Hindawi
Complexity
Volume 2019, Article ID 5353296, 12 pages
https://doi.org/10.1155/2019/5353296

Research Article

An Incremental Learning Ensemble Strategy for Industrial Process Soft Sensors

Huixin Tian,[1,2,3] **Minwei Shuai**,[1] **Kun Li**,[4] **and Xiao Peng**[1]

[1]*School of Electrical Engineering & Automation and Key Laboratory of Advanced Electrical Engineering and Energy Technology, Tianjin Polytechnic University, Tianjin, China*
[2]*State Key Laboratory of Process Automation in Mining & Metallurgy, Beijing 100160, China*
[3]*Beijing Key Laboratory of Process Automation in Mining & Metallurgy Research Fund Project, Beijing 100160, China*
[4]*School of Economics and Management, Tianjin Polytechnic University, Tianjin, China*

Correspondence should be addressed to Kun Li; lk_neu@163.com

Received 13 November 2018; Accepted 25 March 2019; Published 2 May 2019

Guest Editor: Marisol B. Correia

With the continuous improvement of automation in industrial production, industrial process data tends to arrive continuously in many cases. The ability to handle large amounts of data incrementally and efficiently is indispensable for modern machine learning (ML) algorithms. According to the characteristics of industrial production process, we address an ILES (incremental learning ensemble strategy) that incorporates incremental learning to extract information efficiently from constantly incoming data. The ILES aggregates multiple sublearning machines by different weights for better accuracy. When new data set arrives, a new submachine will be trained and aggregated into ensemble soft sensor model according to its weight. The other submachines' weights will be updated at the same time. Then a new updated soft sensor ensemble model can be obtained. The weight updating rules are designed by considering the prediction accuracy of submachines with new arrived data. So the update can fit the data change and obtain new information efficiently. The sizing percentage soft sensor model is established to learn the information from the production data in the sizing of industrial processes and to test the performance of ILES, where the ELM (Extreme Learning Machine) is selected as the sublearning machine. The comparison is done among new method, single ELM, AdaBoost.R ELM, and OS-ELM, and the test of the extensions is done with three test functions. The results of the experiments demonstrate that the soft sensor model based on the ILES has the best accuracy and ability of online updating.

1. Introduction

During industrial processes, plants are usually heavily instrumented with a large number of sensors for process monitoring and control. However, there are still many process parameters that cannot be measured accurately because of high temperature, high pressure, complex physical and chemical reactions and large delays, etc. Soft sensor technology provides an effective way to solve these problems. The original and still the most dominant application area of soft sensors is the prediction of process variables, which can be determined either at low sampling rates or through off-line analysis only. Because these variables are often related to the process product quality, they are very important for process control and quality management. Additionally, the soft sensor application field usually refers to online prediction during the process of production.

Currently, with the continuous improvement of automation in industrial production, large amounts of industrial process data can be measured, collected, and stored automatically. It can provide strong support for the establishment of data-driven soft sensor models. Meanwhile, with the rapid development and wide application of big data technology, soft sensor technology has already been used widely and plays an essential role in the development of industrial process detection and control systems in industrial production. Artificial intelligence and machine learning, as the important core technologies, are getting increasingly more attention. The traditional machine learning algorithm generally refers to the single learning machine model that

is trained by training sets, and then the unknown samples will be predicted based on this model. However, the single learning machine models have to face defects that cannot be overcome by themselves, such as unsatisfactory accuracy and generalization performance, especially for complex industrial processes. Specifically, in the supervised machine learning approach, the model's hypothesis is produced to predict the new incoming instances by using predefined label instances. When the multiple hypotheses that support the final decision are aggregated together, it is called ensemble learning. Compared with the single learning machine model, the ensemble learning technique is beneficial for improving quality and accuracy. Therefore, increasingly more researchers are studying how to improve the speed, accuracy and generalization performance of integrated algorithms instead of developing strong learning machines.

Ensemble algorithms were originally developed for solving binary classification problems [1], and then AdaBoost.M1 and AdaBoost.M2 were proposed by Freund and Schapire [2] for solving multiclassification problems. Thus far, there are many different versions of boosting algorithms for solving classification problems [3–7], such as boosting by filtering and boosting by subsampling. However, for regression problems, it is not possible to predict the output exactly as that in classification. To solve regression problems using ensemble techniques, Freund and Schapire [2] extended AdaBoost.M2 to AdaBoost.R, which projects the regression sample into a classification data set. Drucker [8] proposed the AdaBoost.R2 algorithm, which is an ad hoc modification of AdaBoost.R. Avnimelech and Intrator [9] extended the boosting algorithm for regression problems by introducing the notion of weak and strong learning as well as an appropriate equivalence theorem between the two. Feely [10] proposed BEM (big error margin) boosting method, which is quite similar to the AdaBoost.R2. In BEM, the prediction error is compared with the preset threshold value, BEM, and the corresponding example is classified as either well or poorly predicted. Shrestha and Solomatine [11] proposed an AdaBoost.RT, with the idea of filtering out the examples with relative estimation errors that are higher than the preset threshold value. However, the value of the threshold is difficult to set without experience. For solving this problem, Tian and Mao [12] present a modified AdaBoost.RT algorithm that can adaptively modify the threshold value according to the change in RMSE. Although ensemble algorithms have the ability to enhance the accuracy of soft sensors, they are still at a loss for the further information in the new coming data.

In recent years, with the rapid growth of data size, a fresh research perspective has arisen to face the large amount of unknown important information contained in incoming data streams in various fields. How can we obtain methods that can quickly and efficiently extract information from constantly incoming data? The batch learning is meaningless, and the algorithm needs to have the capability of real-time processing because of the demand of real-time updated data in industrial processes. Incremental learning idea is helpful to solve the above problem. If a learning machine has this kind of idea, it can learn new knowledge from new data sets and can retain old knowledge without accessing the old data set. Thus, the incremental learning strategy can profoundly increase the processing speed of new data while also saving the computer's storage space. The ensemble learning methods can be improved by combining the characteristics of the ensemble strategy and incremental learning. It is an effective and suitable way to solve the problem of stream data mining [13–15]. Learn++ is a representative ensemble algorithm with the ability of incremental learning. This algorithm is designed by Polikar et al. based on AdaBoost and supervised learning [16]. In Learn ++, the new data is also assigned sample weights, which update according to the results of classification at each iteration. Then, a newly trained weak classifier is added to the ensemble classifier. Based on the Learn ++ CDS algorithm, G Ditzler and Polikar proposed Learn ++ NIE [17] to improve the effect of classification on a few categories. Most of research of incremental learning and ensemble algorithm focus on the classification field, while the research in the field of regression is still less. Meanwhile the limitation of ensemble approaches is that they cannot address the essential problem of incremental learning well, what the essential problem is accumulating experience over time then adapting and using it to facilitate future learning process [18–20].

For the process of industrial production, increasingly more intelligent methods are used in soft sensors with the fast development of artificial intelligence. However, the practical applications of soft sensors in industrial production are not good. The common shortages of soft sensors are unsatisfactory, unstable prediction accuracy, and poor online updating abilities. It is difficult to meet a variety of changes in the process of industrial production. Therefore, in this paper, we mainly focus on how to add the incremental learning capability to the ensemble soft sensor modeling method and hopefully provide useful suggestions to enhance both the generation and online application abilities of soft sensors for industrial process. Aiming at the demands of soft sensors for industrial applications, a new detection strategy is proposed with multiple learning machines ensembles to improve the accuracy of the soft sensors based on intelligent algorithms. Additionally, in practical production applications, acquisition of information in new production data is expensive and time consuming. Consequently, it is necessary to update the soft sensor in an incremental fashion to accommodate new data without compromising the performance on old data. Practically, in most traditional intelligent prediction models for industrial process, the updates are often neglected. Some models use traditional updating methods that retrain the models by using all production data or using the updated data and forgo the old data. This kind of methods is not good enough because some good performances have to be lost to learn new information [21]. Against this background, we present a new incremental learning ensemble strategy with a better incremental learning ability to establish the soft sensor model, which can learn additional information from new data and preserve previously acquired knowledge. The update does not require the original data that was used to train the existing old model.

In the rest of this paper, we first describe the details of the incremental learning ensemble strategy (ILES), which

involves the strategy of updating the weights, the ensemble strategy, and the strategy of incremental learning for real-time updating. Then, we design experiments to test the performance of the ILES for industrial process soft sensors. The sizing percentage of the soft sensor model is built by the ILES in the sizing production process. The parameters of the ILES are discussed. We also compare the performance of the ILES to those of other methods. To verify the universal use of the new algorithm, three test functions are used to test the improvement on the predictive performance of the ILES. Finally, we summarize our conclusions and highlight future research directions.

2. The Incremental Learning Ensemble Strategy

The industrial process needs soft sensors with good accuracy and online updating performance. Here, we focus on incorporating the incremental learning idea into the ensemble regression strategy to achieve better performance of soft sensors. A new ensemble strategy called ILES for industrial process soft sensors that combines the ensemble strategy with the incremental learning idea is proposed. The ILES has the ability to enhance soft sensors' accuracy by aggregating the multiple sublearning machines according to their training errors and prediction errors. Additionally, during the iteration process, incremental learning is added to obtain the information from new data by updating the weights. It is beneficial to enhance the real-time updating ability of industrial process online soft sensors. The details of the ILES are described as shown in Algorithm 1.

2.1. Strategy of Updating the Weight. In each iteration of k ($k = 1, 2, \ldots, K$), the initial $D_1(i) = 1/m(k)$ is distributed to each sample (x_i, y_i) with the same values. This means that the samples have the same chance to be included in training dataset or tested dataset at the beginning. In the subsequent iterations, the weight will be calculated as $D_t(i) = w_t / \sum_{i=1}^{m} w_t(i)$ for every sublearning machine (in each iteration of t). In contrast to the traditional AdaBoost.R, here, the testing subdataset is added to test the learning performance in each iteration. It is useful to ensure the generalization performance of the ensemble soft sensors. Then, the distribution will be changed according to the training and testing errors at the end of each iteration. Here, the training subdataset TR_t and the testing subdataset TE_t will be randomly chosen according to D_t (for example, by using the roulette method). The sublearning machine is trained by TR_t, and a hypothesized soft sensor $f_t : x \longrightarrow y$ will be obtained. Then, the training error and testing error of f_t can be calculated as follows:

$$ARE_t(i) = \left| \frac{f(x_i) - y_i}{y_i} \right| \tag{1}$$

The error rate of f_t on $S_k = TR_t + TE_t$ is defined as follows:

$$\varepsilon_t = \sum_{ARE_t(i) > \delta} D_t(i) \tag{2}$$

If $\varepsilon_t > \varphi$, the submodel is regarded as an unqualified and suspicious model. The hypothesis f_t is given up. Otherwise, the power coefficient is calculated as $\beta_t = \varepsilon_t^n$ (e.g., linear, square, or cubic). Here, φ ($0 < \varphi < 1$) is the coefficient of determination. After t ($t = 1, 2, \ldots, T_k$) iterations, the composite hypothesis F_k can be obtained according to the t hypothesized soft sensors (sublearning machines) $f_1, f_2, \ldots, f_t$. The training subdataset error, the testing subdataset error, and the error rate of F_k are calculated similarly to those of f_t. In the same way, if $E_k > \varphi$, the hypothesis F_k is given up. At the end of the iterations, according to the error rate E_t, the weight is updated as follows:

$$w_{t+1} = w_t \times \begin{cases} B_k & ARE_k(i) < \delta \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

where $B_k = E_k/(1 - E_k)$. In the next iteration, the TR_t and TE_t will be chosen again according to the new distribution, which is calculated by the new weight w_{t+1}. During the above process of iterations, the updating of the weights depends on the training and testing performance of the sublearning machines with different data. Therefore, the data with large errors will have a larger distribution for the difficult learning. It means that the "difficult" data will have more chances to be trained until the information in the data is obtained. Conversely, the sublearning machines or hypothesized soft sensors are reserved selectively based on their performance. Therefore, the final hypothesized soft sensors are well qualified to aggregate the composite hypothesis. This strategy is very effective for improving the accuracy of ensemble soft sensors.

2.2. Strategy of Ensemble with Incremental Learning. Aiming at the needs of real-time updates, the incremental learning strategy is integrated into the ensemble process. First, the subdatasets $S_k = [(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)]$, $k = 1, 2, \ldots, K$ are selected randomly from the dataset. In each iteration $k = 1, 2, \ldots, K$, the sublearning machines are trained and tested. Therefore, for each subdataset, when the inner loop ($t = 1, 2, \ldots, T_k$) is finished, the T_k hypothesized soft sensors $f_1, f_2, \ldots, f_{T_k}$ are generated. An ensemble soft sensor is obtained based on the combined outputs of the individual hypotheses, which constitute the composite hypothesis F_k.

$$F_k(x) = \frac{\sum_t \left(\lg\left(1/\beta_t\right)\right) f_t(x)}{\sum_t \left(\lg\left(1/\beta_t\right)\right)} \tag{4}$$

Here, the better hypotheses will be aggregated with larger chances. Therefore, the best performance of ensemble soft sensors is ensured based on these sublearning machines. Then, the training subdataset error and testing subdataset error of F_k can be calculated similarly to the error of f_t.

$$ARE_t(i) = \left| \frac{F_t(x_i) - y_i}{y_i} \right| \tag{5}$$

The error rate of F_k on $S_k = TR_t + TE_t$ is defined as follows:

$$E_k = \sum_{ARE_k(i) > \delta} D_t(i) \tag{6}$$

Input

(i) S_k sub datasets are drawn from original data set. Here, $k = 1, 2, \ldots, K$, $S_k = [(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)]$.
(ii) The number of sub learning machines is T_k.
(iii) The coefficients of determination are δ and φ.

For $k = 1, 2, \ldots, K$

Initialize $D_1(i) = 1/m(k)$, $\forall i = 1, 2, \ldots, K$. Here, $m(k)$, is the amount of data.

For $t = 1, 2, \ldots, T_k$

(1) Calculate $D_t(i) = w_t / \sum_{i=1}^{m} w_t(i)$. D_t is a distribution, and $w_t(i)$ is the weight of (x_i, y_i).
(2) Randomly choose the training sub dataset TR_t and the testing sub dataset TE_t according to D_t.
(3) The sub learning machine is trained by TR_t to obtain a soft sensor model $f_t : x \longrightarrow y$.
(4) Calculate the error of f_t using TR_t and TE_t:

$$ARE_t(i) = \left| \frac{f(x_i) - y_i}{y_i} \right|.$$

(5) Calculate the error rate $\varepsilon_t = \sum_{ARE_t(i) > \delta} D_t(i)$. If $\varepsilon_t > \varphi$, give up f_t, and return to step (2).
(6) Calculate $\beta_t = \varepsilon_t^n$, where $n = 1, 2$ or 3. Obtain the ensemble soft sensor model according to β_t:

$$F_k(x) = \frac{\sum_t (\lg(1/\beta_t)) f_t(x)}{\sum_t (\lg(1/\beta_t))}$$

(7) Calculate the error of F_k using S_k : $E_k = \sum_{ARE_k(i) > \delta} D_t(i)$. If $E_k > \varphi$, give up f_t, and return to step (2).
(8) Calculate $B_k = E_k/(1 - E_k)$ to update the weights:

$$w_{t+1} = w_t \times \begin{cases} B_k & ARE_k(i) < \delta \\ 1 & \text{otherwise} \end{cases}.$$

Output: Obtain the ensemble soft sensor model according to B_k:

$$F_{fin}(x) = \frac{\sum_k (\lg(1/B_k)) F_k(x)}{\sum_k (\lg(1/B_k))}$$

Algorithm 1: Incremental learning ensemble strategy.

After K hypotheses are generated for each subdataset, the final hypothesis F_{fin} is obtained by the weighted majority voting of all the composite hypotheses.

$$F_{fin}(x) = \frac{\sum_k (\lg(1/B_k)) F_k(x)}{\sum_k (\lg(1/B_k))} \tag{7}$$

When new data come constantly during the industrial process, new subdatasets will be generated (they will be the $K + 1$, $K + 2 \ldots$). Based on a new subdataset, a new hypothesized soft sensor can be trained by a new iteration. The new information in the new data will be obtained and added to the final ensemble soft sensor according to (7). As the added incremental learning strategy, the ensemble soft sensor is updated based on the old hypothesis. Therefore, the information in the old data is also retained, and the increment of information from new data is achieved.

Overall, in the above ILES, the ensemble strategy is efficient to improve the prediction accuracy using the changed distribution. Therefore, the ILES will give more attention to the "difficult" data with big errors in every iteration that are attributable to the new distribution. Due to the harder learning for the "difficult" data, more information can be obtained. Therefore, the soft sensor model is built more completely, and the accuracy of perdition is improved. Moreover, the data that is used to train the sublearning machines is divided into a training subdataset and a testing subdataset. The testing error will be used in the following steps: the weight update and the composite hypothesis ensemble. Therefore, the generalization of the soft sensor model based on the ILES can be improved efficiently, especially compared with traditional algorithms. Additionally, when the new data is added, the new ILES with incremental learning ability can learn the new data in real-time and does not give up the old information from the old data. The ILES can save the information of old sublearning machines that have been trained, but it does not need to save the original data. In other words, only a small amount of new production data being saved is enough. This strategy is efficient to save space. Furthermore, the new ILES also may save time compared with the traditional updating method. This strategy is attributed to the conservation of the old B_k and the sublearning machines in composite hypotheses (7).

3. Experiments

In this chapter, the proposed ILES is tested in the sizing production process for predicting the sizing percentage. First, the influence of each parameter on the performance of the proposed algorithm is discussed. Meanwhile, the real industrial process data is used to establish the soft sensor model to verify the incremental learning performance of the algorithm. Finally, to prove its generalization performance, three test functions are used to verify the improvement of the prediction performance. The methods are implemented using MATLAB language and all the experiments are performed on a PC with the Intel Core 7500U CPU (2.70GHZ for each single core) and the Windows 10 operation system.

3.1. Sizing Production Process and Sizing Percentage. The double-dip and double pressure sizing processes are widely used in textile mills, as shown in Figure 1. The sizing percentage plays an important role during the process of sizing for good sizing quality. In addition, the sizing agent control of warp sizing is essential for improving both productivity and product quality. The sizing percentage online detection is a key factor for successful sizing control during the sizing process. The traditional sizing detection methods, which are instruments measurement and indirect calculation, have expensive prices or unsatisfactory accuracy. Soft sensors provide an effective way to predict the sizing percentage and to overcome the above shortages. According to the mechanism analysis of the sizing process, the influencing factors on the sizing percentage are slurry concentration, slurry viscosity, slurry temperature, the pressure of the first Grouting roller, the pressure of the second Grouting roller, the position of immersion roller, the speed of the sizing machine, the cover coefficient of the yarn, the yarn tension, and the drying temperature [22]. In the following soft sensor modeling process, the inputs of soft sensors are the nine influencing factors, and the output is the sizing percentage.

3.2. Experiments for the Parameters of the ILES. Here, we select ELM as the sublearning machine of the ILES, due to its good performance, such as fast learning speed and simple parameter choices [22, 23]; the appendix reviews the process of ELM. Then, experiments with different parameters of the ILES are done to research the performance trend of the ILES when the parameters change.

First, the experiments to assess the ILES algorithm's performance are done with different T_ks. Here, the T_k increases from 1 to 15. Figure 2 shows the results of the training errors and the testing errors with different T_ks. Along with the increasing T_k, the training and testing errors decrease. When T_k increases to 7, the testing error is the smallest. However, when T_k increases to more than 9, the testing error becomes larger again. Furthermore, the training errors only slightly decrease. Therefore, we can draw the conclusion when the parameter T_k is 7 that the performance of ILES is the best. The comparison is also done between AdaBoost.R and the ILES regarding the testing errors with different numbers of ELMs in Figure 3. Although the RMSE means of AdaBoost.R and the ILES are different, their performance trends are similar with the increasing number of ELMs. Here, the RMSE is described as

$$\text{RMSE} = \left(\frac{1}{n} \sum_{i=1}^{n} e^2 (i) \right)^{1/2} \tag{8}$$

Second, we discuss the impact of parameter (δ and φ) changes on the ILES performance. The experiments demonstrate that when δ is too small, the performance of ELM is difficult to achieve the preset goal, and the iteration is difficult to stop. δ is also not larger than 80 percent of the average of the relative errors of ELMs; otherwise the F_k can not be obtained. Furthermore, the value of φ determines the number of "better samples". Here the "better samples" refer to the samples that can reach the expected precision standard of predicted results by submachines. If φ is too small, the ELM soft sensor model (f_t) will not be sufficiently obtained. If φ is too large, the "bad" ELM model (f_t) will be aggregated into the final composite hypothesized $F_{fin}(x)$. Then, the accuracy of the ILES cannot be improved. The relationships among δ, φ, and RMSE are shown in Table 1. When $\delta = 0.06$ and $\varphi = 0.15$, the model has the best performance (the RMSE is 0.3084).

Table 1: The RMSE of the ILES with different parameters Δ, Φ ($T_k = 7$).

δ	φ				
	0.05	0.1	0.15	0.2	0.25
0.02	—	—	—	0.4559	0.4712
0.03	—	—	0.4505	0.4637	0.4614
0.04	—	0.3484	0.3829	0.3888	0.4125
0.05	0.3947	0.3620	0.3765	0.3898	0.3812
0.06	0.4363	0.3734	**0.3084**	0.4036	0.4121
0.07	0.4591	0.3342	0.3323	0.3909	0.3954
0.08	0.4573	0.3682	0.3517	0.4138	0.4332

3.3. Experiments for the Learning Process of the ILES. For establishing the soft sensor model based on the ILES, a total of 550 observations of real production data are collected from Tianjin Textile Engineering Institute Co., Ltd., of which 50 data are selected randomly as testing data. The remaining 500 data are divided into two data sets according to the time of production. The former 450 data are used as the training data set, and the latter 50 data are used as the update data set. The inputs are 9 factors that affect the sizing percentage. The output is the sizing percentage. The parameters of the ILES are $K = 9$, $T_k = 7$, $\delta = 0.06$, and $\varphi = 0.15$. That is to say, the 450 training data are divided into 9 subdatasets $K_1 \sim K_9$, and the number of ELMs is 7. According to the needs of the sizing production, the predictive accuracy of the soft sensors is defined as

$$\text{accuracy} = \frac{N_s}{N_w} \tag{9}$$

where N_s is the number of times with an error < 0.6 and N_w is the total number of testing times.

Since the learning process is similar to the OS-ELM [24] update process. It is an online assessment model that is capable of updating network parameters based on new arriving data without retrains historical data. Therefore, while comparing the accuracy of IELS learning process, it is also compared with OS-ELM. The two columns on the right side of Table 2 show the changes in the soft sensor accuracy during the learning process of the ILES and OS-ELM. It can be seen that the stability and accuracy of ILES are superior to OS-ELM.

3.4. Comparison. In this experiment, we used 10-fold cross validation to test the model's performance. The first 500 data sets are randomly divided into 10 subdatasets $S_1 \sim S_{10}$. The remaining 50 data sets are used as the updated data set S_{11}. The single subdataset from $S_1 \sim S_{10}$ will be retained as the

Table 2: The changes in the soft sensor accuracy during the learning process of the ILES.

Dataset	Iteration										
	1	2	3	4	5	6	7	8	9	Update	OS-ELM
K_1	95%	95%	95%	96%	95%	95%	97%	95%	95%	95%	56%
K_2		94%	94%	96%	94%	95%	95%	94%	94%	94%	68%
K_3			96%	95%	96%	96%	95%	95%	96%	97%	64%
K_4				97%	96%	94%	95%	95%	96%	95%	62%
K_5					96%	94%	96%	95%	95%	96%	66%
K_6						95%	95%	97%	96%	96%	70%
K_7							96%	96%	95%	96%	80%
K_8								93%	94%	94%	88%
K_9									94%	95%	88%
Update set										96%	88%
Testing set	83.2%	85%	85.1%	87%	90%	91%	91.8%	92%	93.2%	95%	90%

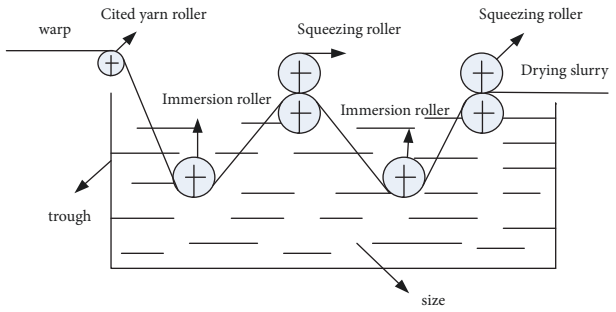


Figure 1: The double-dip and double pressure sizing processes.

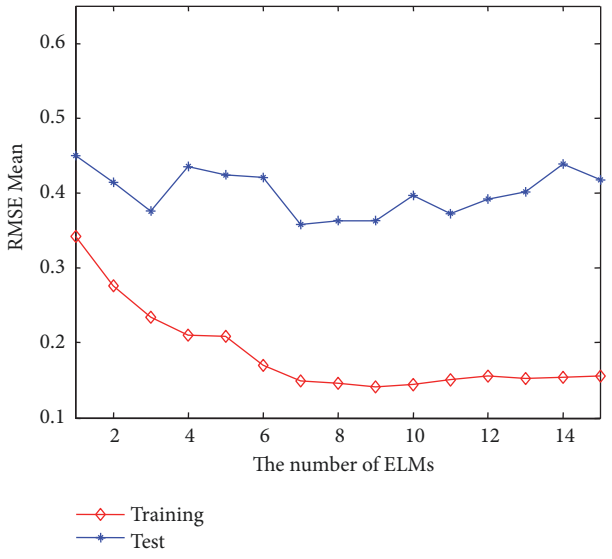


Figure 2: The training and testing errors of the ILES with different parameters T_k.

validation data for testing the model, and the remaining 9 subdatasets are used as the training data. For comparing the new method with other soft sensor methods, the single ELM and ensemble ELM based on AdaBoost.R are also applied to build the sizing percentage soft sensor models as traditional

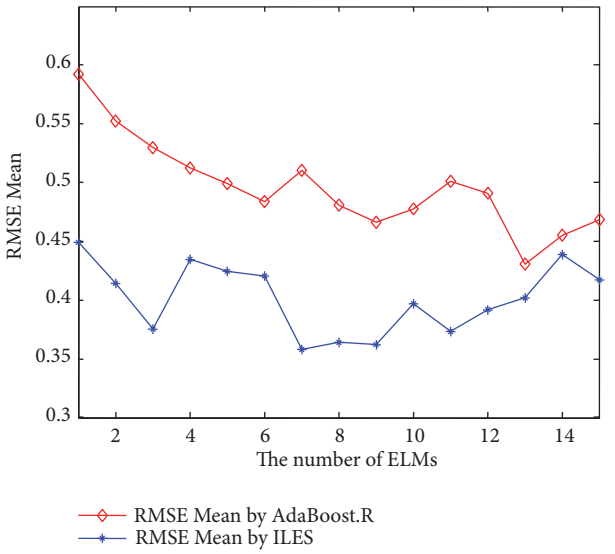


Figure 3: The performance trends of the ILES and AdaBoost.R with different parameters T_k.

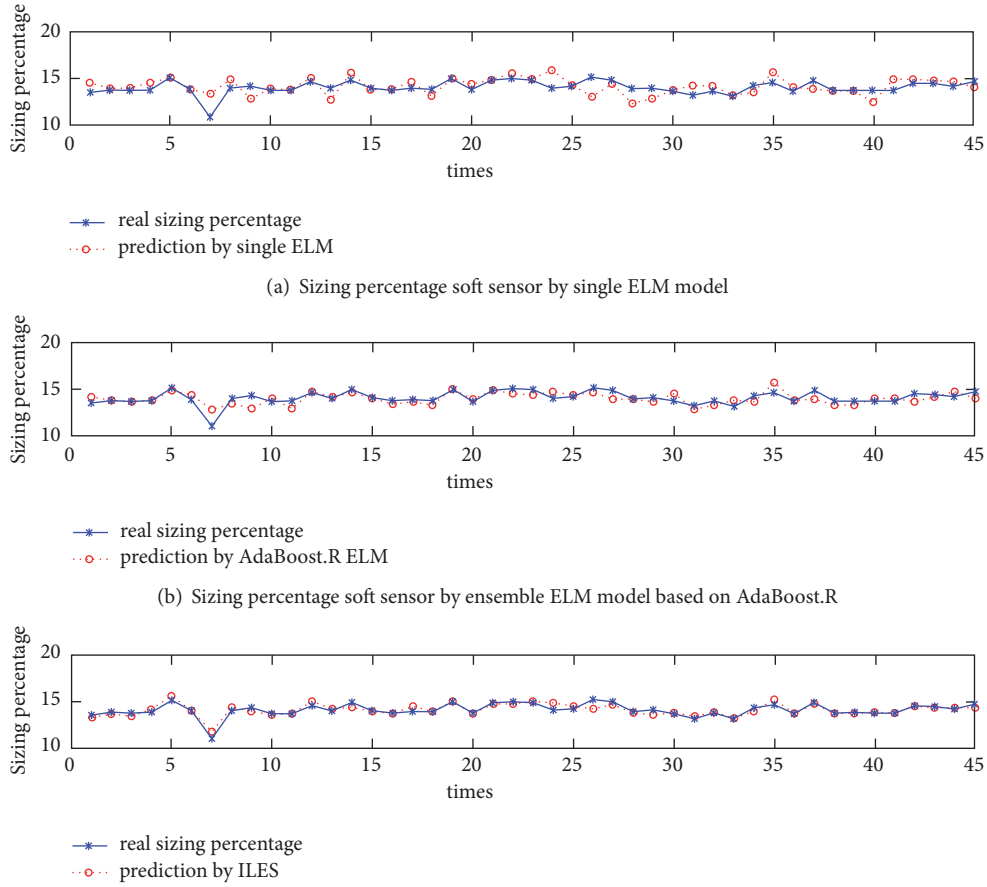
methods with the same data set. The soft sensor models are listed as follows.

Single ELM model: the main factors that affect the sizing percentage are the inputs of the model. The input layer of the ELM has 9 nodes. The hidden layer of the ELM has 2300 nodes, and the output layer of the ELM has one node, which is the sizing percentage.

AdaBoost.R model: the parameters and the structure of the base ELM are the same as those of the single ELM. AdaBoost.R has 13 iterations.

ILES model: the ELMs are same as the single ELM model described above. The parameters of the ILES are $T_k = 7$, $\delta = 0.06$, and $\varphi = 0.15$ (time consuming 163s).

Figures 4(a)–4(c) shows the predicted sizing percentage of different soft sensor models based on different methods. The experiments demonstrate that the strategy of the ILES can improve the accuracy of the soft sensor model. In addition, the training errors of the above three models all



(a) Sizing percentage soft sensor by single ELM model

(b) Sizing percentage soft sensor by ensemble ELM model based on AdaBoost.R

(c) Sizing percentage soft sensor by ILES model

Figure 4: The result of the sizing percentage soft sensor.

can achieve 0.2. However, the testing errors of the prediction models of AdaBoost.R and the ILES are smaller than that of the single ELM. This result means that the ensemble methods have better generalization performance. Table 3 shows the performance of the prediction model based on different methods after updating. The results of the comparison experiments show that the soft sensor based on the new ILES has the best accuracy and the smallest RMSE. This result is attributed to the use of the testing subdataset in the ensemble strategy and the incremental learning strategy during the learning process of the ILES algorithm. Overall, the accuracy of the soft sensor can fit the needs of actual production processes. Moreover, the incremental learning performance can ensure the application of industrial process soft sensors in practical production.

3.5. Experiments for the Performance of the ILES by Test Functions. To verify the universal use of the algorithm, three test functions are used to test the improvement of the prediction performance. These test functions are Friedman#1, Friedman#2, and Friedman#3. Table 4 shows the expression of each test model and the value range of each variable. Friedman#1 has a total of 10 input variables, of which there are five input variables associated with the output variable, and the other five input variables are independent of the output variables. The Friedman#2 and Friedman#3 test functions incorporate the impedance phase change of the AC circuit.

Through continuous debugging, the parameters of each algorithm are determined as shown in Table 5. For every test function, generate a total of 900 data, and 78% of the total samples were selected as training samples, 11% as updating samples and 11% as testing samples, according to the need for different test models. That is to say, the 700 training data are divided into 7 subdatasets $K_1 \sim K_7$. Figures 5–7 show the predicted results of Friedman#1, Friedman#2, and Friedman #3 with different soft sensor models based on different methods (time consuming 227s). The comparison of the performances of the different soft sensors is shown in Table 6. It shows the soft sensor model based on ILES has the best performance.

4. Conclusions

An ILES algorithm is proposed for better accuracy and incremental learning ability for industrial process soft sensors. The sizing percentage soft sensor model is established to test the performance of the ILES. The main factors that influence the sizing percentage are the inputs of the soft sensor model.

Table 3: The performance of the soft sensor model based on different methods with 10-fold cross validation.

Method	datasets			RMSE	ARE	Max ARE
	training	updating	testing			
ELM				0.7987	0.0698	0.2265
AdaBoost.R	$S_1 \sim S_9$	S_{11}	S_{10}	0.5915	0.0313	0.1447
ILES				0.3704	0.0213	0.0689
ELM				0.7386	0.0868	0.3773
AdaBoost.R	$S_1 \sim S_8, S_{10}$	S_{11}	S_9	0.4740	0.0394	0.1846
ILES				0.3309	0.0182	0.0542
ELM				0.8405	0.0772	0.2962
AdaBoost.R	$S_1 \sim S_7, S_9 \sim S_{10}$	S_{11}	S_8	0.5099	0.0295	0.1873
ILES				0.3323	0.0202	0.0572
ELM				0.936	0.0655	0.4255
AdaBoost.R	$S_1 \sim S_6, S_8 \sim S_{10}$	S_{11}	S_7	0.4835	0.219	0.1773
ILES				0.3556	0.0204	0.0525
ELM				0.7342	0.0618	0.3451
AdaBoost.R	$S_1 \sim S_5, S_7 \sim S_{10}$	S_{11}	S_6	0.5198	0.0324	0.1816
ILES				0.3965	0.0221	0.0641
ELM				0.8533	0.0872	0.3546
AdaBoost.R	$S_1 \sim S_4, S_6 \sim S_{10}$	S_{11}	S_5	0.4672	0.0336	0.1724
ILES				0.3531	0.0195	0.0499
ELM				0.7752	0.0749	0.3249
AdaBoost.R	$S_1 \sim S_3, S_5 \sim S_{10}$	S_{11}	S_4	0.4105	0.0362	0.1924
ILES				0.3934	0.0209	0.0748
ELM				0.8430	0.0823	0.3281
AdaBoost.R	$S_1 \sim S_2, S_4 \sim S_{10}$	S_{11}	S_3	0.5773	0.0525	0.1827
ILES				0.3656	0.0203	0.0610
ELM				0.9127	0.1104	0.4302
AdaBoost.R	$S_1, S_3 \sim S_{10}$	S_{11}	S_2	0.5695	0.0580	0.1891
ILES				0.3625	0.0211	0.0545
ELM				0.7905	0.0910	0.2245
AdaBoost.R	$S_2 \sim S_{10}$	S_{11}	S_1	0.5045	0.0436	0.1773
ILES				0.3145	0.0191	0.0756

Table 4: Test function expressions.

Data set	Expression	Variable scope
Friedman#1	$y = 10\sin(\pi x_1 x_2) + 20\left(x_3 - 0.5\right)^2 + 10x_4 + 5x_5 + \delta$	$x_i \sim U[0,1],\ i = 1, 2..., 10$ $\delta \sim U[-4, 4]$
Friedman#2	$y = \sqrt{{x_1}^2 + \left(x_2 x_3 - \frac{1}{x_2 x_4}\right)^2} + \delta$	$x_1 \sim U[0, 100]$ $x_2 \sim U[40\pi, 560\pi]$ $x_3 \sim U[0, 1]$ $x_4 \sim U[1, 11]$ $\delta \sim U[-120, 120]$
Friedman#3	$y = \tan^{-1}\left(\frac{x_2 x_3 - 1/x_2 x_4}{x_1}\right) + \delta$	$x_1 \sim U[0, 100]$ $x_2 \sim U[40\pi, 560\pi]$ $x_3 \sim U[0, 1]$ $x_4 \sim U[1, 11]$ $\delta \sim U[-0.4, 0.4]$

Table 5: Parameters of the algorithmic performance test.

Test function	T_k (the number of ELMs)	K	δ	φ	datasets		
					training	updating	testing
Friedman#1	10	7	0.012	0.1050	700	100	100
Friedman#2	10	7	0.012	0.510	700	100	100
Friedman#3	8	7	0.010	0.1810	700	100	100

Table 6: The performance comparisons of the algorithms.

Data set	Method	RMSE	ARE	Max ARE
Friedman#1	ELM	0.8278	0.7662	4.7880
	AdaBoost.R	0.7987	0.6536	3.8817
	ILES	0.3099	0.3073	2.7132
Friedman#2	ELM	0.6338	0.9975	17.9870
	AdaBoost.R	0.4190	0.4495	7.3544
	ILES	0.2079	0.3683	7.2275
Friedman#3	ELM	1.1409	1.4314	14.9348
	AdaBoost.R	0.8959	1.3461	11.5991
	ILES	0.4624	0.3930	5.2719

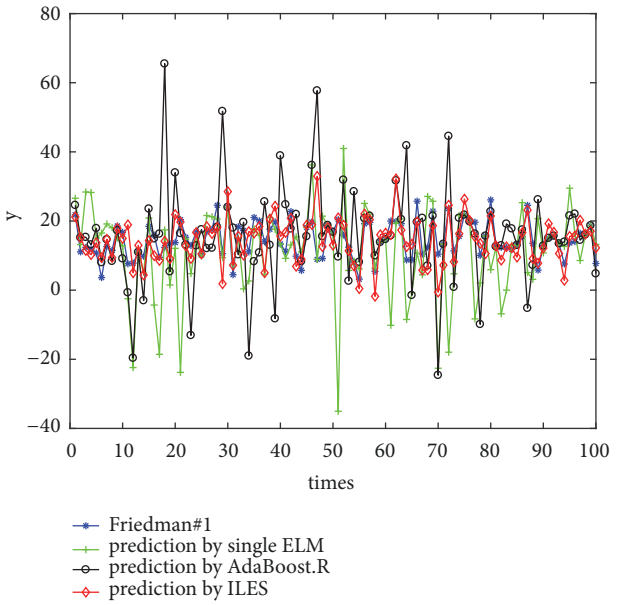


Figure 5: The comparison of Friedman#1 with different models.

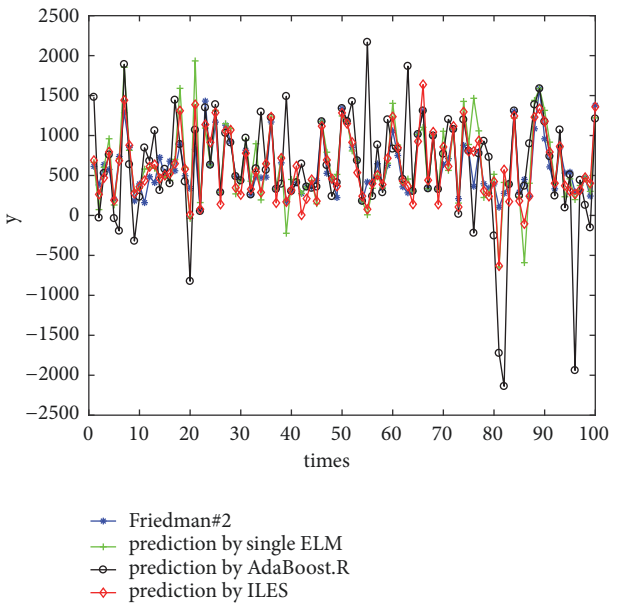


Figure 6: The comparison of Friedman#2 with different models.

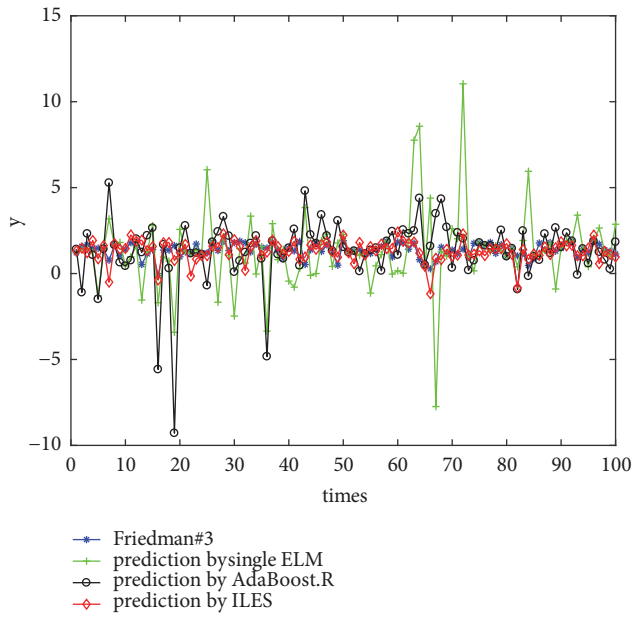


Figure 7: The comparison of Friedman#3 with different models.

Then, the ensemble model is trained with different subtraining dataset, and a soft sensor model with incremental learning performance is obtained by the ILES strategy. When new data add up to a certain number, the model will be updated using an incremental learning strategy. The new sizing percentage soft sensor model is used in Tianjin Textile Engineering Institute Co., Ltd. The experiments demonstrate that the new soft sensor model based on the ILES has good performance. The predictive accuracy of the new soft sensor model could be satisfied for sizing production. Finally, the new ILES is also tested with three testing functions to verify the performance with different data sets for universal use. Because the size of the subdataset is different from the experiment on sizing percentage, it can be conclude from the prediction results that the size of subdataset does not affect the performance of the algorithm. In the future, the ILES can also be used in other complex industry processes that require the use of soft sensors.

Appendix

Review of Extreme Learning Machine

Single Hidden Layer Feedforward Networks (SLFNs) with Random Hidden Nodes

For N arbitrary distinct samples $\{(x_j, t_j)\}_{j=1}^{N}$, where $x_j = [x_{j1}, x_{j2}, \cdots x_{jn}]^T \in R^n$ and $t_j = [t_{j1}, t_{j2}, \cdots t_{jm}]^T \in R^n$, standard SLFNs with $\widetilde{N}$ hidden nodes and the activation function $g(x)$ are mathematically modeled as

$$\sum_{i=1}^{\widetilde{N}} \beta_i g_i\left(x_j\right) = \sum_{i=1}^{\widetilde{N}} \beta_i g\left(w_i \cdot x_j + b_i\right) = o_j, \quad j = 1, \cdots, N \tag{A.1}$$

where $w_i = [w_{i1}, w_{i2}, \cdots w_{in}]^T$ is the weight vector connecting the ith hidden node and the input nodes, $\beta_i = [\beta_{i1}, \beta_{i2}, \cdots \beta_{im}]^T$ is the weight vector connecting the ith hidden node and the output nodes, $o_j = [o_{j1}, o_{j2}, \cdots, o_{jm}]^T$ is the output vector of the SLFN, and b_i is the threshold of the ith hidden node. $w_i \cdot x_j$ denotes the inner product of w_i and x_j. The output nodes are chosen linearly. The standard SLFNs with $\widetilde{N}$ hidden nodes with the activation function $g(x)$ can approximate these N samples with zero error means such that $\sum_{j=1}^{\widetilde{N}} \|o_j - t_j\| = 0$. These N equations can be written compactly as follows:

$$\mathbf{H}\beta = \mathbf{T} \tag{A.2}$$

where

$$\mathbf{H} = \begin{bmatrix} g\left(w_1 \cdot x_1 + b_1\right) & \cdots & g\left(w_{\widetilde{N}} \cdot x_1 + b_{\widetilde{N}}\right) \\ \vdots & \cdots & \vdots \\ g\left(w_1 \cdot x_N + b_1\right) & \cdots & g\left(w_{\widetilde{N}} \cdot x_N + b_{\widetilde{N}}\right) \end{bmatrix}_{N \times \widetilde{N}} \tag{A.3}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\widetilde{N}}^T \end{bmatrix}_{\widetilde{N} \times m} \tag{A.4}$$

and

$$\mathbf{T} = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \tag{A.5}$$

Here, $\mathbf{H}$ is the hidden layer output matrix.

ELM Algorithm. The parameters of the hidden nodes do not need to be tuned and can be randomly generated permanently according to any continuous probability distribution. The unique smallest norm least squares solution of the above linear system is

$$\widehat{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{A.6}$$

where $\mathbf{H}^{+}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$.

Thus, a simple learning method for SLFNs, called extreme learning machine (ELM), can be summarized as follows.

Step 1. Randomly assign input weight w_i and bias b_i, $i = 1, \cdots, \widetilde{N}$,

Step 2. Calculate the hidden layer output matrix $\mathbf{H}$.

Step 3. Calculate the output weight $\beta : \beta = \mathbf{H}^{\dagger}\mathbf{T}$.

The universal approximation capability of the ELM has been rigorously proved in an incremental method by Huang *et al*. [23].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grants nos. 71602143, 61403277, 61573086, and 51607122), Tianjin Natural Science Foundation (no. 18JCYBJC22000), Tianjin Science and Technology Correspondent Project (no. 18JCTPJC62600), the Program for Innovative Research Team in University of Tianjin (nos. TD13-5038, TD13-5036), and State Key Laboratory of Process Automation in Mining & Metallurgy/Beijing Key Laboratory of Process Automation in Mining & Metallurgy Research Fund Project (BGRIMM-KZSKL-2017-01).

References

[1] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.

[2] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[3] S. Liu, S. Wang, J. Chen, X. Liu, and H. Zhou, "Moving larval shrimps recognition based on improved principal component analysis and AdaBoost," *Transactions of the Chinese Society of Agricultural Engineering (Transactions of the CSAE)*, vol. 33, no. 1, pp. 212–218, 2017.

[4] Y. Li, H. Guo, and X. Liu, "Classification of an integrated algorithm based on Boosting in unbalanced data," *Journal of Systems Engineering and Theory*, vol. 01, pp. 189–199, 2016.

[5] L. L. Wang and Z. L. Fu, "AdaBoost algorithm for multi-tag classification based on label correlation," *Journal of Sichuan University (Engineering Science Edition)*, vol. 5, pp. 91–97, 2016.

[6] H. Tian and A. Wang, "Advanced AdaBoost modeling method based on incremental learning," *Control and Decision*, vol. 09, pp. 1433–1436, 2012.

[7] M. Wu, J. Guo, Y. Ju, Z. Lin, and Q. Zou, "Parallel algorithm for parallel selection based on hierarchical filtering and dynamic updating," *Computer Science*, vol. 44, no. 1, pp. 48–52, 2017.

[8] H. Drucker, "Improving regressor using boosting," in *Proceedings of the 14th International Conference on Machine Learning*, pp. 107–115, Morgan Kaufmann Publishers, Burlington, Mass, USA, 1997.

[9] R. Avnimelech and N. Intrator, "Boosting regression estimators," *Neural Computation*, vol. 11, no. 2, pp. 499–520, 1999.

[10] R. Feely, *Predicting Stock Market Volatility Using Neural Networks*, Trinity College Dublin, 2000.

[11] D. L. Shrestha and D. P. Solomatine, "Experiments with AdaBoost.RT, an improved boosting scheme for regression," *Neural Computation*, vol. 18, no. 7, pp. 1678–1710, 2006.

[12] H.-X. Tian and Z.-Z. Mao, "An ensemble ELM based on modified AdaBoost.RT algorithm for predicting the temperature of molten steel in ladle furnace," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 1, pp. 73–80, 2010.

[13] L. Kao, C. Chiu, C. Lu, C. Chang et al., "A hybrid approach by integrating wavelet-based feature extraction with MARS and SVR for stock index forecasting," *Decision Support Systems*, vol. 54, no. 3, pp. 1228–1244, 2013.

[14] X. Qiu, P. N. Suganthan, and G. A. J. Amaratunga, "Ensemble incremental learning Random Vector Functional Link network for short-term electric load forecasting," *Knowledge-Based Systems*, vol. 145, no. 4, pp. 182–196, 2018.

[15] Z. Zhou, J. Chen, and Z. Zhu, "Regularization incremental extreme learning machine with random reduced kernel for regression," *Neurocomputing*, vol. 321, no. 12, pp. 72–81, 2018.

[16] R. Polikar, L. Udpa, S. S. Udpa, and V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 31, no. 4, pp. 497–508, 2001.

[17] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2283–2301, 2013.

[18] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: the accuracy updated ensemble algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 81–94, 2013.

[19] H.-J. Rong, Y.-X. Jia, and G.-S. Zhao, "Aircraft recognition using modular extreme learning machine," *Neurocomputing*, vol. 128, pp. 166–174, 2014.

[20] Q. Lin, X. Wang, and H.-J. Rong, "Self-organizing fuzzy failure diagnosis of aircraft sensors," *Memetic Computing*, vol. 7, no. 4, pp. 243–254, 2015.

[21] X. Xu, D. Jiang, B. Li, and Q. Chen, "Optimization of Gaussian mixture model motion detection algorithm," *Journal of Chemical Industry and Engineering*, vol. 55, no. 6, pp. 942–946, 2004.

[22] H. Tian and Y. Jia, "Online soft measurement of sizing percentage based on integrated multiple SVR fusion by Bagging," *Journal of Textile Research*, vol. 35, no. 1, pp. 62–66, 2014 (Chinese).

[23] G. Huang, Q. Zhu, and C. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *Proceedings of International Joint Conference on Neural Networks*, vol. 2, pp. 985–990, 2004.

[24] N. Liang, G. Huang, H. Rong et al., "On-line sequential extreme learning machine," in *Proceedings of the Iasted International Conference on Computational Intelligence*, pp. 232–237, DBLP, Calgary, Alberta, Canada, 2005.