS), i.e., $y_i = f(\boldsymbol{x}_i) = \sum_{j=1}^{n} \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Find m-1 vectors $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{m-1} \in \mathbb{R}^n$. $\boldsymbol{\alpha}_k (k = 1, \ldots, m-1)$ is the solution the linear equations system:

$$(\boldsymbol{K} + \alpha \boldsymbol{I})\alpha_k = \boldsymbol{y}_k \quad (8)$$

where $\boldsymbol{K}$ is $n \times n$ gram matrix $\boldsymbol{K}_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the Mercer kernel of RKHS $\mathcal{H}_{\mathrm{K}}$. It can be easily verified that function $f(\mathbf{x}) = \sum_{\mathrm{i}=1}^{\mathrm{m}} \alpha_i^k \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i)$ is the solution of the following regularized kernel least square problem [17]:

$$\min_{f \in \mathcal{H}_{\mathrm{K}}} \sum_{i=1}^{m} \left(f(\boldsymbol{x}_i) - y_i^k\right)^2 + \alpha \|f\|_K^2 \quad (9)$$

where α_i^k is the i-th element of vector $\boldsymbol{\alpha}_k$.

Let $\boldsymbol{\Theta} = [\alpha_1, \ldots, \alpha_{c-1}]$, $\boldsymbol{\Theta}$ is a $n \times (m-1)$ transformation matrix. The samples can be embedded into m − 1 dimensional subspace by

$$\boldsymbol{x} \to \boldsymbol{z} = \boldsymbol{\Theta}^T K(:, \boldsymbol{x}) \quad (10)$$

where $K(:, \boldsymbol{x}) = [K(\boldsymbol{x}_1, \boldsymbol{x}), \ldots, K(\boldsymbol{x}_n, \boldsymbol{x})]^T$.

4 Two-Stage Extreme Learning Machine

4.1 Dimensionality reduction using ELM

In SR, a linear function is used to represent the embedding function. Alternatively, if we consider an output function in ELM feature space, i.e., $f_L(x_i) = \sum_{j=1}^{L} \beta_j h_j(\boldsymbol{x}_i) = \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta}$, the embedding function can be acquired by solving the following least square problem:

$$\boldsymbol{\beta} = arg\min_{\boldsymbol{\beta}} \left(\sum_{i=1}^{n} (\boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta} - y_i)^2 + \gamma \|\boldsymbol{\beta}\|^2 \right), \quad (11)$$

where y_i is the i-th element of the label vector $\boldsymbol{y}$.

It can be verified that $\boldsymbol{\beta}$ equals to

$$\boldsymbol{\beta} = \boldsymbol{H}^T\left(\boldsymbol{H}\boldsymbol{H}^T + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{y}, \tag{12}$$

where $\boldsymbol{H}$ is an $n \times L$ output matrix of ELM hidden layer and $\boldsymbol{I}$ is an $n \times n$ identity matrix. If the number of training samples is huge, we have

$$\begin{aligned}\boldsymbol{H}^T\left(\boldsymbol{H}\boldsymbol{H}^T + \gamma\mathrm{I}\right)^{-1} &= \left(\boldsymbol{H}^T\boldsymbol{H} + \gamma\mathrm{I}\right)^{-1}\left(\boldsymbol{H}^T\boldsymbol{H} + \gamma\mathrm{I}\right)\boldsymbol{H}^T\left(\boldsymbol{H}\boldsymbol{H}^T + \gamma\mathrm{I}\right)^{-1} \\ &= \left(\boldsymbol{H}^T\boldsymbol{H} + \gamma\mathrm{I}\right)^{-1}\boldsymbol{H}^T\left(\boldsymbol{H}\boldsymbol{H}^T + \gamma\mathrm{I}\right)\left(\boldsymbol{H}\boldsymbol{H}^T + \gamma\mathrm{I}\right)^{-1} \\ &= \left(\boldsymbol{H}^T\boldsymbol{H} + \gamma\mathrm{I}\right)^{-1}\boldsymbol{H}^T\end{aligned} \tag{13}$$

Substituting (13) into (12), we have

$$\boldsymbol{\beta} = \left(\boldsymbol{H}^T\boldsymbol{H} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{H}^T\boldsymbol{y}, \tag{14}$$

where $\boldsymbol{\beta}$ is an $L \times 1$ transformation vector. The samples can be embedded into 1-dimensional subspace by

$$\mathbf{x} \rightarrow \mathbf{z} = \boldsymbol{\beta}^T\boldsymbol{h}(\boldsymbol{x}), \tag{15}$$

where $\boldsymbol{h}(\boldsymbol{x}) = [h_1(\boldsymbol{x}), \ldots, h_L(\boldsymbol{x})]^T$.

The reason that the samples are embedded into 1-dimensional subspace is that the output weights vector $\boldsymbol{\beta}$ can be repeatedly used in the computation of the final decision function of ELM in the second stage. Thus, the proposed method does not need the training process of ELM, which can speed up training high-dimensional data.

4.2 ELM based on dimensionality reduction

Based on low-dimensional data and the obtained output weights in the first stage, the decision function of the standard ELM model in the second stage can be computed efficiently. Consequently, the proposed method not only handles high-dimensional data effectively, but also has the advantage of fast learning speed. The two-stage ELM framework estimates an unknown function by minimizing [12]

$$f^* = \arg\min_{f\in\mathbb{H}}\left[\frac{C}{2}\sum_{i=1}^{n}(\boldsymbol{f}(x_i) - y_i)^2 + \frac{1}{2}\|f\|_{\mathbb{H}}^2\right] \tag{16}$$

where $\mathbb{H}$ is the random mapping space of ELM, $\|\mathrm{f}\|_{\mathbb{H}}^2$ is the norm penalty in the random mapping space $\mathbb{H}$ and represents the complexity of the function in $\mathbb{H}$, x_i represents a variable of the 1-dimensional subspace generated by the first stage. Eq. (16) can be transformed into the following optimal problem:

$$\boldsymbol{\beta} = arg\min_{\boldsymbol{\beta}}\left(\frac{\mathrm{C}}{2}\sum_{i=1}^{n}(\boldsymbol{h}(x_i)\boldsymbol{\beta} - y_i)^2 + \frac{1}{2}\|\boldsymbol{\beta}\|^2\right) \tag{17}$$

It should be noted that the form of Eq. 17) is the same as that of Eq. (11). Since SR is a kind of Locality Preserving Projections (LPP) in essence, it can maintain the local structure of samples very well. Thus, the output weight vector $\boldsymbol{\beta}$ of Eq. (17) can be approximately acquired by solving Eq. (11) with the original data. Thus, the proposed method has fast learning speed by solving one least square problem. The flow chart of the two-stage ELM method is shown in Fig. 1.

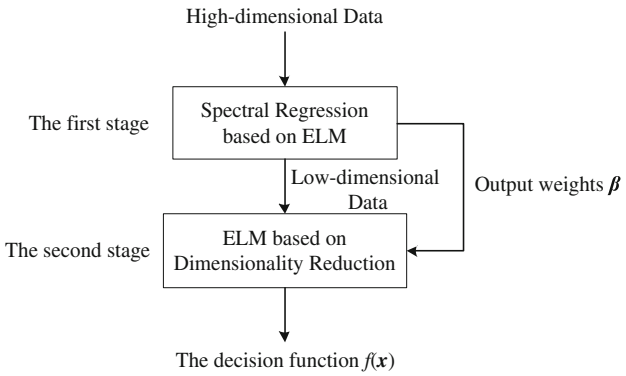

Fig. 1 The flow chart of two-stage ELM

Table 2 The Description of the two-stage ELM Algorithm

Two-stage ELM Algorithm
Input: n labeled high-dimensional data $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$
Output: Decision function $\boldsymbol{f}^*(\boldsymbol{x})$.
Step 1: Choosing a hidden-node function $G(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{x})$, γ and L, randomly generating $\{(a_i, b_i)\}_{i=1}^L$ and computing $\boldsymbol{H} = (G(\boldsymbol{x}_1), G(\boldsymbol{x}_2), \ldots G(\boldsymbol{x}_n))^{\mathrm{T}}$, where $G(\boldsymbol{x}) = [G(a_1, b_1, \boldsymbol{x}), G(a_2, b_2, \boldsymbol{x}), \ldots, G(a_L, b_L, \boldsymbol{x})]$.
Step 2: According to Eq. (11), computing the output weights $\boldsymbol{\beta}$.
Step 3: If the number of the training data sets is very large $n \gg L$, selecting (14) to compute the embedding function $\boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta}$ otherwise, selecting (12).
Step 4: Computing the low-dimensional data according to the embedding function.
Step 5: Based on the low-dimensional data, choosing γ and L, randomly generating$\{(a_i, b_i)\}_{i=1}^L$ and computing the hidden matrix $\boldsymbol{H}'$.
Step 6: Computing the final decision function $\boldsymbol{f}^*(\boldsymbol{x}) = \boldsymbol{H}'\boldsymbol{\beta}$.

Similar to ELM, the proposed model gives unified solutions for regression, binary and multiclass classification. But we mainly discuss the classification problem in this paper. The two-stage ELM algorithm is summarized in Table 2. In the supervised case, the Eigen-problem of the first stage is mainly the cost of Gram-Schmidt method, which requires $(nm^2 - \frac{1}{3}m^3)$ times addition and multiplication operations. If the number of training samples is very large, the computational complexity of the regularized least squares problem in Eq. (11) and (17) is $\mathrm{O}(L^3)$, where $L \ll n$. Thus, the total computational

Table 3 Description of the data sets for classification

Data	Size (n)	Feature (d)	Class
Iris	150	4	3
Ionosphere	351	34	2
SatelliteC1-C2	2,236	36	2
CLL-SUB-111	111	11,340	3
PCMAC	1,943	3,289	2
TOX-171	171	5,748	4
Extended Yale B	2,114	1,024	38
CMU PIE	11,560	1,024	68

complexity of the proposed two-stage ELM is $O(L^3 + nm^2 - \frac{1}{3}m^3)$. Correspondingly, traditional kernel classification methods based on regularized least squares generally need to compute the inverse of a kernel matrix, which has the computational complexity of $O(n^3)$. For the two-stage ELM algorithm, the decision function needs cL^2 times multiplication operation toward a new sample. For traditional kernel classification methods based on regularized least squares, the decision function needs cn^2 times multiplication operation given a new sample Consequently, the proposed method has lower computational complexity and better scalability than traditional kernel classification methods.

5 Experiments

5.1 Data set

To evaluate the accuracy and efficiency of the proposed algorithm, we mainly discuss classification problems and perform the experiments on the several real-world data sets from the UCI machine learning repository and another benchmark repository [18–22]. The basic information about real-world data sets is summarized in Table 3. These data sets include both high-dimensional and low-dimensional data sets, since the proposed method can also be applicable to low-dimensional data.

We obtained 170 images for each individual on the CMU PIE face data set and make each image be 32 × 32 pixels, with 256 gray levels per pixel. For each individual, l (=10, 20, 30, 40) images were randomly selected for training and the rest were used for testing. For the Extended Yale B data set, l (=5, 10, 15, 20, 25, 30) images per individual were taken to form the training set, and the rest were considered to be the testing set. For other data sets, we all used one half data to training the ELM model and the rest for testing.Finally, we averaged the results over 30 random splits.

Table 4 Recognition accuracy rates on low-dimensional data sets (mean ± SD %)

Data sets	SVM	ELM	SRELM	TSELM
Iris	85.3 ± 0.5	86.2 ± 0.6	86.3 ± 0.8	86.7 ± 0.3
Ionosphere	92.6 ± 0.5	94.1 ± 0.4	94.4 ± 0.5	94.7 ± 0.3
SatelliteC1-C2	95.2 ± 0.3	96.6 ± 0.4	96.8 ± 0.4	97.1 ± 0.2

5.2 Parameter settings

We compared the proposed two-stage ELM (TSELM) with standard ELM and ELM based on SR (SRELM). All the experiments were carried out in MATLAB 7.0.1 environment running in a 3.10GHZ Intel Core™ i5-2400 with 4-GB RAM. In our experiments, the Gaussian function $\exp(-b\|\boldsymbol{x} - a\|^2)$ was selected for each algorithm. A grid search of the trade-off constant γ on $\{2^{-18}, 2^{-16}, \ldots, 2^{48}, 2^{50}\}$ and the number of hidden nodes L on $\{2^1, 2^2, \ldots, 2^{14}, 2^{15}\}$ was conducted in seek of the optimal result for each algorithm using ELM. The trade-off constant C of SVM was searched in the range of $\{2^{-18}, 2^{-16}, \ldots, 2^{18}, 2^{50}\}$ by using five-fold cross-validation. Since the search time of each algorithm was too long, the search time of optimal parameters did not be added to the whole training time for each algorithm to facilitate comparison. Finally, we implemented ELM and SVM training and computed the training time of each algorithm with the final optimal parameters.

5.3 Performance comparison

The classification accuracy rates of these algorithms on low-dimensional and high-dimensional data sets are shown in Tables 4 and 5, respectively. We average the results over 30 random splits and report the mean as well as the standard deviation. As can be seen from Table 4, the results of all algorithms are close to each other. This is due to the fact that ELM is not sensitive to low-dimensional data sets. Thus, the performance of standard ELM is close to that of ELM based on dimensionality reduction. The proposed TSELM algorithm achieves the better recognition accuracy than ELM, which shows that the performance of ELM can be improved further by using dimensionality reduction techniques. Since TSELM replaces the linear embedding function of SR by the nonlinear output function of ELM, it can be performed better than SRELM. Recognition accuracy rates of all algorithms on high-dimensional data sets are shown in Table 5. Since SVM and ELM are sensitive to high-dimensional data sets, the performance of them is worse than that of SRELM and TSELM. TSELM still outperforms SRELM by using nonlinear output functions in SR. The running time (second) of different classification methods on high-dimensional data sets are shown in

Table 5 Recognition accuracy rates on high-dimensional data sets (mean ± SD %)

Data sets	SVM	ELM	SRELM	TSELM
TOX-171	79.3 ± 1.4	80.4 ± 1.2	83.7 ± 1.0	84.7 ± 0.8
PCMAC	81.5 ± 0.8	82.6.1 ± 0.8	84.4 ± 0.6	85.7 ± 0.5
CLL-SUB-111	75.6 ± 1.8	77.6 ± 1.4	81.8 ± 1.2	83.3 ± 1.2

Table 6 Running time of different classification methods on high-dimensional data sets (s)

Data sets	SVM	ELM	SRELM	TSELM
TOX-171	20.832	6.673	3.682	3.057
PCMAC	24.625	8.759	4.834	4.249
CLL-SUB-111	32.362	12.247	6.356	5.865

Table 7 Recognition accuracy rates on PIE (mean ± SD%)

Train Size	ELM	SRELM	TSELM
10 × 68	83.2 ± 0.7	85.0 ± 1.3	87.6 ± 0.4
20 × 68	91.1 ± 0.6	92.3 ± 0.7	93.5 ± 0.3
30 × 68	93.4 ± 0.6	93.4 ± 0.7	94.7 ± 0.3
40 × 68	94.6 ± 0.6	94.8 ± 0.4	96.0 ± 0.3

Table 6, where SRELM and TSELM run much faster than SVM and ELM. TSELM has the fastest learning speed among them. Overall, TSELM can have the best performance at a faster learning speed.

We further tested the proposed TSELM algorithm on the high-dimensional CMU PIE face data set. For each given l (the number of training samples per class), we also averaged the results over 30 random splits and reported the mean as well as the standard deviation, which are shown in Table 7.

From Table 7, we can observe that SRELM and TSELM all outperform standard ELM, which shows that standard ELM is sensitive to high-dimensional data and its performance can be improved by utilizing effective DR methods. Since SRELM and TSELM all use the Tikhonov regularizer to improve the smoothness of the projection functions, they are able to achieve better performance than ELM. The performance of SRELM is a little better than that of ELM, this is due to the fact that SRELM obtains the projection function based on linear embedding mapping, which is not applicable to face image data sets containing manifold structure data. Correspondingly, TSELM substitutes the nonlinear embedding functions with the linear ones. Thus, it can outperform SRELM and the result of the first stage of TSELM is better than that of SR, which contributes to improve the whole recognition accuracy.

We further obtained the training and testing time of ELM based on different DR methods, which are listed in Table 8, where the training time of ELM refers to the cost time of computing the final decision function, the training time of SRELM and TSELM includes the computational time of the dimension reduction for the training data and the training time of standard ELM. As can be seen from Table 8, TSELM and SRELM perform faster than ELM, which shows that both the classification accuracy and learning speed of ELM can be enhanced by using the SR method for high-dimensional data. TSELM performs faster than SRELM, this is due to the fact that SRELM solves two least square problems by training the ELM model based on the SR algorithm, while TSELM only solves on least square problem by using the obtained output weights in the first stage directly. The final results all indicate that the decision function of standard ELM model can be computed efficiently based on the low-dimensional data and the obtained output weights in the first stage, which validates the effectiveness of the proposed method. Figure 2 shows the classification accuracy rates as well as the running time (second) for each method on the Extended Yale B data set. We averaged the results over 30 random splits for each given l.

As can be seen from Fig. 2, TSELM and SRELM can efficiently exploit high-dimensional data to discover the intrinsic geometry structure in the data. They perform significantly better than ELM. The speed of TSELM is faster than that of ELM and SRELM as l increases, which validates the effectiveness and efficiency of TSELM further.

Overall, compared with ELM and SRELM, the proposed method can achieve better performance at much faster learning speed. It not only handles high-dimensional data effectively, but also inherits the advantage of fast learning speed of ELM.

6 Conclusion

In this paper, we construct a two-stage ELM model in terms of the SR algorithm for high-dimensional data classification tasks. We incorporate ELM into SR and derive the form of the embedding function applicable to high-dimensional data. A fast ELM learning algorithm by SR based on ELM is proposed to improve the effectiveness and efficiency of the standard ELM method. The proposed method not only overcomes the influence of high-dimensional data, but also maintains the advantage of fast learning speed of ELM. Experimental results show that TSELM has the best performance than ELM and ELM based on SR for both high-dimensional and low-dimensional data sets. But, the choice of trade-off constant γ is

Table 8 Running time of different classification methods on PIE (s)

Train Size	ELM			SRELM			TSELM		
	Training Time	Testing Time	Total	Training Time	Testing Time	Total	Training Time	Testing Time	Total
10 × 68	0.175	0.814	0.989	0.203	0.259	0.462	0.106	0.258	0.364
20 × 68	0.283	0.772	1.055	0.275	0.245	0.520	0.124	0.245	0.369
30 × 68	0.399	0.709	1.108	0.336	0.228	0.564	0.212	0.229	0.441
40 × 68	0.574	0.664	1.238	0.423	0.213	0.636	0.325	0.213	0.538

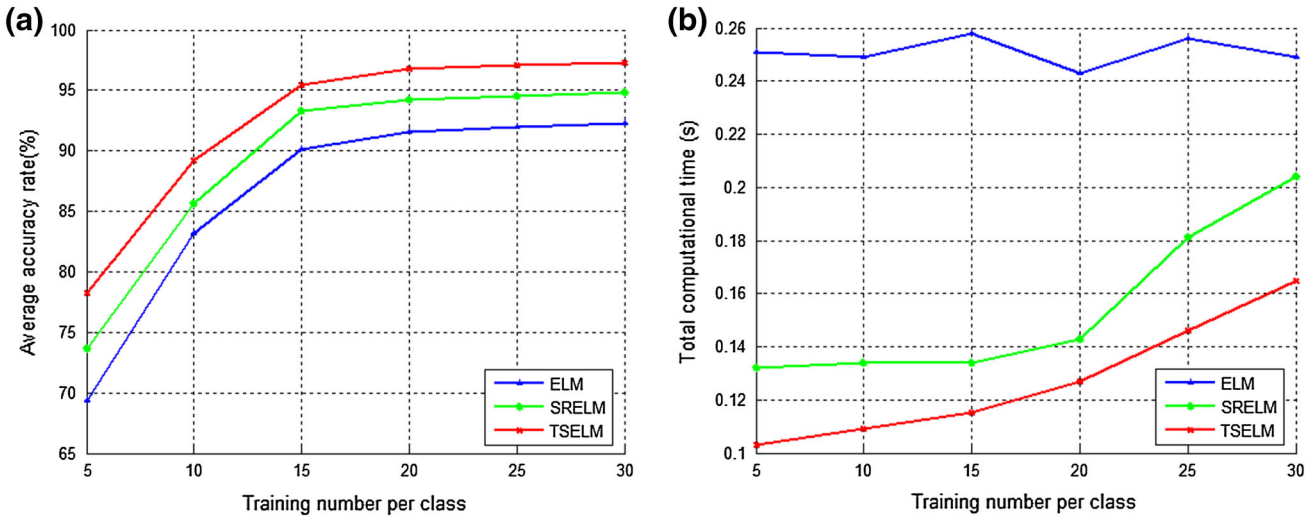

Fig. 2 Results of different methods under the different number of the labeled data on Extended Yale B. **a** Average accuracy rate (%). **b** Total computational time (s)

not discussed further and how to select the optimal value of the number of hidden nodes is another primary challenge of our method. In the near future, we will study how to select the optimal parameters of the proposed model and study the sparse regularization problem for this method.

Acknowledgments The authors would like to thank the editor and the anonymous reviewers for their valuable comments and constructive suggestions. This work is jointly supported by the National High Technology Research and Development Program of China (863 Program) (No. 2013AA06A411), the National Natural Science Foundation of China (No. 41302203), the Fundamental Research Funds for the Central Universities (No. 2011QNB26) and A Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

1. Wang Xizhao, Chen Aixia, Feng Huimin (2011) Upper integral network with extreme learning mechanism. Neurocomputing 74(16):2520–2525
2. Wang Xizhao, Shao Qingyan, Qing Miao, Zhai Junhai (2013) Architecture selection for networks trained with extreme learning machine using localized generalization error model. Neurocomputing 102:3–9
3. Frénay B, Verleysen M (2010). Using SVMs with randomised feature spaces: an extreme learning approach. In: European Symposium on Artificial Neural Networks (ESANN): 315–320
4. Miche Y, Sorjamaa A, Bas P, Simula O, Jutten C, Lendasse A (2010) OP-ELM: optimally pruned extreme learning machine. IEEE Trans Neural Netw 21(1):158–162
5. Deng W, Zheng Q, Chen L (2009). Regularized extreme learning machine. In: IEEE Symposium on Computational Intelligence and Data Mining (CIDM):389–395
6. Huang GB, Wang DH, Lan Y (2011) Extreme learning machines: a survey. Int J Mach Learn Cyber 2(2):107–122
7. Wang G, Zhao Y, Wang D (2008) A protein secondary structure prediction framework based on the Extreme Learning Machine. Neurocomputing 72(1):262–268
8. Lan Y, Soh YC, Huang GB (2008). Extreme Learning Machine based bacterial protein subcellular localization prediction. In: IEEE International Joint Conference on Neural Networks (IJCNN):1859–1863
9. Mohammed AA, Minhas R, Wu Jonathan QM, Sid-Ahmed MA (2011) Human face recognition based on multidimensional PCA and extreme learning machine. Pattern Recogn 44(10):2588–2597
10. Wang Y, Cao F, Yuan Y (2011) A study on effectiveness of extreme learning machine. Neurocomputing 74(16):2483–2490
11. Cao F, Liu B, Sun Park D (2013) Image classification based on effective extreme learning machine. Neurocomputing 102:90–97
12. Belkin M, Niyogi P, Sindhwani V (2006) Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J Mach Learn Res 7:2399–2434

13. Xue H, Chen S, Yang Q (2009) Discriminatively regularized least-squares classification. Pattern Recogn 42(1):93–104
14. Liu J, Chen Y, Liu M, Zhao Z (2011) SELM: semi-supervised ELM with application in sparse calibrated location estimation. Neurocomputing 74(16):2566–2572
15. Li L, Liu D, Ouyang J (2012) A new regularization classification method based on extreme learning machine in network data. J Inf Comput Sci 9(12):3351–3363
16. Huang GB, Zhou H, Ding X, Zhang R (2012) Extreme learning machine for regression and multiclass classification. IEEE T SYST MAN CY B 42(2):513–529
17. Cai D, He X, Han J (2007). Spectral regression for efficient regularized subspace learning. In: IEEE 11th International Conference on Computer Vision (ICCV):1–8
18. He X, Cai D, Shao Y, Bao H, Han J (2011) Laplacian regularized gaussian mixture model for data clustering. IEEE T KNOWL DATA EN 23(9):1406–1418
19. Lee KC, Ho J, Kriegman DJ (2005) Acquiring linear subspaces for face recognition under variable lighting. IEEE T PATTERN ANAL 27(5):684–698
20. Wu J, Wang ST, Chung FL (2011) Positive and negative fuzzy rule system, extreme learning machine and image classification. Int J Mach Learn Cyber 2(4):261–271
21. Belhumeur PN, Hespanha JP, Kriegman DJ (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE T PATTERN ANAL 19(7):711–720
22. Vink JP, de Haan G (2013). Comparison of machine learning techniques for target detection. ARTIF INTELL REV: 1–15