Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 5281512, 15 pages
https://doi.org/10.1155/2020/5281512

Research Article

Deep Forest-Based Fault Diagnosis Method for Chemical Process

Jiaman Ding, Qingbo Luo, Lianyin Jia, and Jinguo You

Kunming University of Science and Technology, Artificial Intelligence Key Laboratory of Yunnan Province, Kunming, 650500 Yunnan, China

Correspondence should be addressed to Lianyin Jia; jlianyin@163.com

Received 30 August 2019; Revised 20 December 2019; Accepted 2 January 2020; Published 25 January 2020

Academic Editor: Sergey A. Suslov

With the rapid expanding of big data in all domains, data-driven and deep learning-based fault diagnosis methods in chemical industry have become a major research topic in recent years. In addition to a deep neural network, deep forest also provides a new idea for deep representation learning and overcomes the shortcomings of a deep neural network such as strong parameter dependence and large training cost. However, the ability of each base classifier is not taken into account in the standard cascade forest, which may lead to its indistinct discrimination. In this paper, a multigrained scanning-based weighted cascade forest (WCForest) is proposed and has been applied to fault diagnosis in chemical processes. In view of the high-dimensional nonlinear data in the process of chemical industry, WCForest first designs a set of relatively suitable windows for the multigrained scan strategy to learn its data representation. Next, considering the fitting quality of each forest classifier, a weighting strategy is proposed to calculate the weight of each forest in the cascade structure without additional calculation cost, so as to improve the overall performance of the model. In order to prove the effectiveness of WCForest, its application has been carried out in the benchmark Tennessee Eastman (TE) process. Experiments demonstrate that WCForest achieves better results than other related approaches across various evaluation metrics.

1. Introduction

Performance improvement and surveillance facilitation have become increasingly important in industrial processes. Accompanied by extreme conditions, modern industrial processes are becoming more and more complex. In the case of underdeveloped monitoring technology and lack of historical fault data, diagnosis technology mainly consists of two types of diagnosis methods based on process and knowledge [1, 2]. They make the diagnosis results easy to understand, but the use cost is too high for systems with many devices and large state variables [3]. However, the modern industries are developing in the direction of large scale and complexity, and with the widespread use of monitoring technology, large volumes of industrial process data have been collected from broadly deployed sensors and other control equipment. Therefore, to maximize use of these massive data to further improve both accuracy and speed of fault diagnosis is significant for a complicated process monitoring system.

With the increase of storage capacity and computing power, data-driven fault diagnosis methods have been widely used in chemical processes [4, 5]. Among these methods, the multivariate statistical method, mainly including principal component analysis (PCA) [6, 7], partial least squares (PLS) [8, 9], independent components analysis (ICA) [10, 11], Fisher discriminant analysis (FDA) [12, 13], random forest (RF) [14], canonical correlation analysis (CCA) [15], exponential discriminant analysis (EDA) [16], and their derivatives [17–22], have also made a rapid progress. Although certain effects have been achieved by these data-driven methods, there are still two shortcomings: On one hand, most of these methods rely on an assumption of a single data distribution (e.g., Gaussian distribution) [23, 24]. But in actual industrial processes, data do not always strictly follow a certain distribution. Therefore, expert experiences will be needed for these methods. Approximate hypothesis can also be used to process these data, but diagnostic errors may be generated. On the other hand, in the context of big data, the above methods are easy to be

saturated for sample data; that is, when sample size increases to a certain scale, it is difficult to further utilize the remaining sample data to improve the fault diagnosis accuracy.

In order to maximize the use of massive data, in recent years, deep learning (DL) has been applied to various fields of big data, and a large number of DL based fault diagnosis methods have emerged [25–28]. Xie and Bai [29] proposed a hierarchical deep neural network (HDNN) to diagnose faults in the benchmark Tennessee Eastman (TE) process. By training a monitoring deep neural network (DNN), the faults are divided into several groups. For each group, a special DNN trained is triggered for further diagnosis. Zhang and Zhao [30] presented an extensible deep belief network- (DBN-) based fault diagnosis model. The features of fault data in spatial and temporal domains are extracted by DBN subnet, and then fault classification is carried out by the global back propagation network. Moreover, a deep convolution neural network- (DCNN-) based fault diagnosis method was also proposed [31], which achieves better results than the former one. However, some shortcomings may limit the application of DNN in fault diagnosis: (1) DNN is mainly used to process the spectrogram of image and speech recognition in computer vision, and in order to extract both spatial and temporal features, the input data in fault diagnosis are usually processed to a two-dimensional data matrix composed of a period of time [29–31]. So, it may result in a low real-time performance. (2) It is well known that the performance of DNN depends largely on parameter adjustment because of a large number of hyperparameters.

In order to alleviate the aforementioned shortcomings of DNN, an alternative of DNN, gcForest [32], was proposed in 2017, which can achieve comparable or even better results than DNN on several domains. gcForest has much fewer hyperparameters than DNN and can be easily trained without too many parameter-adjustment skills. However, in gcForest, two key issues, the diversity of classifiers and the power of each classifier, should be paid attention on. For the former, different forests can be used, such as random forest, completely random forest, and so on. For the latter, in this paper, a weighted cascade forest (WCForest) model is proposed. The main idea of WCForest is to design a strategy to set weight for each forest in cascade structure and to improve the performance of the good forests and to restrain the bad ones.

The remainder of this paper is organized as follows. Section 2 introduces the principle and mathematical model of gcForest. The WCForest-based fault diagnosis model is proposed in Section 3. The applications of WCForest in the TE process and the comparisons with other fault diagnosis methods are discussed in Section 4. Finally, conclusions are drawn in Section 5.

2. Multigrained Cascade Forest

gcForest consists of two integrated components: the multigrained scanning and the cascade forest. The multigrained scanning adopts sliding window to scan local context from high-dimensionality to learn representations of input data by different forests. The cascade forest learns more discriminating representations under the supervision of input representations at each level, so as to give a more accurate prediction according to the ensemble of forests.

2.1. Multigrained Scanning. Inspired by feature relationships of CNN, the cascade forest adopts a sliding window-based multigrained scanning strategy. An illustration of its process is given in Figure 1. Suppose that there are N instances of M classes in the training dataset and the dimension of each instance is m. A sliding window of size n is used to scan each instance, and the $(m-n+1)$ n-dimensional feature vectors can be generated by scanning each raw instance sequentially. All feature vectors extracted from the raw instance are regarded as derived instances of this class. For each n-dimensional derived instance, each forest generates M-dimensional class vector. The $(m-n+1)$-derived instances of each raw instance are input into random forest and completely random forest to generate their class distribution vectors and then to concatenate them into transformed feature vector of $2M*(m-n+1)$-dimensional. As shown in Figure 1, the training dataset includes three classes and each raw instance has 400 dimensions and the sliding window size is 100. Therefore, from the above process, a feature vector of an 1806-dimensional transformed feature vector corresponding to a 400-dimensional raw feature vector is obtained. Compared with the raw vector, the transformed feature vector has much more dimensions and an enhanced feature representation.

2.2. Cascade Forest. In the cascaded forest, each cascade layer assembles many decision forests, receives the features processed by its previous layer, and inputs its processing results to its next layer. In fact, each layer is designed to include different types of forests to encourage overall diversity. Figure 2 shows the schematic of an example cascade forest, in which two types of forests (random forest in green and completely random forest in blue) are used. The number of forests per layer and the number of trees in each forest are hyperparameters in practice. The instances are input to the cascade layer, and each forest produces an estimate of class distribution. The class distribution outputs of all forests in the same layer form a class vector, which is then connected with the raw vector as an input of the next cascade. Cross-validation is used to evaluate the overall expansion performance of the new layer. When there is no performance improvement, the expansion progress will be automatically terminated.

For each instance, each forest will generate an estimated vector of class distribution by averaging classification probability of all trees in the same forest. The classification probability of a tree is obtained by calculating the proportion of different classes of training instances at the leaf node where the concerned instance falls. The process of the distribution characteristics of random forests is shown in Figure 3. Suppose the class-distribution vector obtained by the ith tree in the forest is $X_i = (x_{1i}, x_{2i}, \ldots, x_{Mi})$, where M represents the number of classes and each forest contains t trees, then the class distribution vector generated by the

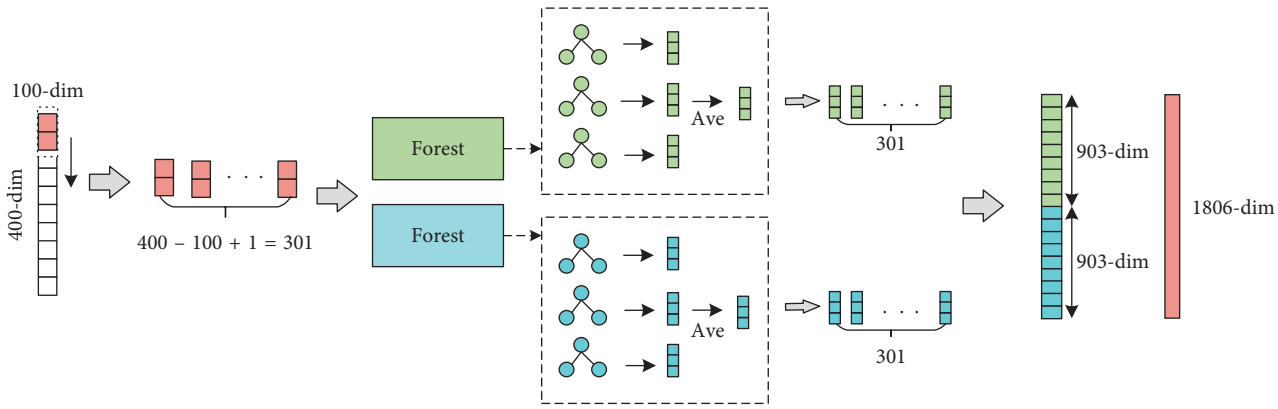

Figure 1: Illustration of multigrained scanning.

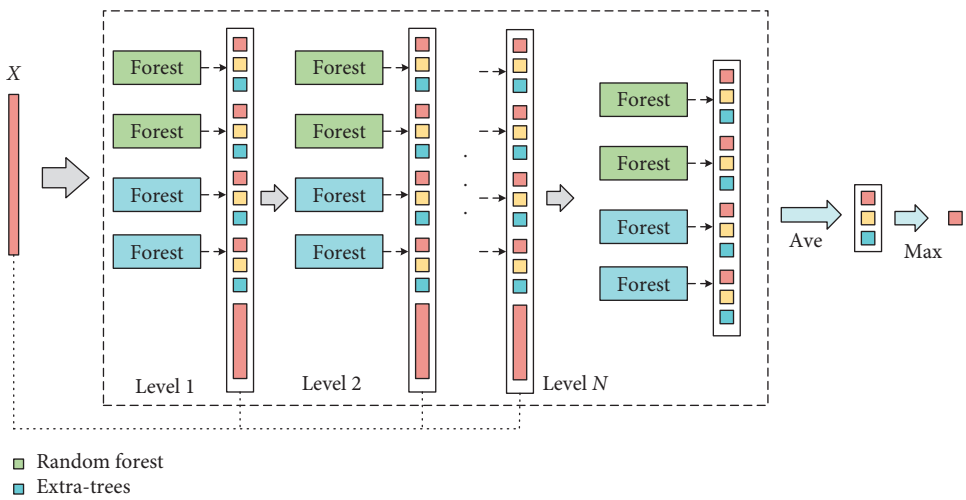

Figure 2: Illustration of cascade forest structure.

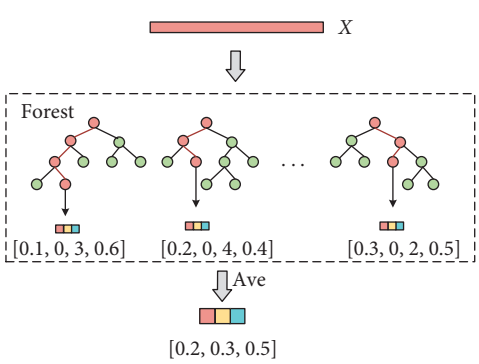

Figure 3: Illustration of process of generating class vector of random forest.

forest is $X' = (x_1', x_2', \ldots, x_M')$, where $x_j' = (1/t) \sum_{i=1}^{t} x_{ji}$, $j = 1, 2, \ldots, M; i = 1, 2, \ldots, t$.

3. WCForest-Based Fault Diagnosis Method

3.1. Weighted Cascade Forest. As a substitute for DNN, the cascade forest learns hyperlevel representation in a low cost. It does not learn hidden variables based on complex forward or backward propagation algorithms in DNN. Instead, it directly learns class-distribution features by assembling a large number of decision-tree-based forests under the supervision of input. The layer-wise supervised learning strategy makes cascade forests easy to be trained. Moreover, the ensemble of forests can acquire more precise class-distribution features, owing to its powerful ability in most classification applications. However, in a standard cascade forest model, all forests in each cascade structure contribute equally to the final prediction, which may result in a sensitive estimation of classification distribution to the amount of forests fitting. In order to alleviate this problem, based on cascade forest, this section introduces a new variant of cascade forest, WCForest.

Inspired by weighted voting, we give higher weights to excellent classifiers than poor classifiers in the training process of the cascade structure. Obviously, it is difficult to define rules to set weights to the forests in the cascade structure. On one hand, the sample set of training forests are random, but the result of a single classification is not suitable for measuring the quality of the forest. On the other hand, extensive calculation and estimation of weights may bring additional costs. In this study, we attempt to set weights for the forests as objectively as possible without additional costs.

Specifically, the performance of the forest can be measured by training results of different subtraining samples. In order to mitigate the risk of overfitting, cross-validation is used to evaluate the overall performance of each layer. Therefore, the classification accuracy of cross-validation for each forest can be used to estimate its weights. The reasons for using cross-validation to calculate weights are as follows: (1) Cross-validation itself is a default way to evaluate the performance of new layer in the cascade forest, so using it as a strategy for calculating weights of forests does not incur additional computational costs. (2) Cross-validation determines the weight of the forest classification quality through multiple verifications, which eliminates the contingency of verification.

Assuming that there are M classes in the training set, the weight of each forest in each layer is estimated by k-fold cross-validation. The training set is divided into k subsample sets, one of which is retained as a verification set, and the other $k-1$ subsample sets are used to train the forest. Cross-validation is repeated k times. Each subsample set validates a random forest at one time, leading to a classification accuracy. After training and verifying each level of the cascade forest, the classification accuracy matrix ACC can be generated as follows:

$$\mathrm{ACC} = \begin{bmatrix} \mathrm{acc}_{11} & \mathrm{acc}_{12} & \cdots & \mathrm{acc}_{1k} \\ \mathrm{acc}_{21} & \mathrm{acc}_{22} & \cdots & \mathrm{acc}_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{acc}_{t1} & \mathrm{acc}_{t2} & \cdots & \mathrm{acc}_{tk} \end{bmatrix}, \tag{1}$$

where acc_{ij}, $i = 1, 2, \ldots, t$; $j = 1, 2, \ldots, k$ denotes the accuracy of the jth cross-validation of the ith forest, t represents the number of random forests in each cascade structure, and k is the number of cross-validations.

According to ACC, the average classification accuracy of each forest is as follows:

$$\overline{\mathrm{acc}_i} = \frac{1}{k}\sum_{j=1}^{k} \mathrm{acc}_{ij}, \quad i = 1, 2, \ldots, t, \tag{2}$$

the weight matrix W is defined as follows:

$$W = \begin{pmatrix} w_1 & w_2 & \cdots & w_t \end{pmatrix}^T, \tag{3}$$

where $w_i\,(i = 1, 2, \ldots, t)$ represented the weight of ith forest, which is calculated as follows:

$$w_i = \frac{\overline{\mathrm{acc}_i}}{\sum_{i=1}^{t}\overline{\mathrm{acc}_i}}. \tag{4}$$

Given a new instance, each forest produces an estimate of the class distribution as described in [29]. Assuming that the class distribution vector obtained by the ith random forest in a cascade forest is $X_i = (x_{i1}, x_{i2}, \ldots, x_{iM})$, then the weighted class probability vector of the next cascade structure is $[w_1X_1, w_2X_2, \ldots, w_tX_t]$ and is connected with the raw vector together as an input to the next layer.

If the current layer is the last layer of the model, the class distribution matrix of the cascade forest is as follows:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1M} \\ x_{21} & x_{22} & \cdots & x_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t1} & x_{t2} & \cdots & x_{tM} \end{bmatrix}. \tag{5}$$

The weighted class probability results can be calculated as follows:

$$(WX)_j = \sum_{i=1}^{t}\left(w_i * x_{ij}\right), \tag{6}$$

where $(WX)_j$, $j = 1, 2, \ldots, M$, represents the total probability of class j.

Finally, the class with the maximum probability is chosen as the fault classification results, as shown in Figure 4.

$$\mathrm{Final}\,(j) = \arg\max_j\left\{(WX)_j\right\}, \quad j = 1, 2, \ldots, M. \tag{7}$$

3.2. WCForest-Based Fault Diagnosis Model. The process data of industrial processes are usually high-dimensional and noisy. Generally, the original input space is mapped to the feature space by feature extraction. However, the effect of feature extraction will directly affect the performance of the classifier. The two randomness of random forest make it to have better antinoise ability, and when the input data have high dimensionality, the representative learning ability can be further enhanced by multigrained scanning, which may make WCForest have a context or structure awareness. Based on the WCForest fault diagnosis model, the data extracted from each monitor in the industrial process are diagnosed and will get the evaluation status of data at each time.

In this paper, the process of the model consists of two parts: multigrained scanning-based feature extraction and weighted cascade forest-based fault diagnosis. After the data are collected, we use multigrained scanning to extract representation vectors from training and testing sets. Then, the weighted cascade forest classification model is trained by the representation vectors of training set and validated by the representation vectors of testing set. Finally, we obtain the classification results of testing set. The flow chart of the model is shown in Figure 5. Its diagnostic procedures include offline modeling and online diagnosis, described as follows:

Offline stage:

Step 1. Historical data are collected and preprocessed from the chemical process.

Step 2. Data collected at each time are composed into m-dimension vectors and labeled with their corresponding classes, including "normal" and their fault types.

Step 3. The samples including their corresponding labels are divided into the training set and the testing set.

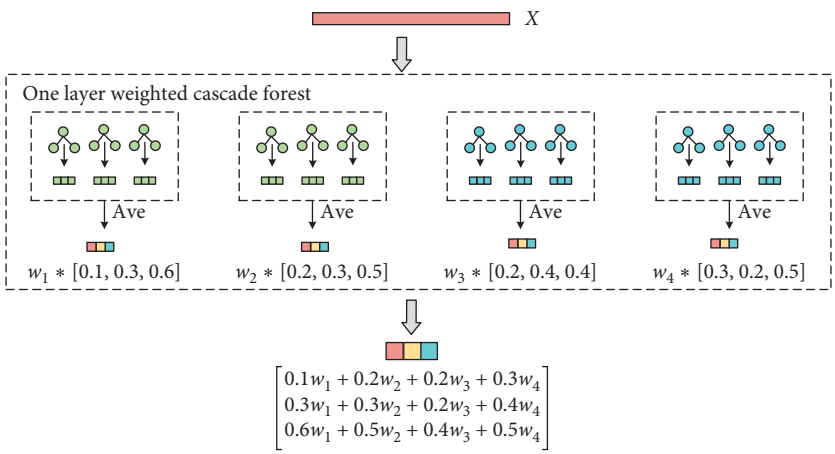

Figure 4: Illustration of generation of weighted class vectors.

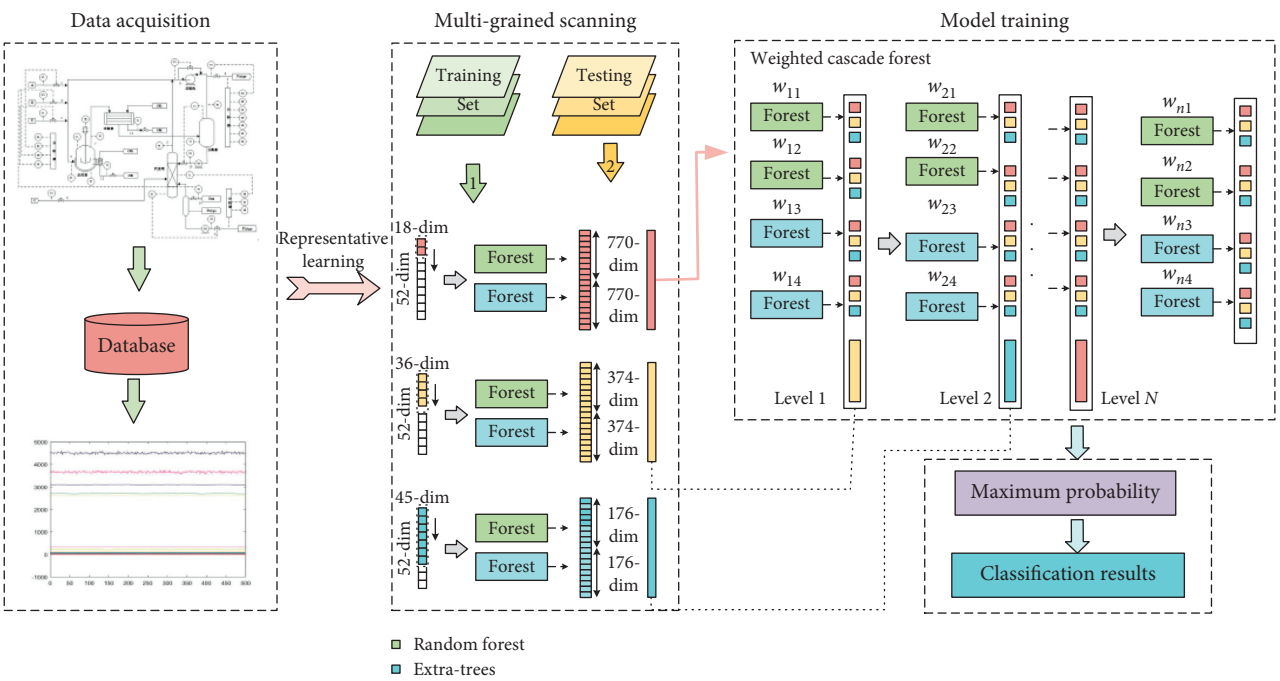

Figure 5: The flow chart of fault diagnosis method based on WCForest.

Step 4. Given several different sets of windows, use the training set to select a set of windows from them for multigrained scanning.

Step 5. The class probability vectors of training set and testing set is obtained by multigrained scanning of selected set of windows.

Step 6. In training set, the k-fold cross-validation is used to train the WCForest model, and obtain the weight vector $W = (w_1 \ w_2 \ \cdots \ w_t)^T$ of each layer. Verify the WCForest model using the class probability vectors of testing set.

Step 7. The fault diagnosis result is outputted and visualized.

Online stage:

Step 1. Online data are collected from the chemical process.

Step 2. Online sample vectors are input to the WCForest, which can give a predicted diagnosis result for each sample vector. The diagnosis result is either "normal" or one specific fault type.

4. Experiment Result

In this section, the proposed WCForest-based fault model is applied to the TE process. Furthermore, the results of the proposed method are compared with other decision tree-based ensemble methods (RF, XGBoost, AdaBoost), gcForest, and existing literature.

4.1. Tennessee Eastman Process. As a real industrial process-based simulation platform, the Tennessee Eastman process is widely used to evaluate the performance of monitoring

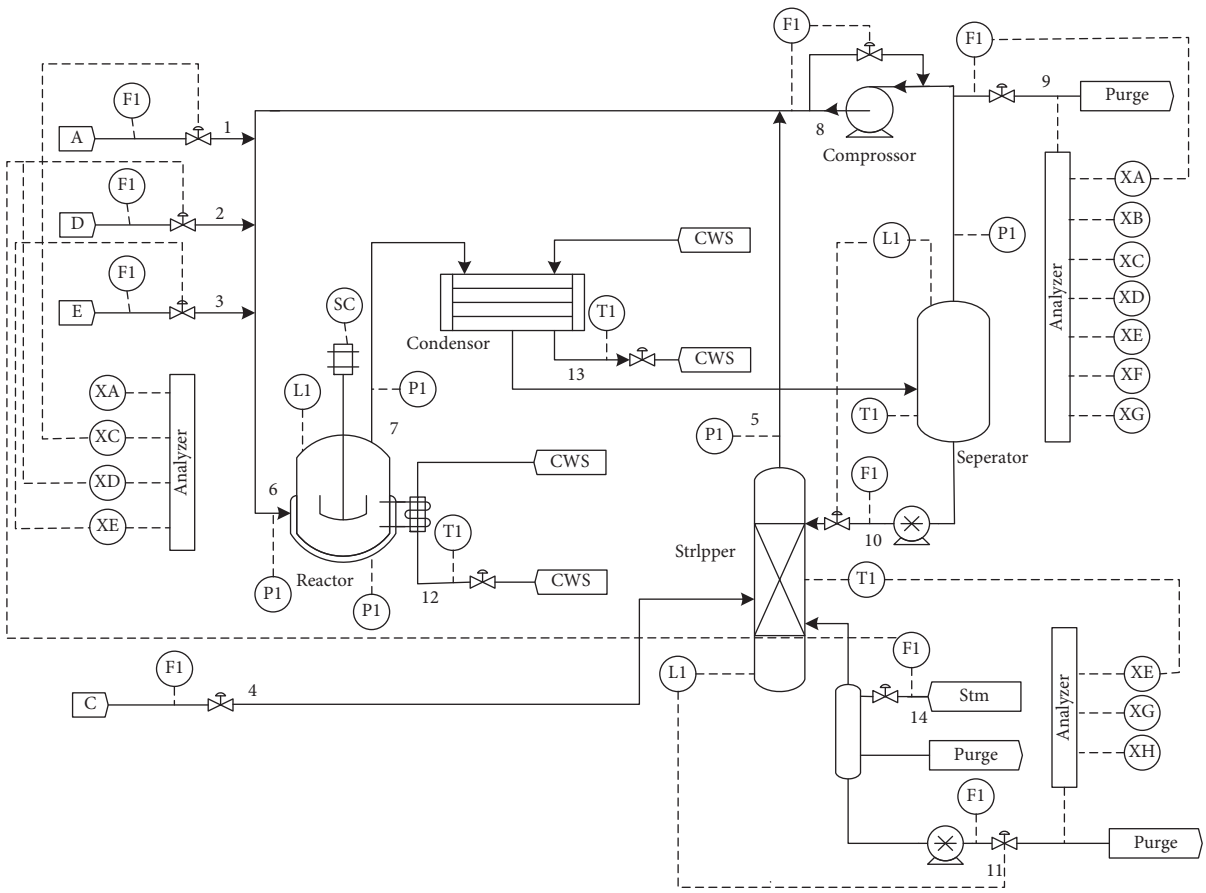

Figure 6: The flow diagram of TE process.

methods in the field of data-driven fault detection research. Figure 6 shows a flow diagram of the TE process. The process consists of 5 major unit operations: reactor, product condenser, vapor-liquid separator, recycle compressor, and product stripper. In addition, the process consists of four reactants—A, C, D, and E—and an inertia component B. The 4 reactants and the inertia component are sent to the reactor, and then the process produces liquid products G, H, and a byproduct F. The reaction process is irreversible, exothermic, and approximately first-order with respect to concentrations.

The TE process includes 41 measured variables and 12 manipulated variables. However, one of the manipulated variables, reactor speed, is always constant, and does not need to be analysed. The remaining 52 variables are used as research variables [33], which are all listed in Table 1. The first 41 variables are measured variables and the last 11 variables are manipulated variables. The TE process contains 21 faults which are listed in Table 2. The data used for faults classification of TE simulation system can be downloaded from http://web.mit.edu/braatzgroup. Each state (normal state and 21 different fault states) is divided into training and testing sections. The data are sampled once every three minutes. The training data are sampled 500 times for 25 hours and faults are introduced after one hour, so the simulation only uses the remaining 480 fault samples to a fault diagnosis model. The testing data are sampled 960 times for 48 hours, and faults are introduced after 8 hours; that is, the fault samples were collected from the 161th sampling point. With these normal sample set and fault sample sets, a completed WCForest model can be trained and tested.

4.2. WCForest Model for TE Process. The WCforest model suitable for TE process fault diagnosis is designed and constructed, in which the construction of forest and the setting of some hyperparameters need to be selected experimentally, such as the number of trees in each forest, the number and types of forest, and the setting of feature window. To find a suitable model, we tried the following experiment.

In the process of constructing a decision tree, information gain and Gini index are generally used as heuristic functions for feature selection. In this paper, we tested these two feature selection rules, respectively, and the test accuracy obtained was not significantly different. Therefore, relatively good Gini index was selected as the node splitting rule of the random forest model in this paper.

For hyperparameters in the model, the numbers of RF in the multigrained scanning and the cascade structure are set according to the setting in the literature [29]. 2 random forests (1 completely random forest and 1 random forest) and 8 random forests (4 completely random forests and 4

Table 1: Measurement and manipulation variables of the TE process.

No.	Process measurements
1	A feed
2	D feed
3	E feed
4	Total feed
5	Recycle flow
6	Reactor feed rate
7	Reactor pressure
8	Reactor level
9	Reactor temperature
10	Purge rate
11	Product separator temperature
12	Product separator level
13	Product separator pressure
14	Product separator underflow
15	Stripper level
16	Stripper pressure
17	Stripper underflow
18	Stripper temperature
19	Stripper steam flow
20	Compressor work
21	Reactor cooling water outlet temperature
22	Separator cooling water outlet temperature
23	A in reactor feed
24	B in reactor feed
25	C in reactor feed
26	D in reactor feed
27	E in reactor feed
28	F in reactor feed
29	A in reactor feed
30	B in reactor feed
31	C in reactor feed
32	D in reactor feed
33	E in reactor feed
34	F in reactor feed
35	G in reactor feed
36	H in reactor feed
37	D in product flow
38	E in product flow
39	F in product flow
40	G in product flow
41	H in product flow
42	D feed flow valve
43	E feed flow valve
44	A feed flow valve
45	Total feed flow valve
46	Compressor recycle valve
47	Purge valve
48	Separator pot liquid flow valve
49	Stripper liquid product flow valve
50	Stripper steam valve
51	Reactor cooling water flow
52	Condenser cooling water flow

random forests) were set up in each layer of the weighted cascade forest, respectively. However, the number of trees in forest N_{tree} and the setting of scanning window have a great influence on the diagnostic accuracy, so its parameters need to be optimized. First of all, we discuss the value of N_{tree}, which has little effect on the diagnostic accuracy when $400 \sim 1000$ trees. Considering the relationship of time complexity, fixed $N_{\text{tree}} = 500$. It is a common issue that there is no scientific guidance for the setting of windows. In order to find a proper setting, we tried several window settings: [15, 30], [20, 45], [13, 30, 42], [18, 36, 45], [16, 27, 35, 42], and [18, 25, 36, 47].

Here the dataset samples with 400 samples of one class are randomly selected; 80% of each class samples are training dataset, and others are testing dataset. The fault diagnosis testing is on one sample each time. The testing average diagnostic accuracy and the training/testing time of the different window settings are listed in Table 3. The window setting of [18, 25, 36, 47] has the highest testing average diagnostic accuracy (75.9%) and takes 83.4 min for training. With a little decrease of the testing average diagnostic accuracy (75.6%), the window setting of [18, 36, 45] takes 20 min less than the setting of [18, 25, 36, 47]. In the following discussion, [18, 36, 45] is chosen as the best window setting.

4.3. Fault Diagnosis Result. Experimental results of fault diagnosis for display, two commonly used indicators, fault detection rate (FDR) and false positive rate (FPR), are considered to evaluate the diagnostic performance of the model, and they can be calculated by the general confusion matrix defined in Table 4, which are shown in the following equation:

$$\text{FDR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{8}$$

Table 5 shows the FDRs of 21 faults in the TE process obtained by WCForest, gcForest, and three decision tree-based integration methods: Random Forest (RF), XGBoost, and AdaBoost. The setting parameter of gcForest is the same as WCForest. The parameters of the remaining algorithms are set as follows (which are all set by multiple parameter adjustments):

(1) RF: 400 decision trees, Gini index as classification rule.

(2) XGBoost: 400 decision trees, that is, the number of iterations, learning rate = 0.1, softmax loss function as objective function.

(3) AdaBoost: 500 decision trees, learning rate = 0.6.

Compared with the diagnostic results of RF, the FDRs of most faults have increased in varying degrees, which is of great significance in industrial production and theoretical research. RF and XGBoost have a good performance for faults with obvious feature differences, such as 1, 2, 6, 7, and so on, and a low performance for well-known faults, such as 3, 9, and 15. AdaBoost generally has a low diagnostic rate, with the exception of fault 2 reaching nearly 100%. gcForest enhances the perception of differences between features in cascade structure and improves the classification ability of the model through multigrained scanning representation

Table 2: Faults details of the TE process.

No.	Description	Type
1	A/C feed ratio, B composition constant (Stream 4)	Step
2	B composition, A/C ratio constant (Stream 4)	Step
3	D feed temperature (Stream 2)	Step
4	Reactor cooling water inlet temperature	Step
5	Condenser cooling water inlet temperature	Step
6	A feed loss (Stream 1)	Step
7	C header pressure loss–reduced availability (Stream 4)	Step
8	A, B, C feed composition (Stream 4)	Random variation
9	D feed temperature (Stream 2)	Random variation
10	C feed temperature (Stream 4)	Random variation
11	Reactor cooling water inlet temperature	Random variation
12	Condenser cooling water inlet temperature	Random variation
13	Reaction kinetics	Slow drift
14	Reactor cooling water valve	Sticking
15	Condenser cooling water valve	Sticking
16	Unknown	Unknown
17	Unknown	Unknown
18	Unknown	Unknown
19	Unknown	Unknown
20	Unknown	Unknown
21	The valve for Stream 4 was fixed at the steady state position	Constant position

Table 3: The testing average FDR and the training and testing time.

Window setting	Testing average diagnostic accuracy (%)	Training time (min)	Testing time for one sample (ms)
[15, 30]	56.3	42.1	0.7
[20, 45]	68.7	38.3	0.5
[13, 30, 42]	72.4	70.6	1.3
[18, 36, 45]	75.6	61.5	1.1
[16, 27, 35, 42]	74.8	79.7	1.4
[18, 25, 36, 47]	75.9	83.4	1.6

Table 4: Confusion matrix.

	Prediction value 1	Prediction value 0
Actual value 1	TP	FN
Actual value 0	FP	TN

learning. And the weights in WCForest can improve the robustness and sparsity of the model.

WCForest has the best performance in these five methods, with an average FDR of 84.13%, which is about 62.18% higher than AdaBoost. The FDR of fault 6, 7, and 21 increased by 100%, the highest improvement among all 21 faults, and the improvements of more than 70% faults exceed 50%. For RF and XGBoost, their diagnostic rates for all faults are similar. The average FDR increased by 24.21% and 15.77%, respectively. Compared with gcForest, the performance of WCForest is improved by nearly 2%. The FDR for fault 2, 6, 7, and 21 is 100%, which means there are no false alarms and missing alarms. Furthermore, the FDR for 11 faults exceed 90% and the FDR for 6 faults exceed 95%, which is an important achievement.

A performance comparison of the five methods is shown in Figure 7. Obviously, WCForest and corset outperform RF, XGBoost, and AdaBoost. Compared with gcForest, the performance of WCForest is slightly improved.

To further demonstrate the validity of WCForest for the TE process, the FPR is shown in Table 6. In addition, RF, XGBoost, Adaboost, and gcForest are compared.

In Table 5, WCForest has an average FPR of 2.45%, a 27.58% decrease compared to AdaBoost and a 0.37% decrease compared to gcForest. The FPRs of fault 1, 2, 6, 7, 8, 17, 18, and 21 are zero, which is of great significance in industry. In addition, the FAR of a half of faults is reduced by more than 40% compared to AdaBoost. Figure 8 shows the detailed comparison results.

In order to examine the performance of WCForest, we compare it with methods listed in Table 7, which shows that our fault diagnosis model has a better performance than the others. Except for faults 3 and 15, the other 19 faults have a diagnostic rate of more than 50%, especially faults 3 and 9 are 20% better than the other models (except literature [30]). Compared with a DBN-based fault diagnosis model proposed in [30], the FDRs of the 21 faults have no much difference, so the average FDR is only 1.23% higher. It should be noted that fault 15 has a relatively poor diagnostic effect and needs to be further investigated.

To thoroughly evaluate the quality of the proposed method, F_1 score is selected as the evaluation indicator. It is a classical index in machine learning field [38], which analyses classifiers based on recall and precision and is calculated by their harmonic means.

Table 5: FDR obtained by RF, XGBoost, AdaBoost, gcForest, and WCForest.

FDR (%)	RF	XGBoost	AdaBoost	gcForest	WCForest	Percentage increase of WCForest compared to RF
Fault 1	100	100	1.83	99.42	99.17	−0.83
Fault 2	100	100	97.83	98.17	100	0
Fault 3	15.17	16.5	0	41.59	42.33	27.16
Fault 4	99.17	98.83	0	98.26	98.83	−0.34
Fault 5	28.83	99.67	50.67	83.74	90.97	62.14
Fault 6	100	100	0	100	100	0
Fault 7	100	100	0	100	100	0
Fault 8	62.50	69.17	52.67	93.36	96	33.50
Fault 9	5.83	5	0	53.18	51.50	45.67
Fault 10	30.50	40.67	9.17	78.21	75.83	45.33
Fault 11	61.33	67.67	49.33	83.45	82.00	20.67
Fault 12	80.00	77.83	29	87.29	93.50	13.50
Fault 13	45.67	50.33	26.67	89.21	94.83	49.16
Fault 14	94.83	96.67	61.33	97.03	96.83	2.00
Fault 15	0.83	3.67	0	6.48	7.17	6.34
Fault 16	35.00	49.67	0	72.37	78.00	43.00
Fault 17	87.17	87.83	42.17	94.00	94.50	7.33
Fault 18	99.33	99.83	0.5	90.84	99.67	0.34
Fault 19	50.50	58	2.33	85.32	89.50	39.00
Fault 20	61.00	67.83	37.5	76.05	76.00	15.00
Fault 21	0.67	46.33	0	98.87	100	99.33
Average	59.92	68.36	21.95	82.23	84.13	24.21

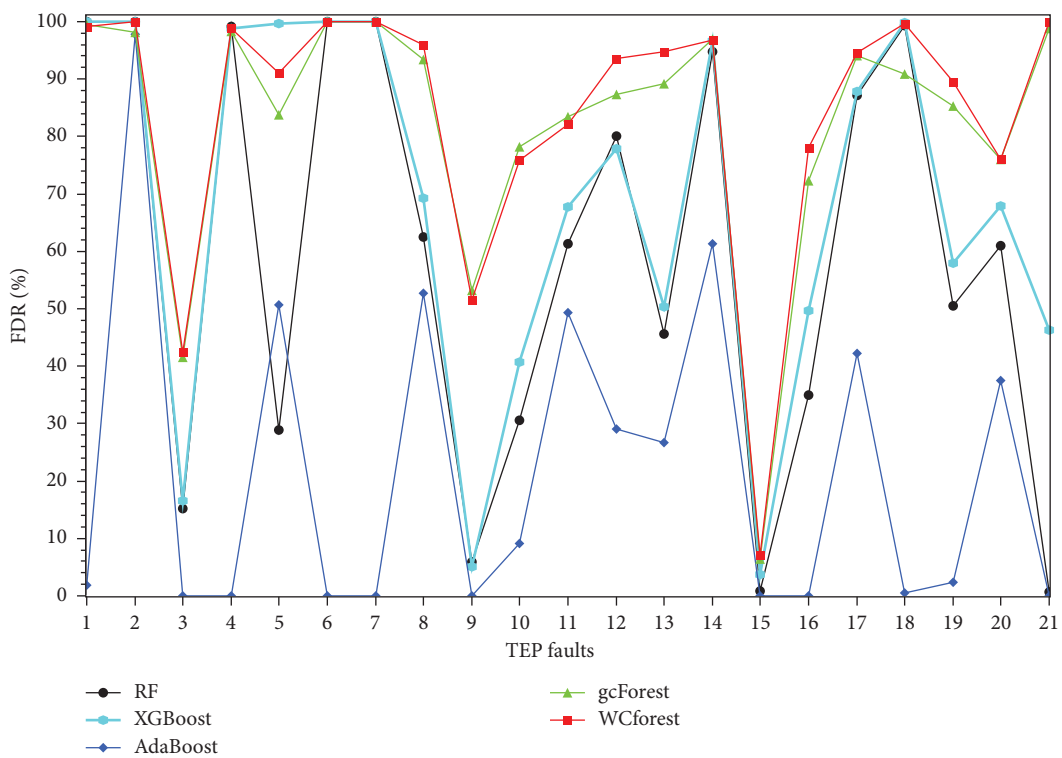

Figure 7: Performance of FDR on the TE process.

From the general confusion matrix in Table 3, the formulas for calculating recall and precision are as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{9}$$

Thus, the F_1 score can be calculated as follows:

$$F_1 = \frac{2 \times \text{Rcell} \times \text{Precision}}{\text{Rcell} + \text{Precision}}. \tag{10}$$

The F_1 score of WCForest is shown in Table 8, reflecting the diagnostic ability of the model. The values of recall and precision almost achieve 100% on faults 1, 2, 6, 7, and 21,

Table 6: FAR obtained by RF, XGBoost, AdaBoost, gcForest, and WCForest.

FPR (%)	RF	XGBoost	AdaBoost	gcForest	WCForest
Fault 1	0	0	0	0	0
Fault 2	0	0	0	0.5	0
Fault 3	47.6	62.3	83.8	9.7	10.7
Fault 4	0.3	0	0.2	0	0
Fault 5	41.2	0.2	42	0.3	2.4
Fault 6	0	0	0	0	0
Fault 7	0	0	0	0	0
Fault 8	0	0	1.8	0.3	0
Fault 9	42.3	51.8	81.7	21.3	16.3
Fault 10	18.7	21.5	62	4.6	4.1
Fault 11	10.5	11.3	38	1.3	1.6
Fault 12	0	0	1	0.3	0.4
Fault 13	0	0	0	0	0
Fault 14	0	0	0	0	0
Fault 15	50.3	54.8	74.3	12.1	7.2
Fault 16	3.2	4.3	59.2	2.0	3.9
Fault 17	1.0	1.2	10.2	0.1	0
Fault 18	0	0	0	0.1	0
Fault 19	34.0	28.7	87.3	3.8	2.3
Fault 20	13.1	15.5	45.7	2.9	2.6
Fault 21	10.0	9.2	40.8	0	0
Average	12.96	12.42	30.03	2.82	2.45

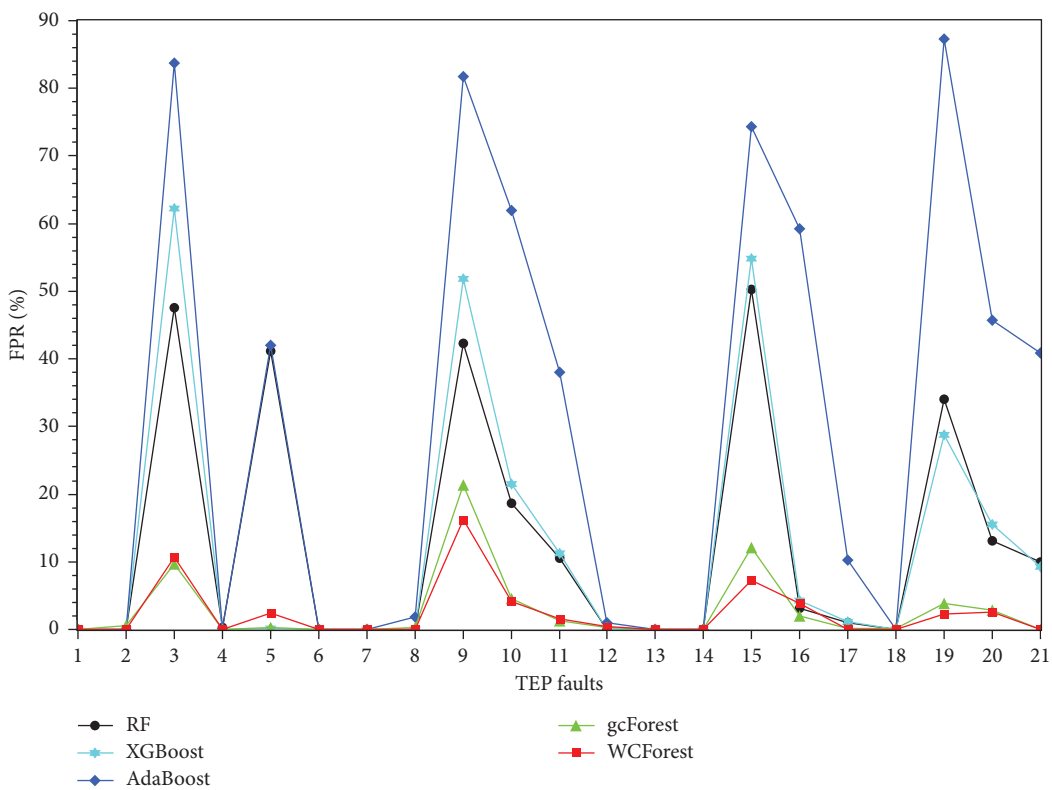

Figure 8: Performance of FPR on the TE process.

which means the great performance of true positive rate and false positive values. Finally, Figure 9 shows the recall and precision of WCForest which indicates the proposed method has good performance.

4.4. Hierarchical Representation Learning Visualization. In order to understand the characterization process of WCForest and the hierarchical representation of its learning process, it is very important to observe the diagnostic results

Table 7: Performance comparison of different faults diagnosis methods.

FDR (%)	(a)	(b)	(c)	(d)	(e)	(f)
Fault 1	99.87	96.37	100	100	99.88	99.17
Fault 2	97.87	97.62	99	99	99.13	100
Fault 3	2.37	20.62	6	95	6.13	42.33
Fault 4	100	82.75	100	98	100	98.83
Fault 5	99.87	96	100	86	99.88	90.97
Fault 6	99.5	100	100	100	100	100
Fault 7	100	100	100	100	100	100
Fault 8	96.62	96.87	99	78	97.00	96
Fault 9	3.37	12.12	3	57	5.75	51.50
Fault 10	82.25	88.25	84	98	92.13	75.83
Fault 11	64.75	73.5	82	87	74.00	82.00
Fault 12	99	93.62	100	85	99.75	93.50
Fault 13	95	72.25	95	88	95.63	94.83
Fault 14	100	95.87	100	87	100	96.83
Fault 15	9.75	21.12	17	0	10.25	7.17
Fault 16	81.62	78.12	89	0	90.00	78.00
Fault 17	84.87	80.25	96	100	97.13	94.50
Fault 18	89.5	86.37	90	98	90.25	99.67
Fault 19	76.12	96.12	52	93	92.00	89.50
Fault 20	66.37	86.75	88	93	85.25	76.00
Average	77.44	78.73	80	82.1	80.53	83.33

Note: (a) optimized variable selection-based PCA [34]; (b) supervised local multilayer perceptron [35]; (c) Bayesian method [36]; (d) DBN-based model [30]; (e) residual subspace associated with PCA [37]; (f) WCForest-based model (proposed in this paper).

Table 8: The F_1 score of 22 states on WCForest.

	Precision (%)	Recall (%)	F_1 score
Fault 0	44.46	40.66	42.47
Fault 1	100	99.17	99.58
Fault 2	99.16	100	99.57
Fault 3	41.78	42.33	42.05
Fault 4	96.19	98.83	97.49
Fault 5	96.88	90.97	93.83
Fault 6	100	100	100
Fault 7	100	100	100
Fault 8	95.05	96	95.52
Fault 9	32.12	51.50	39.56
Fault 10	68.84	75.83	72.16
Fault 11	84.10	82.00	83.04
Fault 12	88.63	93.50	91.00
Fault 13	98.61	94.83	96.69
Fault 14	97.81	96.83	97.32
Fault 15	16.73	7.17	10.04
Fault 16	73.82	78.00	75.85
Fault 17	94.19	94.50	94.34
Fault 18	97.77	99.67	98.71
Fault 19	80.03	89.50	84.50
Fault 20	77.29	76.00	76.64
Fault 21	98.68	100	99.34
Average	80.56	82.15	81.35

of each layer intuitively. As features learned are high dimensional, the diagnostic results of each layer are difficult to be visualized. To address this problem, we use t-distributed stochastic neighbour embedding (t-SNE) [39] as a tool to visualize the hierarchical representation learning process of the WCForest model.

The t-SNE method is a variant of stochastic neighbour embedding (SNE) [40]. It uses symmetric SNE to replace conditional probability with joint probability between data points in high-dimensional space and low-dimensional space. Meanwhile, Gauss probability distribution is used in high-dimensional space and t distribution with 1 freedom degree is used in low-dimensional space, which solves the problem of data point congestion in SNE. Therefore, t-SNE can better express the complex nonlinear relationship between high-dimensional data in the process of dimension reduction.

We use t-SNE method to embed high-dimensional features of each layer into two-dimensional or three-dimensional space, which can be visualized in scatter plots. The feature learning process can be easily visualized by using the 2D or 3D maps corresponding to each layer. Through experiments, we found that 3D maps are not suitable for visualization of WCForest-based fault diagnosis models. Therefore, the high-dimensional output features of each layer are embedded in the 2D map and then plotted in the subgraph of Figure 10.

600 samples of 22 classes (one normal and 21 faults) were randomly selected from testing set for visualization. The size of input data is 600×52, and then these 52-dimensional vectors are transformed into 600 vectors of 2-dimensional by using the t-SNE method. In each subgraph of Figure 10, these points are marked with their actual class labels, "Normal" with "0" and "Fault 01" with "1," and so on. In addition, in order to distinguish clusters for viewing, different colors are used to represent their classes. The output of each layer is converted to a vector of 2-dimensional by using t-SNE, so that it can be visualized in 2D (Figures 10(b)–10(h)).

As shown in Figure 10(a), the raw process data samples of all classes are mixed. Distribution of feature samples of

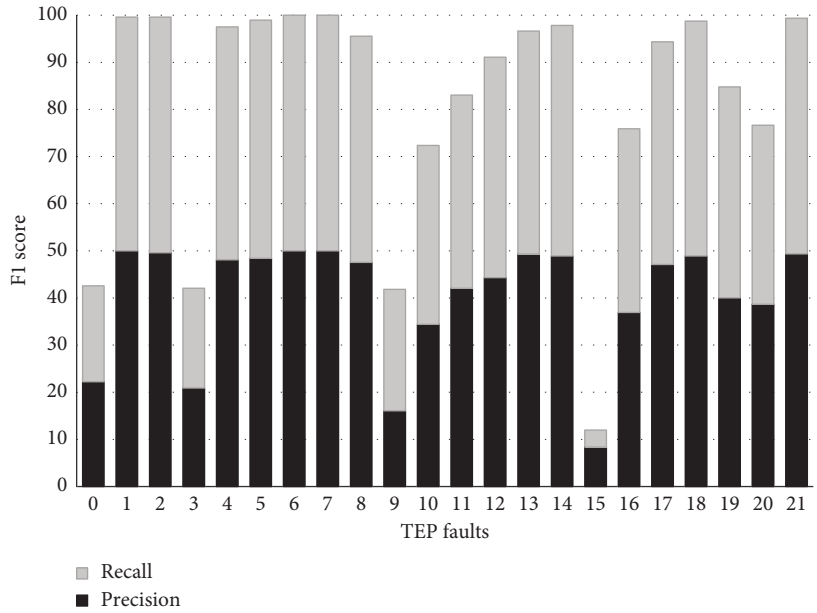

Figure 9: Precision and recall of F_1 score on the TE process.

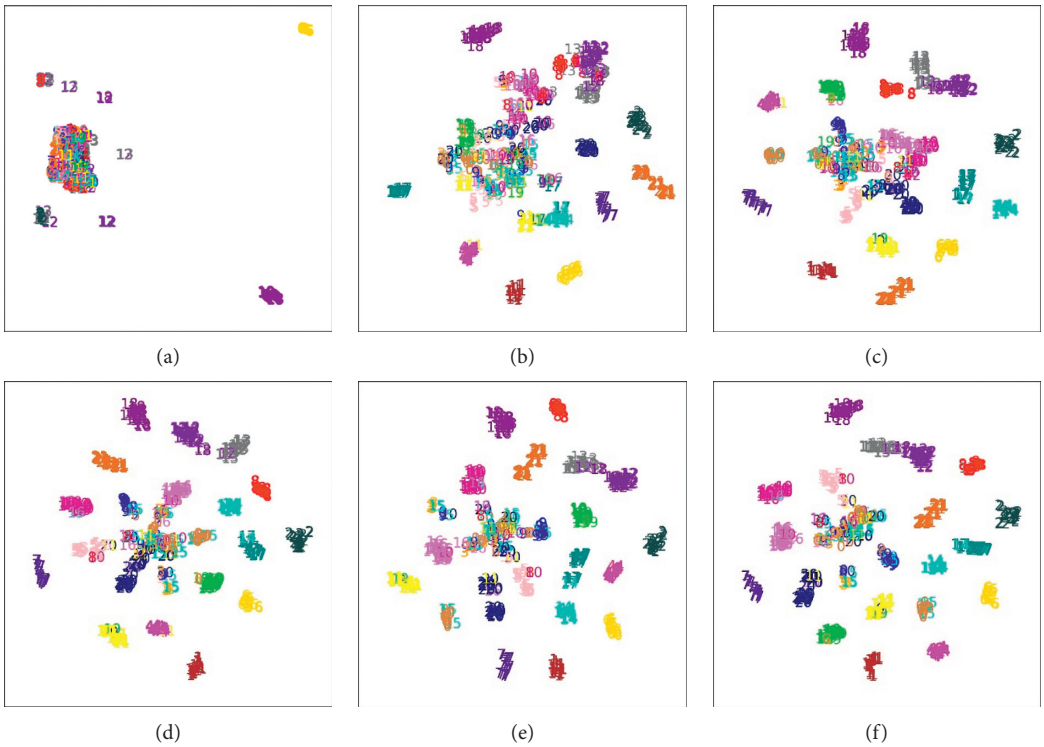

Figure 10: Continued.

(g)

Figure 10: WCForest model visualization using t-SNE. (a) Raw simples. (b) t-SNE of the output of level-1. (c) t-SNE of the output of level-2. (d) t-SNE of the output of level-3. (e) t-SNE of the output of level-4. (f) t-SNE of the output of level-5. (g) t-SNE of the output of level-6.

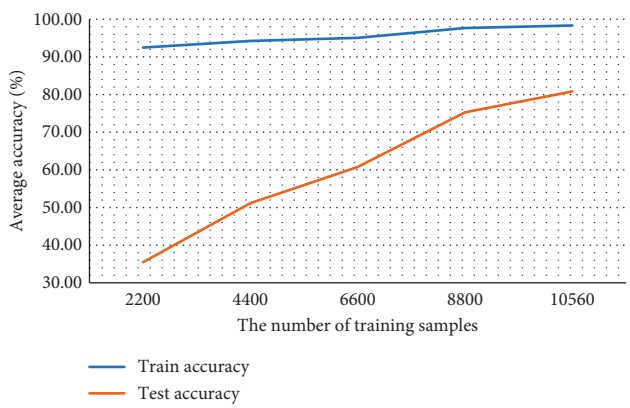

Figure 11: Comparison of average accuracy on training data with different quantity.

multigrained scanning is shown in Figure 10(b). Then, by learning the representation of weighted cascaded forests at different levels, we can find that the samples are clustered gradually through class labels in t-SNE mapping (see Figures 10(c)–10(f)). This indicates that the nonlinear expression ability of the WCForest model increases with the increase of the number of layers. WCForest maps the indivisible features to the nonlinear separable space by deepening the number of layers in the cascade forest. It also verifies the rationality of the WCForest model to deepen the design of the forest layer. Finally, these subgraphs strongly prove that the WCForest model is effective for fault diagnosis tasks.

4.5. Model Performance. Is the number of training samples crucial for obtaining good diagnosis performance? To answer the question, we compared the average accuracy for training and testing on different training datasets with 10560, 8800, 6600, 4400, and 2200 training samples and 600 testing samples, and demonstrated the result in Figure 11. The figure shows that the test accuracy of the WCForest model is greatly affected by the number of training samples, especially in the early stage. Although the late rise is relatively low, it has continued to increase, while the train accuracy is not greatly affected by the number of training samples.

5. Conclusions

In this paper, an improved deep forest model, WCForest, is proposed for fault diagnosis of chemical processes to improve accuracy, reduce false alarm rate, and process high-dimensional and nonlinear data. The main performance is that, without increasing the computational complexity, k-fold cross-validation is used to calculate the weight of each forest in the cascade structure in order to boost the good performance of forests and weaken the bad ones, so as to improve the overall performance of the cascade random forest.

To show the performance of the proposed model, RF, XGBoost, AdaBoost, gcForest, and WCForest were applied to the benchmark TE process, containing 16 known faults and 5 unknown faults for testing. The WCForest model predicts an average FDR of 84.13% and a FPR of 2.45%, with a high accuracy and a low false positive rate, which is comparable to the average diagnostic rate reported in other literatures. To provide more information about the performance of the model, the F_1 score is also chosen as an evaluation measurement for the integrity and purity of the classifier. Our work shows the validity and efficiency of WCForest, which can predict fault diagnosis in the TE process and can provide a reference for other chemical processes. In addition, most data samples are clearly and correctly clustered by WCForest in the t-SNE map.

Because of its excellent fault diagnosis rate and false positive rate, this method has industrial prospects. The data-driven fault diagnosis methods depend on the collection of a large amount of various process malfunction samples. Inevitably, our WCForest-based fault diagnosis model suffers from the same drawback. In the near future, research also will be focused on fault diagnosis with limited number of fault samples available.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The paper was supported by grants from the National Natural Science Foundation of China (NSFC) (61562054).

References

[1] S. Rajaraman, J. Hahn, and M. S. Mannan, "A methodology for fault detection, isolation, and identification for nonlinear processes with parametric uncertainties," *Industrial & Engineering Chemistry Research*, vol. 43, no. 21, pp. 6774–6786, 2004.

[2] H. Sun, S. Zhang, C. Zhao, and F. Gao, "A sparse reconstruction strategy for online fault diagnosis in nonstationary processes with no a priori fault information," *Industrial & Engineering Chemistry Research*, vol. 56, no. 24, 2017.

[3] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[4] Q. Jiang, X. Yan, and B. Huang, "Review and perspectives of data-driven distributed monitoring for industrial plant-wide processes," *Industrial & Engineering Chemistry Research*, vol. 58, no. 29, pp. 12899–12912, 2019.

[5] M. Alauddin, F. Khan, I. Syed, and S. Ahmed, "A bibliometric review and analysis of data-driven fault detection and diagnosis methods for process systems," *Industrial and Engineering Chemistry Research*, vol. 57, pp. 10719–10735, 2018.

[6] M. Misra, H. H. Yue, S. J. Qin, and C. Ling, "Multivariate process monitoring and fault diagnosis by multi-scale PCA," *Computers & Chemical Engineering*, vol. 26, no. 9, pp. 1281–1293, 2002.

[7] Q. Jiang, X. Yan, and J. Li, "PCA-ICA integrated with bayesian method for non-gaussian fault diagnosis," *Industrial & Engineering Chemistry Research*, vol. 55, no. 17, pp. 4979–4986, 2016.

[8] Y. Zhang and C. Ma, "Fault diagnosis of nonlinear processes using multiscale KPCA and multiscale KPLS," *Chemical Engineering Science*, vol. 66, no. 1, pp. 64–72, 2011.

[9] S. Yin, X. Zhu, and O. Kaynak, "Improved PLS focused on key-performance-indicator-related fault diagnosis," *IEEE Transactions on Industrial Electronics*, vol. 62, pp. 1651–1658, 2015.

[10] J.-M. Lee, J. Qin, and I.-B. Lee, "Fault detection and diagnosis of multivariate process based on modified independent component analysis," *AIChE Journal*, vol. 52, no. 10, pp. 3501–3514, 2006.

[11] Y. Zhang and C. Ma, "Decentralized fault diagnosis using multiblock kernel independent component analysis," *Chemical Engineering Research and Design*, vol. 90, no. 5, pp. 667–676, 2012.

[12] J. Yu, "Localized Fisher discriminant analysis based complex chemical process monitoring," *AIChE Journal*, vol. 57, no. 7, pp. 1817–1828, 2011.

[13] Z.-B. Zhu and Z.-H. Song, "Fault diagnosis based on imbalance modified kernel Fisher discriminant analysis," *Chemical Engineering Research and Design*, vol. 88, no. 8, pp. 936–951, 2010.

[14] Z. Chai and C. Zhao, "Enhanced random forest with concurrent analysis of static and dynamic nodes for industrial fault classification," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 54–66, 2019.

[15] Q. Jiang, F. Gao, H. Yi, and X. Yan, "Multivariate statistical monitoring of key operation units of batch processes based on time-slice CCA," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 1368–1375, 2019.

[16] W. Yu and C. Zhao, "Sparse exponential discriminant analysis and its application to fault diagnosis," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5931–5940, 2018.

[17] C. Zhao, W. Wang, and F. Gao, "Probabilistic fault diagnosis based on Monte Carlo and nested-loop Fisher discriminant analysis for industrial processes," *Industrial & Engineering Chemistry Research*, vol. 55, no. 50, pp. 12896–129081, 2016.

[18] J. Wang, B. Zhong, and J. Zhou, "Quality-relevant fault monitoring based on locality preserving partial least squares statistical models," *Industrial & Engineering Chemistry Research*, vol. 56, no. 24, pp. 7009–7020, 2017.

[19] M. Galiaskarov, V. V. Kurkina, and L. A. Rusinov, "Online diagnostics of time-varying nonlinear chemical processes using moving window kernel principal component analysis and Fisher discriminant analysis: diagnostics of time-varying nonlinear chemical processes," *Journal of Chemometrics*, vol. 31, no. 8, Article ID e2866, 2017.

[20] S. He, X. Liu, Y. Wang et al., "An effective fault diagnosis approach based on optimal weighted least squares support vector machine," *The Canadian Journal of Chemical Engineering*, vol. 95, no. 12, pp. 2357–2366, 2017.

[21] Y. Xu, S.-Q. Shen, Y.-L. He, and Q.-X. Zhu, "A novel hybrid method integrating ICA-PCA with relevant vector machine for multivariate process monitoring," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 4, pp. 1780–1787, 2019.

[22] H. Chen, B. Jiang, N. Lu, and Z. Mao, "Deep PCA based real-time incipient fault detection and diagnosis methodology for electrical drive in high-speed trains," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 4819–4830, 2018.

[23] M. Basseville and I. Nikiforov, *Detection of Abrupt Change Theory and Application*, Vol. 15, Prentice Hall, Upper Saddle River, NJ, USA, 1993.

[24] J. Chen and A. K. Gupta, *Parametric Statistical Change Point Analysis: with Applications to Genetics, Medicine, and Finance*, Springer, Cambridge, MA, USA, 2012.

[25] L. Wen, X. Li, L. Gao, and Y. Zhang, "A new convolutional neural network based data-driven fault diagnosis method," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5990–5998, 2017.

[26] G. Jiang, H. He, J. Yan, and P. Xie, "Multiscale convolutional neural networks for fault diagnosis of wind turbine gearbox," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 4, pp. 3196–3207, 2018.

[27] Z. Chen, S. Deng, X. Chen, C. Li, R.-V. Sanchez, and H. Qin, "Deep neural networks-based rolling bearing fault diagnosis," *Microelectronics Reliability*, vol. 75, pp. 327–333, 2017.

[28] K. B. Lee, S. Cheon, and C. O. Kim, "A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes," *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, pp. 135–142, 2017.

[29] D. Xie and L. Bai, "A hierarchical deep neural network for fault diagnosis on Tennessee-Eastman process," in *Proceedings of the IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 745–748, IEEE, Miami, FL, USA, December 2015.

[30] Z. Zhang and J. Zhao, "A deep belief network based fault diagnosis model for complex chemical processes," *Computers & Chemical Engineering*, vol. 107, pp. 395–407, 2017.
[31] H. Wu and J. Zhao, "Deep convolutional neural network model based chemical process fault diagnosis," *Computers & Chemical Engineering*, vol. 115, pp. 185–197, 2018.
[32] Z.-H. Zhou and Ji Feng, "Deep forest: towards an alternative to deep neural networks," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Melbourne, Australia, August 2017.
[33] J. J. Downs and E. F. Vogel, "A plant-wide industrial process control problem," *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.
[34] K. Ghosh, M. Ramteke, and R. Srinivasan, "Optimal variable selection for effective statistical process monitoring," *Computers & Chemical Engineering*, vol. 60, pp. 260–276, 2014.
[35] M. Ali Ayubi Rad and M. Javad Yazdanpanah, "Designing supervised local neural network classifiers based on em clustering for fault diagnosis of Tennessee Eastman process," *Chemometrics and Intelligent Laboratory Systems*, vol. 146, pp. 149–157, 2015.
[36] Q. Jiang and B. Huang, "Distributed monitoring for large-scale processes based on multivariate statistical analysis and bayesian method," *Journal of Process Control*, vol. 46, pp. 75–83, 2016.
[37] C. Zhang, X. Gao, T. Xu, Li Yuan, and Y. Pang, "Fault detection and diagnosis strategy based on a weighted and combined index in the residual subspace associated with PCA: fault detection and diagnosis based on PCA-RS2 in residual subspace," *Journal of Chemometrics*, vol. 32, no. 11, Article ID e2981, 2018.
[38] G. Ning, S. Nakajima, and M. Pantel, "Online diagnosis of accidental faults for real-time embedded systems using a hidden markov model," *Simulation*, vol. 91, no. 10, pp. 851–868, 2015.
[39] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, November 2018.
[40] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermayer, Eds., vol. 15, pp. 857–864, MIT Press, Cambridge, MA, USA, 2003.