Postprint: Improved Particle Swarm Optimization-Based Extreme Learning Machine Soft Sensor Modeling Method

Authors: Sheng Xiaochen, Shi Xudong, Xiong Weili

Date: 2019-05-10T00:00:00+00:00

Abstract

Industrial processes frequently exhibit significant nonlinearities, time-varying characteristics, and other complex features. Conventional extreme learning machines may not fully exploit the information contained within the data, leading to suboptimal predictive performance of the constructed soft sensor models. To enhance the generalization capability and prediction accuracy of extreme learning machines, a soft sensor modeling approach based on improved particle swarm optimization for extreme learning machines is proposed. First, the characteristics of the Gaussian function' s normal distribution are utilized to achieve adaptive updating of the inertia weight, while the learning factors are varied linearly, thereby improving the convergence speed and search performance of the particle swarm optimization algorithm. Subsequently, this algorithm is employed to optimize the penalty coefficient and kernel width of the extreme learning machine, yielding a set of optimal hyperparameters. Finally, the proposed method is applied to soft sensor modeling for the debutanizer column process. Simulation results demonstrate that the optimized extreme learning machine model exhibits significant improvement in prediction accuracy, thereby verifying that the proposed method is not only feasible but also possesses favorable prediction accuracy and generalization performance.

Full Text

Preamble

Soft Sensor Modeling for Extreme Learning Machine Based on Improved Particle Swarm Optimization

Sheng Xiaochen[1], Shi Xudong[1], Xiong Weili[1,2]
([1]School of Internet of Things Engineering, Jiangnan University; [2]Key Labora-

tory of Advanced Process Control for Light Industry (Ministry of Education), Jiangnan University, Wuxi, Jiangsu 214122, China)

Abstract: Industrial processes often exhibit significant nonlinear and time-varying characteristics. Traditional extreme learning machine (ELM) based soft sensors sometimes fail to effectively utilize data information, resulting in poor prediction performance. To enhance the generalization capability and prediction accuracy of ELM, this paper proposes a soft sensor modeling approach for ELM based on an improved particle swarm optimization algorithm. First, leveraging the characteristics of Gaussian normal distribution, the algorithm achieves adaptive updating of inertia weight while linearly varying learning factors to improve convergence speed and search performance. Then, the algorithm is employed to optimize the penalty coefficient and kernel width of ELM, yielding a set of optimal hyperparameters. Finally, the proposed method is applied to soft sensor modeling for a debutanizer column process. Simulation results demonstrate that the optimized ELM model achieves significantly improved prediction accuracy, verifying that the proposed method is not only feasible but also exhibits good prediction precision and generalization performance.

Keywords: soft sensor modeling; extreme learning machine; particle swarm optimization; adaptive weight

0 Introduction

In actual industrial production processes, many quality variables closely related to product quality are difficult to monitor and control online in real-time due to economic or technical limitations. Examples include acetic acid concentration in azeotropic distillation of acetic acid [1], oil viscosity and composition, and product concentration in penicillin fermentation processes [2]. Soft sensing technology addresses this challenge by constructing mathematical models between easily measurable auxiliary variables and difficult-to-measure dominant variables [3]. Commonly used soft sensor models such as principal component regression (PCR) [4], partial least squares (PLS) [5], artificial neural networks (ANN) [6], support vector machines (SVM) [7], and Gaussian process regression (GPR) [8] can all achieve satisfactory prediction results.

Among soft sensor modeling methods, extreme learning machine (ELM) [9] has been widely applied to industrial processes in recent years due to its advantages of high prediction accuracy, fast learning speed, and few parameters. Compared with traditional neural networks [10] and least squares support vector machines [11], the ELM model overcomes the drawback of gradient descent learning algorithms that require multiple iterations, enabling faster neural network learning while meeting required accuracy demands. However, ELM also has certain limitations, such as difficulty in precisely locating optimal hyperparameters and room for improvement in generalization performance and prediction accuracy. In particular, improper parameter selection can lead to severely inadequate net-

work learning performance and degraded prediction capability for query samples, failing to meet practical industrial requirements.

Currently, the penalty coefficient and kernel width parameters are often selected manually, a process that requires extensive experimentation, consumes considerable time, and may result in low model accuracy due to improper settings. Consequently, some researchers have applied swarm intelligence optimization algorithms to select these parameters for ELM. Wang et al. [12] proposed using grey wolf optimization (GWO) to select the penalty coefficient and kernel width of ELM, achieving higher classification accuracy. Lyu et al. [13] applied improved bacterial foraging optimization (IBFO) to ELM hyperparameter selection, successfully applying it to diagnose somatization disorder patients. However, the GWO algorithm suffers from slow convergence and susceptibility to local optima, while the IBFO algorithm requires numerous parameter settings, and its optimization performance is heavily influenced by the chemotactic step size. If the optimization algorithm parameters are improperly selected, the goal of improving ELM model generalization performance cannot be achieved.

Particle swarm optimization (PSO) [14], first proposed by Kennedy and Eberhart in 1995, features clear principles, few adjustable parameters, and fast convergence, making it widely applicable for enhancing machine learning algorithm performance. Lin et al. [15] utilized PSO for feature selection and optimal hyperparameter selection in SVM, significantly improving SVM generalization capability and classification accuracy. Tao et al. [16] proposed an improved PSO-K-means clustering algorithm that leverages the strong global search capability of PSO to compensate for the tendency of K-means to fall into local optima. Grimaldi et al. [17] applied PSO to neural network weight configuration, substantially improving network learning performance. While the standard PSO algorithm demonstrates good optimization performance for simple function optimization and parameter selection problems, it exhibits premature convergence, low precision, and poor stability when solving complex nonlinear and multimodal optimization problems, sometimes failing to obtain global optima.

This paper introduces a Gaussian descent-based inertia weight adjustment strategy combined with linearly varying learning factors to improve the PSO algorithm, enabling faster escape from local optima, improved network convergence speed, and accelerated search for the global optimum, resulting in significantly enhanced optimization performance. The improved PSO algorithm is then applied to select the penalty coefficient and kernel width of ELM, yielding an ELM network structure determined by optimal hyperparameters. This approach not only overcomes the poor convergence caused by improper parameter selection but also substantially improves the generalization capability and prediction accuracy of the ELM model. The effectiveness and precision of the proposed method are verified through simulation studies on a debutanizer column process and comparisons with different modeling approaches.

1 Extreme Learning Machine

Extreme learning machine is a learning algorithm for single-hidden layer feedforward networks (SLFNs) proposed by Huang et al. in 2004 [18]. The algorithm randomly assigns input layer weights and hidden node biases—only requiring the number of hidden nodes—and obtains output weights by solving the least squares solution of a linear system of equations to achieve optimal performance.

For a training set containing N distinct samples $\{(x_i, y_i)\}_{i=1}^{N}$, where $x_i = [x_{i1}, x_{i2}, \dots, x_{in}]^T \in \mathbb{R}^n$ represents the n-dimensional input sample and $y_i = [y_{i1}, y_{i2}, \dots, y_{im}]^T \in \mathbb{R}^m$ represents the m-dimensional output sample, the neural network output can be expressed as:

$$f(x_i) = h(x_i)\beta, \quad i = 1, 2, \dots, N$$

where $h(x_i)$ denotes the output vector of the hidden layer relative to x_i, i.e., the hidden layer output matrix, and β represents the output weights.

The training objective of ELM is to minimize both training error and output weight norm, which can be formulated as a constrained optimization problem:

$$\begin{aligned} &\min_{\beta, e_i} \frac{1}{2}\|\beta\|^2 + \frac{C}{2}\sum_{i=1}^{N}\|e_i\|^2 \\ &\text{s.t.} \quad h(x_i)\beta = y_i - e_i, \quad i = 1, 2, \dots, N \end{aligned}$$

where $e_i \in \mathbb{R}^m$ is the training error for the i-th training sample and C is the penalty coefficient.

Solving this optimization problem yields:

$$\beta = \left(\frac{I}{C} + H^T H\right)^{-1} H^T Y$$

where H is the hidden layer output matrix:

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \cdots & h_{\tilde{N}}(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_N) & \cdots & h_{\tilde{N}}(x_N) \end{bmatrix}$$

and $Y = [y_1, \dots, y_N]^T$. I is an identity matrix of dimension $\tilde{N}$. Applying Mercer' s condition defines the kernel matrix [19]:

$$\Omega_{ELM} = HH^T : \Omega_{ELM_{i,j}} = h(x_i) \cdot h(x_j) = K(x_i, x_j)$$

In this paper, the Gaussian kernel function is selected:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

where σ is the kernel width parameter that determines the radial influence range of the kernel function. In this case, the ELM output can be expressed by Equation (6):

$$f(x) = \begin{bmatrix} K(x, x_1) \\ \vdots \\ K(x, x_N) \end{bmatrix}^T \left(\frac{I}{C} + \Omega_{ELM}\right)^{-1} Y$$

2.1 Standard PSO Algorithm

In a D-dimensional space, a particle swarm with population size N can be represented as $X = \{x_1, x_2, \ldots, x_N\}$. The position and velocity of particle i can be expressed as $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ and $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$, respectively. After k iterations, the historical best positions obtained by the particle and the entire swarm can be represented as individual extremum $p_{best_i} = (p_{i1}, p_{i2}, \ldots, p_{iD})$ and global extremum $g_{best} = (g_1, g_2, \ldots, g_D)$.

During the iteration process for finding optimal values, each particle i adjusts its position and velocity through individual and global extrema:

$$v_{id}^{k+1} = v_{id}^k + c_1 r_1 (p_{id}^k - x_{id}^k) + c_2 r_2 (g_d^k - x_{id}^k)$$

$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1}$$

where c_1 and c_2 are learning factors; r_1 and r_2 are random numbers between (0, 1); $i = 1, 2, \ldots, N$; $d = 1, 2, \ldots, D$; and k is the current iteration number.

To regulate the influence of a particle' s historical velocity on its current state for better optimization results, Shi et al. [14] introduced inertia weight ω into Equation (7). Equations (7) and (9) constitute the standard particle swarm optimization (SPSO) algorithm:

$$v_{id}^{k+1} = \omega v_{id}^k + c_1 r_1 (p_{id}^k - x_{id}^k) + c_2 r_2 (g_d^k - x_{id}^k)$$

The inertia weight ω primarily balances local and global search capabilities, typically set to $\omega = 0.8$. The learning factors c_1 and c_2 regulate the influence of individual and global extrema on particles, usually set to $c_1 = c_2 = 2$.

2.2 Improved PSO

While the standard PSO algorithm demonstrates good optimization effects in searching for optimal solutions, it also has certain shortcomings, such as premature convergence to local optima [20], which affects the search for global optimal solutions. To address this issue, numerous improved PSO algorithms have been proposed by domestic and international scholars through research on inertia weight and learning factors.

Inertia weight is a crucial parameter in PSO. A larger inertia weight enhances global search capability, while a smaller one facilitates local search. Shi [14] proposed a linearly decreasing weight PSO algorithm (LDWPSO) where inertia weight decreases linearly with iterations. Amoshahy et al. [21] proposed a flexible exponential nonlinear weight adjustment strategy. Liang et al. [22] introduced a new dynamic nonlinear variable inertia weight adaptive mutation PSO algorithm. Reference [23] describes a new inertia weight setting method that improves upon linearly decreasing inertia weight, resulting in a nonlinear decreasing weight PSO algorithm (denoted as w1PSO).

Compared with SPSO, LDWPSO exhibits better global search capability in the early stage and better local search capability in the later stage. However, this algorithm maintains a fixed ratio between global and local search during iterations, often yielding unsatisfactory results for complex nonlinear problems. The w1PSO algorithm shows certain improvements in optimization performance, stability, and precision. To further enhance PSO optimization performance—ensuring fast convergence speed in the early search stage while avoiding premature convergence to local optima in the later stage—this paper draws inspiration from the characteristics of Gaussian normal distribution to introduce a Gaussian descent-based inertia weight adjustment strategy combined with linearly varying learning factors c_1 and c_2 [24], thereby obtaining an improved PSO algorithm (denoted as IPSO) with the following settings:

$$\omega(k) = \omega_{\max} \exp\left(-\left(\frac{k}{d \times k_{\max}}\right)^2\right)$$

$$\begin{cases} c_1(k) = c_{1\,\min} + (c_{1\,\max} - c_{1\,\min}) \times \frac{k}{k_{\max}} \\ c_2(k) = c_{2\,\min} + (c_{2\,\max} - c_{2\,\min}) \times \frac{k}{k_{\max}} \end{cases}$$

where $c_{1\,\min}, c_{1\,\max} \in [c_{1\,\min}, c_{1\,\max}]$; $c_{2\,\min}, c_{2\,\max} \in [c_{2\,\min}, c_{2\,\max}]$; and d is a constant (set to $d = 0.8$ in this paper).

The IPSO algorithm employs dynamically decreasing inertia weight while simultaneously linearly varying learning factors, balancing global exploration and local exploitation capabilities, ensuring population diversity, and achieving better optimization results while improving convergence speed. To construct a high-precision soft sensor model, we define a "particle" as a potential solution in the

optimization solution space (C, σ), and use the root mean square error (RMSE) shown in Equation (13) as the fitness function for the particle swarm algorithm [26], with the calculated fitness value denoted as $fitness(i)$:

$$RMSE = \sqrt{\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} (y_{\text{test}} - y_{\text{pred}})^2}$$

where N_{test} is the number of test samples; y_{test} is the true value of test samples; and y_{pred} is the model prediction output.

[Figure 1: see original paper] shows the inertia weight variation curves for three different PSO algorithms.

The inertia weight ω of the IPSO algorithm exhibits a Gaussian decreasing trend as iteration number increases. Compared with the other two algorithms, IPSO has a larger inertia weight in the early iteration stage, resulting in faster convergence speed and stronger global search capability for extensive search range. In the later stage when the approximate optimal region is located, a smaller inertia weight is set to emphasize local exploitation capability for fine-grained search. Meanwhile, the learning factor c_1 of IPSO decreases with iteration number, while c_2 increases. This arrangement ensures that particle flight states are primarily influenced by individual optimal positions in the early search stage, enhancing development capability across the entire search space, while in the later stage, states are mainly affected by neighboring particles' optimal positions, facilitating local exploitation.

To test the performance of the IPSO algorithm in solving complex problems, the Rastrigin function from standard test functions is selected for simulation comparison among four algorithms: SPSO, LDWPSO, w1PSO, and IPSO. This function features no correlation between variables and numerous local minima with sinusoidal fluctuations, representing a typical complex multimodal function [25]. The function expression is given in Equation (12):

$$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$$

We now seek the minimum value of the Rastrigin function in two-dimensional space $[-5, 5] \times [-5, 5]$. Algorithm parameters are set as follows: population size is 20, dimension is 2, $c_{1\,\max} = c_{2\,\max} = 2.5$, $c_{1\,\min} = c_{2\,\min} = 0.5$, and $d = 0.8$. Maximum iterations are set to 200, with each algorithm tested 50 times. The test results are shown in .

As shown in , compared with the other three algorithms, the proposed IPSO algorithm can converge to the global optimum with higher probability after multiple iterations, demonstrating better optimization performance. This algorithm can therefore be used to optimize the penalty coefficient and kernel

width of ELM to obtain a training model with higher prediction accuracy and improved regression performance.

---

3 Extreme Learning Machine Based on Improved Particle Swarm Optimization

To test the influence of ELM's penalty coefficient and kernel width on its learning performance, regression analysis is performed on the function $f(x) = \sin(x)/x$. In this experiment, 200 data sets $\{(x_i, f(x_i) + \zeta_i)\}$ are randomly generated, where 100 sets serve as training samples and 100 as test samples, with ζ_i being noise following a normal distribution $\zeta_i \sim N(0, 0.05)$. RMSE and tracking precision (TP) [27] are selected as performance metrics to evaluate ELM fitting effectiveness. TP is defined in Equation (14):

$$TP = 1 - \frac{\sum_{i=1}^{N_{\text{test}}} (y_{\text{test}} - y_{\text{pred}})^2}{\sum_{i=1}^{N_{\text{test}}} (y_{\text{test}} - \bar{y}_{\text{test}})^2}$$

The generalization performance of ELM largely depends on the reasonable selection of its parameters, including the penalty coefficient and kernel width. Therefore, this paper employs the improved particle swarm algorithm to optimize ELM hyperparameters to obtain a more accurate ELM prediction model. The training steps for IPSO-optimized ELM parameters are as follows:

a) Initialize PSO parameters, including population size N, space dimension D, learning factors c_1 and c_2, inertia weight ω, maximum iterations $k_{\max}$, velocity limits $[v_{\min}, v_{\max}]$, and position limits $[d_{\min}, d_{\max}]$. Set the optimal fitness value $fitness(best) = fitness(1)$ and randomly generate a set of data x_1 as the initial solution space.

b) Randomly initialize input weights ω_i and hidden layer biases b_i to construct the ELM neural network structure. Use each particle x_i as the corresponding optimized ELM model parameters (C, σ) to train on samples and calculate fitness values. Particles adjust their positions and velocities by comparing fitness values of individual and global extrema.

c) Update particle positions and velocities using the particle swarm algorithm.

d) Check if maximum iterations are reached. If so, terminate the algorithm and output results; otherwise, continue iterating to obtain a new particle swarm.

e) Use the final optimized particle x_{best} as the penalty coefficient and kernel width of ELM to establish the optimized ELM model.

The algorithm flowchart for IPSO-optimized ELM parameters is shown in [Figure 2: see original paper].

---

4 Simulation Research

4.1 ELM Example Simulation

Considering result fluctuations, ten experiments are conducted and averaged. The RMSE and TP of ELM are tested separately, with results shown in and .

**** Effect of different penalty coefficient and kernel parameter on RMSE
**** Effect of different penalty coefficient and kernel parameter on TP

As shown in and , ELM generalization performance is significantly affected by the selected penalty coefficient and kernel width. In this test, ELM demonstrates good fitting performance for the function when $C = 200$ and $\sigma = 0.5$. With appropriate parameter selection, ELM exhibits strong learning capability, achieving high prediction accuracy for function values even in the presence of noise. Conversely, improper parameter selection leads to poor ELM generalization performance.

4.2 Debutanizer Column Process Simulation

The debutanizer column process is a crucial component in desulfurization and naphtha separation units in petroleum refining [28]. The bottom butane concentration significantly impacts petroleum refining but is difficult to measure directly with instruments, necessitating soft sensor modeling for this industrial process. Seven measurable variables are selected as auxiliary variables: top temperature (x_1), top pressure (x_2), reflux flow rate (x_3), flow rate to the next process (x_4), 6th tray temperature (x_5), top temperature 1 (x_6), and top temperature 2 (x_7). The dominant variable is the bottom butane concentration.

[Figure 3: see original paper] shows a schematic diagram of the debutanizer column process with auxiliary variable measurement points annotated.

A total of 2,000 sets of debutanizer column process data are collected, with 1,000 sets as the training sample set and 1,000 as the test sample set. Three different soft sensor models are established for simulation comparison: the ELM model, SPSO-optimized ELM model (denoted as PSOELM), and IPSO-optimized ELM model (denoted as IELM). Population size is set to 100, maximum iterations to 100, velocity limit $v \in [-0.5, 0.5]$, and other parameters remain the same as in Section 2.2. Performance metrics include RMSE [Equation (13)], TP [Equation (14)], and Correlation Coefficient (COR) [Equation (15)]:

$$COR = \frac{\mathrm{Cov}(y_{\mathrm{test}}, y_{\mathrm{pred}})}{\sqrt{\mathrm{var}(y_{\mathrm{test}}) \cdot \mathrm{var}(y_{\mathrm{pred}})}}$$

**** Prediction results comparison of three modeling methods

The results show that the IELM model optimized by IPSO achieves significantly higher prediction accuracy for bottom butane concentration than the traditional ELM model. Compared with the PSOELM model, the IELM model further reduces prediction error for butane concentration and demonstrates improved generalization performance, indicating that the proposed method can effectively enhance model learning capability and prediction accuracy for butane concentration prediction.

To more intuitively demonstrate the learning effectiveness of the IELM algorithm, [Figure 4: see original paper] shows the prediction results of the three modeling methods for butane concentration. The IELM model' s predicted value curve follows the true value curve more closely, with prediction errors fluctuating within a smaller range around zero, indicating that the IELM model' s butane concentration predictions are closer to true values with smaller errors. Compared with traditional ELM simulation results, the proposed method exhibits stronger fitting capability for butane concentration and superior model generalization with higher prediction accuracy.

[Figure 5: see original paper] presents scatter plots of prediction results from the three modeling methods. The IELM model' s scatter points are closer to the black diagonal line, while those from traditional ELM and PSOELM models are relatively more dispersed, demonstrating that the model established by the proposed method yields butane concentration predictions closer to true values with significantly improved prediction accuracy.

---

5 Conclusion

This paper employs an improved particle swarm optimization algorithm to simultaneously optimize the penalty coefficient and kernel width of extreme learning machine, obtaining an optimized ELM soft sensor model with enhanced performance. The approach improves both the generalization capability of the model and the prediction accuracy for quality variables in nonlinear industrial processes. Application simulation on a debutanizer column process verifies the effectiveness and precision of the proposed method, providing valuable reference for soft sensor modeling in complex industrial processes.

References

[1] Pan Hongfang. Research on extreme learning machine and its application in acetic acid soft sensor modeling [D]. East China University of Science and Technology, 2014.

[2] Xiong Weili, Yao Le, Xu Baoguo. Multi-stage fusion modeling of penicillin fermentation process based on EM algorithm [J]. CIESC Journal, 2014, 65(12): 4935-4941.

[3] Cao Pengfei, Luo Xionglin. Modeling of soft sensor for chemical process [J]. CIESC Journal, 2013, 64(3): 788-800.

[4] Niu Zhiguang, Wang Chong, Zhang Ying, et al. Leakage rate model of urban water supply networks using principal component regression analysis [J]. Journal of Tianjin University, 2018, 24(2): 172-181.

[5] Zheng Junhua, Song Zhihuan. Semisupervised learning for probabilistic partial least squares regression model and soft sensor application [J]. Journal of Process Control, 2018, 64: 123-131.

[6] Willis M J, Montague G A, Massimo C D, et al. Artificial neural networks in process estimation and control [J]. Automatica, 1992, 28(6): 1181-1187.

[7] Cortes C. Support vector network [J]. Machine Learning, 1995, 20(3): 273-297.

[8] Xiong Weili, Shi Xudong. Soft sensor modeling with a selective updating strategy for Gaussian process regression based on probabilistic principle component analysis [J]. Journal of the Franklin Institute, 2018, 355(12): 5336-5349.

[9] Huang Guangbin, Zhu Qinyu, Siew C K. Extreme learning machine: a new learning scheme of feedforward neural networks [C]// Proc of IEEE International Joint Conference on Neural Networks. 2005: 985-990.

[10] Sun Kai, Liu Jialin, Kang Jialin, et al. Development of a variable selection method for soft sensor using artificial neural network and nonnegative garrote [J]. Journal of Process Control, 2014, 24(7): 1041-1048.

[11] Zhao Yantao, Shan Zeyu, Chang Yuejin, et al. Soft sensor modeling for cement fineness based on least squares support vector machine and mutual information [J]. Chinese Journal of Scientific Instrument, 2017(2): 487-496.

[12] Wang Mingjing, Chen Huiling, Li Huaizhong, et al. Grey wolf optimization evolving kernel extreme learning machine: application to bankruptcy prediction [J]. Engineering Applications of Artificial Intelligence, 2017, 63: 54-68.

[13] Lyu Xinen, Chen Huiling, Zhang Qian, et al. An improved bacterial-foraging, optimization-based machine learning framework for predicting the severity of somatization disorder [J]. Algorithms, 2018, 11(2): 17.

[14] Shi Yuhui, Eberhart R. Modified particle swarm optimizer [C]// Proc. of IEEE ICEC Conference. 1999: 69-73.

[15] Lin S W, Ying K C, Chen S C, et al. Particle swarm optimization for parameter determination and feature selection of support vector machines [J]. Expert Systems with Applications, 2008, 35(4): 1817-1824.

[16] Tao Xinmin, Xu Jing, Yang Libiao, et al. Improved cluster algorithm based on K-means and particle swarm optimization [J]. Journal of Electronics & Information Technology, 2010, 32(1): 92-97.

[17] Grimaldi E A, Grimaccia F, Mussetta M, et al. PSO as an effective learning algorithm for neural network applications [C]// Proc of International Conference on Computational Electromagnetics and ITS Applications. 2004: 557-560.

[18] Huang Guangbin, Song Shiji, Gupta J N, et al. Semi-supervised and unsupervised extreme learning machines [J]. IEEE Trans on Cybernetics, 2014, 44(12): 2405-2417.

[19] Ma Chao, Xu Shouxiang, Liu Yuandong. Optimized kernel extreme learning machine based on ensemble method for heart diseases [J]. Application Research of Computers, 2017, 34(6): 1671-1676.

[20] Wang Cunrui, Duan Xiaodong, Liu Xiangdong, et al. A modified basic particle swarm optimization algorithm [J]. Computer Engineering, 2004, 30(21): 35-37.

[21] Amoshahy M J, Shamsi M, Sedaaghi M H. A novel flexible inertia weight particle swarm optimization algorithm [J]. Plos One, 2016, 11(8): e0161558.

[22] Liang Hongtao, Kang Fengju. Adaptive mutation particle swarm algorithm with dynamic nonlinear changed inertia weight [J]. Optik-International Journal for Light and Electron Optics, 2016, 127(19): 8036-8042.

[23] Shi Feng, Wang Hui, Yu Lei, et al. 30 case analysis of MATLAB intelligent algorithm [M]. Beijing: Beihang University Press, 2011.

[24] Ratnaweera A, Halgamuge S K, Watson H C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients [J]. IEEE Trans on Evolutionary Computation, 2004, 8(3): 240-255.

[25] Lu Shuying. Some improvements and applications of particle swarm optimization algorithm [D]. Xuzhou: China University of Mining and Technology, 2014.

[26] Wang Jie, Bi Haoyang. An extreme learning machine based on particle swarm optimization [J]. Journal of Zhengzhou University: Engineering Science, 2013, 45(1): 100-104.

[27] Zhang Wei, Li Yanjun, Xiong Weili, et al. Adaptive soft sensor for online prediction based on enhanced moving window GPR [C]// Proc of IEEE International Conference on Control, Automation and Information Sciences. 2015: 291-296.

[28] Fortuna L, Graziani S, Xibilia M G. Soft sensors for product quality monitoring in debutanizer distillation columns [J]. Control Engineering Practice, 2005, 13(4): 499-508.

Note: Figure translations are in progress. See original paper for figures.