This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2020.3025805, IEEE Sensors Journal

A just-in-time fine-tuning framework for deep learning of SAE in adaptive data-driven modeling of time-varying industrial processes

Yijun Wu, Diju Liu, Xiaofeng Yuan, Yalin Wang

School of Automation, Central South Univeristy, Changsha, China, 410083

***Abstract*—In modern industrial processes, soft sensors have played increasingly important roles for effective process monitoring, control and optimization. Deep learning has shown excellent ability for hierarchical nonlinear feature representation in soft sensors. However, the existing deep learning based soft sensors are mostly trained offline and applied online without updating mechanism. This may cause their performance degradation in time-varying processes. To deal with this problem, an adaptive updating framework is proposed for deep learning, which is based on just-in-time fine-tuning of stacked autoencoder (JIT-SAE). In JIT-SAE, an offline SAE model is first trained with layer-wise unsupervised pre-training and supervised fine tuning. For online prediction, the network is dynamically fine-tuned upon the query sample. For each query sample, the most relevant labeled samples are selected to form a fine-tuning dataset from the historical labeled database, which is regularly augmented once new labeled samples are available from laboratory analysis. Moreover, each relevant sample is assigned with a weight according to its similarity with the query sample. Then, the deep network is fine-tuned with these relevant labeled samples by designing a weighted loss function. Thus, JIT-SAE is able to track the newest process running state timely and match the data pattern accurately. Case study on an industrial hydrocracking process is provided to demonstrate the effectiveness of the JIT-SAE framework.**

***Index Terms*—Soft sensor; Quality prediction; Stack autoencoders (SAE); Deep learning; Just-in-time learning**

I. INTRODUCTION

The monitoring of key quality variables in process industries plays an important role in process control and optimization, which is indispensable for keeping product quality and improving production efficiency [1-5]. However, most process quality variables cannot be measured on-line by traditional hard sensors due to technical and economical limitations. Instead, they are often tested by offline lab analyzers, which usually causes large time delay and irregular sampling frequencies. Therefore, it is generally difficult to meet the real-time measurement requirements for fast control strategies. Even for some variables that can be measured online, the online analyzers should be taken off regularly for examination and maintenance [6-8].

Alternatively, soft sensors have been developed to handle the above problems, which seek to use a set of measurable process variables to construct mathematical prediction models for estimating the difficult-to-measure quality variables [9]. In recent years, soft sensor technology has gained a lot of popularity in the process industry with its economical savings, reliable performance, fast response, and easy maintenance. Moreover, soft sensors have achieved many successful results in different processes [10-12]. Generally speaking, soft sensors can be divided into two types: first-principle models [13] and data-driven models [14]. First-principle models require a deep understanding of the complex physicochemical principles behind the industrial processes. Thus, it is very difficult to build first-principle models for large complex industrial processes. Differently, data-driven models are constructed based on plant running data and require little process prior knowledge. In recent years, with the development of hard sensor and data storage technologies, many industrial processes have stored a large amount of process historical data. This provides a very favorable condition for the realization of data-driven soft sensors. Therefore, data-driven soft sensors have been largely applied to provide accurate and reliable on-line prediction of key variables in complex industrial processes [15-17].

Some of the most common approaches to data-driven models are based on multivariate statistical methods such as principal component analysis (PCA) [18], partial least squares (PLS) [19], canonical correlation analysis [20], etc. The disadvantage of such models is that they are linear models and thus cannot handle process nonlinearity. Therefore, kernel techniques have been introduced to deal with the nonlinearity to construct kernel PCA (KPCA) [21], kernel PLS (KPLS) [22]. Moreover, artificial neural networks (ANN) [23] can also deal with the continuous nonlinear approximation problems to arbitrary accuracy with a sufficient number of hidden layers. However, the gradient vanishing and gradient explosion have been the limiting problems in network training when the structure becomes large with many hidden layers [24]. In 2006, Hinton et al. proposed the deep learning technique, which could alleviate these problems by using the greedy layer-wise unsupervised pre-training and supervised fine-tuning in large network

This work was supported in part by the National Key R&D Program of China (2018AAA0101603, 2018YFB1701100), the Program of National Natural Science Foundation of China (61988101, 61590921, 61703440, 61621062). (Co-first author: Y. Wu and D. Liu, Corresponding author: X. Yuan and Y. Wang)

Y. Wu, D. Liu, X. Yuan and Y. Wang are with the School of Automation, Central South University, Changsha 410083, China (1129274213@qq.com, 1358807706@qq.com, yuanxf@csu.edu.cn, ylwang@csu.edu.cn).

training [25]. Moreover, deep learning has natural advantages in feature representation from massive raw data for different tasks. Since then, deep learning has witnessed explosive development in many areas. For example, stack autoencoders (SAE) [26], convolutional neural network (CNN) [27], deep belief network (DBN) [28], long-short-term memory network (LSTM) [29] and other popular deep learning models have made significant achievements in image processing, natural language processing, and other fields [30, 31].

Deep learning has also been introduced into process industry for soft sensor modeling due to its strong feature extraction and nonlinear processing ability [32-34]. For example, Shang et al. first exploited deep belief network to build soft sensor model for a crude distillation unit. They discussed the unique advantages of deep learning for industrial processes [35]. Yao et al. developed a semi-supervised deep learning framework for inferential modeling, which was based on the hierarchical extreme learning machine [32]. Liu et al. developed a soft sensor based on ensemble deep kernel learning to predict the quality variable in industrial polymerization processes. Moreover, new deep networks such as stacked quality-driven autoencoder [36] were proposed to extract quality-relevant features from the input data by introducing the quality information to guide the pretraining procedure. These deep learning methods have achieved good results in the industrial systems such as debutanizer column and hydrocracking processes.

There are two critical issues should be tackled in the soft sensor modeling of modern industrial processes, which are the process nonlinearity and time-varying characteristics. From the above examples, it can be seen that deep learning has shown impressive performance in dealing with process nonlinearity. However, most of the aforementioned models like SAE pay little attention to the process time-varying problems. The common modeling procedure of these works is that they train the networks offline with process historical data and then use them online for quality forecasting of new samples. With the running of processes, there are time-varying problems because of reasons like the aging of equipment, deactivation of catalyst, changing of operation conditions, etc. Thus, the characteristics of data and their relationship are constantly changing all the time. However, the offline model cannot update the model timely according to the process changes, which results in the performance degradation of deep networks. Re-training the deep network with the available dataset is one optional way to solve this problem. However, it is difficult to meet the real-time prediction requirements in the actual industrial processes. This is mainly because it takes a lot of computational resources and time to re-train the deep networks with a huge amount of data. Hence, it is necessary to utilize a more efficiently adaptive strategy for deep learning to deal with the performance degradation in time-varying processes.

Inspired by the just-in-time learning (JITL) [37], which is an effective framework that can be used for adaptive modeling by building online local models upon demand, an adaptive just-in-time fine-tuning framework is proposed for deep learning of SAE (JIT-SAE) in time-varying processes. Although the characteristics of industrial processes are time-varying, there is a high degree of similarity between the possible process states. Therefore, the pre-training and fine-tuning with historical data can roughly extract the general data characteristics of various working conditions. When a query sample arrives, some similar data samples can be selected to fine-tune the model so that the model can accommodate to the current working conditions quickly. Thus, there are two main stages in the JIT-SAE framework. The first stage is offline modeling by unsupervised pre-training with massive unlabeled data and fine-tuning with historical labeled data. The second stage is online just-in-time fine-tuning by selecting the most relevant labeled samples for network updating. The pre-training in the offline stage is just a general step to extract historical data features for the running processes. The offline fine-tuning obtains a rough elementary prediction network using all available labeled data. In the online stage, the historical labeled dataset is dynamically augmented since new labeled samples can be added to it timely once they are obtained from laboratory analysis. Then, the online just-in-time fine-tuning stage selects a small part of relevant labeled samples that are most similar to the query samples. Moreover, these relevant samples are assigned with different weights according to their similarities with the query sample. A more relevant sample indicates that it is has a closer data pattern with the query sample from a similar running state. They are then used to fine-tune the trained network for adaptive modeling to improve the prediction accuracy. Therefore, the off-line trained model is acted as a basic one for online updating and the network is dynamically updated whenever a query sample comes. This method can not only improve the processing ability of the model for time-varying problems, but also greatly reduce the computation costs compared with the re-training of the deep networks.

The remaining parts of this paper are organized as follows. In Section II, stack autoencoder (SAE) is briefly reviewed. In Section III, the modeling procedure of the JIT-SAE framework is introduced in detail. Next, the JIT-SAE framework is applied to an industrial hydrocracking process for quality prediction in Section IV. In Section V, we summarize the main contributions of this paper and discuss future research directions.

II. STACKED AUTOENCODER

A. Autoencoder

Autoencoder (AE), which was first proposed by Rumelhart in 1986 [38], is a typical unsupervised neural network. An autoencoder is a three-layer neural network containing an encoder and a decoder. It seeks to reconstruct the original input as accurately as possible at the output layer. Fig. 1 shows the basic structure of an AE model.

Suppose the inputs of the AE is $x = \left[x_{(1)}, x_{(2)}, \cdots, x_{(d_x)} \right]^T \in R^{d_x}$,where d_x is the dimension of the input vector. The input x is first mapped to the hidden

representation $h=\left[h_{(1)},h_{(2)},\cdots,h_{(d_h)}\right]^T \in R^{d_h}$ by nonlinear function f

$$h = f(x) = s_f(Wx+b) \quad (1)$$

where s_f is the activation function at the hidden layer, W is a $d_h \times d_x$ weight matrix, $b \in R^{d_h}$ is the bias vector, $d_{(h)}$ is the dimension of the hidden variable vector.

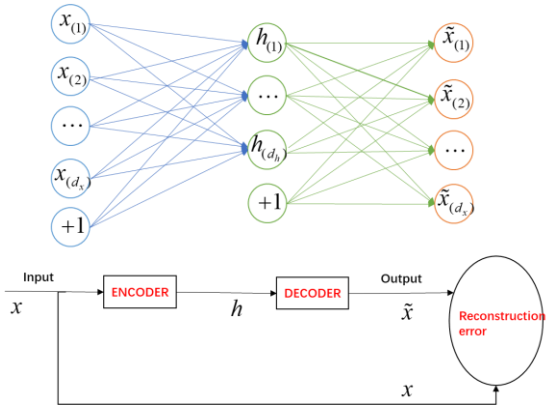


Fig. 1. Model structure of autoencoder

In the decoder, the hidden feature variable vector h is decoded to the reconstructed input vector at the output layer as

$$\tilde{x} = \tilde{f}(h) = s_{\tilde{f}}(\tilde{W}h+\tilde{b}) \quad (2)$$

where $\tilde{W}$ is a $d_x \times d_h$ weight matrix, the $\tilde{b} \in R^{d_x}$ is the bias vector, $s_{\tilde{f}}$ is the nonlinear activation function at the output layer. s_f and $s_{\tilde{f}}$ can be selected as the sigmoid or other activation function such as tanh and the rectified linear unit function. Hence, the parameter set of an autoencoder is $\theta=\left\{W,\tilde{W},b,\tilde{b}\right\}$. AE seeks to obtain feature representation from its inputs with the aim that the reconstructed variable $\tilde{x}$ can be mostly identical to the original variable x. That is to say, AE seeks to learn a function $g_\theta(x)=\tilde{f}(f(x)) \approx x$ by imposing constraints on the network such as limiting the number of hidden units. Given the raw training dataset of N training samples $X=\left\{x_1,x_2,\cdots,x_N\right\}$, the corresponding hidden data are denoted as $H=\left\{h_1,h_2,\cdots,h_N\right\}$ and the reconstructed input data are denoted as $\tilde{X}=\left\{\tilde{x}_1,\tilde{x}_2,\cdots,\tilde{x}_N\right\}$. To obtain the model parameters, the reconstructed loss function is minimized by calculating the mean squared reconstructed error as

$$J(W,\tilde{W},b,\tilde{b}) = \sum_{i=1}^{N}\left\|\tilde{x}_i - x_i\right\|^2 \Big/ 2N = \sum_{i=1}^{N}\left\|g_\theta(x_i)-x_i\right\|^2 \Big/ 2N \quad (3)$$

The parameters of AE can be updated by the back-propagation based gradient descent algorithm.

B. *Stacked autoencoder*

Stack autoencoder is a neural network composed of multiple layers of autoencoder [39], the basic structure of SAE is shown in Fig. 2. The raw input data are first transmitted to the input layer of the whole SAE network with *K* autoencoders. Thus, the first AE is pre-trained by minimizing the reconstruction error for the raw input data. Then, the outputs of the first hidden layer are served as the inputs for the second AE, which is also pre-trained by minimizing the reconstruction error for the first-hidden feature data. Similarly, the entire SAE network can obtain the initial weight through this layer-wise unsupervised pre-training. After unsupervised pre-training, an output layer can be added to the top hidden layer for fine-tuning in order to carry out classification or prediction tasks. For regression prediction task, the neurons of this output layer just represent the predicted quality variables. The weight obtained by unsupervised pre-training is taken as the initial weight of the network. Then, the back-propagation algorithm is used to fine-tune the whole network to further adjust the parameters with the historical labeled data.

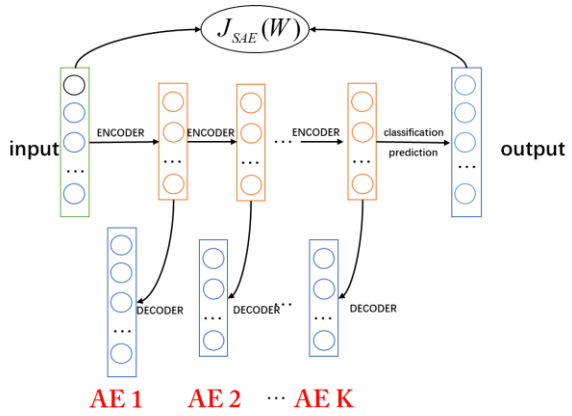


Fig. 2. Model structure of SAE

III. JUST-IN-TIME FINE-TUNING FRAMEWORK FOR SAE

For most deep learning -based applications, the networks are trained offline and used online without updated mechanisms. However, the situation is different for soft sensor modeling in process industries. In the actual industrial processes, there are many factors causing the fluctuation in working conditions like the aging of equipment, deactivation of the catalyst, and operation adjustment. Thus, the deep learning -based soft sensor models with only offline training have degradation problems in prediction performance. However, it is computationally expensive and time consuming to re-train the deep learning -based soft sensors. In this case, it is necessary to update the soft sensors effectively so that the models can quickly track the newest process working conditions. Therefore, an online just-in-time fine-tuning framework is designed for SAE-based deep learning for adaptive updating in time-varying processes. There are two main stages in the JIT-SAE modeling framework with offline modeling and online adaptive fine-tuning.

A. *Offline modeling*

In the offline modeling stage, a basic deep SAE model is obtained with process historical data. there are two steps in this stage: layer-wise greedy unsupervised pre-training and supervised fine-tuning.

Assume $\{x_u\}_{u=1,2,...,U}$ and $\{x_l, y_l\}_{l=1,2,...,L}$ are the historical unlabeled and labeled dataset, respectively. Here, U and L are the corresponding number of samples in the two datasets. Usually, the number of labeled data is much smaller than that of the unlabeled data. Thus, it is often the case that $L \ll U$. In the pre-training stage of SAE, the input data $\{x_u\}_{u=1,2,...,U}$ and $\{x_l\}_{l=1,2,...,L}$ are transmitted to the input layer and mapped to the first-layer hidden representation by function $h^1 = f(Wx+b)$. Thus, the weight and bias parameters $\{W_1, b_1\}$ from the input layer to the first hidden layer need to be optimized. To this end, the first AE (AE 1) is constructed by designing an output layer connecting to the first hidden layer and reconstructing the raw input at the output layer of AE 1. Thus, x, h^1 and $\tilde{x}^1$ are the input, hidden and output layer vectors in AE 1, respectively. The encoder maps x to h^1 with parameter set $\{W_1, b_1\}$, which is denoted as $h^1 = f(W_1 x + b_1)$, and the decoder reconstructs $\tilde{x}^1$ from h^1 with parameter set $\{\tilde{W}_1, \tilde{b}_1\}$, which is expressed as $\tilde{x}^1 = \tilde{f}(\tilde{W}_1 h^1 + \tilde{b}_1)$. To pre-train AE 1, the following objective function of the reconstruction error is minimized on $\{x_u\}_{u=1,2,...,U}$ and $\{x_l\}_{l=1,2,...,L}$ as

$$J_1(W_1, b_1, \tilde{W}_1, \tilde{b}_1) = \frac{1}{2(U+L)}\left(\sum_{u=1}^{U}\left\|\tilde{x}_u^1 - x_u\right\|^2 + \sum_{l=1}^{L}\left\|\tilde{x}_l^1 - x_l\right\|^2\right) \quad (4)$$

After AE 1 is pre-trained with back propagation algorithm, the encoder part of AE 1 is retained in the SAE network structure. Moreover, the first layer feature data can be obtained as $\{h_u^1\}_{u=1,2,...,U}$ and $\{h_l^1\}_{l=1,2,...,L}$ for the unlabeled and labeled datasets, respectively. Then, the feature states at the remaining hidden layer feature, h^2, h^3, ..., h^K, can be progressively obtained in a similar way.

Assume AE k has already been constructed and pre-trained. Thus, its hidden data are obtained as $\{h_u^k\}_{u=1,2,...,U}$ and $\{h_l^k\}_{l=1,2,...,L}$ for the unlabeled and labeled datasets. Then, they are transmitted to the next hidden layer to extract the higher hidden feature h^{k+1}. For this purpose, the parameter set $\{W_{k+1}, b_{k+1}\}$ should be optimized by constructing AE (k+1), in which h^k, h^{k+1} and $\tilde{h}^k$ are its input, hidden and output variable vectors, respectively. The mapping relationships among them are expressed as

$$h^{k+1} = f(W_{k+1} h^k + b_{k+1}) \quad (5)$$

$$\tilde{h}^k = \tilde{f}(\tilde{W}_{k+1} h^{k+1} + \tilde{b}_{k+1}) \quad (6)$$

Then, the parameter set of AE (k+1) is pre-trained by minimizing the reconstruction error as

$$J_{k+1}(W_{k+1}, b_{k+1}, \tilde{W}_{k+1}, \tilde{b}_{k+1}) = \frac{1}{2(U+L)}\left(\sum_{u=1}^{U}\left\|\tilde{h}_u^k - h_u^k\right\|^2 + \sum_{l=1}^{L}\left\|\tilde{h}_l^k - \tilde{h}_l^k\right\|^2\right) \quad (7)$$

In a progressive way, the deep SAE network can be pre-trained layer by layer.

After the pre-training, an output layer for the quality variable is added to the top hidden layer to construct the SAE-based prediction network, in which the weight and bias are denoted as w_o and b_o for the output layer, respectively. In the beginning of offline fine-tuning, the pre-trained parameter set $\{W_k, b_k\}_{k=1,2,...,K}$ are used to initialize the weights and bias of each hidden layer in SAE network. Moreover, w_o and b_o are first randomly initialized.

The SAE-based prediction network can be only fine-tuned with the labeled dataset $\{x_l, y_l\}_{l=1,2,...,L}$ since there is no output label in the unlabeled dataset. The top hidden feature data are obtained as $\{h_l^K\}_{l=1,2,...,L}$ for the labeled dataset after pretraining. Then, the predicted output is computed by forward propagation (FP) as

$$\tilde{y}_l = f(w^o h_l^K + b^o)_{l=1,2,...,L} \quad (8)$$

The prediction loss for the labeled training data is calculated as

$$J = \sum_{l=1}^{L}(y_l - \tilde{y}_l)^2 / 2L \quad (9)$$

By applying the back-propagation algorithm to minimize the prediction error, the parameters of the whole SAE prediction network are updated once. Then, the forward and back propagation algorithms can be alternately carried out to fine tune the network parameters until some convergence conditions are met.

B. Just-in-time Fine-tuning

The SAE network possesses good predictive performance in the training dataset after offline training. However, the frequent changes in operating conditions may cause model performance degradation when it is applied online. In order to keep the performance of the model and track the working conditions timely, a just-in-time fine-tuning strategy is designed for fast adaptive network updating. In the online JIT fine-tuning, the model only selects a small number of labeled samples that are most relevant to the query sample from the current historical labeled dataset. These relevant labeled samples are then used to form the fine-tuning dataset for this query sample. Moreover, these relevant samples are assigned with different weights according to their similarities with the query sample. Then, fine-tuning dataset is used to make online adaptive fine-tuning for the offline SAE model.

In detail, when a query sample comes to be predicted, assume x_q is the input part of this testing query sample. The historical labeled dataset is denoted as $\{x_l, y_l\}_{=1,2,...,L_o}$, where L_o is the number of labeled samples in the historical labeled dataset when x_q arrives. It is worth noting that the history labeled dataset is dynamically augmented with the running of process plants. This is because new labeled data can be added to the historical labeled dataset timely once it is available from lab test or other approaches. Thus, the number of historical labeled samples is becoming larger with the running of the process.

In order to select similar samples from the historical labeled dataset that are mostly related to the query sample x_q, Euclidean distance is used as a benchmarked measurement for the similarity between two samples. Thus, the Euclidean distance between each historical labeled sample and the query sample x_q is calculated as

$$D_l = \sqrt{(x_l - x_q)^T (x_l - x_q)},\ l = 1, 2, ..., L_o \tag{10}$$

A smaller distance indicates this sample is closer to the query sample. Thus, they may have more relevant data patterns from similar running states. In order to distinguish different sample relevance, a similarity weight function is designed with regard to the distance as

$$\lambda_l = \exp(-(D_l)^2 / \sigma^2)_{,\ l=1,2,...,L_o} \tag{11}$$

It is easily seen that weight is a decreasing function with the distance. σ is an adjustable parameter to control the decreasing rate with the distance. The increase of parameter σ will reduce the influence of distance D_l on the weight and the discrimination ability of the weight.

After calculating the weights, they are sorted in descending order from the largest to the smallest values. To apply the online adaptive fine-tuning, only N most relevant samples with the largest weight values are selected to form the fine-tuning dataset (FD). Assume the index of these relevant samples are $\{r_1, ..., r_n, ..., r_N\}$. Thus, the online fine-tuning dataset is

$$FD = \{(x_{r_1}, y_{r_1}), ..., (x_{r_n}, y_{r_n}), ..., (x_{r_N}, y_{r_N})\} \tag{12}$$

Correspondingly, their similarity weights are $\{\lambda_{r_1}, ..., \lambda_{r_n}, ..., \lambda_{r_N}\}$. Thus, the fine-tuning dataset and the similarity weight set are used to update the SAE model.

By forward propagation, the output variable can be obtained for the fine-tuning dataset as $\{\tilde{y}_{r_1}, ..., \tilde{y}_{r_n}, ..., \tilde{y}_{r_N}\}$. However, these relevant samples are different from each other since they have different similarities with the query sample. To pay more importance to relevant samples that are highly similar to the query sample, the following weighted loss function is designed to fine tune the network online through back propagation

$$J = \frac{1}{2N} \sum_{n=1}^{N} \lambda_{r_n} (y_{r_n} - \tilde{y}_{r_n})^2 \tag{13}$$

By carrying out forward propagation and backpropagation in turn, the SAE network can be updated online quickly. Then, the output can be easily predicted for the query sample. After the prediction is finished for the current query sample, the historical labeled dataset is dynamically updated by adding new collected available labeled data $\{x_{newt}, y_{newt}\}$ until the next query sample comes. For the next query sample, the above procedure of distance calculation, weight calculation, sample selection, just-in-time fine-tuning and quality prediction, will be carried out once again.

As can be seen, the online fine-tuning is based on the previous offline-trained SAE model with relatively general performance. Only a small number of similar samples are selected as the online fine-tuning dataset in the historical labeled database, in which the selected samples have very similar data patterns with the query sample. Thus, it only takes a few computational costs and updating iterations to achieve better performance for SAE model with an online adaptive fine-tuning strategy.

C. JIT-SAE based soft sensor

The just-in-time fine-tuning framework is very suitable for soft sensor modeling in complex nonlinear time-varying processes. In this part, the procedure is given for the JIT-SAE based soft sensor. In the offline training stage, JIT-SAE model inherits the advantages of offline SAE model in complex hierarchical nonlinear feature representation. More importantly, the JIT-SAE model can automatically perform online adaptive fine-tuning by adopting the just-in-time learning framework. Also, the computational and running-time costs of the just-in-time fine-tuning framework is less than the traditional model re-training strategies. Thus, it can optimize the model parameters and track the running changes quickly in real-time. This makes that the JIT-SAE model can be easily applied for soft sensor modeling of various actual industrial production processes.

Fig. 3 shows the basic offline modeling and online updating procedure of the JIT-SAE based soft sensor. First, the offline SAE model is established through layer-by-layer pre-training with raw input data and fine-tuning with labeled data. Next, the historical labeled dataset is dynamically augmented whenever new available ones are collected. When a query sample comes, relevant samples are selected from the historical labeled dataset to form the fine-tuning dataset. Meanwhile, the similarity weights are calculated for these relevant samples. Thus, online adaptive fine-tuning is carried out by using the fine-tuning dataset. Finally, the just-in-time fine-tuned model can predict the quality variable for the query sample with fast response.

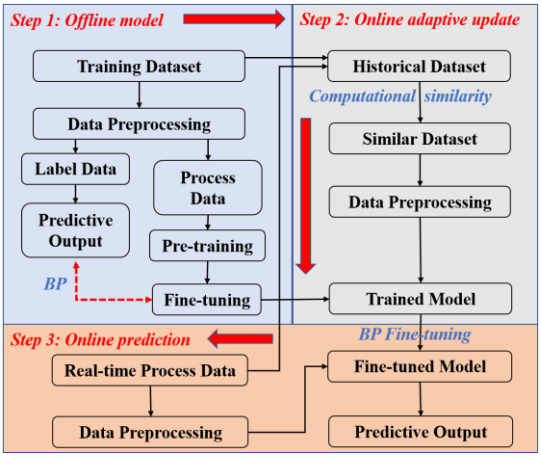


Fig. 3. The flowchart of the proposed JIT-SAE-based soft sensor model

To assess the prediction performance, the determination coefficient (R^2), root-mean-square error (RMSE), and mean absolute error (MAE) are used to calculate the performance accuracy of the different soft sensor models in this paper. The

prediction accuracy and performance of different models can be reflected by the three indicators on the testing dataset. They are defined as follows.

$$R^2 = 1 - \sum_{t=1}^{T}\left(y_t - \tilde{y}_t\right)^2 \Big/ \sum_{t=1}^{T}\left(y_t - \bar{y}\right)^2 \tag{14}$$

$$RMSE = \sqrt{\sum_{t=1}^{T}\left(y_t - \tilde{y}_t\right)^2 \Big/ T} \tag{15}$$

$$MAE = \sum_{t=1}^{T}\left|y_t - \tilde{y}_t\right| \Big/ T_n \tag{16}$$

where y_t and $\tilde{y}_t$ are the real output value and predicted output values of the t th query sample in the testing dataset, respectively; $\bar{y}$ is the mean of all real output value in the testing dataset, which is denoted as $\bar{y} = \sum_{t=1}^{T} y_t / T$; T is the total number of query samples in the testing dataset.

IV. CASE STUDY

In this section, the proposed JIT-SAE-based soft sensor is applied to an industrial hydrocracking process to evaluate its model performance .

A. Hydrocracking process

In the petroleum refining processes, hydrocracking is an improvement of the cracking technology, in which the catalytic cracking reaction is accompanied by hydrocarbon hydrogenation. The hydrocracking process studied in this paper are from a real industrial refinery in China. Fig. 4 shows the basic flowchart of this hydrocracking process. In this process, the heavy oil serves as the raw feeding material and undergoes hydrogenation, cracking and isomerization reactions under specific condition of high temperature, high pressure, hydrogen, and catalyst. Finally, it will be converted into light oil such as gasoline, kerosene, naphtha and diesel oil, etc.

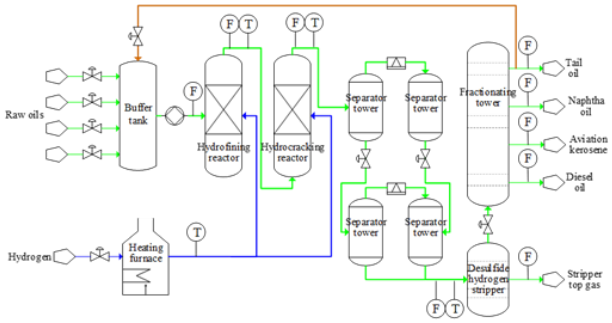


Fig. 4. Flowchart for the hydrocracking process

There are many complicated physical and chemical reactions during the whole production process, which result in it that the quality of the final product will be affected by many different factors. Therefore, in order to improve the quality and added-value of products, and increase economic benefits, it is particularly important to monitor the production process and evaluate the production status. Monitoring the real-time production status can help control and optimize the process in time, which will greatly improve the product qualification rate and reduce economic expenditure. These largely depend on the real-time measurement of the quality attributes of key products in the production process. However, most of the quality variables are difficult to measure online. In this paper, the C5 content in the light naphtha and the initial boiling point in the heavy naphtha are used as two examples of the key quality variables. In the actual hydrocracking process, temperature probes, pressure sensors and flowmeters are installed at different locations to measure the temperature, pressure and flow parameters, many of which have strong relationship with the quality variables. On one hand, the two key quality variables can only be obtained by laboratory test due to the harsh environment and technical reasons, which causes a huge time delay. On the other hand, there are a lot of easy-to-measure process variables that can be obtained in real time. Therefore, the process variables are used to construct soft sensors to predict the key quality variables. Moreover, this process is a highly nonlinear and time-varying one. Hence, the JIT-SAE is used as the modeling framework for the soft sensors.

B. C5 content in the light naphtha

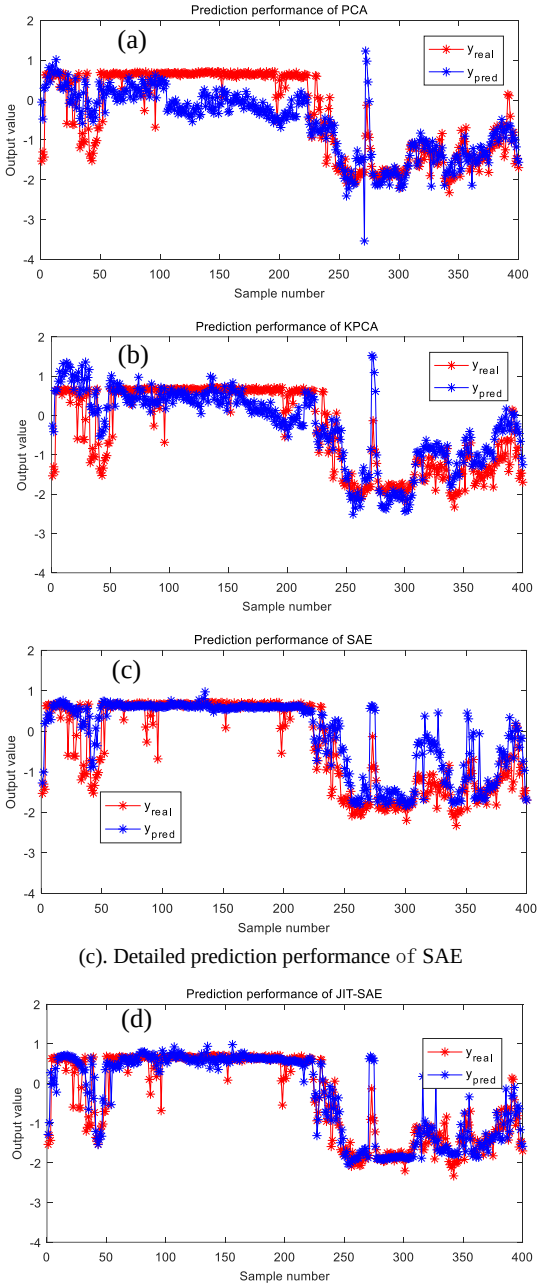
First, the JIT-SAE model is developed to predict the C5 content in the light naphtha. By analyzing the mechanism of hydrocracking reaction, 43 routinely measured process variables that are closely related to it are selected as the auxiliary variables to construct the JIT-SAE soft sensor model. The detailed description of these variables can be found in reference [40]. To eliminate the effect of unit difference between variables, all data are standardized by zero-mean normalization. A total of 1900 labeled data were collected in the plant from December 3, 2015 to November 30, 2018. For model training and testing, the first 1500 samples were used as the training dataset, while the last 400 samples were used as testing dataset. First, the hyperparameters of the neural network are determined by trial and error technique. The structure of the whole network is [43 20 5 1]. The sigmoid function is used as the activation function. First, the SAE is trained by means of greedy layer-wise pre-training to obtain a better initial weight and bias. Each AE is trained with random batch gradient descent algorithm. The batch size is selected as 250. After pre-training, an additional output layer is added to the top hidden layer for fine-tuning. When a testing query sample comes, 50 historical samples similar to the query samples will be selected to carry out just-in-time fine-tuning of the network. Then the fine-tuned model is used to predict the output of the query sample. In order to verify the performance of JIT-SAE model, traditional machine learning algorithms such as PCA and KPCA, as well as the original SAE algorithms without adaptive updating mechanism are also tested on the same data set.

TABLE 1. Prediction performance on testing dataset of the two methods for C5 content in the light naphtha

Method	RMSE	R2	MAE
PCA	0.6106	0.6639	0.4893
KPCA	0.5667	0.7104	0.4392
SAE	0.5384	0.7387	0.3291
JIT-SAE	0.2542	0.9418	0.1419

After applying the four different models on the dataset, the

prediction results are shown in Table 1. It can be seen from Table 1 that PCA has the worst performance as a linear model. KPCA has a certain degree of improvement compared with PCA. SAE can further improve the performance as a deep learning algorithm that has strong ability to extract nonlinear features. However, the JIT-SAE model with adaptive updating mechanism is significantly superior to the other three methods in terms of the three evaluation indexes. This is due to the JIT-SAE can adapt the model to the time-varying conditions.



(c). Detailed prediction performance of SAE

Fig. 5. Detailed prediction performance on the testing dataset for C5 content in the light naphtha: (a). PCA, (b). KPCA. (c)SAE (d)JIT-SAE

The detailed prediction results on the testing dataset are shown in Fig. 5 for the four methods. It can be seen from Fig.5(a) and Fig.5(b) that when the data fluctuation is small, the predictions of PCA and KPCA have large deviations from the actual values. As shown in Fig.5(c) and Fig.5(d), both SAE and JIT-SAE models can reduces the deviatins. For SAE, it can fit well with the first half of the data samples while the prediction error of the model gradually becomes large after a period of time. The main reason for this phenomenon is that SAE is a global offline model, which cannot automatically update the model parameters according to data patterns. This leads to the fact that the original model of SAE cannot adapt well to the varing working conditions. While for the JIT-SAE model, it can fit the actual quality curve well all the time because of its excellent adaptive mechanism.

Moreover, the boxplots of the absolute prediction errors are shown in Fig. 6 for the four methods. From Fig. 6, the results further illustrate that the JIT-SAE model has much tighter prediction error distribution around zero. Thus, compared with other three methods, the JIT-SAE framework has better modeling capacity for the time-varying problems.

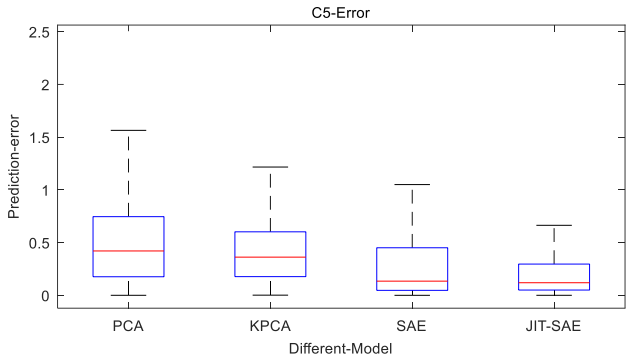


Fig. 6. Boxplot of the absolute prediction errors on the testing dataset of C5 content in the light naphtha with four methods

C. Initial boiling point of the heavy naphtha

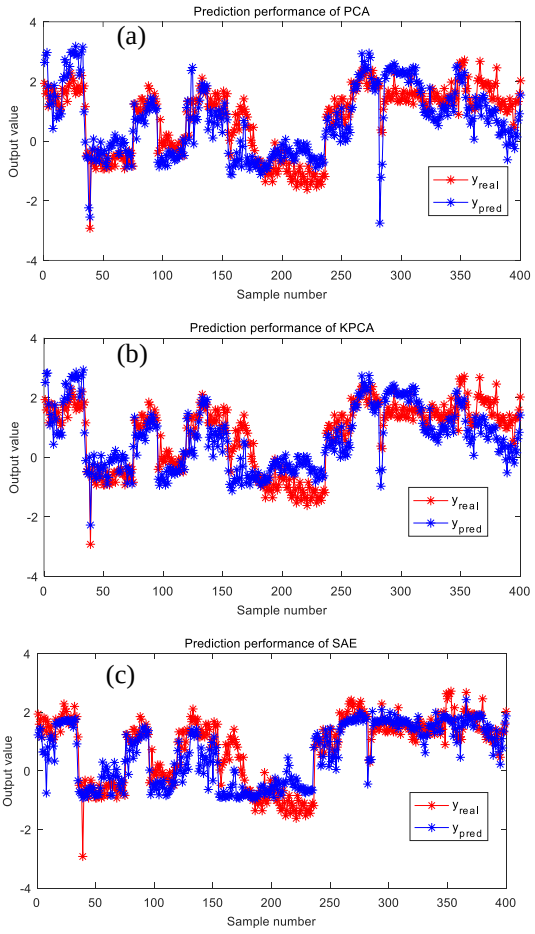
Then, the JIT-SAE model is applied to predict the initial boiling point of the heavy naphtha. The initial distillation point of heavy naphtha can partly reflect the composition of the components of distilled naphtha. A total of 1300 labeled samples are collected from this process. Among them, the first 900 samples are used as the training dataset and the last 400 are for the testing dataset. Similar to the parameter selection for the first indicator, the network structure of SAE is selected as [43-20-5-1].

Then, the prediction accuracy of the four methods are shown in Table 2 on the testing dataset. It can be seen from the evaluation indices that the prediction accuracy of JIT-SAE model is much better than that of the other three methods. This is because PCA is just a linear model that cannot extract the nonlinear data features. Compared with PCA, KPCA has better performance due to its nonlinear processing capability. SAE can improve the performance than the PCA and KPCA. However, its performance is much worse than JIT-SAE. This is mainly due to that the SAE is just a global offline model without updating. Thus, it is difficult to adapt to the changing conditions of the time-varying process. Differently, JIT-SAE adopts the just-in-time adaptive fine-tuning strategy to update the network dynamically upon the query sample. It can select the most relevant labeled samples to fine tune the network to the newest process running state quickly. Hence, it can largely outperform the original SAE in the three indices.

TABLE 2. Prediction performance on the testing dataset for initial boiling point of the heavy naphtha

Method	RMSE	R2	MAE
PCA	0.7449	0.5562	0.6035
KPCA	0.7201	0.5852	0.6068
SAE	0.6467	0.6654	0.4787
JIT-SAE	0.4877	0.8097	0.3572

Fig. 7 further shows the detailed predictions for the initial boiling point of the heavy naphtha on the testing dataset with the four models. As can be seen from Fig. 7, PCA and KPCA models only have a good fitting effect in a few specific regions. They can only keep up with the general trend of data changes, and there are certain large errors between the predicted and actual values. Compared with PCA and KPCA, SAE has better fitting performance, but it is still inadequate in the region where the data changes rapidly. For the JIT-SAE model, it can match the actual curve well. Moreover, Fig. 8 displays the boxplot of the absolute prediction errors for the four methods. The boxplot shows that the absolute prediction errors of the JIT-SAE method is more closely concentrated around 0 than the others. Compared with the other three methods, JIT-SAE shows more excellent prediction accuracy. This provides more accurate data for operational guidance and process monitoring at the industrial site.



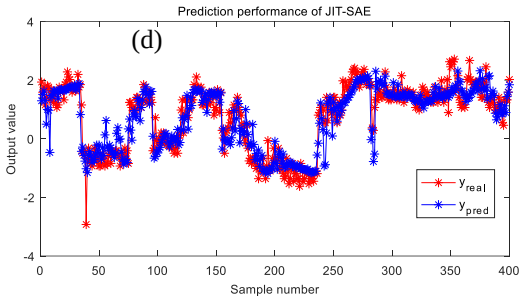


Fig. 7. Detailed prediction performance on the testing dataset for initial boiling point of the heavy naphtha: (a). PCA, (b). KPCA (c).SAE(d).JIT-SAE

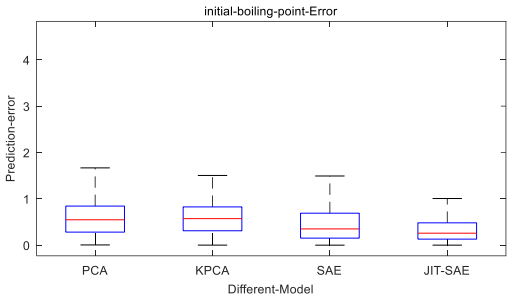


Fig. 8. Boxplot of the absolute prediction errors on the testing dataset of initial boiling point of the heavy naphtha with four methods

V. CONCLUSION

This paper proposes an adaptive model updating framework based on stack autoencoders for time-varying processes, which can effectively update model parameters and improve performance. The JIT-SAE model adopts the just-in-time fine-tuning strategy to adapt the deep learning to the process running states quickly for accurate prediction. Experiments are carried out on an industrial hydrocracking process for prediction of two quality indicators. Compared with the PCA, KPCA and original SAE, the JIT-SAE model shows excellent predictive performance. However, there are still some aspects needing to be improved in this JIT-SAE framework. For example, the Euclidean distance is not sufficient to select the most relevant samples for JIT fine-tuning of the deep networks. For the future work, we will further investigate it and find solutions to alleviate this problem.

References

[1] B. Bidar, F. Shahraki, J. Sadeghi, and M. M. Khalilipour, "Soft Sensor Modeling Based on Multi-State-Dependent Parameter Models and Application for Quality Monitoring in Industrial Sulfur Recovery Process," *IEEE Sensors Journal*, vol. 18, no. 11, pp. 4583-4591, 2018.

[2] F. Xu, Y. Wang, and X. Luo, "An Adaptive Soft Sensor Based on Nonlinear Differential-Algebraic Observer for Chemical Processes," *IEEE Sensors Journal*, vol. 15, no. 6, pp. 3249-3257, 2015.

[3] W. Shao, L. Yao, Z. Ge, and Z. Song, "Parallel Computing and SGD-Based DPMM For Soft Sensor Development With Large-Scale Semisupervised Data," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 8, pp. 6362-6373, 2019.

[4] L. Yao, and Z. Ge, "Online Updating Soft Sensor Modeling and Industrial Application Based on Selectively Integrated Moving Window Approach," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 8, pp. 1985-1993, 2017.

[5] X. Yuan, L. Li, and Y. Wang, "Nonlinear Dynamic Soft Sensor Modeling With Supervised Long Short-Term Memory Network," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3168-3176, 2020.

[6] T. Cheng, F. Harrou, Y. Sun, and T. Leiknes, "Monitoring Influent Measurements at Water Resource Recovery Facility Using Data-Driven

Soft Sensor Approach," *IEEE Sensors Journal,* vol. 19, no. 1, pp. 342-352, 2019.

[7] Z. Ge, Z. Song, and F. Gao, "Review of Recent Research on Data-Based Process Monitoring," *Ind. Eng. Chem. Res.,* vol. 52, no. 10, pp. 3543-3562, 2013.

[8] P. Kadlec, R. Grbić, and B. Gabrys, "Review of adaptation mechanisms for data-driven soft sensors," *Comput. Chem. Eng.,* vol. 35, no. 1, pp. 1-24, 2011.

[9] S. Khatibisepehr, B. Huang, and S. Khare, "Design of inferential sensors in the process industry: A review of Bayesian methods," *J. Process Contr.,* vol. 23, no. 10, pp. 1575-1596, 11, 2013.

[10] C. Gao, L. Jian, and S. Luo, "Modeling of the Thermal State Change of Blast Furnace Hearth With Support Vector Machines," *IEEE Transactions on Industrial Electronics,* vol. 59, no. 2, pp. 1134-1145, 2012.

[11] F. Souza, and R. Araújo, "Online Mixture of Univariate Linear Regression Models for Adaptive Soft Sensors," *IEEE Transactions on Industrial Informatics,* vol. 10, no. 2, pp. 937-945, 2014.

[12] D. Wang, "Robust Data-Driven Modeling Approach for Real-Time Final Product Quality Prediction in Batch Process Operation," *IEEE Transactions on Industrial Informatics,* vol. 7, no. 2, pp. 371-377, 2011.

[13] B. Huang, Y. Qi, and M. Murshed, "Dynamic modeling and predictive control in solid oxide fuel cells," John Wiley & Sons,, 2013, p. 1 online resource.

[14] X. Yuan, C. Ou, Y. Wang, C. Yang, and W. Gui, "A layer-wise data augmentation strategy for deep learning networks and its soft sensor application in an industrial hydrocracking process," *IEEE T. Neur. Net. Lear. Syst.,* pp. DOI: 10.1109/TNNLS.2019.2951708, 2019.

[15] L. Fortuna, P. Giannone, S. Graziani, and M. G. Xibilia, "Virtual Instruments Based on Stacked Neural Networks to Improve Product Quality Monitoring in a Refinery," *IEEE Transactions on Instrumentation and Measurement,* vol. 56, no. 1, pp. 95-101, 2007.

[16] L. Fortuna, S. Graziani, and M. G. Xibilia, "Comparison of Soft-Sensor Design Methods for Industrial Plants Using Small Data Sets," *IEEE Transactions on Instrumentation and Measurement,* vol. 58, no. 8, pp. 2444-2451, 2009.

[17] W. Shao, Z. Ge, and Z. Song, "Quality variable prediction for chemical processes based on semisupervised Dirichlet process mixture of Gaussians," *Chemical Engineering Science,* vol. 193, pp. 394-410, 2019/01/16/, 2019.

[18] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometr. Intell. Lab. Syst,* vol. 2, no. 1, pp. 37-52, 1987.

[19] P. Geladi, and B. R. Kowalski, "Partial least-squares regression: a tutorial," *Anal. Chim. Acta,* vol. 185, pp. 1-17, 1986.

[20] Z. Chen, S. X. Ding, T. Peng, C. Yang, and W. Gui, "Fault Detection for Non-Gaussian Processes Using Generalized Canonical Correlation Analysis and Randomized Algorithms," *IEEE Transactions on Industrial Electronics,* vol. 65, no. 2, pp. 1559-1567, 2018.

[21] J. Dai, N. Chen, X. Yuan, W. Gui, and L. Luo, "Temperature prediction for roller kiln based on hybrid first-principle model and data-driven MW-DLWKPCR model," *ISA T.,* vol. 98, pp. 403-417, 2020.

[22] R. Rosipal, and L. J. Trejo, "Kernel partial least squares regression in reproducing kernel hilbert space," *Journal of machine learning research,* vol. 2, no. Dec, pp. 97-123, 2001.

[23] X. Yuan, Y. Gu, Y. Wang, C. Yang, and W. Gui, "A deep supervised learning framework for data-driven soft sensor modeling of industrial processes," *IEEE T. Neur. Net. Lear. Syst.,* pp. DOI: 10.1109/TNNLS.2019.2957366, 2019.

[24] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[25] G. E. Hinton, and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science,* vol. 313, no. 5786, pp. 504-507, Jul, 2006.

[26] X. Yuan, C. Ou, Y. Wang, C. Yang, and W. Gui, "Deep quality-related feature extraction for soft sensing modeling: A deep learning approach with hybrid VW-SAE," *Neurocomputing,* vol. 396, pp. 375-382, 2020.

[27] 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, no. 2, pp. 135-142, 2017.

[28] Y. Wang, Z. Pan, X. Yuan, C. Yang, and W. Gui, "A novel deep learning based fault diagnosis approach for chemical process with extended deep belief network," *ISA Transactions,* vol. 96, pp. 457-467, 2020/01/01/, 2020.

[29] X. Yuan, L. Li, Y. Wang, C. Yang, and W. Gui, "Deep learning for quality prediction of nonlinear dynamic process with variable attention-based long short-term memory network," *Can. J. Chem. Eng.,* vol. 98, no. 6, pp. 1377-1389, 2020.

[30] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Appl. Intell.,* vol. 42, no. 4, pp. 722-737, 2015.

[31] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep Learning Face Representation by Joint Identification-Verification," pp. 1988--1996, 2014.

[32] L. Yao, and Z. Ge, "Deep Learning of Semisupervised Process Data With Hierarchical Extreme Learning Machine and Soft Sensor Application," *IEEE. T. Ind. Electron.,* vol. 65, no. 2, pp. 1490-1498, 2018.

[33] L. Yao, and Z. Ge, "Distributed parallel deep learning of Hierarchical Extreme Learning Machine for multimode quality prediction with big process data," *Eng. Appl. Artif. Intel,* vol. 81, pp. 450-465, 2019.

[34] X. Yuan, L. Li, Y. Shardt, Y. Wang, and C. Yang, "Deep learning with spatiotemporal attention-based LSTM for industrial soft sensor model development," *IEEE Transactions on Industrial Electronics,* pp. 1-1, 2020.

[35] C. Shang, F. Yang, D. Huang, and W. Lyu, "Data-driven soft sensor development based on deep learning technique," *J. Process Contr.,* vol. 24, no. 3, pp. 223-233, 2014.

[36] X. Yuan, J. Zhou, B. Huang, Y. Wang, C. Yang, and W. Gui, "Hierarchical quality-relevant feature representation for soft sensor modeling: a novel deep learning strategy," *IEEE. T. Ind. Inf.,* vol. 16, no. 6 pp. 3721-3730, JUN 2020, 2020.

[37] B. Pan, H. Jin, L. Wang, B. Qian, X. Chen, S. Huang, and J. Li, "Just-in-time learning based soft sensor with variable selection and weighting optimized by evolutionary optimization for quality prediction of nonlinear processes," *Chemical Engineering Research and Design,* vol. 144, pp. 285-299, 2019/04/01/, 2019.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature,* vol. 323, no. 6088, pp. 533-536, 1986.

[39] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, "Autoencoder for words," *Neurocomputing,* vol. 139, pp. 84-96, 2014.

[40] X. Yuan, J. Zhou, and Y. Wang, "A Comparative Study of Adaptive Soft Sensors for Quality Prediction in an Industrial Refining Hydrocracking Process," in 2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS), Enshi, China, 2018, pp. 1064-1068.