Research Article

Novel Model Based on Stacked Autoencoders with Sample-Wise Strategy for Fault Diagnosis

Diehao Kong and Xuefeng Yan

Key Laboratory of Advanced Control and Optimization for Chemical Processes of Ministry of Education, East China University of Science and Technology, Shanghai 200237, China

Correspondence should be addressed to Xuefeng Yan; xfyan@ecust.edu.cn

Received 22 February 2019; Revised 26 April 2019; Accepted 20 May 2019; Published 4 June 2019

Academic Editor: Alessandro Lo Schiavo

Copyright © 2019 Diehao Kong and Xuefeng Yan. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Autoencoders are used for fault diagnosis in chemical engineering. To improve their performance, experts have paid close attention to regularized strategies and the creation of new and effective cost functions. However, existing methods are modified on the basis of only one model. This study provides a new perspective for strengthening the fault diagnosis model, which attempts to gain useful information from a model (teacher model) and applies it to a new model (student model). It pretrains the teacher model by fitting ground truth labels and then uses a sample-wise strategy to transfer knowledge from the teacher model. Finally, the knowledge and the ground truth labels are used to train the student model that is identical to the teacher model in terms of structure. The current student model is then used as the teacher of next student model. After step-by-step teacher-student reconfiguration and training, the optimal model is selected for fault diagnosis. Besides, knowledge distillation is applied in training procedures. The proposed method is applied to several benchmarked problems to prove its effectiveness.

1. Introduction

As a result of the advancement of process control systems, modern industries have switched to automation. With advantages such as low cost, high efficiency, and improved safety, automated control systems have become increasingly popular worldwide, especially in power plants, the aerospace industry, and chemical plants [1]. However, in abnormal situations, these systems are unable to cope on their own and thus require the intervention of human operators. Fault detection and diagnosis (FDD) has been developed to determine where problems lie and prevent machine damage or accidents.

In practical chemical processes, FDD comprises two vital procedures that influence the life of devices and operators. In fault diagnosis, a model is built to distinguish fault types accurately. Corresponding measures are taken to overcome these faults when necessary. In the early application of FDD, mathematical-based models elicited the most attention and thus developed rapidly. However, model-based FDD methods require priori knowledge about chemical processes before application. Therefore, the development of this kind

of method was limited. With the support of related technology, voluminous data can be collected easily. Although the data collected are high-dimensional, they remain useful for some FDD methods. This approach is called the data-driven method. Basic data-driven methods include principal component analysis (PCA) [2, 3], canonical correlation analysis [4], independent component analysis, and Fisher discriminant analysis [5, 6]. These data-driven methods greatly contribute to FDD, but they are not suitable for certain situations. For example, all the aforementioned methods fail to detect faults exactly in nonlinear process monitoring and fault diagnosis.

In the early 1990s, neural networks (NNs) were merged into FDD as a feature extraction tool [7]. The NNs at that time were shallow and narrow. These NNs were also prone to overfitting once they became deep and wide. Hence, NNs were never widely used for FDD during this period. Since Hinton proposed pretraining and fine tuning [8], deep learning (DL) has received much attention. As a result, DL has undergone unprecedented application and development. DL has been applied in computer vision, speech recognition,

text processing, medical, finance advertising, and other fields. Stacks of restricted Boltzmann machines (RBMs), referred to as a deep belief network (DBN), have been found to behave better than other traditional data-driven methods (e.g., PCA) in data dimension reduction [9]. As pretraining helps deep neural networks (DNNs) jump out of the local minimum, the fact that DBN outperforms PCA may be easily disregarded. As for FDD, RBMs and DNNs have been applied to some specific processes to initialize offline fault detection models and detect online time-related samples automatically. Ren et al. used multiple DBN models to correspond to each working condition and set an adaptive threshold to detect faulty conditions [10]. Zhang et al. considered extracting features in spatial and temporal domains [11]. They built two subnetworks by mutual information and trained them with a backpropagation algorithm. In their model, each class has its own network. The result is a considerable improvement in model performance. Sun et al. trained a DBN with normal samples in offline modeling. While online, this DBN can discriminate faults according to reconstruction errors [9]. The convolutional neural network (CNN) is widely applied in image processing based on convolution calculation. Roy et al. used deep CNN with layer-wise training to recognize handwritten Bangla isolated compound character [12]. In addition, RMSProp algorithm is applied to achieve faster convergence. CNN also can be employed in FDD. Lee et al. applied CNN to multivariate time-series data and built a connection between the output of the CNN and the structural meaning of data to fault classification [13]. Wen et al. proposed a new method on the basis of CNN [14]. Instead of using time-domain data, they converted raw data into image data. After such transformation, the CNN can easily extract the features of faults. An autoencoder (AE) and a stacked autoencoder (SAE) are trained to extract valuable features to determine whether a machine is faulty. Lee et al. employed a stacked denoising AE for fault classification [15]. Compared with models with SAEs, their model adds noise to the samples before training to prevent overfitting. In addition, their model realizes improved generalization capability. Zhang et al. considered self-correlation in data. They first extended the data matrix by correlation analysis [16]. Then, they applied a deep SAE to extract the necessary features. Different from these existing studies, the current work highlights the results obtained from a trained DL model. This work takes advantage of learned knowledge and reuses it to train an identical model.

Much research has explored knowledge distillation (KD). Yim et al. calculated the inner products between two identical DNN models and created a new loss function using L2 norm to improve classification performance in image processing [17]. Similarly, Furlanello et al. used two identical models. In addition, they explored two distillation objectives: confidence weighted by Teacher Max and dark knowledge with permuted predictions [18]. They found that KD is suitable for almost all machine learning (ML) models. Xu et al. focused on one ML model and, trained it for several times [19]. After the first training, they softened the correct targets and reset the mistaken targets to 1. Bae et al. used residual networks as basic models to transfer knowledge in a teacher-student framework

[20]. They proposed a layer-wise hint training method and improved the recognition results on several typical datasets.

The rest of the paper is organized as follows. Section 2 provides an overview of SAE and KD. Section 3 introduces the novel model based on SAE with the sample-wise strategy (SAE-SWS). Section 4 discusses the application of the method to several industrial benchmarked problems. Finally, Section 5 summarizes the methodology.

2. Preliminaries

2.1. Stacked Autoencoders. SAE refers to a stack of autoencoders. Nevertheless, SAEs cannot be formulated by connecting AEs directly. An SAE's training process includes two procedures: unsupervised pretraining and fine tuning. In unsupervised pretraining, the features of a dataset are extracted. As shown in Figure 1(a), the input dataset is assumed to be $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^T \in \mathbf{R}^{n \times u}$, n is the number of samples, and u is the number of observed variables. $\mathbf{x}_i \in \mathbf{R}^{u \times 1}$ ($i = 1, 2, \dots, n$). $\mathbf{x}_i$ undergoes encoding and decoding and can be presented as

$$\mathbf{d}_i = f_a(\mathbf{W}_a \mathbf{x}_i + \mathbf{b}_a), \quad (1)$$

$$\hat{\mathbf{x}}_i = f_s(\mathbf{W}_s \mathbf{d}_i + \mathbf{b}_s), \quad (2)$$

where $\mathbf{d}_i \in \mathbf{R}^{v \times 1}$ is the hidden layer feature of $\mathbf{x}_i$. $\mathbf{W}_a \in \mathbf{R}^{v \times u}$ and $\mathbf{b}_a \in \mathbf{R}^{v \times 1}$ are the parameters of the encoding stage, where v is the hidden number of nodes. $\mathbf{W}_s \in \mathbf{R}^{u \times v}$ and $\mathbf{b}_s \in \mathbf{R}^{u \times 1}$ are the parameters of the decoding stage. They can be expressed together by $\boldsymbol{\theta}$. $f_a(\cdot)$ and $f_s(\cdot)$ denote the activations of the encoding and decoding stages, respectively. Several activations are available, and rectified linear unit is employed here. $\hat{\mathbf{x}}_i \in \mathbf{R}^{u \times 1}$ is the predictive estimate of the input $\mathbf{x}_i$. The process, including (1) and (2), is trained as

$$\min_{\boldsymbol{\theta}} J^{(1)}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2. \quad (3)$$

Feature $\mathbf{d}_i$ is used to dig deep for feature $\mathbf{a}_i$, which is shown in Figure 1(b). Figures 1(a) and 1(b) belong to the feature extraction stage.

In the fine-tuning stage, the whole network is trained for classification or regression. The structure is shown in Figure 1(c). Regarding classification, the SAE assumes the label dataset $\mathbf{L} = [\mathbf{l}_1, \mathbf{l}_2, \dots, \mathbf{l}_n]^T \in \mathbf{R}^{n \times c}$ (c is the number of classes) corresponding to $\mathbf{X}$, and it uses one hot encoding to build $\mathbf{l}_i$ ($i = 1, 2, \dots, n$). The parameters of the SAE are transferred from the previous pre-training, similar to that shown in the figure. Then a softmax layer is added with c nodes after the network. $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n]^T \in \mathbf{R}^{n \times c}$ represents the output. $\mathbf{y}_i$ ($i = 1, 2, \dots, n$) denotes the SAE classification probabilities of input $\mathbf{x}_i$ ($i = 1, 2, \dots, n$), which satisfy $y_{ij} \in [0, 1]$ ($i = 1, 2, \dots, n, j = 1, 2, \dots, c$). The SAE is then trained as

$$\min_{\boldsymbol{\theta}} J^{(2)}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n \mathcal{L}(\mathbf{y}_i, \mathbf{l}_i), \quad (4)$$

where $\mathcal{L}(\mathbf{y}_i, \mathbf{l}_i) = -\sum_{j=1}^c [y_{ij} \ln l_{ij} + (1 - y_{ij}) \ln(1 - l_{ij})]$.

FIGURE 1: (a), (b): Layer-by-layer learning of AE; (c): structure of SAE.

2.2. Knowledge Distillation. KD means transferring “knowledge” between ML models. It was first proposed by Hinton to transfer knowledge learned by a cumbersome model to a much simpler one [21]. Before interpreting KD, the soft targets and hard targets must be understood. For example, for binary classification, label $[1, 0]$ is a hard target, whereas the label whose value for a certain dimension is not 1, such as $[0.8, 0.2]$, and is a soft target. The latter clearly contains more information. KD tries to soften hard targets to gain messages for classification.

KD is closely related to the “softmax” function. Softmax converts multicategory output values into relative probabilities, making them easy to understand and compare. It is usually connected after the ML model. For example, assume that the softmax layer input is $\mathbf{h} \in \mathbf{R}^{c \times 1}$, where c denotes the number of classes. The probability that $\mathbf{h}$ belongs to the i th class can be calculated by

$$p_i = \frac{\exp(h_i)}{\sum_{j=1}^c \exp(h_j)} \quad (i = 1, 2, \dots, c), \quad (5)$$

where i, j denote the category indexes. By contrast, if softmax in KD (softmax-KD) is applied, then the probability is calculated by

$$p_i = \frac{\exp(h_i/t)}{\sum_{j=1}^c \exp(h_j/t)} \quad (i = 1, 2, \dots, c), \quad (6)$$

where t denotes the temperature of softmax. A high t equates to soft targets. When t equals to 1, softmax-KD becomes original softmax. In this paper, t is set within the range of 2.5–4 during model training.

3. FDD Model Based on SAE with Sample-Wise Strategy

3.1. Model Construction with Sample-Wise Strategy. Two identical models, namely, a teacher model and a student

FIGURE 2: SAE used with the sample-wise strategy in the teacher model.

model, are used in the algorithm. Both models consist of SAEs for feature extraction and softmax-KD layers for classification. Following the variables used in Section 2.1, the teacher model is shown in Figure 2, where $x_i^{(T)} = x_i (i = 1, 2, \dots, n)$ and $q_i^{(T)} \in \mathbf{R}^{c \times 1}$ are the input and output of the teacher model T using temperature t , respectively.

$\mathbf{X}^{(T)} = [x_1^{(T)}, x_2^{(T)}, \dots, x_n^{(T)}]^T \in \mathbf{R}^{n \times u}$ represents the set of $x_i^{(T)}$. Given the label dataset $\mathbf{L}^{(T)} = [l_1^{(T)}, l_2^{(T)}, \dots, l_n^{(T)}]^T \in \mathbf{R}^{n \times c}$, where $l_i^{(T)} = l_i (i = 1, 2, \dots, n)$, then the teacher model

T is trained by minimizing the cross-entropy cost function, which can be presented by

$$\min_{\theta^{(T)}} J^{(T)}(\theta^{(T)}) = \frac{1}{n} \sum_{i=1}^n \mathcal{L}(q_i^{(T)}, l_i^{(T)}), \quad (7)$$

where $\theta^{(T)}$ denotes weights and biases in the teacher model T . $\mathcal{L}(q_i^{(T)}, l_i^{(T)})$ is calculated by

$$\begin{aligned} \mathcal{L}(q_i^{(T)}, l_i^{(T)}) \\ = - \sum_{j=1}^c [q_{ij}^{(T)} \ln l_{ij}^{(T)} + (1 - q_{ij}^{(T)}) \ln (1 - l_{ij}^{(T)})]. \end{aligned} \quad (8)$$

The SWS is conducted after training the teacher model T . Considering the i th sample $\mathbf{x}_i^{(T)}$, its predicted category c_{pred} and true category c_{true} are obtained according to $q_i^{(T)}$ and $l_i^{(T)}$, respectively. If $c_{pred} = c_{true}$, $\mathbf{x}_i^{(T)}$ is considered to have been sorted into the correct category. As a result, all the n samples are divided into two parts: the correct samples (CSs) and the mistaken samples (MSs). Assume m MSs. In terms of MSs, the teacher model does not extract the needed features accurately. Hence, the strategy SWS improves the weight of the MSs by adding rm MSs to the student model's dataset; that is, the input set of student model S_1 is $\mathbf{X}^{(S_1)} = [\mathbf{x}_1^{(S_1)}, \mathbf{x}_2^{(S_1)}, \dots, \mathbf{x}_{n+rm}^{(S_1)}]^T \in \mathbf{R}^{(n+rm) \times u}$. r is a multiple. That is, rm is r times the number of MSs. Then the label set of student model S_1 is $\mathbf{L}^{(S_1)} = [l_1^{(S_1)}, l_2^{(S_1)}, \dots, l_{n+rm}^{(S_1)}]^T \in \mathbf{R}^{(n+rm) \times c}$. Put the student model's input sample $\mathbf{x}_i^{(S_1)}$ ($i = 1, 2, \dots, n + rm$) into the teacher model; then $\mathbf{Q}^{(T-S_1)} = [q_1^{(T-S_1)}, q_2^{(T-S_1)}, \dots, q_{n+rm}^{(T-S_1)}] \in \mathbf{R}^{(n+rm) \times c}$ is obtained. In other words, $\mathbf{Q}^{(T-S_1)}$, which represents the distribution of the output of teacher model T , is obtained from teacher model T by SWS and then used for training S_1 . Thus, the student model S_1 can be trained with the modified loss function, which can be presented as

$$\begin{aligned} \min_{\theta^{(S_1)}} J^{(S_1)}(\theta^{(S_1)}) \\ = \frac{1}{n + rm} \sum_{i=1}^{n+rm} [\alpha \mathcal{L}(q_i^{(S_1)}, q_i^{(T-S_1)}) + \mathcal{L}(q_i^{(S_1)}, l_i^{(S_1)})], \end{aligned} \quad (9)$$

where $q_i^{(S_1)}$ denotes the softmax-KD output of S_1 using the temperature t . $q_i^{(T-S_1)}$ is called the knowledge between model T and S_1 using the temperature t . α denotes the weight of the cross-entropy between $q_i^{(S_1)}$ and $q_i^{(T-S_1)}$. The novel loss function is composed of two parts: cross-entropy between the student model's output $q_i^{(S_1)}$ and the knowledge $q_i^{(T-S_1)}$ and cross-entropy between the student model's output $q_i^{(S_1)}$ and the ground truth label $l_i^{(S_1)}$. $\alpha \mathcal{L}(q_i^{(S_1)}, q_i^{(T-S_1)})$ can be regarded as a regularizing part for boosting the student model's performance. The ground truth label information with MSs provides strong capability for further improving discriminative capability. The two parts make full use of

FIGURE 3: Entire training procedure with KD and SWS.

the information from the result of the teacher model and strengthen the MSs in the student model training.

The above procedures are contents of step 1 in Figure 3. Besides, the SWS can be applied continuously, as shown in Figure 3. S_i ($i = 1, 2, \dots, k$) denotes the i th student model. k denote the number of steps. Each model used in this paper is SAE with softmax-KD layer. For step 1, model T is trained. Then SAE-SWS employs the SWS to mine further knowledge. Finally, the further knowledge is applied to train model S_1 . After k steps, the optimal student model is selected according to the classification accuracy on several datasets.

3.2. Flow Chart of the Model for FDD. On the basis of the SWS above, the ML model for FDD is introduced. FDD models include two parts: offline modeling and online monitoring. The procedures of offline modeling and online monitoring are shown in Figure 4.

Offline modeling:

- (1) Dataset $\mathbf{X} \in \mathbf{R}^{n \times u}$ and label set $\mathbf{L} \in \mathbf{R}^{n \times c}$ are collected, where n refers to the number of samples, u denotes the number of variables, and c denotes the number of classes. Then $\mathbf{X}$ is divided into $\mathbf{X}^{train} \in \mathbf{R}^{n_1 \times u}$ for training and $\mathbf{X}^{test} \in \mathbf{R}^{n_2 \times u}$ for testing, where $n_1 + n_2 = n$. Correspondingly, $\mathbf{L}$ is divided into $\mathbf{L}^{train} \in \mathbf{R}^{n_1 \times c}$ for training and $\mathbf{L}^{test} \in \mathbf{R}^{n_2 \times c}$ for testing.
- (2) Every component of $\mathbf{X}^{train}$ is normalized to zero mean and unit variance, respectively. Then every component of $\mathbf{X}^{test}$ is normalized by using the mean value and variance of $\mathbf{X}^{train}$. $\bar{\mathbf{X}}^{train}$ and $\bar{\mathbf{X}}^{test}$ denote the normalized training and testing dataset, respectively.

FIGURE 4: Flow chart of offline modeling and online monitoring.

- (3) Letting $\mathbf{X}^{(T)} = \tilde{\mathbf{X}}^{train}$ and $\mathbf{L}^{(T)} = \mathbf{L}^{train}$, then the teacher model T , which consists of SAE and a softmax-KD layer, is trained by matching the distribution of ground truth labels. Letting $i = 1$, i denotes the index of steps.
- (4) Transfer knowledge and obtain the input set and label set for next student model by SWS.
- (5) Train the student model by matching the distribution of the teacher's output and ground truth labels.
- (6) If $i > k$, jump to (7); or let the current student becomes teacher of next student. And $i = i + 1$. Then go back to (4).
- (7) Use $\tilde{\mathbf{X}}^{test}$ and $\mathbf{L}^{test}$ to calculate the classification accuracy of all trained model. Select the optimal model with the highest accuracy.

Online monitoring:

- (1) The new sample $\mathbf{x}_{test} \in \mathbf{R}^{u \times 1}$ is normalized by using the mean value and variance of training data $\mathbf{X}^{train}$.
- (2) Integrate $\mathbf{x}_{test}$ into the optimal model and obtain its predicted class.

4. Application to Benchmark Problems

4.1. Continuous Stirred Tank Reactor Process

4.1.1. CSTR Description. The CSTR process is an anaerobic treatment that is widely used in industrial effluent disposal

[22]. As a result of the high research value of the CSTR, many modeling methods have been proposed to fill the gaps. In this experiment, the model proposed by Belevi is adopted [23]. The flow chart of the CSTR is shown in Figure 5. C_{AF} denotes feed concentration. Q_F denotes feed flow. T_F denotes feed temperature. Q_C denotes coolant flow. T_{CF} denotes the coolant inlet temperature. T denotes the temperature in the reactor. h denotes the reactor level. T_C denotes the coolant outlet temperature. C_A denotes the concentration of component A in the reactor. Q denotes the flow of component A. The variables listed above are the 10 operational variables of the CSTR. They are selected as the monitored variables and are recorded in detail. Table 1 lists and describes 11 faults we set for the experiment. Except for the listed variables, other operational variables are within the allowable range.

4.1.2. Parameter Setting and Results. White noise is added to every variable to simulate actual working conditions. Then the parameter is set [24]. We select 1,000 samples with 0.2 min as the simulation step for the experiment: of these samples, 70% comprise the training set, and 30% comprise the testing set. Normal classification algorithms, such as support vector machine (SVM), k-nearest neighbor (kNN), and basic SAE, are chosen as competitive algorithms. For the proposed algorithm, SAE-SWS, the performance of the teacher model of SAE-SWS (SAE-SWS-t) and the optimal student model of SAE-SWS (SAE-SWS-s) are listed. Besides, SAE-KD, which removes the SWS from SAE-SWS, is applied as comparison. It shares the same teacher with SAE-SWS and then goes

TABLE 1: Lists of faults and description.

Fault number	Fault description
1	C_{AF} drift
2	C_{AF}, Q_F drift
3	C_{AF}, Q_F, T_F drift
4	E/R abnormal
5	Q_F drift
6	Q_F step
7	T step
8	T_{CF} drift
9	T_F drift
10	T_F step
11	UA_C drift

FIGURE 5: CSTR and related variables.

through successive teacher-student training. SAE-KD-s is the optimal student model. In this experiment, SVM uses radial basis function (RBF) as kernel function. Penalty factor is set to 1. As for kNN, number of neighbors is 5. Euclidean distance is the metric. For SAE-based algorithms, their structures are [100-50-25]. Adam optimizer is used and learning rate is set to 0.001. The size of batch is 10. Epoch is set to 1000. In SAE-SWS, t is set to 2.5 during training according to the reference [22]. In equation (9), α is set to $[0.1, 0.2, \dots, 1.0]$ to obtain the optimal parameter. It is set to 0.4 based on the results shown in Figure 6. How to set r remains a problem in the SWS. Hence, an experiment is conducted to decide r and its result is shown in Figure 7. According to Figure 7, r is set to 2. Theoretically, k can be very large. However, limited by equipment, in this paper, k is set to 4. The classification accuracy of each model in SAE-SWS is shown in Figure 8. According to Figure 8, S_1 is chosen as the optimal student model.

The comparison results presented in Table 2 show that the SAE-SWS algorithm has the best performance among all algorithms. By comparing SAE-SWS-t with SAE-SWS-s, we find that the latter outperforms the former by 1.69%. This value implies that the proposed SWS improves feature

FIGURE 6: Relationship between parameter α and classification accuracy for the CSTR. FIGURE 7: Relationship between parameter r and classification accuracy for the CSTR.

TABLE 2: Classification accuracy of some algorithms for the CSTR.

Algorithms	Accuracy (%)
SVM	64.05
kNN	78.61
SAE	81.30
SAE-SWS-t	81.45
SAE-SWS-s	83.14
SAE-KD-s (without SWS)	82.11

extraction capability. SAE-SWS-s outperforms SAE-KD-s by 1.03%, which shows the effective of SWS on knowledge transfer. Table 3 shows the detailed classification accuracy for each fault of the SAE-SWS-s.

4.2. TE Process. The TE process is an authoritative benchmark problem in FDD. Since it was proposed by Downs and Vogel, it has emerged as a popular tool to test the performance of the FDD model [25]. The entire process consists of five operating units: reactor, condenser, recycle compressor, separator, and stripper. It has 53 variables, including 12

FIGURE 8: Classification accuracy of the series of student models for the CSTR.

FIGURE 9: Relationship between parameter α and classification accuracy in the TE process.

TABLE 3: Detailed results of SAE-SWS-s for the CSTR.

Fault Number	Training Accuracy (%)	Testing Accuracy (%)
Fault 1	97.00	92.50
Fault 2	95.50	92.30
Fault 3	97	90.33
Fault 4	87.5	75.50
Fault 5	97.33	85.33
Fault 6	98.83	96.67
Fault 7	99.67	96.50
Fault 8	97.67	74.67
Fault 9	96.5	78.67
Fault 10	100	100
Fault 11	87.83	70.50
Normal state	93.33	85.67

manipulated variables and 41 measuring variables. When a fault occurs, it affects almost all variables [26]. The dataset that includes a normal state and 17 faults for training and testing is collected, respectively, to build a fault classification model. IDV3, IDV9, IDV15, and IDV21 are removed. The fault testing set removes the first 160 normal samples; hence, the test sample of a certain fault includes 800 samples. In this experiment, kernel function of SVM is RBF. Penalty factor is set to 1. Number of neighbors in kNN is 8. Euclidean distance is the metric. As for the SAE-based methods, the structure is set to [200-100-50-30]. Adam optimizer is applied and its learning rate is set to 0.002. The batch size and epoch are 10 and 1500, respectively. Similar to that for the CSTR, a number of experiments are conducted to confirm the related parameters in SAE-SWS. The results are shown in Figures 9, 10, and 11. Accordingly, α is set to 0.3. r is set to 2. The optimal student model is S_1 .

SAE-SWS is compared with other algorithms. Table 4 shows the performance of the algorithms for the TE dataset. The SAE series outperforms the conventional methods. The comparison of the SAE series shows that SAE-SWS-s achieves

FIGURE 10: Relationship between parameter r and classification accuracy in the TE process.

TABLE 4: Classification accuracy of some algorithms in the TE process.

Algorithms	Accuracy (%)
SVM	60.52
kNN	40.35
SAE	74.39
SAE-SWS-t	75.02
SAE-SWS-s	76.64
SAE-KD-s (without SWS)	75.84

the best performance for the TE dataset. SAE-SWS-s outperforms SAE-KD-s by 0.8%, which confirms the effectiveness of the SWS. Table 5 lists the specific accuracy values for SAE-SWS-s for training and testing.

4.3. Sensorless Drive Diagnosis. In sensorless drive diagnosis (SDD), variables are extracted from electric current drive signals. The drive has intact and defective components. There are 11 classes with different conditions in this dataset. The current signals are measured with a current probe and an oscilloscope on two phases. The empirical mode

FIGURE 11: Classification accuracy of series student mode in the TE process.

TABLE 5: Detailed results of SAE-SWS-s in the TE process.

Fault Number	Training Accuracy (%)	Testing Accuracy (%)
IDV 1	99.23	99.13
IDV 2	97.02	96.38
IDV 4	99.11	97.75
IDV 5	99.70	99.25
IDV 6	99.70	86.25
IDV 7	100	99.88
IDV 8	95.53	45.13
IDV 10	94.64	43.88
IDV 11	83.33	69.13
IDV 12	94.64	76.75
IDV 13	97.62	20.50
IDV 14	95.53	95.13
IDV 16	97.92	83.25
IDV17	92.28	91.38
IDV 18	85.42	72.50
IDV 19	89.58	75.75
IDV 20	82.74	71.00
Normal	63.14	37.25

decomposition was used to generate the database for the generation of 48 variables. More information can be found in <http://archive.ics.uci.edu/ml/datasets/Dataset+for+Sensor-less+Drive+Diagnosis>.

In this experiment, kernel function of SVM is RBF. Penalty factor is set to 1. Number of neighbors in kNN is 6. Euclidean distance is the metric. As for SAE-based methods, structure is set to [100-50-25]. Adam optimizer is applied and its learning rate is set to 0.001. The batch size and epoch are 10 and 1000, respectively. Similarly, a number of experiments are conducted to confirm the related parameters in SAE-SWS. The results are shown in Figures 12, 13, and 14. Accordingly, α is set to 0.3. r is set to 2. The optimal student model is S_3 .

FIGURE 12: Relationship between parameter α and classification accuracy in the SDD dataset.

FIGURE 13: Relationship between parameter r and classification accuracy in the SDD dataset.

TABLE 6: Classification accuracy of some algorithms in the SDD dataset.

Algorithms	Accuracy (%)
SVM	86.24
kNN	77.53
SAE	93.92
SAE-SWS-t	93.62
SAE-SWS-s	96.75
SAE-KD-s (without SWS)	94.94

The comparison results presented in Table 6 indicate that the SAE-SWS algorithm has the best performance among all algorithms. Besides, results of SAE-SWS-t and SAE-SWS-s show that the student learns a lot from teacher. In addition, SAE-SWS-s outperforms SAE-KD-s by 1.81%, which implies the effectiveness of SWS on improving classification performance. Detailed classification accuracy for each fault of the SAE-SWS-s can be seen in Table 7.

5. Conclusions

In this work, a novel strategy called the SWS is proposed for SAEs to deal with fault classification problems. The proposed

FIGURE 14: Classification accuracy of series student mode in the SDD dataset.

TABLE 7: Detailed results of SAE-SWS-s in the SDD dataset.

Class Number	Train Accuracy (%)	Test Accuracy (%)
Class 1	100	98.62
Class 2	97.02	96.54
Class 3	95.79	93.25
Class 4	100	97.39
Class 5	99.70	96.56
Class 6	100	100
Class 7	94.53	93.42
Class 8	95.12	94.41
Class 9	99.63	97.83
Class 10	100	100
Class 11	98.56	96.25

strategy tries to transfer knowledge between two identical models. After training a teacher model, the strategy SWS is applied to obtain “knowledge”, which denotes the output distribution of the teacher model, for student model training. Then the student model becomes next student’s teacher. After step-by-step teacher-student training, the optimal model, which is selected among all the trained models, is applied for classifying real-time data. The experiments on several datasets prove the effectiveness of SWS strategy as well as the proposed SAE-SWS algorithm. Besides, the improvements of proposed method are summarized as follows. (1) A powerful DL model SAE is used for feature extraction in FDD. (2) By successive teacher-student training, SWS boosts the probability of achieving superior performance by increasing the weights of the mistaken samples. (3) KD helps to obtain more useful information about the correlation among classes. However, in practical applications, there are the following limitations. (1) Processes with insufficient training sample size are not conducive to feature extraction in the proposed SAE-SWS, resulting in the extracted features not being the intrinsic representation of data. (2) As the step increases, the number of parameters in SAW-SWS increases dramatically. So the application of SAE-SWS requires good hardware

equipment and a much longer off-line modeling time than traditional monitoring methods. Therefore, further study will focus on reducing number of parameters as well as model complexity.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (21878081) and the Fundamental Research Funds for the Central Universities under Grant of China (222201917006).

References

- [1] R. Isermann, “Model-based fault-detection and diagnosis—status and applications,” *Annual Reviews in Control*, vol. 29, no. 1, pp. 71–85, 2005.
- [2] Y. Shin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, “A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process,” *Journal of Process Control*, vol. 22, no. 9, pp. 1567–1581, 2012.
- [3] Q. Jiang and X. Yan, “Chemical processes monitoring based on weighted principal component analysis and its application,” *Chemometrics and Intelligent Laboratory Systems*, vol. 119, pp. 11–20, 2012.
- [4] J. Tang and X. Yan, “DPLS-MSOM modeling for visual industrial fault diagnosis and monitoring based on variation data from normal to anomalous,” *Industrial and Engineering Chemistry Research*, vol. 56, no. 49, pp. 14518–14529, 2017.
- [5] M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski, “Face recognition by independent component analysis,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 13, no. 6, pp. 1450–1464, 2002.
- [6] J. Tang and X. Yan, “Neural network modeling relationship between inputs and state mapping plane obtained by FDA-t-SNE for visual industrial process monitoring,” *Applied Soft Computing*, vol. 60, pp. 577–590, 2017.
- [7] Y. Maki and K. A. Loparo, “A neural-network approach to fault detection and diagnosis in industrial processes,” *IEEE Transactions on Control Systems Technology*, vol. 5, no. 6, pp. 529–541, 1997.
- [8] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *The American Association for the Advancement of Science: Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [9] J. Sun, R. Wyss, A. Steinecker, and P. Glocker, “Automated fault detection using deep belief networks for the quality inspection of electromotors,” *Technisches Messen*, vol. 81, no. 5, pp. 255–263, 2014.
- [10] H. Ren, Y. Chai, J. Qu, X. Ye, and Q. Tang, “A novel adaptive fault detection methodology for complex system using deep

- belief networks and multiple models: A case study on cryogenic propellant loading system,” *Neurocomputing*, vol. 275, pp. 2111–2125, 2018.
- [11] 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.
 - [12] S. Roy, N. Das, M. Kundu, and M. Nasipuri, “Handwritten isolated Bangla compound character recognition: A new benchmark using a novel deep learning approach,” *Pattern Recognition Letters*, vol. 90, pp. 15–21, 2017.
 - [13] K. B. Lee, S. Cheon, and O. Chang, “A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, no. 99, pp. 135–142, 2017.
 - [14] 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, 2018.
 - [15] H. Lee, Y. Kim, and C. O. Kim, “A deep learning model for robust wafer fault monitoring with sensor measurement noise,” *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, no. 1, pp. 23–31, 2017.
 - [16] N. Zhang, X. Tian, and L. Cai, “Nonlinear dynamic fault diagnosis method based on autoencoder,” in *Proceedings of the Fifth International Conference on Measuring Technology and Mechatronics Automation*, pp. 729–732, IEEE Computer Society, China, January 2013.
 - [17] J. Yim, D. Joo, J. Bae, and J. Kim, “A gift from knowledge distillation: Fast optimization, network minimization and transfer learning,” in *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 7130–7138, IEEE Computer Society, Honolulu, July 2017.
 - [18] T. Furlanello, Z. C. Lipton, M. Tschannen, L. Itti, and A. Anandkumar, “Born again neural networks,” in *Proceedings of the International Conference on Machine Learning*, International Machine Learning Society, Stockholm, Sweden, 2018.
 - [19] C. Xu, J. Yang, J. Gao, H. Lai, and S. Yan, “SRNN: Self-regularized neural network,” *Neurocomputing*, vol. 273, pp. 260–270, 2018.
 - [20] J.-H. Bae, J. Yim, N.-S. Kim, C.-S. Pyo, and J. Kim, “Layer-wise hint-based training for knowledge transfer in a teacher-student framework,” *ETRI Journal*, vol. 41, no. 2, pp. 242–253, 2019.
 - [21] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” in *Proceedings of the Twenty-eighth Conference on Neural Information Processing Systems*, Foundation of Neural Information Processing Systems, Montreal, 2014.
 - [22] V. Balakotaiah and D. Luss, “Analysis of the multiplicity patterns of a CSTR,” *Chemical Engineering Communications*, vol. 13, no. 1–3, pp. 111–132, 1981.
 - [23] H. Belevi, J. R. Bourne, and P. Rys, “Mixing and fast chemical reaction-II. Diffusion-reaction model for the CSTR,” *Chemical Engineering Science*, vol. 36, no. 10, pp. 1649–1654, 1981.
 - [24] S. Yoon and J. F. MacGregor, “Fault diagnosis with multivariate statistical models part I: using steady state fault signatures,” *Journal of Process Control*, vol. 11, no. 4, pp. 387–400, 2001.
 - [25] J. J. Downs and E. F. Vogel, “Plant-wide industrial process control problem,” *Computers & Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.
 - [26] P. R. Lyman and C. Georgakis, “Plant-wide control of the tennessee eastman problem,” *Computers & Chemical Engineering*, vol. 19, no. 3, pp. 321–331, 1995.

Submit your manuscripts at
www.hindawi.com