

Data-driven prediction model for adjusting burden distribution matrix of blast furnace based on improved multilayer extreme learning machine

Xiaoli Su^{1,2} · Sen Zhang^{1,2} · Yixin Yin^{1,2} · Yanan Liu^{1,2} · Wendong Xiao^{1,2}

Published online: 18 April 2018
© Springer-Verlag GmbH Germany, part of Springer Nature 2018

Abstract

Reasonable burden distribution matrix is one of important requirements that can realize low consumption, high efficiency, high quality and long campaign life of the blast furnace. This paper proposes a data-driven prediction model of adjusting the burden distribution matrix based on the improved multilayer extreme learning machine (ML-ELM) algorithm. The improved ML-ELM algorithm is based on our previously modified ML-ELM algorithm (named as PLS-ML-ELM) and the ensemble model. It is named as EPLS-ML-ELM. The PLS-ML-ELM algorithm uses the partial least square (PLS) method to improve the algebraic property of the last hidden layer output matrix for the ML-ELM algorithm. However, the PLS-ML-ELM algorithm may have different results in different trails of simulations. The ensemble model can overcome this problem. Moreover, it can improve the generalization performance. Hence, the EPLS-ML-ELM algorithm is consisted of several PLS-ML-ELMs. The real blast furnace data are used to testify the data-driven prediction model. Compared with other prediction models which are based on the SVM algorithm, the ELM algorithm, the ML-ELM algorithm and the PLS-ML-ELM algorithm, the simulation results demonstrate that the data-driven prediction model based on the EPLS-ML-ELM algorithm has better prediction accuracy and generalization performance.

Keywords Extreme learning machine · Multilayer extreme learning machine · Blast furnace · Burden distribution matrix · Ensemble model · Data-driven prediction model

1 Introduction

Ironmaking process is a multivariant and nonlinear industrial process, and it has numerous chemical reactions (Peacey and Davenport 2016). Blast furnace is an essential step for ironmaking process (Radhakrishnan and Ram 2001; Geerdes et al. 2009). The smooth operation of the blast furnace is very important for iron and steel companies. Burden distribution matrix is a significant operation system of the blast furnace (Liu 2012). It is consisted of the angels of rotating chute

and the rotational numbers corresponding to each angel of rotating chute. The burden distribution matrix is expressed as

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_i & \cdots & \alpha_n \\ N_1 & N_2 & \cdots & N_i & \cdots & N_n \end{bmatrix}, \quad (1)$$

where α_i represents the angel of rotating chute, N_i is the rotational numbers corresponding to each angel, $i = 1, 2, \dots, n$.

Reasonable burden distribution matrix can obtain reasonable gas flow distribution and make full use of energy of gas, and reasonable burden distribution matrix is one of important requirements that can realize low consumption, high efficiency, high quality and long campaign life of the blast furnace (Liu 2012; Shi et al. 2016). In practical production process, the burden distribution matrix is not fixed. The angles of rotating chute and the rotational numbers corresponding to each angle of rotating chute can be changed. When the blast furnace has the abnormal conditions or the blast furnace condition parameters are not good, the burden

Communicated by X. Wang, A.K. Sangaiah, M. Pelillo.

✉ Sen Zhang
zhangsen@ustb.edu.cn

¹ School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China

² Key Laboratory of Knowledge Automation for Industrial Processes of Ministry of Education, School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China

distribution matrix should be adjusted. Moreover, the burden distribution matrix can be measured by the blast furnace condition parameters. In real blast furnace operation, operators use blast furnace condition parameters to determine whether the burden distribution matrix needs to be adjusted. There are seven blast furnace condition parameters (the blast volume, the blast pressure, the blast velocity, the top pressure, the permeability index, the gas utilization rate and the utilization coefficient). According to these blast furnace condition parameters and operation experience, this paper will establish a data-driven prediction model based on machine learning algorithm to determine whether the burden distribution matrix needs to be adjusted.

Extreme learning machine (ELM) is a fast learning algorithm for single-hidden layer feedforward neural networks (SLFNs) (Huang et al. 2004; Li et al. 2005; Huang et al. 2006). The input weights and hidden biases of the ELM algorithm are randomly generated, respectively. They do not need to be fine-tuned. Moreover, the ELM algorithm has better generalization performance and faster learning rate than conventional gradient-based algorithms. Besides, it may get rid of sinking into the local minima. As a result, the ELM algorithm has been widely applied in many fields, such as image processing, face recognition, fault diagnose and human-computer interaction.

More and more researchers have pay attention to the ELM algorithm, and it has obtained huge development. Many variants of ELM have been improved the specific aspects of performance of the original algorithm. For example, the differential evolutionary (DE) (Storn and Price 1997) is used to optimize the input weights and hidden biases for ELM (Zhu et al. 2005); the online sequential extreme learning machine (OS-ELM) algorithm can learn data one-by-one or chunk-by-chunk (Liang et al. 2006); a dynamic ensemble ELM based on sample entropy is proposed to overcome the over-fitting problem (Zhai et al. 2012); voting method is introduced into ELM for classification applications (Cao et al. 2012); a weighted ELM (W-ELM) is proposed to deal with data with imbalanced class distribution (Zong et al. 2013); multi-ELM is proposed to approximate any target continuous function and classify disjointed regions (Yang et al. 2015). In addition, the ELM algorithm and improved ELM algorithm are widely used. For instance, the upper integral network with the ELM algorithm (Wang et al. 2011) has better performance than the single upper integral classifier; an algorithm for architecture selection of SLFNs trained by the ELM algorithm-based initial localized generalization error (LGEM) can automatically determine the number of hidden nodes (Wang et al. 2013); three ELM-based discriminative clustering methods are proposed by Huang et al. (Huang et al. 2015); a single classifier is trained by ELM to obtain optimal and generalized solution for multiclass traffic sign recognition (TSR) (Huang et al. 2017); based on MapReduce and ensemble of

ELM classifiers, a classification algorithm is proposed for imbalanced large data datasets (Zhai et al. 2017); OS-ELM with sparse weighting is proposed to increase the classification accuracy of minority class samples and reduce the accuracy loss of majority class samples (Mao et al. 2017). In addition, a semi-supervised low-rank kernel learning method based on ELM is proposed (Liu et al. 2017); unsupervised ELM based on embedded features extreme learning machine autoencoder (ELM-AE) (Kasun et al. 2013) is proposed to handle the multicluster clustering (Ding et al. 2017). It is worth noting that the multilayer extreme learning machine (ML-ELM) algorithm (Kasun et al. 2013) has been proposed. It is based on the ELM-AE algorithm. ELM-AE is used to initialize the whole hidden layer weights of ML-ELM. ML-ELM does not need to be fine-tuned. Hence, it costs less training time than deep learning (Bengio 2009). There are some improvements for ML-ELM. For example, the ML-ELM with subnetwork nodes is proposed for representation learning (Yang and Wu 2016); a new architecture based on multilayer network framework is proposed for dimension reduction and image reconstruction (Yang et al. 2016).

Like ELM, the output weights of the ML-ELM algorithm is obtained by using $\beta = (H^T H)^{-1} H^T T$, where H is the output matrix of the last hidden layer for ML-ELM or the output matrix of the hidden layer for ELM. For ELM, multicollinearity problem can deteriorate its generalization performance (Zhang et al. 2016). Due to multicollinearity problem, $H^T H$ may not always be nonsingular or may tend to be singular in some applications and $\beta = (H^T H)^{-1} H^T T$ may not perform well (Huang et al. 2006). In order to get more stable resultant solutions and better generalization performance, $\beta = (\frac{1}{\lambda} + H^T H)^{-1} H^T T$ is used to obtain the output weights of ELM (Huang et al. 2012), including ML-ELM. PLS-ML-ELM (Su et al. 2016) uses the partial least square (PLS) method to overcome the multicollinearity problem. It has better generalization performance than the ML-ELM algorithm. However, the PLS-ML-ELM algorithm may have different results in different trails of simulations. Hence, this paper further improves the PLS-ML-ELM algorithm. The ensemble model (Hansen and Salamon 1990) can overcome this problem. And it is used to improve the PLS-ML-ELM algorithm in this paper. This algorithm is named as EPLS-ML-ELM. The EPLS-ML-ELM algorithm is consisted of several PLS-ML-ELMs.

The blast furnace process belongs to the process industry, and it has a high demand of timeliness. Hence, in this paper, ML-ELM is selected as the classification algorithm of the data-driven prediction model of determining whether the burden distribution matrix needs to be adjusted. Moreover, the proposed EPLS-ML-ELM algorithm has better generalization performance and prediction accuracy than the ML-ELM algorithm and the PLS-ML-ELM algorithm. Hence, the proposed EPLS-ML-ELM algorithm is applied to establish the

data-driven prediction model of determining whether the burden distribution matrix needs to be adjusted. And the real industrial data are used to verify the data-driven prediction model. Compared with the SVM algorithm, the ELM algorithm, the ML-ELM algorithm and the PLS-ML-ELM algorithm, simulation results are shown that the data-driven prediction model based on the proposed EPLS-ML-ELM algorithm has better prediction accuracy and generalization performance.

The rest of this paper is as follows: Sect. 2 briefly introduces the ELM algorithm, the ELM-AE algorithm and the ML-ELM algorithm, and the PLS-ML-ELM algorithm and the EPLS-ML-ELM algorithm are introduced in detail. The data-driven prediction model of determining whether the burden distribution matrix needs to be adjusted is described in Sect. 3. The simulation results are shown in Sect. 4. Section 5 represents the summarization of this paper.

2 The improved multilayer extreme learning machine algorithm

Because the last hidden layer of ML-ELM has the multicollinearity problem which can affect the prediction accuracy, the PLS-ML-ELM algorithm uses the PLS method to overcome this problem. However, the PLS-ML-ELM algorithm may have different results in different trails of simulations. So this paper proposes the EPLS-ML-ELM algorithm to obtain better generalization performance than the PLS-ML-ELM algorithm. EPLS-ML-ELM is consisted of several PLS-ML-ELMs. This section briefly introduces the ELM algorithm, the ELM-AE algorithm and the ML-ELM algorithm. The PLS-ML-ELM algorithm and the proposed EPLS-ML-ELM algorithm are presented in detail, respectively.

2.1 Extreme learning machine (ELM)

ELM, proposed by Huang et al, is a fast machine learning algorithm of SLFNs (Huang et al. 2006). It has input layer, one single-hidden layer and output layer. Figure 1 gives the structure of the ELM algorithm. There are N arbitrarily distinct samples $(x_i, t_i) \in R^n \times R^m$, where $x_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T$ represents that the i th sample has n -dimensional features, and $t_i = [t_{i1}, t_{i2}, \dots, t_{im}]^T$ is the target vector. ELM with l hidden layer nodes and the activation function $g(x)$ can be mathematically expressed as

$$\sum_{i=1}^l \beta_i g_i(x_j) = \sum_{i=1}^l \beta_i g(w_i \cdot x_j + b_i) = o_j, \quad j = 1, 2, \dots, N, \tag{2}$$

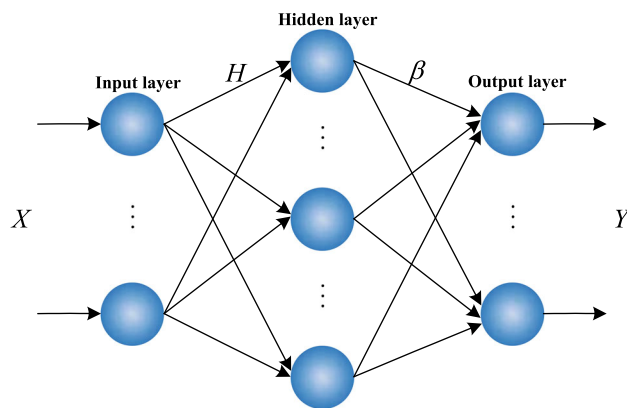


Fig. 1 Structure of the ELM algorithm

where $x_j = [x_{j1}, x_{j2}, \dots, x_{jn}]$ is the input vector, $w_i = [w_{i1}, w_{i2}, \dots, w_{in}]^T$ is the weight vector of connecting the i th hidden layer node and the whole input nodes, b_i is the bias of the i th hidden layer node, $\beta_i = [\beta_{i1}, \beta_{i2}, \dots, \beta_{im}]^T$ is the weight vector which connects the i th hidden node and the whole output nodes.

Equation (2) can be briefly expressed as

$$H\beta = T, \tag{3}$$

$$\text{where } H = \begin{bmatrix} g(w_1 \cdot x_1 + b_1) & \dots & g(w_l \cdot x_1 + b_l) \\ \vdots & \ddots & \vdots \\ g(w_1 \cdot x_N + b_1) & \dots & g(w_l \cdot x_N + b_l) \end{bmatrix}_{N \times l},$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_l^T \end{bmatrix}_{l \times m}, \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m}.$$

H is the output matrix of the hidden layer for ELM. The i th column of H is the i th hidden node output vector with respect to the inputs x_1, x_2, \dots, x_N .

For ELM, the least square method is used to calculate the output weight β , and the representation is shown as

$$\beta = H^\dagger T = (H^T H)^{-1} H^T T, \tag{4}$$

where the H^\dagger is the Moore–Penrose generalized inverse of H .

2.2 Extreme learning machine autoencoder (ELM-AE)

ELM-AE is an unsupervised learning algorithm (Kasun et al. 2013). The inputs of ELM-AE are also used as the outputs. It has input layer, one single-hidden layer and output layer. The input weights and the hidden biases of ELM-AE are randomly selected and orthogonal, respectively. Structure of the ELM-AE algorithm is shown in Fig. 2. There are N distinctive samples $x_i \in R^n \times R^j, i = 1, 2, \dots, N$, where j is

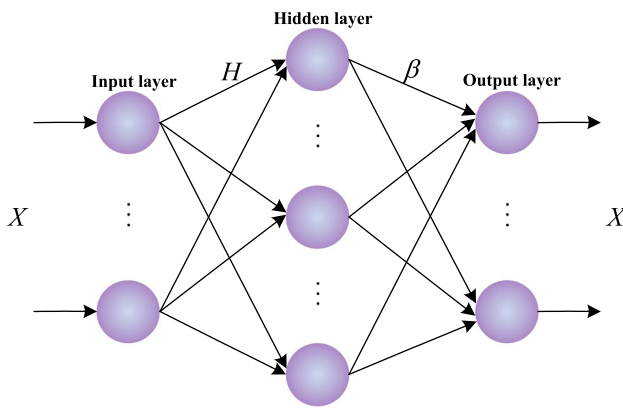


Fig. 2 Structure of the ELM-AE algorithm

the number of input nodes. The outputs of ELM-AE hidden layer can be written as

$$h = g(ax + b), \quad (5)$$

where $a^T a = I$, $b^T b = 1$.

The mathematical relationship for the outputs of hidden layer and the outputs of output layer can be represented as

$$h(x_i)\beta = x_i^T, \quad i = 1, 2, \dots, N, \quad (6)$$

where β represents the output weights of output layer. Moreover, there have three cases of calculating the β (Ding et al. 2015).

Case 1 The number of hidden layer nodes is less than the number of training data.

$$\beta = \left(\frac{I}{C} + H^T H \right)^{-1} H^T X. \quad (7)$$

Case 2 The number of hidden layer nodes is more than the number of training data.

$$\beta = H^T \left(\frac{I}{C} + H H^T \right)^{-1} X. \quad (8)$$

Case 3 The number of hidden layer nodes is equal to the number of training data.

$$\beta = H^{-1} X. \quad (9)$$

2.3 Multilayer extreme learning machine (ML-ELM)

ML-ELM uses the unsupervised learning to train parameters for every hidden layer, and ELM-AE is used to train parameters of every hidden layers for ML-ELM (Kasun et al. 2013). The ML-ELM algorithm does not need to be fine-tuned, and

then it can save time in terms of training network (Tang et al. 2016). Figure 3 shows structure of the ML-ELM algorithm.

For ML-ELM, the activation function of every hidden layer can be either linear or nonlinear piecewise (Kasun et al. 2013). Note: if the number of nodes for the k th hidden layer is equal to the number of nodes for the $k - 1$ th hidden layer, the activation function is linear; otherwise, it is nonlinear piecewise. The output of the k th hidden layer can be represented as

$$H_k = g \left((\beta^k)^T H_{k-1} \right), \quad (10)$$

where H_k and H_{k-1} are the output matrix and the input matrix of the k th hidden layer, respectively. $g(\cdot)$ represents the activation function of every hidden layer. Note: $k - 1 = 0$, we explain that H_0 is the input matrix of the first hidden layer (it is also the output matrix of the input layer) and H_1 is the output matrix of the first hidden layer.

2.4 The PLS-ML-ELM algorithm

PLS method brings together the characters of the multiple linear regression and principal component analysis, and it is an effective method to model between multi-independent variables and multidependent variables (Wold et al. 2001). Moreover, PLS can deal with the multicollinearity problem. Hence, the PLS-ML-ELM algorithm uses PLS method to get rid of the multicollinearity problem and noise. In addition, the output weights between the last hidden layer and the output layer also can be calculated directly through partial least square.

Suppose that the output matrix of the last hidden layer for ML-ELM is $H_{\text{Last}} \in R^{N \times m}$, where N is the number of sample data, m is the number of nodes for the last hidden layer. And the outputs of output layer are $Y \in R^{N \times l}$, where l represents the number of nodes for the output layer. PLS method is used to establish the relationship between H_{Last} and Y (Su et al. 2016). The mathematical representation is shown as

$$Y = H_{\text{Last}} \beta_{\text{PLS}} + \xi, \quad (11)$$

where β_{PLS} can be calculated by PLS method, and ξ is noise.

The main idea of the PLS-ML-ELM algorithm is as follows: the first component u_1 from H_{Last} (u_1 is the linear combination of h_1, h_2, \dots, h_m , where $H_{\text{Last}} = [h_1, h_2, \dots, h_m]_{N \times m}$), and the first component v_1 from Y (v_1 is the linear combination of y_1, y_2, \dots, y_l) is, respectively, extracted. In the same time, the correlation degree between u_1 and v_1 is maximum; then the regression model between y_1, y_2, \dots, y_l and u_1 is established. If the regression function reaches the satisfactory accuracy, the algorithm can be

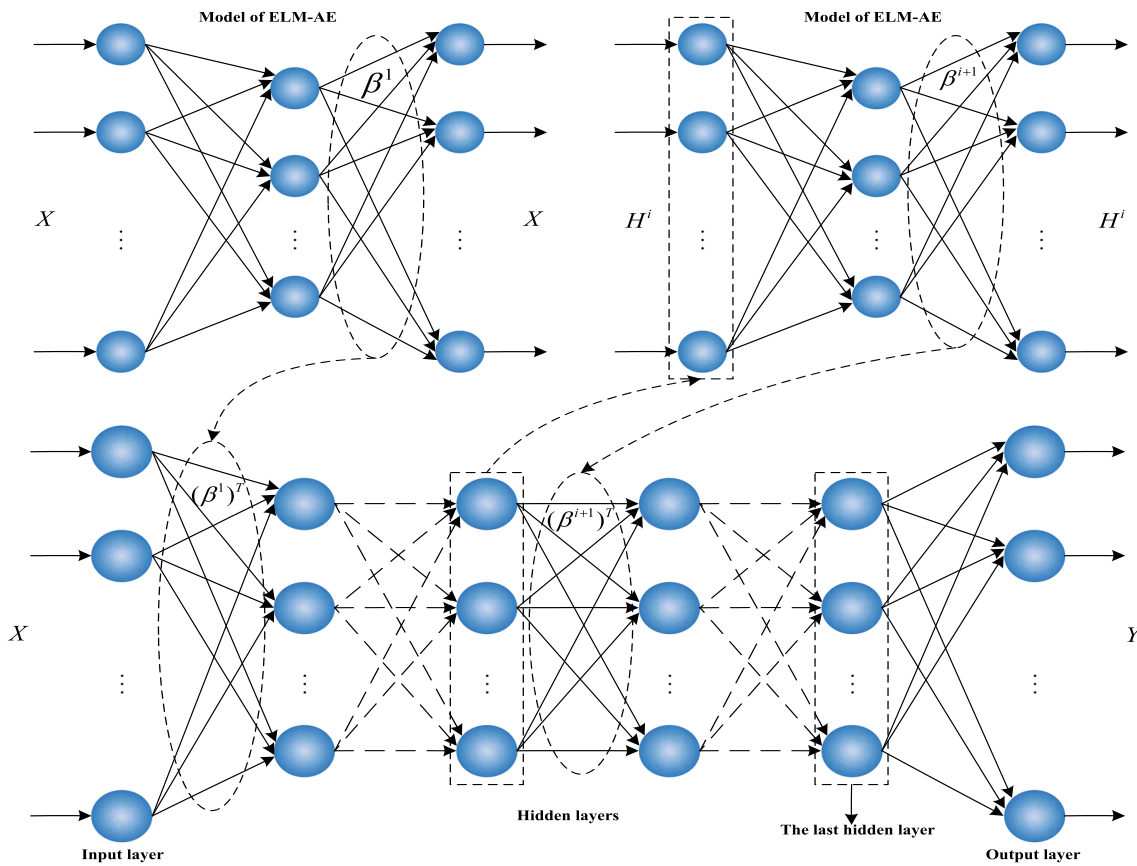


Fig. 3 Structure of the ML-ELM algorithm

terminated; otherwise, the u_2 and v_2 from the residual matrix of H_{Last} and Y are continuously extracted. Hence, the regression equation is established until it achieves the satisfactory accuracy (Geladi and Kowalski 1986).

Assume that there have r extracted components u_1, u_2, \dots, u_r , and the regression equation between y_1, y_2, \dots, y_l and u_1, u_2, \dots, u_r is established. Furthermore, the regression equation between y_1, y_2, \dots, y_l and h_1, h_2, \dots, h_m is established; then the output weights β_{PLS} is obtained. The detailed steps are as follows.

Step 0 Given a data set (x_i, t_i) which has N sample data, where $x_i = [x_{i1}, x_{i2}, \dots, x_{im}]^T \in R^m$ represents the i th sample data has m features, $t_i = [t_{i1}, t_{i2}, \dots, t_{il}]^T \in R^l$ is the target vector. Each of the hidden layer weights is initialized through applying ELM-AE which performs layer-wise unsupervised training. According to Eq. (10), the output matrixes of every hidden layer can be calculated, until H_{Last} , the output matrix of the last hidden layer, is obtained.

Step 1 Extract the first pair components u_1 and v_1 , and make the correlation degree between y_1, y_2, \dots, y_l and u_1 the largest; moreover, u_1 and v_1 should meet two requests:

1. u_1 and v_1 extract as much as possible information of variables from variable group, respectively.

2. Correlation degree between u_1 and v_1 reaches the largest.

And u_1 and v_1 can be represented as

$$\begin{cases} u_1 = H_{Last} p_1 = \begin{bmatrix} h_{11} & \cdots & h_{1m} \\ \vdots & \ddots & \vdots \\ h_{N1} & \cdots & h_{Nm} \end{bmatrix} \begin{bmatrix} p_{11} \\ \vdots \\ p_{1m} \end{bmatrix} = \begin{bmatrix} u_{11} \\ \vdots \\ u_{N1} \end{bmatrix}, \\ v_1 = Y q_1 = \begin{bmatrix} y_{11} & \cdots & y_{1l} \\ \vdots & \ddots & \vdots \\ y_{N1} & \cdots & y_{Nl} \end{bmatrix} \begin{bmatrix} q_{11} \\ \vdots \\ q_{1l} \end{bmatrix} = \begin{bmatrix} v_{11} \\ \vdots \\ v_{N1} \end{bmatrix}, \end{cases} \quad (12)$$

where $p_1 = [p_{11}, \dots, p_{1m}]^T$ and $q_1 = [q_{11}, \dots, q_{1l}]^T$ are the loading factor of H_{Last} and Y , respectively.

So requests (1) and (2) can be transformed as condition extremum problem

$$\begin{aligned} \max (u_1, v_1) &= (H_{Last} p_1, Y q_1) = p_1^T H_{Last}^T Y q_1, \\ \text{s.t.} \quad &\begin{cases} p_1^T p_1 = \|p_1\|^2 = 1, \\ q_1^T q_1 = \|q_1\|^2 = 1. \end{cases} \end{aligned} \quad (13)$$

Applying the Lagrange multiplier method, that is

$$L = p_1^T H_{Last}^T Y q_1 - \lambda_1 (p_1^T p_1 - 1) - \lambda_2 (q_1^T q_1 - 1). \quad (14)$$

Obtain the derivation of L about $p_1, q_1, \lambda_1, \lambda_2$

$$\begin{cases} \frac{\partial L}{\partial p_1} = H_{Last}^T Y q_1 - 2\lambda_1 p_1 = 0, \\ \frac{\partial L}{\partial q_1} = Y^T H_{Last} p_1 - 2\lambda_2 q_1 = 0, \\ \frac{\partial L}{\partial \lambda_1} = -(p_1^T p_1 - 1) = 0, \\ \frac{\partial L}{\partial \lambda_2} = -(q_1^T q_1 - 1) = 0. \end{cases} \tag{15}$$

According Eq. (15), Eq. (16) can be obtained,

$$2\lambda_1 = 2\lambda_2 = p_1^T H_{Last}^T Y q_1 = \langle H_{Last} p_1, Y q_1 \rangle. \tag{16}$$

Note: $\theta_1 = 2\lambda_1 = 2\lambda_2 = p_1^T H_{Last}^T Y q_1$, so θ_1 is the objective function value of the condition extremum problem. Then, there have

$$H_{Last}^T Y q_1 = \theta_1 p_1, \tag{17}$$

$$Y^T H_{Last} p_1 = \theta_1 q_1. \tag{18}$$

According to Eq. (17) and Eqs. (18), (19) can be obtained

$$H_{Last}^T Y Y^T H_{Last} p_1 = \theta_1^2 p_1. \tag{19}$$

Observe Eq. (19), p_1 is the eigenvector of matrix

$H_{Last}^T Y Y^T H_{Last}$, θ_1^2 is the corresponding eigenvalue, and θ_1 is the objective function value. According to Eq. (19), q_1 can be calculated by p_1 ; then the score vectors can be represented by the loading factors p_1 and q_1 ,

$$\begin{cases} u_1 = H_{Last} p_1, \\ v_1 = Y q_1. \end{cases} \tag{20}$$

Step 2 Establish the regression equation between h_1, h_2, \dots, h_m and $u_1, y_1, y_2, \dots, y_l$ and v_1 , respectively. Note $E_0 = H_{Last}, F_0 = Y$. So the regression model is

$$\begin{cases} E_0 = u_1 \alpha_1^T + E_1, \\ F_0 = v_1 \gamma_1^T + F_1, \end{cases} \tag{21}$$

where $\alpha_1 = [\alpha_{11}, \alpha_{12}, \dots, \alpha_{1m}]^T$ and $\gamma_1 = [\gamma_{11}, \gamma_{12}, \dots, \gamma_{1l}]^T$ are vectors of parameter, E_1 and F_1 are residual matrixes. The least square estimation of regression parameter vectors α_1 and γ_1 is

$$\begin{cases} \alpha_1 = \frac{E_0^T u_1}{\|u_1\|^2}, \\ \gamma_1 = \frac{F_0^T v_1}{\|v_1\|^2}. \end{cases} \tag{22}$$

Step 3 Substitute E_1 and F_1 for E_0 and F_0 . If elements of F_1 are close to zero, the regression equation which is established by using the first pair score vector meets the accuracy, and

cease. Otherwise, repeat Step 1 and Step 2; then the second score vector can be expressed as

$$\begin{cases} u_2 = E_1 p_2, \\ v_2 = F_1 q_2, \end{cases} \tag{23}$$

so the regression model is displayed as

$$\begin{cases} E_0 = u_1 \alpha_1^T + u_2 \alpha_2^T + E_2, \\ F_0 = v_1 \gamma_1^T + v_2 \gamma_2^T + F_2. \end{cases} \tag{24}$$

where α_2 and γ_2 are vectors of regression parameter, they can be represented as

$$\begin{cases} \alpha_2 = \frac{E_1^T u_2}{\|u_2\|^2}, \\ \gamma_2 = \frac{F_1^T v_2}{\|v_2\|^2}. \end{cases} \tag{25}$$

Step 4 Repeat Step 2 and Step 3, until r principal components are reserved. Meanwhile, the rest, $m - r$ components have little variation, so they can be seen as the noise or reason of generating the multicollinearity of the last hidden layer. Furthermore, the biases of E_r and F_r are extremely small. So there has the regression model

$$\begin{cases} E_0 = \sum_{i=1}^r u_i \alpha_i^T + u_2 \alpha_2^T + \dots + u_r \alpha_r^T + E_r \\ \quad = U \alpha^T + E_r, \\ F_0 = \sum_{i=1}^r v_i \gamma_i^T + v_2 \gamma_2^T + \dots + v_r \gamma_r^T + F_r \\ \quad = V \gamma^T + F_r. \end{cases} \tag{26}$$

Moreover, there has inner relationship for u_k and v_k (Geladi and Kowalski 1986); then the relationship can be described as

$$v_k = u_k b_k, k = 1, 2, \dots, r. \tag{27}$$

So the equation of F_0 can be rewritten as

$$F_0 = V \gamma^T + F_r = \sum_{i=1}^r u_i b_i \gamma_i^T + u_2 b_2 \gamma_2^T + \dots + u_r b_r \gamma_r^T + F_r = U B \gamma^T + F_r, \tag{28}$$

where $\hat{U} = E_0 P$, the regression equation can be expressed as

$$\hat{F}_0 = E_0 P B \gamma^T + F_r. \tag{29}$$

According to the above analysis, the output weight of the output layer can be represented as

$$\hat{\beta}_{PLS} = P B \gamma^T, \tag{30}$$

where P is component matrix, B is diagonal matrix, γ^T is the weight matrix of F_0 .

2.5 The proposed EPLS-ML-ELM algorithm

For PLS-ML-ELM, it may have different results in different trails of simulations. Hence, in order to overcome this problem, this paper constructs L PLS-ML-ELM networks to form the EPLS-ML-ELM algorithm. EPLS-ML-ELM has better generalization performance than PLS-ML-ELM. For the whole PLS-ML-ELMs of EPLS-ML-ELM, they have same number of hidden layers, and they have same number of every hidden layer nodes. In addition, for each PLS-ML-ELM, every hidden layer weights are initialized through applying ELM-AE. The detailed steps of the EPLS-ML-ELM algorithm are as follows.

Step 1 Assemble L PLS-ML-ELMs with same number of hidden layers and same number of every hidden layer nodes, and same action function for each hidden layer node.

Step 2 For each PLS-ML-ELM, every hidden layer weights are initialized by applying ELM-AE. And the output weights of each PLS-ML-ELM can be obtained according to Eq. (30).

Step 3 The output matrix of output layer for each PLS-ML-ELM network is obtained.

Step 4 There are two cases for prediction result of the EPLS-ML-ELM algorithm.

Case 1 For the regression problem, the average value of the whole prediction results obtained by L PLS-ML-ELM networks is used as the final prediction result of the EPLS-ML-ELM algorithm. It can be represented as

$$O_{\text{final}} = \frac{1}{L} \sum_{i=1}^L O_i, i = 1, 2, \dots, L, \tag{31}$$

where i indicates the i th PLS-ML-ELM network.

Case 2 For the classification problem, the prediction result of each sample is determined by the highest vote (Xue et al. 2014). Each sample has a class label which is a vector v . The dimension of vector v is equal to the whole number of classes (suppose there has p classes). For EPLS-ML-ELM, if the prediction result of the i th PLS-ML-ELM network is the k th class, then the k th number of the corresponding vector v_i is set to one; otherwise, it is set to zero, where $i = 1, 2, \dots, L$ and $k = 1, 2, \dots, p$. When each sample is predicted by the whole PLS-ML-ELMs, the prediction vector v_{final} of each sample can be calculated as

$$v_{\text{final}} = \sum_{i=1}^L v_i, \tag{32}$$

where L is the number of PLS-ML-ELMs in EPLS-ML-ELM. The biggest label in v_{final} is used as the prediction label.

3 Data-driven prediction model of adjusting the burden distribution matrix for blast furnace

Burden distribution matrix is extremely important for smooth operation of the blast furnace. For example, adjusting burden distribution matrix is an effective measure to control radial distribution of gas flow in blast furnace; adjusting burden distribution matrix can improve gas utilization rate and utilization coefficient. Moreover, adjusting burden distribution matrix can make the relationship between the blast pressure and the blast volume stable. Reasonable pressure difference between the blast pressure and the top pressure can be obtained by adjusting the burden distribution matrix. Besides, reasonable permeability index can be obtained by adjusting burden distribution matrix. In practical operation process, operators determine whether the burden distribution matrix needs to be adjusted according to the blast furnace condition parameters and operation experience. At the blast furnace operation site, these blast furnace condition parameters (the blast volume, the blast pressure, the blast velocity, the top pressure, the permeability index, the gas utilization rate and the utilization coefficient) are used to determine whether the burden distribution matrix needs to be adjusted. Data of these seven parameters can be obtained. Moreover, for the burden distribution matrix, there have two class labels: class 1 represents that the burden distribution matrix needs to be adjusted, and class 0 represents that the burden distribution matrix does not need to adjusted. According to the above explanation, the two class labels of the burden distribution matrix are the dependent variables, and these seven blast furnace condition parameters are the independent variables. Hence, based on the collected data, operation experience and machine learning algorithm, this paper will establish a data-driven prediction model to determine whether the burden distribution matrix needs to be adjusted.

In this paper, the proposed EPLS-ML-ELM algorithm is used as the prediction algorithm for data-driven prediction model. The structure of this data-driven prediction model is shown in Fig. 4. The detailed modeling steps are introduced as follows.

Step 1 Determine the parameters of data-driven prediction model.

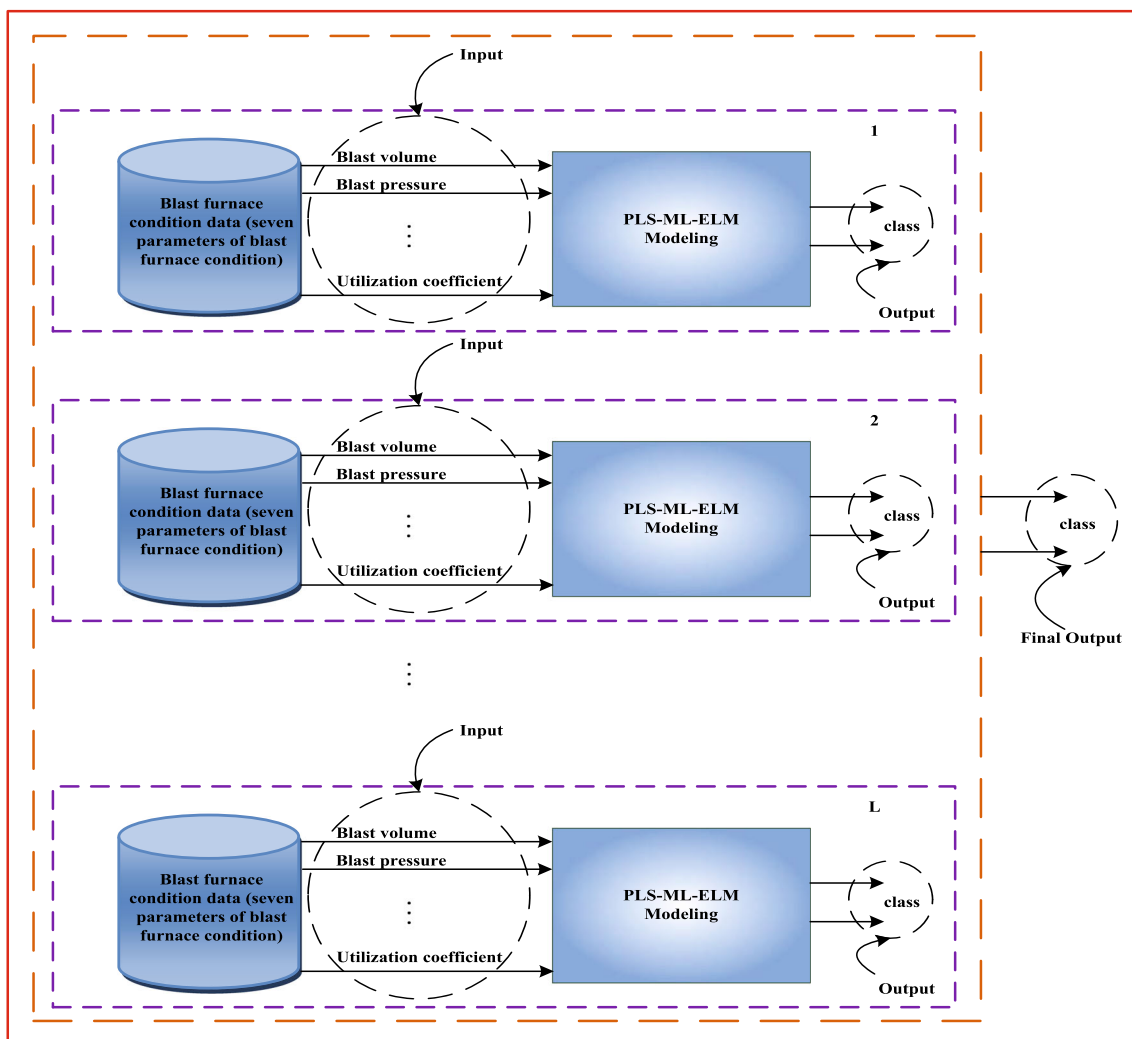


Fig. 4 Structure of the data-driven prediction model of adjusting burden distribution matrix based on the EPLS-ML-ELM algorithm

According to the above analysis, there are seven blast furnace condition parameters (the blast volume, the blast pressure, the blast velocity, the top pressure, the permeability index, the gas utilization rate and the utilization coefficient) are used as the input parameters of data-driven prediction model. And class labels (1 represents that the burden distribution matrix needs to be adjusted, and 0 represents that the burden distribution matrix does not need to be adjusted) are used as the output parameters of data-driven prediction model.

Step 2 Establish the EPLS-ML-ELM structure.

The EPLS-ML-ELM algorithm is consisted of several PLS-ML-ELMs, and the number of all PLS-ML-ELMs is represented as L . Moreover, for EPLS-ML-ELM, each PLS-ML-ELM uses the whole dataset.

Step 2 includes seven small parts.

(2-1) In this paper, the EPLS-ML-ELM algorithm is used as the prediction algorithm for data-driven prediction model. And EPLS-ML-ELM is consisted of L PLS-ML-

ELMs. According to Step 1, for each PLS-ML-ELM, there are seven input nodes of input layer and two output nodes of output layer.

(2-2) Determine the number of hidden layers and the number of every hidden layer nodes. For all PLS-ML-ELMs, they have same number of hidden layers and same number of every hidden layer nodes. In this paper, both the number of hidden layers and the number of each hidden layer nodes are determined through many times simulation testing. Moreover, for each PLS-ML-ELM, every hidden layer weight is initialized by applying ELM-AE.

(2-3) The activation function of every hidden layer is sigmoid function.

(2-4) Determine the ensemble number L . The EPLS-ML-ELM algorithm is consisted of several PLS-ML-ELMs. The number of the whole PLS-ML-ELMs needs to be chosen.

Table 1 Information of sampling data

No.	Blast furnace condition parameter (input)	Average	Min	Max	SD	Dimension
1	Blast volume	4210.50	3159	4591	214.80	m ³ /min
2	Blast pressure	327.90	216	356	20.01	kPa
3	Blast velocity	256.38	231	284	7.33	m/s
4	Top pressure	180.43	110	210	15.78	kPa
5	Permeability index	0.74	0.6	0.97	0.05	m ³ /(kPa · min)
6	Gas utilization rate	48.87	43.80	51.93	1.51	%
7	Utilization coefficient	2.14	0.59	2.49	0.29	t/(m ³ · day)

(2–5) For each PLS-ML-ELM, the output matrix of every hidden layer H_i is calculated, where i represents the i th hidden layer.

(2–6) For each PLS-ML-ELM, the connection weight H_{Last} between the last hidden layer and the output layer is calculated by Eq. (30).

(2–7) For EPLS-ML-ELM, the prediction result can be obtained by Case 2 of Step 4 in Sect. 2.5. Based on these two steps, the structure of data-driven prediction model of adjusting the burden distribution matrix is established. *Step 3* Verify the data-driven prediction model.

If evaluation criterions of this data-driven prediction model meet the required precision, then the data-driven prediction model has been established. Otherwise, return (2–2) to adjust the number of hidden layers and the number of every hidden layer nodes, and return (2–4) to adjust the number of PLS-ML-ELMs in EPLS-ML-ELM, and then reestablish the data-driven prediction model.

4 Simulation results

In order to testify the rationality and the prediction accuracy of the data-driven prediction model, this paper adapts the production data of the Blast Furnace with 2500 m³ to testify. There are 1000 data pairs (502 data pairs for class 1, 498 data pairs for class 0). Eight hundred data pairs are used as the training data, and the rest data pairs are the testing data. The detailed information of data for these seven blast furnace condition parameters is shown in Table 1. From Table 1, there is huge difference for the dimension of these seven parameters. The difference may provide fluctuations and influence the accuracy of the data-driven prediction model. Therefore, data of these seven parameters are normalized in this paper. Besides, in order to indicate the data-driven prediction model based on the EPLS-ML-ELM algorithm that has better prediction accuracy and generalization performance, this paper also uses the SVM algorithm, the ELM algorithm, the ML-ELM algorithm and the PLS-ML-ELM algorithm to establish the data-driven prediction model. And these data-driven pre-

diction models are compared with the data-driven prediction model based on the EPLS-ML-ELM algorithm. The whole simulation experiments have been conducted in MATLAB 8.3.0 software.

For the whole data-driven prediction models based on algorithms, this paper adopts the training time, the testing time, the accuracy and the F -score as evaluation criterions. Accuracy is a percentage and can illustrate the good generalization performance of data-driven prediction model when it is close to 100%. F -score ranges from 0 to 1, and it contains precision and recall. F -score reaches the best value at 1 and the worst value at 0. The representations of the accuracy and the F -score are shown as

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}, \quad (33)$$

$$\begin{aligned} F\text{-score} &= 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \\ &= \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}, \end{aligned} \quad (34)$$

where TP is the number of samples that correctly predicted to “1”, FP is the number of samples that falsely predicted to “1”, FN is that the number of samples that falsely predicted to “0”, and TN is the number of samples that correctly predicted to “0”; precision represents precision ratio (precision = $\frac{\text{TP}}{\text{TP} + \text{FP}}$), and recall represents recall ratio (recall = $\frac{\text{TP}}{\text{TP} + \text{FN}}$).

In order to ensure the data-driven prediction model to reach the optimal target, this paper repeats many times experiments to determine the number of hidden layers and the number of nodes for these algorithms. Moreover, this paper also repeats many times experiments to determine the numbers of nodes in all the hidden layers. For ELM, the number of hidden layer nodes is 100. For ML-ELM, two hidden layers, three hidden layers and four hidden layers are adopted to test in this paper, respectively. The numbers of every hidden layer nodes are shown in Table 2. And the prediction results are given in Table 3. It can be shown that the prediction accuracy of the ML-ELM algorithm with three hidden layers or four hidden layers is better than the ML-ELM algorithm with two hidden layers. In addition, evaluation

Table 2 Numbers of nodes of ML-ELM which has different numbers of hidden layers

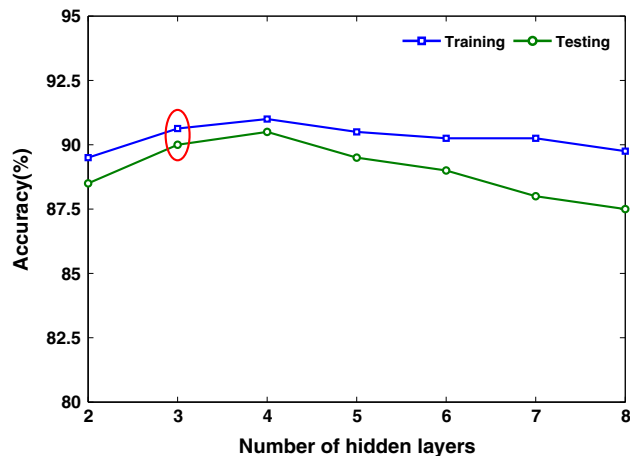
Number of hidden layers	1st hidden layer	2nd hidden layer	3rd hidden layer	4th hidden layer
2	200	150	–	–
3	200	150	100	–
3	200	200	150	–
3	300	200	150	–
4	200	150	100	70
4	200	200	150	100
4	300	200	150	100

Table 3 Comparison of prediction results using the ML-ELM which has different numbers of hidden layers

Number of hidden layers	Number of nodes for every hidden layer	Time (s)		Accuracy (%)		<i>F</i> -score	
		Training	Testing	Training	Testing	Training	Testing
2	200, 150	0.2028	0.0156	89.50	88.50	0.8955	0.8856
3	200, 150, 100	0.3588	0.0312	90.63	90.00	0.9071	0.9019
3	200, 200, 150	0.4524	0.0312	90.75	90.50	0.9077	0.9064
3	300, 200, 150	0.6552	0.0468	91.13	90.50	0.9116	0.9073
4	200, 150, 100, 70	0.4056	0.0312	91.00	90.50	0.9102	0.9055
4	200, 200, 150, 100	0.5616	0.0312	91.25	91.00	0.9129	0.9091
4	300, 200, 150, 100	0.7332	0.0468	91.38	91.00	0.9134	0.9109

criteria of the ML-ELM algorithm containing three hidden layers are approximate with the ML-ELM algorithm with four hidden layers. To further demonstrate the influence of the number for hidden layers in the ML-ELM algorithm about prediction accuracy, the five hidden layers, the six hidden layers, the seven hidden layers and the eight hidden layers are adopted, respectively. However, as the increase in the number for hidden layers, the prediction accuracy is not significantly improved. Hence, it can be clearly known that the ML-ELM algorithm with three hidden layers can well establish the data-driven prediction model. Moreover, the simulation results of ML-ELM with different hidden layers are displayed in Fig. 5. Based on the above analysis, this paper adopts the ML-ELM algorithm with three hidden layers. For PLS-ML-ELM, it also adopts three hidden layers structure. And the number of every hidden layer nodes is same with ML-ELM's.

For EPLS-ML-ELM, each PLS-ML-ELM also adopts three hidden layers structure. And the number of every hidden layer nodes is also same with ML-ELM's. It is worth noting that the ensemble parameter L of the EPLS-ML-ELM algorithm needs to be determined. In this paper, L is set as 5, 10, 15, 20 and 25, respectively. Prediction results of different number for PLS-ML-ELMs are shown in Table 4 and Fig. 6. From Table 4 and Fig. 6, it can be seen that the best result is obtained when the number of PLS-ML-ELMs in EPLS-ML-ELM is 15, and the prediction result is better

**Fig. 5** Accuracy of different numbers for hidden layers of ML-ELM

than single PLS-ML-ELM as well. Hence, the EPLS-ML-ELM algorithm which is consisted of 15 PLS-ML-ELMs is used as the prediction algorithm of the data-driven prediction model.

For determination of the burden distribution matrix whether needs to be adjusted, this paper uses different algorithms to establish the data-driven prediction model. And all the algorithms are, respectively, run 50 times. The standard deviation (SD) is used as an evaluation criterion, and it is written as

Table 4 Comparison of prediction results using the EPLS-ML-ELM which has different numbers of PLS-ML-ELMs

Algorithm	Ensemble number	Time (s)		Accuracy (%)		F-score	
		Training	Testing	Training	Testing	Training	Testing
PLS-ML-ELM 200, 150, 100 → 70	–	0.4485	0.0396	92.13	91.50	0.9204	0.9146
EPLS-ML-ELM 200, 150, 100 → 70	5	2.5369	0.1806	90.88	90.50	0.9089	0.9045
	10	4.9631	0.3412	91.38	91.00	0.9142	0.9126
	15	7.3106	0.5117	93.75	93.50	0.9382	0.9359
	20	9.1416	0.6573	92.50	91.50	0.9152	0.9163
	25	11.5308	0.8425	91.88	91.00	0.9189	0.9010

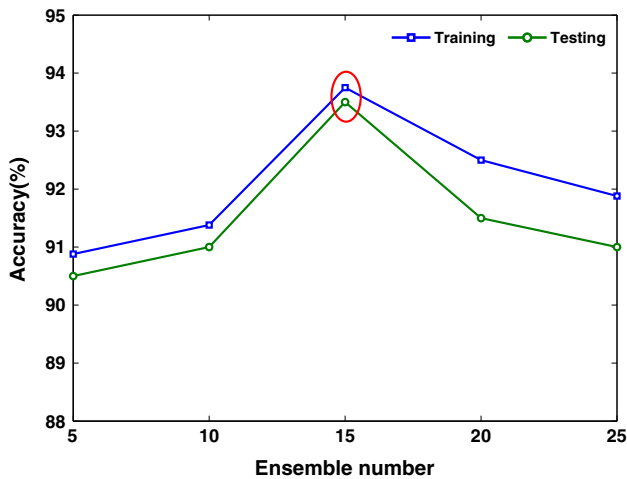


Fig. 6 Accuracy of EPLS-ML-ELM with different numbers of PLS-ML-ELMs

$$SD = \sqrt{\frac{\sum_{i=1}^N (X_i - \bar{X})^2}{n - 1}} \tag{35}$$

where X_i is the accuracy of the i th simulation, \bar{X} is the average value of all the accuracies in the whole simulations, and n is the number of all the simulations.

Prediction results of the whole data-driven prediction models based on different algorithms (the SVM algorithm, the ELM algorithm, the ML-ELM algorithm, the PLS-ML-

ELM algorithm and the proposed EPLS-ML-ELM algorithm) are shown in Table 5 and Fig. 7. From Table 5 and Fig. 7, the data-driven prediction model based on the EPLS-ML-ELM algorithm has better prediction results than others. In Table 5, the training accuracy and the testing accuracy of SVM are 75.88 and 74.00%; the training accuracy and the testing accuracy of ELM are 88.13 and 86.50%; the training accuracy and the testing accuracy of ML-ELM are 90.63 and 90.00%; the training accuracy and the testing accuracy of PLS-ML-ELM are 92.13 and 91.50%; the training accuracy and the testing accuracy of the proposed EPLS-ML-ELM algorithm are 93.75 and 93.50%. As the above results shown, the prediction accuracy of the proposed EPLS-ML-ELM algorithm is better than other algorithms. In terms of training time and testing time, the SVM algorithm costs 0.3609 and 0.0342 s; the ELM algorithm costs 0.0936 and 0.0033 s; the ML-ELM algorithm costs 0.3588 and 0.0312 s; the PLS-ML-ELM algorithm costs 0.4485 and 0.0396 s; the proposed EPLS-ML-ELM algorithm costs 7.3106 and 0.5117 s. Although the data-driven prediction model based on the proposed EPLS-ML-ELM algorithm costs the most training time and testing time, its prediction accuracy is more precise than data-driven prediction models based on other algorithms. In addition, the training time and the testing time of the proposed EPLS-ML-ELM algorithm are measured by seconds, which meet the requirement of the blast furnace process. For the training F -score and

Table 5 Comparison of prediction results using different algorithms

Algorithm	Number of hidden layers	Time (s)		Accuracy (%)		F-score		SD	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing
SVM	–	0.3609	0.0342	75.88	74.00	0.7602	0.7474	0.0253	0.0280
ELM	100	0.0936	0.0033	88.13	86.50	0.8708	0.8696	0.0153	0.0176
ML-ELM	200, 150, 100	0.3588	0.0312	90.63	90.00	0.9071	0.9019	0.0110	0.0134
PLS-ML-ELM	200, 150, 100 → 70	0.4485	0.0396	92.13	91.50	0.9204	0.9146	0.0079	0.0094
EPLS-ML-ELM ensemble number = 15	200, 150, 100 → 70	7.3106	0.5117	93.75	93.50	0.9382	0.9359	0.0053	0.0062

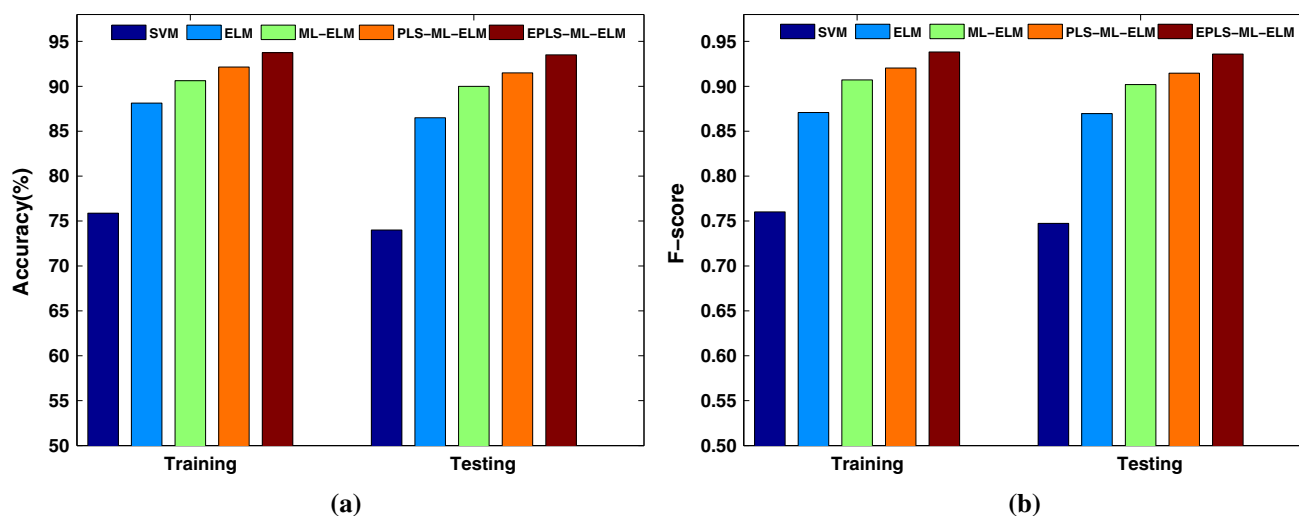


Fig. 7 Results of data-driven prediction models based on different algorithms. **a** Accuracy for different algorithms. **b** F -score for different algorithms

the testing F -score, the data-driven prediction model based on the proposed EPLS-ML-ELM algorithm is 0.9382 and 0.9359. They are greater than data-driven prediction models based on other algorithms. The data-driven prediction model based on the proposed EPLS-ML-ELM algorithm is 0.0053 and 0.0062 in the aspects of the training SD and the testing SD. And they are smaller than the data-driven prediction models based on other algorithms, which indicates that the proposed EPLS-ML-ELM algorithm is better and more stable than other algorithms. According to the above analysis, the proposed EPLS-ML-ELM algorithm has better prediction accuracy and generalization performance than other algorithms. Compared with data-driven prediction models based on other algorithms, the data-driven prediction model based on the proposed EPLS-ML-ELM algorithm can better predict the burden distribution matrix whether needs to be adjusted. In order to further indicate the proposed EPLS-ML-ELM algorithm has better generalization performance, the standard data sets are used to testify EPLS-ML-ELM. Moreover, it is compared with other algorithms. It is shown in Appendixes.

5 Conclusions

Reasonable burden distribution matrix of the blast furnace can realize the smooth operation of the blast furnace. It is extremely important for the blast furnace. Based on the collected data of blast furnace production site, operation experience and machine learning algorithm, this paper establishes a data-driven prediction model. This data-driven prediction model can determine whether the burden distribution matrix needs to be adjusted. In this paper, the proposed EPLS-ML-ELM algorithm is used to establish the data-driven prediction

model. This proposed algorithm is based on the PLS-ML-ELM algorithm and the ensemble model. For PLS-ML-ELM, the PLS method is used to overcome the multicollinearity problem. However, PLS-ML-ELM may have different results in different trails of simulations. Hence, the ensemble model is introduced to overcome this problem. Then the EPLS-ML-ELM is consisted of several PLS-ML-ELMs. The blast furnace production data are used to validate the data-driven prediction model based on the EPLS-ML-ELM algorithm. Compared with data-driven prediction models based on other algorithms, simulation results illustrate that the data-driven prediction model based on the EPLS-ML-ELM algorithm can better determine whether the burden distribution matrix needs to be adjusted. Furthermore, this data-driven prediction model can offer decision for the subsequent operation of the blast furnace. In the future, we will continuously improve the ML-ELM algorithm and make the data-driven prediction model more precise.

Acknowledgements This work was supported by the National Nature Science Foundation of China under Grants No. 61673056, the Key Program of National Nature Science Foundation of China under Grant No. 61333002, the Beijing Natural Science Foundation (4182039), the National Nature Science Foundation of China under Grants No. 61673055 and the Beijing Key Discipline Construction Project (XK100080537).

Compliance with ethical standards

Conflict of interest The authors declare no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Appendixes

In Appendixes, the proposed EPLS-ML-ELM algorithm is verified by using standard data sets. The ELM algorithm, the ML-ELM algorithm and the PLS-ML-ELM algorithm are also used to compare with the proposed EPLS-ML-ELM algorithm. There are two parts. Appendix A is for the regression problem, and Appendix B is for the classification problem.

Appendix A

For regression problem, the Abalone data set (UCI 1995) and the California Housing data set (StatLib 1997) are used to testify. The benchmark problems are shown in Table 6. The number of hidden layer nodes for the ELM algorithm is 100. The numbers of the whole hidden layers nodes for the ML-ELM algorithm with three hidden layers are 200, 150 and 100, respectively. For PLS-ML-ELM, the number of hidden layers and the number of every hidden layer nodes are same with ML-ELM's, respectively. For EPLS-ML-ELM, the number of the whole PLS-ML-ELMs is 10. Each PLS-ML-ELM has same number of hidden layers and same number of every hidden layer nodes with ML-ELM's. In addition, all the experiments are carried out 50 trials. In

Table 6 Information of simulation data for regression problem

Data set	Attributes	Training data	Testing data
Abalone	8	3177	1000
California Housing	9	16,460	4000

Table 7 Comparison of prediction results using different algorithms for regression problem

Data set	Algorithm	Number of hidden layers	Time(s)		RMSE	
			Training	Testing	Training	Testing
Abalone	ELM	100	0.4212	0.0312	0.0771	0.0740
	ML-ELM	200, 150, 100	1.0296	0.0784	0.0282	0.0270
	PLS-ML-ELM	200, 150, 100 → 70	1.2165	0.1072	0.0259	0.0246
	EPLS-ML-ELM ensemble number = 10	200, 150, 100 → 70	12.2118	1.1032	0.0225	0.0213
California Housing	ELM	100	1.2792	0.0780	0.1395	0.1350
	ML-ELM	200, 150, 100	3.0732	0.1872	0.0382	0.0338
	PLS-ML-ELM	200, 150, 100 → 70	3.1368	0.2056	0.0353	0.0313
	EPLS-ML-ELM ensemble number = 10	200, 150, 100 → 70	31.4326	2.1328	0.0298	0.0267

Table 8 Information of simulation data for classification problem

Data set	Attributes	Class	Training data	Testing data
Image segmentation	19	7	1810	500
Letter	16	26	18,000	2000
MNIST	784	10	60,000	10,000

this part, the root mean square error (RMSE) is used as the evaluation criterion, and the representation is shown as

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (X_i - \hat{X}_i)^2}{n}}, \tag{36}$$

where n is the total number of testing data; X_i and \hat{X}_i are actual data and prediction data at the i th sample.

The comparison results are shown in Table 7. These comparison results illustrate that the proposed EPLS-ML-ELM algorithm has better generalization performance than other algorithms.

Appendix B

For classification problem, the image segmentation data set (UCI 1990), the letter data set (UCI 1991) and the MNIST data set (LeCun et al. 1998) are used to testify. The benchmark problems are shown in Table 8. All the experiments are carried out 50 trials. The comparison results are shown in Tables 9, 10 and Fig. 8. These comparison results illustrate that the proposed EPLS-ML-ELM algorithm has better generalization performance than other algorithms.

For the letter data set, the parameter set of the whole algorithms is different from Appendix A. The number of hidden layer nodes for the ELM algorithm is 200. The numbers of nodes for the whole hidden layers for the ML-ELM algorithm with three hidden layers are 200, 200 and 400, respectively. For PLS-ML-ELM, the number of hidden layers and the number of every hidden layer nodes are same with ML-ELM's in Appendix B, respectively. For EPLS-ML-ELM, the number of the whole PLS-ML-ELMs is 10, and each PLS-ML-ELM

Table 9 Comparison of prediction results using different algorithms for classification problem

Data set	Algorithm	Number of hidden layer nodes	Time (s)		Accuracy (%)	
			Training	Testing	Training	Testing
Image segmentation	ELM	100	0.4212	0.0312	94.16	93.13
	ML-ELM	200, 150, 100	1.0296	0.0784	95.56	94.70
	PLS-ML-ELM	200, 150, 100 → 70	1.2165	0.1072	96.11	95.49
	EPLS-ML-ELM ensemble number = 10	200, 150, 100 → 70	12.2118	1.1032	96.78	96.27
Letter	ELM	200	1.1076	0.0624	79.42	78.05
	ML-ELM	200, 200, 400	13.3562	0.1716	80.56	79.55
	PLS-ML-ELM	200, 200, 400 → 300	15.6953	0.2028	82.21	81.55
	EPLS-ML-ELM ensemble number = 10	200, 200, 400 → 300	168.6483	2.1995	83.79	83.45

Table 10 Comparison of prediction results using different algorithms on MNIST data set

Algorithm	Number of hidden layer nodes	Accuracy (%)	
		Training	Testing
ELM	1000	92.92	92.36
	1500	93.57	92.89
	2000	94.28	93.51
ML-ELM	700, 700, 1000	95.00	94.62
	700, 700, 1500	95.73	95.29
	700, 700, 2000	96.51	96.03
PLS-ML-ELM	700, 700, 1000 → 900	95.18	94.81
	700, 700, 1500 → 1400	95.92	95.49
	700, 700, 2000 → 1900	96.73	96.26
EPLS-ML-ELM ensemble number = 5	700, 700, 1000 → 900	95.35	95.02
	700, 700, 1500 → 1400	96.11	95.71
	700, 700, 2000 → 1900	96.94	96.53

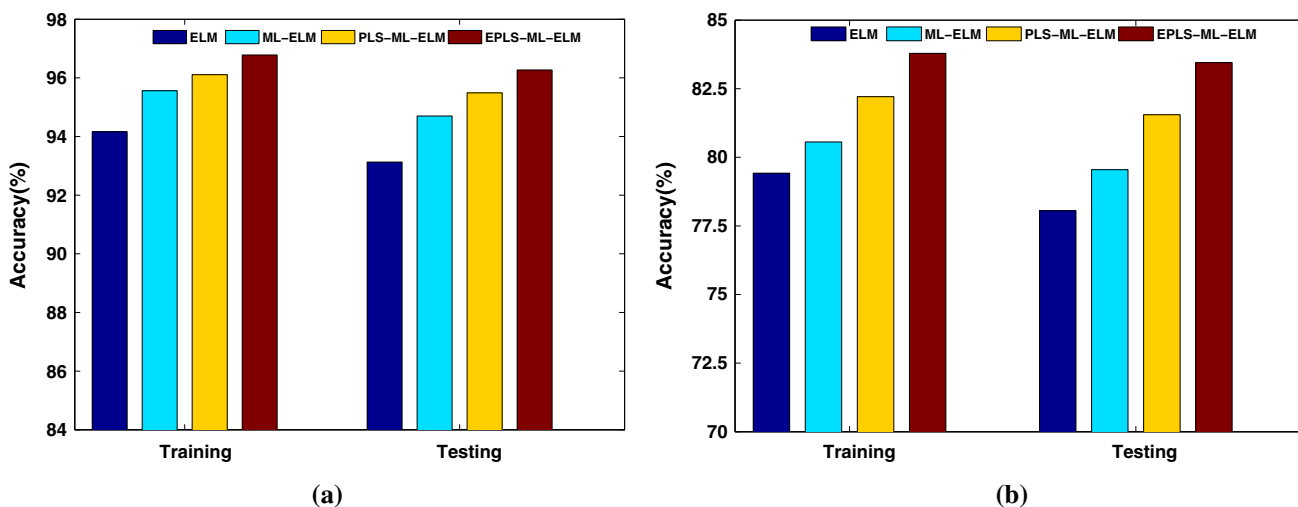


Fig. 8 Results of different data sets based on different algorithms. **a** Accuracy for image segmentation data set based on different algorithms. **b** Accuracy for letter data set based on different algorithms

has same number of hidden layers and same number of every hidden layer nodes with ML-ELM's in Appendix B.

For the MNIST data set, the parameter set of the whole algorithms is also different from Appendix A. The number

of hidden layer nodes for the ELM algorithm is, respectively, set as 1000, 1500 and 2000. For ML-ELM with three hidden layers, the numbers of nodes for the whole hidden layers are, respectively, set as [700–700–1000], [700–700–1500] and [700–700–2000]. For PLS-ML-ELM, the number of hidden layers and the number of every hidden layer nodes are same with ML-ELM's in Appendix B, respectively. For EPLS-ML-ELM, the number of the whole PLS-ML-ELMs is 5, and each PLS-ML-ELM has the same number of hidden layers and the same number of every hidden layer nodes with ML-ELM's in Appendix B. In addition, the whole MNIST data set is used in each PLS-ML-ELM of EPLS-ML-ELM.

References

- Bengio Y (2009) Learning deep architectures for ai. *Found Trends Mach Learn* 2(1):1–127
- Cao JW, Lin ZP, Huang GB, Liu N (2012) Voting based extreme learning machine. *Inf Sci* 185(1):66–77
- Ding S, Zhang N, Xu X, Guo LL, Zhang J (2015) Deep extreme learning machine and its application in EEG classification. *Math Probl Eng* 11. <https://doi.org/10.1155/2015/129,021> (Article ID 129021)
- Ding SF, Zhang N, Zhang J, Xu XZ, Shi ZZ (2017) Unsupervised extreme learning machine with representational features. *Int J Mach Learn Cybern* 8(2):587–595
- Geerdes M, Toxopeus H, der Vliet CV, Chaigneau R, Vander T (2009) *Modern blast furnace ironmaking: an introduction*, vol 4. IOS Press, Amsterdam
- Geladi P, Kowalski BR (1986) Partial least-squares regression: a tutorial. *Anal Chim Acta* 185:1–17
- Hansen LK, Salamon P (1990) Neural network ensembles. *IEEE Trans Pattern Anal Mach Intell* 12(10):993–1001
- Huang G, Liu TC, Yang Y, Lin ZP, Song SJ, Wu C (2015) Discriminative clustering via extreme learning machine. *Neural Netw* 70:1–8
- Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE international joint conference on neural networks, 2004. Proceedings, vol 2. pp 985–990
- Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. *Neurocomputing* 70(1):489–501
- Huang GB, Zhou H, Ding X, Zhang R (2012) Extreme learning machine for regression and multiclass classification. *IEEE Trans Syst Man Cybern B Cybern* 42(2):513–529
- Huang ZY, Yu YL, Gu J, Liu HP (2017) An efficient method for traffic sign recognition based on extreme learning machine. *IEEE Trans Cybern* 47(4):920–933
- Kasun LLC, Zhou H, Huang GB, Vong CM (2013) Representational learning with elms for big data. *IEEE Intell Syst* 28(6):31–34
- LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. *Proc IEEE* 86(11):2278–2324
- Li MB, Huang GB, Saratchandran P, Sundararajan N (2005) Fully complex extreme learning machine. *Neurocomputing* 68:306–314
- Liang NY, Huang GB, Saratchandran P, Sundararajan N (2006) A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Trans Neural Netw* 17(6):1411–1423
- Liu MM, Liu B, Zhang C, Wang WD, Sun W (2017) Semi-supervised low rank kernel learning algorithm via extreme learning machine. *Int J Mach Learn Cyb* 8(3):1039–1052
- Liu YC (2012) *The law of blast furnace*. Metallurgical Industry Press, Beijing
- Mao WT, Wang JW, Xue ZN (2017) An elm-based model with sparse-weighting strategy for sequential data imbalance problem. *Int J Mach Learn Cybern* 8(4):1333–1345
- Peacey JG, Davenport WG (2016) *The iron blast furnace: theory and practice*. Elsevier, Amsterdam
- Radhakrishnan VR, Ram KM (2001) Mathematical model for predictive control of the bell-less top charging system of a blast furnace. *J Process Control* 11(5):565–586
- Shi L, Zhao GS, Li MX, Ma X (2016) A model for burden distribution and gas flow distribution of bell-less top blast furnace with parallel hoppers. *Appl Math Model* 40(23):10254–10273
- StatLib (1997) California housing data set. http://www.dcc.fc.up.pt/Itorgo/Regression/cal_housing
- Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim* 11(4):341–359
- Su XL, Yin YX, Zhang S (2016) Prediction model of improved multi-layer extreme learning machine for permeability index of blast furnace. *Control Theory Appl* 33(12):1674–1684
- Tang J, Deng C, Huang GB (2016) Extreme learning machine for multilayer perceptron. *IEEE Trans Neural Netw Learn Syst* 27(4):809–821
- UCI (1990) Image segmentation data set. <http://archive.ics.uci.edu/ml/datasets/Image+Segmentation>
- UCI (1991) Letter recognition data set. <http://archive.ics.uci.edu/ml/datasets/Letter+Recognition>
- UCI (1995) Abalone data set. <http://archive.ics.uci.edu/ml/datasets/Abalone>
- Wang XZ, Chen AX, Feng HM (2011) Upper integral network with extreme learning mechanism. *Neurocomputing* 74(16):2520–2525
- Wang XZ, Shao QY, Miao Q, Zhai HH (2013) Architecture selection for networks trained with extreme learning machine using localized generalization error model. *Neurocomputing* 102(1):3–9
- Wold S, Trygg J, Berglund A, Antti H (2001) Some recent developments in PLS modeling. *Chemometr Intell Lab* 58(2):131–150
- Xue XW, Yaon M, Wu ZH, Yang JH (2014) Genetic ensemble of extreme learning machine. *Neurocomputing* 129:175–184
- Yang YM, Wu QMJ (2016) Multilayer extreme learning machine with subnetwork nodes for representation learning. *IEEE Trans Cybern* 46(11):2570–2583
- Yang YM, Wu QMJ, Wang YN, Zeeshan KM, Lin XF, Yuan XF (2015) Data partition learning with multiple extreme learning machines. *IEEE Trans Cybern* 45(8):1463–1475
- Yang YM, Wu QMJ, Wang YN (2016) Autoencoder with invertible functions for dimension reduction and image reconstruction. *IEEE Trans Syst Man Cybern Syst*. <https://doi.org/10.1109/TSMC.2016.2637279>
- Zhai JH, Xu HY, Wang XZ (2012) Dynamic ensemble extreme learning machine based on sample entropy. *Soft Comput* 16(9):1493–1502
- Zhai JH, Zhang SF, Wang CX (2017) The classification of imbalanced large data sets based on mapreduce and ensemble of elm classifiers. *Int J Mach Learn Cybern* 8(3):1009–1017
- Zhang HG, Yin YX, Zhang S (2016) An improved elm algorithm for the measurement of hot metal temperature in blast furnace. *Neurocomputing* 174:232–237
- Zhu QY, Qin AK, Suganthan PN, Huang GB (2005) Evolutionary extreme learning machine. *Pattern Recognit* 38(10):1759–1763
- Zong WW, Huang GB, Chen YQ (2013) Weighted extreme learning machine for imbalance learning. *Neurocomputing* 101:229–242