

# Hierarchical quality-relevant feature representation for soft sensor modeling: a novel deep learning strategy

Xiaofeng Yuan, *Member, IEEE*, Jiao Zhou, Biao Huang, *Fellow, IEEE*, Yalin Wang, *Member, IEEE*, Chunhua Yang, *Member, IEEE*, Weihua Gui, *Member, IEEE*

**Abstract**—Deep learning is a recently developed feature representation technique for data with complicated structures, which has great potential for soft sensing of industrial processes. However, most deep networks mainly focus on hierarchical feature learning for the raw observed input data. For soft sensor applications, it is important to reduce irrelevant information and extract quality-relevant features from the raw input data for quality prediction. To deal with this problem, a novel deep learning network is proposed for quality-relevant feature representation in this paper, which is based on stacked quality-driven autoencoder (SQAE). First, quality-driven autoencoder (QAE) is designed by exploiting the quality data to guide feature extraction with the constraint that the potential features should largely reconstruct the input layer data and the quality data at the output layer. In this way, quality-relevant features can be captured by QAE. Then, by stacking multiple QAEs to construct the deep SQAE network, SQAE can gradually reduce irrelevant features and learn hierarchical quality-relevant features. Finally, the high-level quality-relevant features can be directly applied for soft sensing of the quality variables. The effectiveness and flexibility of the proposed deep learning model are validated on an industrial debutanizer column process.

**Index Terms**—Deep learning, Artificial neural network, Quality-driven autoencoder (QAE), Stacked QAE, Soft sensor.

## I. INTRODUCTION

WITH the fast development of data measuring and storing techniques, as well as the wide use of distributed control systems (DCS), huge amounts of data are collected in modern industrial processes. For effective process monitoring, control and optimization, it largely depends on the real-time identification of process key quality variables, most of which are often difficult to measure online [1-5]. Therefore, it is imperative to develop data-driven soft sensor models to estimate these quality variables by those routinely measured process variables based on massive data within the industrial processes [6-8]. Up to now, a number of data-driven methods have been successfully developed for soft sensor applications,

like principal component analysis (PCA) [9], partial least squares (PLS) [10], artificial neural networks (ANN) [11] and support vector machine (SVM) [12], to name just a few.

Generally, there are complex correlations, nonlinearities and redundancies in the massive high-dimensional process data. They can easily lead to poor robustness, instability and even failure of the soft sensor models. To better model the complicated real-world process data, it is necessary to carry out feature representation for them, which can capture useful information for soft sensor modeling. However, it is often labor-intensive, time-consuming and expensive to design domain-specific hand-crafted features for complex data patterns. Alternatively, many different feature extractors, like PCA and PLS, have been designed to capture the underlying features from observed data automatically [13, 14].

Though the aforementioned methods can be utilized to learn different features for data modeling tasks. Their expressive ability is still insufficient to describe the complex data patterns in large-scale processes due to the shallow network architectures. Recently, deep learning [15, 16] has been developed for hierarchical feature representation of complex data patterns, which has made impressive progress in many fields like natural language processing, image recognition and computer vision, etc [17-19]. Especially, it became a milestone that AlexNet (a large deep learning model) won the champion in the famous ImageNet Large Scale Visual Recognition Challenge (ILSVRC) contest, which showed considerable breakthrough improvement in image recognition accuracy in 2012. Since then, deep learning has drawn more and more attention and become a mainstream of machine learning. Up to now, many different deep networks like deep belief network (DBN) [20, 21], stacked autoencoder (SAE) [22] and convolutional neural network (CNN) [23] have been designed and applied in industry extensively. Moreover, they can outperform traditional state-of-the-art machine learning models like SVM on different real-world problems [24-26].

Deep learning can learn hierarchical features from raw observed data progressively with the greedy layer-wise unsupervised pretraining technique. These features can be easily utilized for different tasks like classification and regression problems, which can usually achieve more effective results. Recently, deep learning has been introduced for process data modeling like fault classification and soft sensor in industrial plants [20, 27-35]. Jiang et al. proposed a dynamic sparse SAE (DSSAE) network to extract discriminative features for fault classification in dynamic processes [32]. Shang et al. applied DBN to predict the 95% cut point of the

This paper is supported in part by National Key R&D Program of China (2018YFB1701100), in part by National Natural Science Foundation of China (61590921, 61703440, 61860206014, 61621062), in part by the 111 project (B17048), in part by the Natural Science Foundation of Hunan Province of China (2018JJ3687), and in part by Innovation-driven plan in Central South University (2018CX011). (Corresponding author: Yalin Wang)

X. Yuan, J. Zhou, Y. Wang, C. Yang and W. Gui are with School of Automation, Central South University, Changsha 410083, China (e-mail: yuanxf@csu.edu.cn, 754712094@qq.com, ylwang@csu.edu.cn, ychh@csu.edu.cn, gwh@csu.edu.cn).

B. Huang is the Department of Chemical and Materials Engineering, University of Alberta, Edmonton, AB T6G2G6, Canada. (e-mail: biao.huang@ualberta.ca).

heavy diesel in a crude distillation unit, which showed impressive advantages of deep learning for soft sensor modeling [20]. Yao et al. developed a semi-supervised deep learning framework for inferential modeling, which was based on the hierarchical extreme learning machine [28].

As can be seen, most of the aforementioned deep networks focus on learning features for the raw data by stacking basic modules, which reconstructs the input by feature representation while constrains the representation to have certain desirable properties. For example, regular autoencoder seeks to keep the reconstruction input error as small as possible. However, AE is usually not stable and robust to the corruption of the inputs. Denoising autoencoder was developed to recover the original undistorted input from partially corrupted one. Features learned in this way are more robust to noises than the ordinary autoencoder [22]. When there are more hidden neurons than the inputs, sparse autoencoder can be adopted to make sure that only a small number of hidden neurons are active at the same time [36]. The learned features are good representation for the raw observed input data since these networks are unsupervised autoencoders. However, it may not be a good way of feature learning for soft sensor modeling, in which it is necessary and important to learn quality-relevant features from the raw observed input data with the guidance of the quality data. This idea has attracted a lot of attention in traditional shallow learning methods [37-39]. For example, PLS is just such a counterpart of PCA for quality-related feature learning. In reference [27], a deep learning model was first proposed for quality-related feature representation based on variable-wise weighted stacked autoencoder (VWSAE) network. In each autoencoder of VWSAE, a variable weighted objective function is designed to learn more abstract features by prior reconstruction for important quality-related input/feature variables of its input layer. However, there are two main limitations of VWSAE. First, it needs to measure the pair-wise Pearson correlation between each dimension of the variables at the input layer and the quality variable for each autoencoder, which may be inefficient and cumbersome. More importantly, the Pearson relationship is only a linear correlation measure, which does not take the important nonlinear relationships into consideration. As the nonlinear variable correlation is not measure sufficiently, it may still miss important quality-related information in VWSAE.

To alleviate the aforementioned problems, a new deep learning model is proposed to learn hierarchical quality-relevant features, which is based on stacked quality-driven autoencoder (SQAE). Different from VWSAE, SQAE directly utilizes the quality variables to construct a new deep learning network, which consists of multiple stacked quality-driven autoencoders (QAE). In each QAE, the output layer is composed of two parts: the hidden representation from its low layer and the quality variables. Therefore, high-level features are learned from their low-level ones by the objective that the learned features can simultaneously reconstruct their previous low-level features and predict the quality variables as good as possible. In this way, the learned features should not

only try to represent its input representation, but also largely predict the quality variables. Therefore, quality-relevant features can be effectively learned since they should largely predict the quality data. In a hierarchical stacked way, irrelevant information can be gradually reduced, and quality-relevant features can be progressively focused from the raw input data. After that, regression models can be constructed between the quality variables and the learned features for prediction. Therefore, SQAE network is more capable of improving the performance of soft sensors. The main contribution of this paper lies in the following aspects. First, a novel QAE network is proposed for quality-relevant feature learning, which is very different from other unsupervised self-learning autoencoders. Then, multiple QAEs are layer-wise stacked to construct a deep network for learning of hierarchical features that are largely relevant to the quality variables. At last, the deep quality-relevant features are used to build predictive regression models for accurate prediction. In this stage, traditional regression methods like least squares regression and support vector machine can be directly used. Alternatively, an additional output layer for quality variables can be added to the top layer of SQAE for fine tuning and prediction of the quality variables.

The rest of the paper is structured as follows. In Section II, the basic stacked autoencoder is briefly revisited. Section III introduces the novel SQAE and its soft sensor application. Then, case study is carried out to validate the effectiveness of the proposed method in Section IV. At last, conclusions are given in Section V.

Some important notations that are used throughout the paper are first given as follows.

$x$ : raw input variable vector.

$y$ : quality output variable vector.

$z, h$ : general notations for the input and hidden variable vectors of an autoencoder, respectively. Here,  $z$  can be the raw input variable vector or extracted feature vector at certain layer.

$h^k$ : hidden variable vector at the  $k$ th hidden layer of deep networks.

## II. STACKED AUTOENCODER

SAE is a deep learning neural network with multiple autoencoders hierarchically. An autoencoder is a three-layer unsupervised feature learning network with one input layer, a hidden layer and one output layer. The hidden layer transfers the input vector into hidden feature representation by nonlinear mapping. In the output layer, AE aims to reconstruct its input data by the hidden features. Fig. 1(a) shows a network illustration of AE.

Generally, the input and hidden variable vectors of the AE can be denoted as  $z \in R^{d_z}$  and  $h \in R^{d_h}$ , respectively. The reconstructed input variable vector at the output layer can then be denoted as  $\tilde{z}$ . At the hidden layer,  $z$  is transformed to obtain the hidden feature representation  $h$  by an encoder as

$$h = f(Wz + b) \quad (1)$$

where  $W$  and  $b$  are the weight and bias parameters, respectively;  $f$  is the activation function like sigmoid function. Then, the hidden feature variable vector  $h$  is decoded to the reconstructed input vector at the output layer as

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

where  $\tilde{W}$ ,  $\tilde{b}$  and  $\tilde{f}$  are the weight matrix, bias vector and activation function at the output layer, respectively. To obtain feature representation for its inputs, AE tries to ensure that the reconstructed  $\tilde{z}$  can be mostly identical to the real value  $z$ . Given a dataset of  $N$  training samples  $z_i \in \{z_1, z_2, \dots, z_N\}$ , the corresponding transformed features and reconstructed input data are denoted as  $h_i \in \{h_1, h_2, \dots, h_N\}$  and  $\tilde{z}_i \in \{\tilde{z}_1, \tilde{z}_2, \dots, \tilde{z}_N\}$ , respectively. To train the model parameters and obtain the feature data, the following objective function of a mean squared error term is minimized on the training data by back-propagation algorithm.

$$\begin{aligned} J(W, \tilde{W}, b, \tilde{b}) &= \frac{1}{2N} \sum_{i=1}^N \|\tilde{z}_i - z_i\|^2 \\ &= \frac{1}{2N} \sum_{i=1}^N \|g(z_i, W, b, \tilde{W}, \tilde{b}) - z_i\|^2 \end{aligned} \quad (3)$$

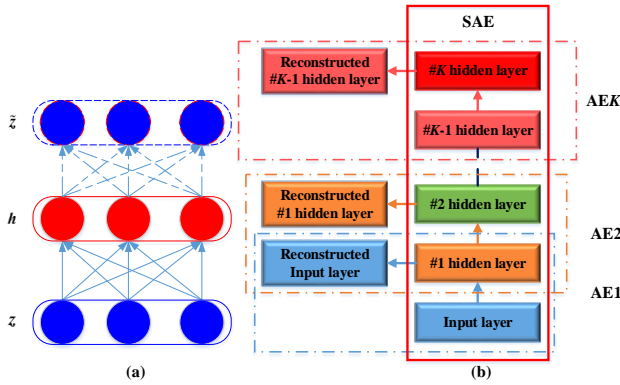


Fig. 1 The structure for AE and SAE. (a) an AE network; (b) SAE network with  $K$  autoencoders

SAE is a deep network consisting of multiple autoencoders with hierarchically stacked structure. The basic structure of SAE is illustrated in Fig. 1(b). For a SAE with  $K$  autoencoders, the raw input vector  $x \in R^{d_x}$  is fed to the input layer of the first AE (AE1) to train its parameter set  $\{W_1, b_1\}$  and obtain the first-layer features by minimizing the reconstruction error for  $x$  on the training samples. Then, the first-layer feature vector  $h^1$  is used as the input vector for AE2. Also, the parameter set  $\{W_2, b_2\}$  and the second-level feature  $h^2$  vector can be obtained by minimizing the reconstruction error for  $h^1$  on the training samples. In a successive way, the whole SAE can be pretrained layer by layer until AEK is pretrained for its parameter set  $\{W_K, b_K\}$  and feature data  $h^K$ .

### III. STACKED QUALITY-DRIVEN AUTOENCODER

As can be seen, the original SAE-based deep network learns hierarchical features that can represent the raw input data as

good as possible. Moreover, high-level abstract features are progressively obtained from low-level ones layer by layer. Such features may be good representation for the raw input data. However, they may not be good representation for prediction of the quality variables since irrelevant information may be retained in the obtained features. To handle this problem, a stacked quality-driven autoencoder (SQAE) is designed to learn hierarchical quality-relevant features in this section. In each quality-driven autoencoder (QAE), the quality variables are also used to guide the feature learning by adding them to the output layer. Each QAE generates new features from its inputs, then reconstructs the quality data and its input data simultaneously. In this way, the features are learned with the constraint that they can predict the output as good as possible. Hence, QAE can learn quality-relevant features for output prediction. By constructing deep network with stacked QAEs, hierarchical quality-relevant features can be consecutively extracted from the raw input data.

#### A. Quality-driven autoencoder (QAE)

QAE is also a three-layer network structure with the input, hidden and output layers. Still,  $z$  and  $h$  are used to denote the input and hidden variable vectors, respectively. The quality variable vector is denoted as  $y \in R^{d_y}$ . In the original AE, the output is just the reconstructed input  $\tilde{z}$ . Differently, QAE intends to reconstruct both its input data and the quality data simultaneously, which are denoted as  $\tilde{z}$  and  $\tilde{y}$ , respectively. Fig. 2 shows the network illustration of QAE. For the sake of simplicity,  $\{W, b\}$  and  $\{\tilde{W}, \tilde{b}\}$  are still used to represent the parameter sets of the encoder and decoder, respectively.  $f$  and  $\tilde{f}$  are the corresponding activation functions. As the decoder consists of two parts, its parameter sets and activation function can be decomposed to two parts with regard to the input layer data and quality data. That is to say, its parameters can be decomposed as

$$\tilde{W} = \begin{bmatrix} \tilde{W}_z \\ \tilde{W}_y \end{bmatrix}, \tilde{b} = \begin{bmatrix} \tilde{b}_z \\ \tilde{b}_y \end{bmatrix}, \tilde{f} = \begin{bmatrix} \tilde{f}_z \\ \tilde{f}_y \end{bmatrix} \quad (4)$$

Thus, the reconstructed input and quality data of QAE can be written as

$$\begin{bmatrix} \tilde{z} \\ \tilde{y} \end{bmatrix} = \tilde{f}(f(z)) = \begin{bmatrix} \tilde{f}_z(f(z, W, b), \tilde{W}_z, \tilde{b}_z) \\ \tilde{f}_y(f(z, W, b), \tilde{W}_y, \tilde{b}_y) \end{bmatrix} \quad (5)$$

Then, given the training dataset of the input layer data  $z_i \in \{z_1, z_2, \dots, z_N\}$  and the corresponding quality data  $\{y_1, y_2, \dots, y_N\}$ , QAE can be trained to minimize the reconstruction error at the output layer, which is

$$\begin{aligned} J(W, \tilde{W}, b, \tilde{b}) &= \frac{1}{2N} \left( \sum_{i=1}^N \left\| \begin{bmatrix} \tilde{z}_i \\ \tilde{y}_i \end{bmatrix} - \begin{bmatrix} z_i \\ y_i \end{bmatrix} \right\|^2 \right) \\ &= \frac{1}{2N} \sum_{i=1}^N (\|\tilde{z}_i - z_i\|^2 + \|\tilde{y}_i - y_i\|^2) \end{aligned} \quad (6)$$

The objective function of QAE comprises of two parts. The first one is the reconstruction error for the data at its input layer. This is used to ensure that the learned feature can capture the main input data patterns. The second one is the prediction error for the quality data. With this constraint, it ensures the hidden features can largely predict the quality variables. With this constraint, quality-relevant features can be extracted in QAE.

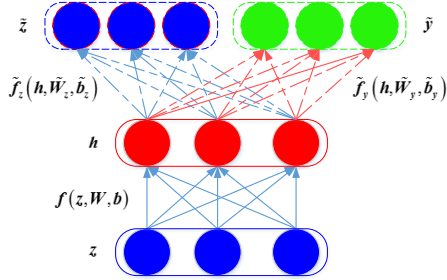


Fig. 2 The illustration of quality-driven autoencoder

Compared to the original AE, QAE does not increase the computational complexity too much. Taking the forward propagation stage for example, it totally computes  $d_z * d_h + d_h * d_z$  times of multiplications in one iteration for a sample. While for QAE, the number of multiplications is  $d_z * d_h + d_h * (d_z + d_y)$ . As the dimension  $d_y$  of the quality variable vector is usually much smaller than  $d_z$ , the computational complexity of AE and QAE is comparable for the forward propagation. In a similar way, it can be evaluated for the computational complexity for the back propagation stage.

As can be seen, the original AE and the proposed QAE are different from each other. AE is an unsupervised feature learning network, which just reconstructs its input variable vector. Different from AE, the output layer of QAE consists of not only its input variables, but also the quality variables. Hence, it is partly a supervised learning network to some degree. The relationship between AE and QAE is very similar to that of PCA and PLS. Though both QAE and VVAE [27] try to learn quality-relevant features from the input data, they are different from each other. VVAE has the same network structure as the original AE, in which the output layer is just the input variable vector. The improvement of VVAE lies in that the output layer reconstructs important quality-related input variables with large weights. However, QAE directly utilizes the quality variables to construct a different new network, in which the output layer consists of not only its input variables, but also the quality variables.

### B. Deep stacked QAE (SQAE)

Deep SQAE network is constructed by hierarchically stacking multiple QAEs. In SQAE, hierarchical quality-relevant features are layer-wise learned from the raw observed input data. At first, the raw observed input data is fed to the input layer of SQAE. By nonlinear transformation, the first-level features are generated at the first hidden layer of SQAE. In a successive way, high-level features can be obtained from its adjacent low-level ones with nonlinear transformation.

However, this cannot be directly computed since the connecting weight and bias parameters are unknown. Hence, layer-wise pretraining is carried out for the SQAE network. Fig. 3 shows the stacking flowchart of SQAE. At first, QAE1 is designed for the first hidden layer. The input of QAE1 is just the raw input data. In order to obtain the first-layer quality-relevant feature, the raw input data and the quality data are simultaneously reconstructed at its output layer. By minimizing the reconstruction error for both the raw input data and the quality data, the first QAE can be pretrained by the gradient-descent back-propagation algorithm. After that, the first-layer feature vector is connected to the input layer of QAE2 to obtain the second-layer feature. At the output layer of QAE2, the first-level feature data and the quality data are also reconstructed by the second-level features. Then, BP algorithm can be utilized to pretrain QAE2. In this way, all the  $K$  QAEs are progressively constructed and pretrained one after another until the top layer of SQAE is reached.

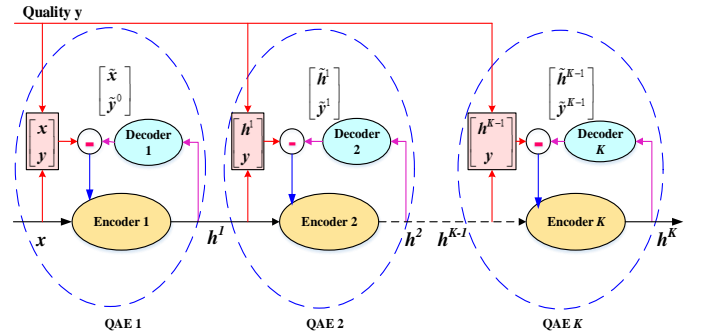


Fig. 3 The stacking flowchart of SQAE

Hence, given the raw observed input dataset  $\{x_1, x_2, \dots, x_N\}$  and the corresponding quality dataset  $\{y_1, y_2, \dots, y_N\}$ , SQAE can be pretrained to obtain hierarchical quality-relevant features. The detailed pretraining procedure for the SQAE can be illustrated in Fig. 4, which is described as follows.

At first,  $\{x_1, x_2, \dots, x_N\}$  are transmitted to the input layer of SQAE to generate the corresponding first-level feature data  $\{h_1^1, h_2^1, \dots, h_N^1\}$  by nonlinear activation function  $f$  with parameter set  $\{W_1, b_1\}$  at the first hidden layer. As  $\{h_1^1, h_2^1, \dots, h_N^1\}$  and  $\{W_1, b_1\}$  are unknown, this cannot be calculated directly. Hence, QAE1 is designed to reconstruct the raw input data and the quality data simultaneously at its output layer. The pretraining of QAE1 is carried out by minimizing the reconstruction error on the training data as

$$J_1(W_1, b_1, \tilde{W}_1, \tilde{b}_1) = \frac{1}{2N} \sum_{i=1}^N (\|\tilde{x}_i - x_i\|^2 + \|\tilde{y}_i^0 - y_i\|^2) \quad (7)$$

After QAE1 is pretrained, its encoder part is kept in the whole QAE network and  $\{h_1^1, h_2^1, \dots, h_N^1\}$  can be calculated. Moreover, they are then utilized to extract the second-level feature data  $\{h_1^2, h_2^2, \dots, h_N^2\}$ . For the remaining levels of features, they can be progressively obtained by the following similar way. Here, assume the  $k$ th ( $k=1, 2, \dots, K-1$ ) layer of feature data

$\{h_1^k, h_2^k, \dots, h_N^k\}$  has already been learned. They will be utilized to obtain the  $(k+1)$ th-level feature data  $\{h_1^{k+1}, h_2^{k+1}, \dots, h_N^{k+1}\}$  by nonlinear activation function  $f$  with parameter set  $\{W_{k+1}, b_{k+1}\}$ . To this end, QAE( $k+1$ ) is built to reconstruct both the  $k$ th-level feature data and the quality data at its output layer, which can be denoted as  $\{\tilde{h}_1^k, \tilde{h}_2^k, \dots, \tilde{h}_N^k\}$  and  $\{\tilde{y}_1^k, \tilde{y}_2^k, \dots, \tilde{y}_N^k\}$ , respectively. Similarly, this QAE is trained by minimizing the reconstruction error for the  $k$ th-level feature data and the quality data as

$$J_{k+1}(W_{k+1}, b_{k+1}, \tilde{W}_{k+1}, \tilde{b}_{k+1}) = \frac{1}{2N} \sum_{i=1}^N \left( \|\tilde{h}_i^k - h_i^k\|^2 + \|\tilde{y}_i^k - y_i\|^2 \right) \quad (8)$$

$$, k = 2, 3, \dots, K-1$$

In this way, SQAE is pretrained from the bottom to the top hidden layers. In each layer, the features are learned from their adjacent low-level ones with additional constraint that the learned features can largely predict the quality variables. Hence, the irrelevant features are gradually reduced and quality-relevant features are progressively learned layer by layer.

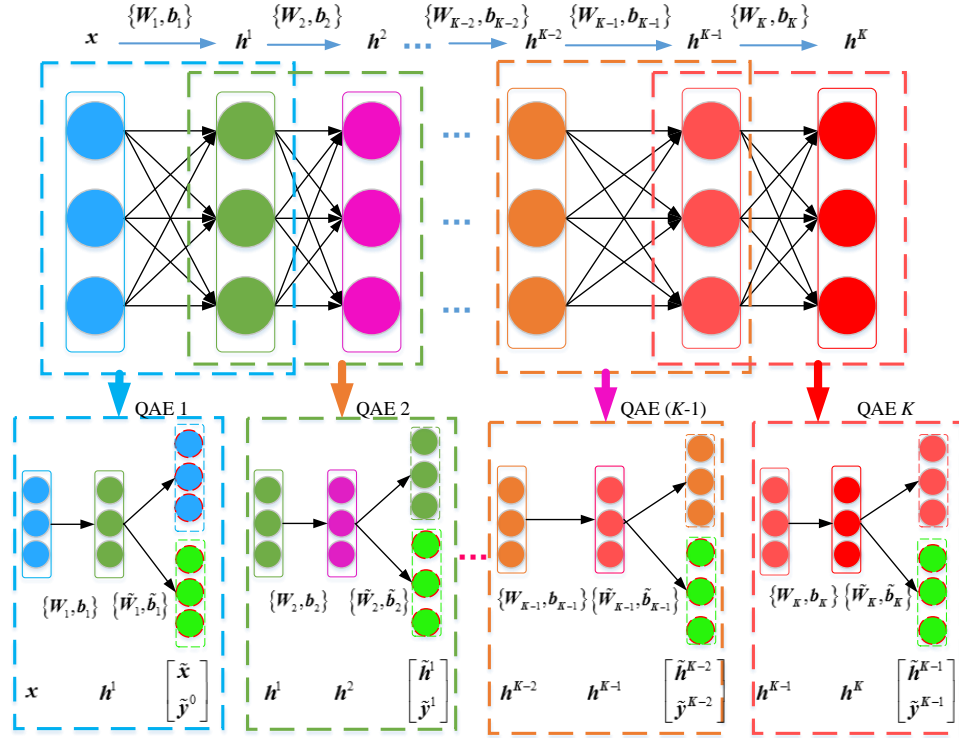


Fig. 4 The pretraining procedure of deep SQAE

### C. SQAE for soft sensor modeling

With SQAE network, it is much more effective and efficient to utilize the deep quality-relevant features for soft sensor modeling. For soft sensing, regression models can be directly constructed between the quality variables and the learned quality-relevant features. To build the regression models, the least square regression (LSR), two-layer neural network (NN) [27] and least square support vector machine (LSSVM) [40] are utilized as three classical regression algorithms. Given the raw training input data  $X = \{x_1, x_2, \dots, x_N\}$  and the quality data  $Y = \{y_1, y_2, \dots, y_N\}$ , as well as the testing input data  $X_t = \{x_{t1}, x_{t2}, \dots, x_{tN_t}\}$ , the detailed SQAE-based feature learning and soft sensor modeling procedures are list in TABLE I as follows.

TABLE I The training and testing procedure of SQAE-based soft sensor modeling

#### Training stage:

>Determine the number of feature layers  $K$  in SQAE.

>Train QAE1 with  $X$  and  $Y$  by BP algorithm to obtain its encoder network parameter  $\{W_1, b_1\}$  and the first-level quality-related feature data  $H^1 = \{h_1^1, h_2^1, \dots, h_N^1\}$ .

> set  $k=2$ ;

>Main loop:

While  $k \leq K$ , repeat:

-Train QAE  $k$  with  $H^{k-1}$  and  $Y$ . After that, the weight parameter of its encoder part is  $\{W_k, b_k\}$ . Meanwhile, the  $k$ th-level feature data are  $H^k = \{h_1^k, h_2^k, \dots, h_N^k\}$ .

-Set  $k=k+1$ ;

> After SQAE is layer-wise pretrained, the top-layer features  $H^K = \{h_1^K, h_2^K, \dots, h_N^K\}$  are used to predict the quality data.

> Build and train LSR, NN and LSSVM regression models between  $Y$  and  $H^K$ , respectively.

#### Testing stage:



> Substitute the testing input data into the input layer of the SQA network. Then, obtain its top-layer feature data  $H_t^K = \{h_{t_1}^K, h_{t_2}^K, \dots, h_{t_{N_t}}^K\}$  by forward propagation.

> The quality variable can be predicted as  $\hat{Y}_t = \{\hat{y}_{t_1}, \hat{y}_{t_2}, \dots, \hat{y}_{t_{N_t}}\}$  for the testing dataset with the trained LSR, NN and LSSVM models.

To assess the performance of the soft sensor model, the root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE) and coefficient of determination  $R^2$  are calculated on the testing dataset, the definition of which can be found in reference [27, 41].

#### IV. CASE STUDY

To assess the performance of the SQA network, it is applied to an industrial debutanizer column process in this section [42]. Firstly, a general description is given for the debutanizer column. Then, deep SQA-based soft sensors are utilized to predict the butane content.

##### A. Debutanizer column

The petrochemical industry, especially the refining process, is a production facility that composes of a number of connected chemical processing units and operations to produce specific materials or convert raw materials into valuable products. It is often characterized as a highly complex process equipped with modern control systems. The debutanizer column is one important part of the industrial refinery process, which is designed for desulfuration and naphtha split [42]. The flowchart of the debutanizer column is shown in Fig. 5. The debutanizer column mainly consists of six devices, which are the heat exchanger, bottom reboiler, overhead condenser, feed pump to the LPG splitter, head reflux pump and reflux accumulator. To keep stable production and ensure product quality, it is required to maximize the stabilized gasoline content (C5) in the overheads and minimize the butane content (C4) in the debutanizer bottoms, which are the feeds of the subsequent light pressure gas splitter and naphtha splitter, respectively. Hence, it is necessary to identify the C4 content online for effective control. Nevertheless, C4 concentration is actually not measured at the bottom flow of the debutanizer column directly, but on the overheads of the subsequent deisopentanizer column by gas chromatograph, which often results in large time delay.

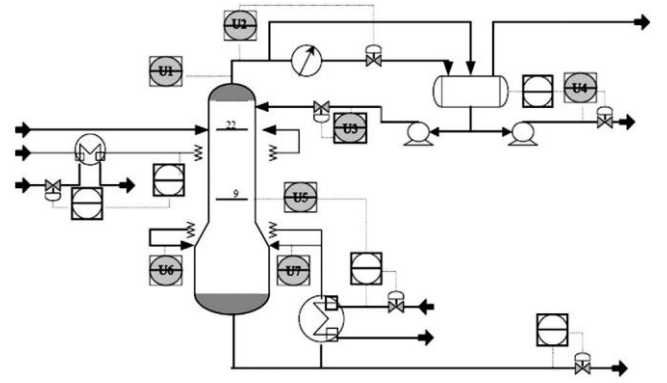


Fig. 5 The schematic flowchart of the debutanizer column[43]

Usually, the measurement technique with gas chromatograph cannot ensure the real-time control and optimization of the debutanizer column. Moreover, the measuring devices also suffer from offline examination and maintenance. To alleviate these problems, soft sensor strategies are adopted to predict the C4 concentration in the bottom. Thus, seven routinely measured process variables are selected for data modeling, which are given in TABLE II for their descriptions.

TABLE II Variable description for the debutanizer column

Input variables	Variable description
$u_1$	Top temperature
$u_2$	Top pressure
$u_3$	Reflux flow
$u_4$	Flow to next process
$u_5$	6 <sup>th</sup> tray temperature
$u_6$	Bottom temperature A
$u_7$	Bottom temperature B

##### B. Results and discussion

To deal with complex nonlinearities, deep learning is adopted for hierarchical feature learning. To take process dynamics into consideration, the augmented variables  $[u_1(k), u_2(k), u_3(k), u_4(k), u_5(k), u_5(k-1), u_5(k-2), u_5(k-3)]^T$  are used as the raw observed input variables of the deep SQA network at the  $k$ th sampling instant [42]. A total of 2390 data samples are collected in this process. For model training and testing, 1000 samples are used as the training dataset and the remaining ones are for the testing dataset.

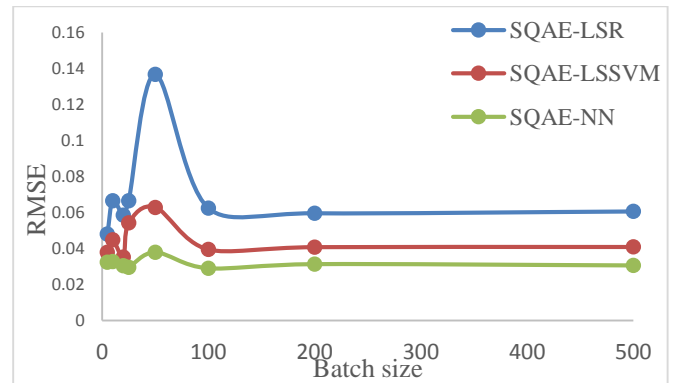


Fig. 6 The prediction RMSE for the three SQA-based models on the testing dataset with different pretraining batch size

To construct the SQA-based soft sensor model, it is necessary to first configure the network structure, which is

determined as [13 10 7 4] from reference [27]. The number of neurons in the input layer of SQAE is thirteen since there are 13 variables in the augmented variable vector. The practical guide of Bengio [44] is a good way to determine some commonly used hyperparameters. For example, the sigmoid function is used as the activation function for the neurons. The learning rate is set to 1. For some other important parameters like the batch size of pretraining, the trial and error technique is exploited to set proper parameter values. Here, different batch sizes from the range [5 10 20 25 50 100 200] in the pretraining stage are taken for example to show the trend of performance changes of the three SQAE-based soft sensor models, which is shown in Fig. 6. As can be seen, the general trend of the prediction RMSE first gets large with the increase of the batch size. When the batch size is 50, the performance is the worst for all the three methods. After that, the performance will decrease and then keep at a relatively stable level. From this, a proper batch size is better to set in a range from 5 to 25. In this study, it is set to 20.

With this batch size, convergence analysis is first carried out for the pretraining procedure of SQAE. Fig. 7 gives the learning loss in the layer-wise pretraining of each QAE within 50 epochs. As can be seen, the three QAEs can converge very fast in the pretraining stage. For QAE1 and QAE2, they can reach the convergent state after about 25 iterations. While for QAE3, it converges within 10 iterations. Furthermore, convergence performance is analyzed for the fine tuning procedure after pretraining is finished. Fig. 8 shows the learning training error for the SQAE, in which the whole network can converge very fast after about 50 epochs.

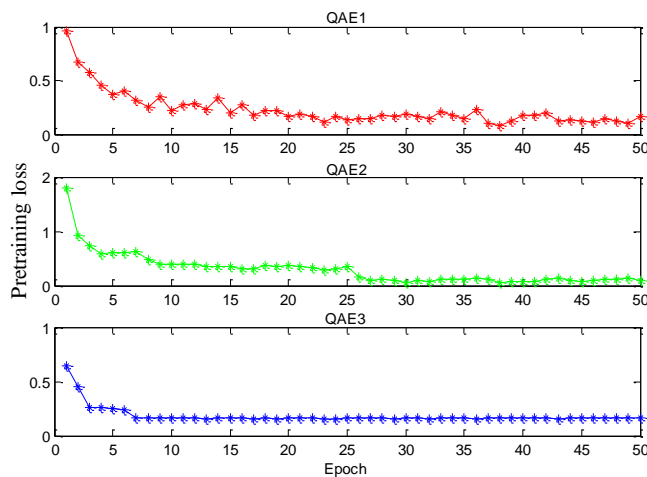


Fig. 7 The convergence analysis for pretraining of SQAE

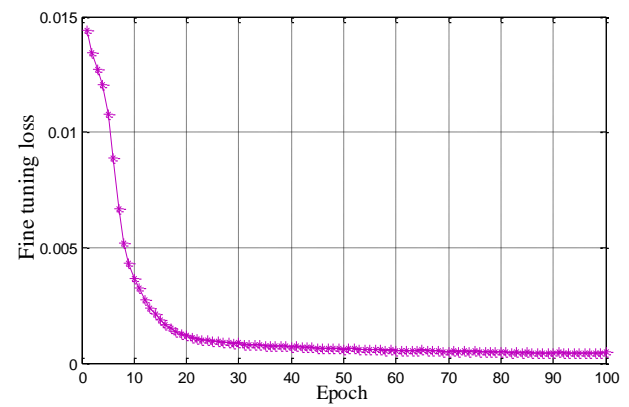


Fig. 8 The learning loss curve of fine-tuning for SQAE

After feature learning, the quality values can be predicted by substituting the features to the trained LSR, NN and LSSVM models. For performance comparison, the original SAE and VWSAE [27] are also adopted for feature representation for the raw observed input data, followed by LSR, NN and LSSVM regression models. Denote the nine approaches as SAE-LSR, SAE-NN, SAE-LSSVM, VWSAE-LSR, VWSAE-NN, VWSAE-LSSVM, SQAE-LSR, SQAE-NN and SQAE-LSSVM. TABLE III shows the prediction performance of the nine methods on the testing dataset.

TABLE III Comparison of prediction performance with nine different modeling methods

Method	RMSE	MAE	MAPE	R <sup>2</sup>
SAE-LSR	0.1105	0.0767	0.4401	0.5287
VWSAE-LSR	0.0597	0.0471	0.3261	0.8622
SQAE-LSR	0.0584	0.0435	0.3221	0.8682
SAE-NN [27]	0.0438	0.0304	0.3154	0.9260
VWSAE-NN [27]	0.0379	0.0277	0.2680	0.9444
SQAE-NN	0.0303	0.0220	0.2331	0.9646
SAE-LSSVM	0.0573	0.0419	0.3209	0.8734
VWSAE-LSSVM	0.0396	0.0342	0.2764	0.9395
SQAE-LSSVM	0.0352	0.0284	0.2552	0.9521

From the comparative results of RMSE, MAPE, MAE and R<sup>2</sup> indices, it can be easily seen that SQAE-LSR can achieve the best prediction accuracy among the first three models in TABLE III. Also, when the same NN regression model is used, SQAE-NN achieves better performance than VWSAE-NN and SAE-NN. Moreover, SQAE-LSSVM outperforms SAE-LSSVM and VWSAE-LSSVM in the four prediction indices. As for SAE, it only focuses on feature representation for the raw observed input data itself. The learned feature may contain a lot of irrelevant information with the quality variable, which can deteriorate the prediction performance of regression models. By taking the variable correlations with the quality variable into account, the weighted reconstruction objective is designed to pretrain each autoencoder in VWSAE. Thus, VWSAE can reduce irrelevant features by giving small weights to less important variables in the reconstruction objective. In this way, VWSAE-based models can provide much better prediction performance than the corresponding SAE-based approaches when the same regression model is used. However, VWSAE only measures the Pearson linear variable correlations, the nonlinear correlations are not fully considered in the

variable-wise weighted reconstruction objective. Hence, it is not sufficient for the learning of deep quality-relevant features that typically have nonlinear relation with the quality variable. Differently, SQAQ learns the features directly with the reference of the quality data by predicting it at the output layer of each QAE. It does not need to measure the variable correlations at each layer. Hence, the quality-relevant features are extracted more sufficiently. In a stacking way, the learned hierarchical features are more suitable for prediction of the quality variable. Therefore, when the same regression method is used, SQAQ-based methods can provide better predictions than the corresponding SAE-based and VWSAE-based ones. Also, one can easily conclude that for the same feature extraction network, LSSVM and NN can achieve much better prediction accuracy than LSR since they can deal with more complex nonlinear data relationships.

Furthermore, the detailed prediction results of SAE-LSSVM, VWSAE-LSSVM and SQAQ-LSSVM are taken as examples to investigate the prediction performance on the testing dataset. Fig. 9 shows the scatterplots of the predicted and labeled values for the three methods. In this figure, each data point represents a testing sample. The point values of horizontal and vertical axes represent the corresponding labeled and predicted values of the quality variable, respectively. From Fig. 9, it is easily seen that the prediction points of SQAQ-LSSVM lie much tighter alongside the diagonal line than these of SAE-LSSVM and VWSAE-LSSVM. Thus, SQAQ is more capable of learn quality-relevant features for soft sensor regression.

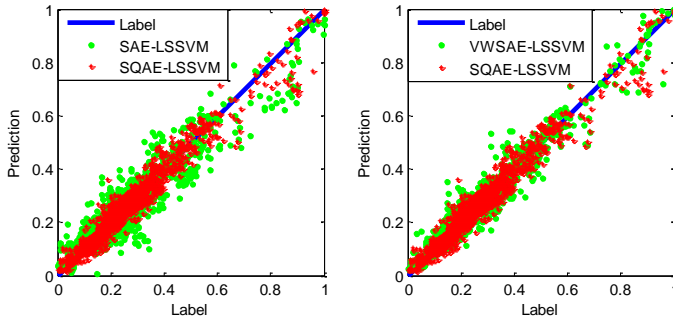


Fig. 9 The scatterplots of predicted and labeled values for the quality variable with SAE-LSSVM, VWSAE-LSSVM and SQAQ-LSSVM

Also, the computational performance is analyzed for the pretraining of SAE, VWSAE and SQAQ. Here, the batch gradient descent algorithm is used to pretraining each layer of the deep networks. The batch size is set to 20. For pretraining of each layer, different epochs are adopted to investigate the computational cost. TABLE IV shows the pretraining running time for the three deep networks with different epochs. As can be seen, for different epochs, SAE has the smallest pretraining time. VWSAE and SQAQ have comparable running time. This may be because VWSAE needs to carry out pair-wise variable correlation analysis in the pretraining of each layer. While for SQAQ, the quality variable is additionally utilized to guide feature learning. It enlarges the network structure and increases the number of parameters at the decoder part in the pretraining, which may also result in the increase of computational cost. Moreover, the total pretraining time is almost in proportion to

the epochs in pretraining stage.

TABLE IV Pretraining time of the three different deep networks

Epochs	SAE	VWSAE	SQAQ
1	0.0062	0.0789	0.0785
10	0.0591	0.6722	0.7435
20	0.1103	1.3338	1.2857
50	0.2664	2.8775	3.2128
100	0.6325	6.1196	6.2972
200	1.3372	11.5767	13.2530

As can be seen, QAE is a very different model from the widely used regular AE, denoising AE and sparse AE, etc. For these traditional AE-based models, they have the same network structure of basic autoencoder, in which the output layer tries to reconstruct its input data with certain desirable properties like robustness or sparsity. Hence, they are unsupervised learning networks, in which they cannot ensure the features are quality-relevant. Different from them, QAE directly utilizes the quality variables to construct a new partially supervised network as shown in Fig. 4. In QAE, the output layer consists of not only its input variables but also the quality variables. With such a structure, the learned features should not only represent their input data, but also largely predict the quality variables. In this way, QAE can learn quality-relevant features due to its partially supervised characteristic. Therefore, it can obtain different performance when QAE are used to construct deep SQAQ network. Here, the performance of SQAQ is further compared with traditional SAE, stacked denoising AE (SDAE), stacked sparse AE (SSAE).

TABLE V RMSE performance comparison of SQAQ with traditional AE-based deep networks

Method	SAE	SDAE	SSAE	SQAQ
RMSE	0.0438	0.0427	0.0412	0.0303

Taking the RMSE index for example, TABLE V shows the prediction performance on the testing dataset with the four different methods. For SDAE, the performance achieves the best with 20% zero marking noise for the inputs. For SSAE, the sparsity parameter is set to 0.05. As can be seen, the RMSE performance is very similar for SAE, SDAE and SSAE. This is because they are constructed in an unsupervised way at the pretraining stage. They cannot learn features that are related to the quality information. Differently, SQAQ can improve the performance effectively since it utilizes the quality data to guide the layer-wise feature learning in each layer. Thus, hierarchical quality-relevant features can be properly learned for better quality prediction.

## V. CONCLUSION

In this paper, a novel deep network is developed for quality-relevant feature learning and soft sensor model development, which is based on stacked quality-driven autoencoder. As most existing deep learning networks mainly focus on the feature representation for the raw observed input data, it may contain irrelevant information and cannot discover quality-relevant features for soft sensor modeling. To handle this problem, SQAQ aims to utilize the



quality data to guide the learning of potential features with constraint that the features can reconstruct both the quality data and the input-layer data as good as possible at each QAE. Thus, quality-relevant features can be well preserved and irrelevant information is gradually reduced with the hierarchically stacked structure of SQAEE. Then, the learned quality-relevant deep features can be directly used to build regression models for quality prediction. The industrial application of SQAEE-based soft sensor shows its effectiveness and flexibility over the original SAE-based and VWSAE-based ones. In real implementation, since the dimension of the quality variables may be much smaller than that of the input variables in each QAE, the second part of prediction error often have less influence on the total pretraining objective function in Eq. 6. To enhance the importance of the quality prediction error, an adjustable weight can be added to the second part of the loss function in order to learn quality-relevant features.

# REFERENCES

- [1] Z. Ge, "Review on data-driven modeling and monitoring for plant-wide industrial processes," *Chemometr. Intell. Lab. Syst.*, vol. 175, pp. 16-25, 2017.
- [2] B. Huang, and R. Kadali, *Dynamic Modeling, Predictive Control and Performance Monitoring*: Springer London, 2008.
- [3] X. Wang, B. Huang, and T. Chen, "Multirate Minimum Variance Control Design and Control Performance Assessment: A Data-Driven Subspace Approach," *IEEE. T. Contr. Syst. T.*, vol. 15, no. 1, pp. 65-74, 2006.
- [4] L. Zhou, J. Zheng, Z. Ge, Z. Song, and S. Shan, "Multimode Process Monitoring Based on Switching Autoregressive Dynamic Latent Variable Model," *IEEE. T. Ind. Electron.*, vol. 65, no. 10, pp. 8184-8194, 2018.
- [5] 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. T. Ind. Electron.*, vol. 65, no. 2, pp. 1559-1567, 2018.
- [6] 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.
- [7] P. Kadlec, B. Gabrys, and S. Strandt, "Data-driven soft sensors in the process industry," *Comput. Chem. Eng.*, vol. 33, no. 4, pp. 795-814, 2009.
- [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] Y. Wang, K. Sun, X. Yuan, Y. Cao, L. Li, and H. Koivo, "A Novel Sliding Window PCA-IPF based Steady-State Detection Framework and Its Industrial Application," *IEEE Access*, vol. 6, pp. 20995 - 21004 2018.
- [10] K. Hazama, and M. Kano, "Covariance-based locally weighted partial least squares for high-performance adaptive modeling," *Chemometr. Intell. Lab. Syst.*, vol. 146, pp. 55-62, 2015.
- [11] 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 T. Instrum. Meas.*, vol. 56, no. 1, pp. 95-101, 2007.
- [12] H. Kaneko, and K. Funatsu, "Adaptive soft sensor based on online support vector regression and Bayesian ensemble learning for various states in chemical plants," *Chemometr. Intell. Lab. Syst.*, vol. 137, pp. 57-66, 2014.
- [13] N. Chen, J. Dai, X. Yuan, W. Gui, W. Ren, and H. N. Koivo, "Temperature Prediction Model for Roller Kiln by ALD-Based Double Locally Weighted Kernel Principal Component Regression" *IEEE T. Instrum. Meas.*, vol. 67, no. 8, pp. 2001 - 2010 2018.
- [14] W. Shao, Z. Ge, and Z. Song, "Quality variable prediction for chemical processes based on semisupervised Dirichlet process mixture of Gaussians," *Chem. Eng. Sci.*, vol. 193, pp. 394-410, 2019.
- [15] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural. Comput.*, vol. 18, no. 7, pp. 1527-1554, 2006.
- [16] 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.
- [17] Y. Bengio, A. Courville, and P. Vincent, "Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives," 2012.
- [18] P. N. Druzhkov, and V. D. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognition & Image Analysis*, vol. 26, no. 1, pp. 9-15, 2016.
- [19] R. Sarikaya, G. E. Hinton, and A. Deoras, "Application of Deep Belief Networks for Natural Language Understanding," *IEEE/ACM Transactions on Audio Speech & Language Processing*, vol. 22, no. 4, pp. 778-784, 2014.
- [20] 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.
- [21] 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 T.*, pp. DOI: 10.1016/j.isatra.2019.07.001, 2019.
- [22] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol, "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371-3408, 2010.
- [23] L. O. Chua, and T. Roska, "CNN paradigm," *IEEE Transactions on Circuits & Systems I Fundamental Theory & Applications*, vol. 40, no. 3, pp. 147-156, 1993.
- [24] Y. Sun, X. Wang, and X. Tang, "Deep Learning Face Representation by Joint Identification-Verification," *Advances in Neural Information Processing Systems*, vol. 27, pp. 1988-1996, 2014.
- [25] 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.
- [26] D. Yu, G. Hinton, N. Morgan, and J. T. Chien, "Introduction to the Special Section on Deep Learning for Speech and Language Processing," *IEEE Transactions on Audio Speech & Language Processing*, vol. 20, no. 1, pp. 4-6, 2012.
- [27] X. Yuan, B. Huang, Y. Wang, C. Yang, and W. Gui, "Deep learning based feature representation and its application for soft sensor modeling with variable-wise weighted SAE," *IEEE. T. Ind. Inf.*, vol. 14, no. 7, pp. 3235-3243, 2018.
- [28] L. Yao, and Z. Ge, "Deep Learning of Semi-supervised Process Data with Hierarchical Extreme Learning Machine and Soft Sensor Application," *IEEE. T. Ind. Electron.*, vol. 65, no. 2, pp. 1490-1498, 2018.
- [29] W. Yan, D. Tang, and Y. Lin, "A Data-Driven Soft Sensor Modeling Method Based on Deep Learning and its Application," *IEEE. T. Ind. Electron.*, vol. 64, no. 5, pp. 4237-4245, 2017.
- [30] X. Yuan, L. Li, and Y. Wang, "Nonlinear dynamic soft sensor modeling with supervised long short-term memory network," *IEEE. T. Ind. Inf.*, vol. 10.1109/TII.2019.2902129 2019.
- [31] Y. Liu, C. Yang, Z. Gao, and Y. Yao, "Ensemble deep kernel learning with application to quality prediction in industrial polymerization processes," *Chemometr. Intell. Lab. Syst.*, vol. 174, pp. 15-21, 2018.
- [32] L. Jiang, Z. Ge, and Z. Song, "Semi-supervised fault classification based on dynamic Sparse Stacked auto-encoders model," *Chemometrics & Intelligent Laboratory Systems*, vol. 168, pp. 72-83, 2017.
- [33] H. Wu, and J. Zhao, "Deep convolutional neural network model based chemical process fault diagnosis," *Comput. Chem. Eng.*, 2018.
- [34] V. Gopakumar, S. Tiwari, and I. Rahman, "A Deep Learning based Data Driven Soft Sensor for Bioprocesses," *Biochem. Eng. J.*, vol. 136, pp. 28-39, 2018.
- [35] B. Andò, S. Graziani, and M. G. Xibilia, "Low-order Nonlinear Finite-Impulse Response Soft Sensors for Ionic Electroactive Actuators Based on Deep Learning" *IEEE T. Instrum. Meas.*, vol. 68, no. 5, pp. 1637 - 1646 2018.
- [36] A. Makhzani, and B. Frey, "k-Sparse Autoencoders," *Comput. Sci.*, 2014.
- [37] Z. Ge, "Supervised Latent Factor Analysis for Process Data Regression Modeling and Soft Sensor Application," *IEEE. T. Contr. Syst. T.*, vol. 24, no. 3, pp. 1004-1011, 2016.
- [38] Z. Ge, and X. Chen, "Supervised linear dynamic system model for quality related fault detection in dynamic processes," *J. Process Contr.*, vol. 44, pp. 224-235, 2016.
- [39] L. Zhou, J. Chen, Z. Song, Z. Ge, and A. Miao, "Probabilistic latent variable regression model for process-quality monitoring," *Chem. Eng. Sci.*, vol. 116, pp. 296-305, 2014.

- [40] H. Q. Wang, F. C. Sun, Y. N. Cai, L. G. Ding, and N. Chen, "An unbiased LSSVM model for classification and regression," *Soft Computing*, vol. 14, no. 2, pp. 171-180, 2010.
- [41] Y. Xu, W. Yang, and J. Wang, "Air quality early-warning system for cities in China," *Atmos. Environ.*, vol. 148, pp. 239-257, 2017.
- [42] L. Fortuna, S. Graziani, and M. G. Xibilia, "Soft sensors for product quality monitoring in debutanizer distillation columns," *Control Eng. Pract.*, vol. 13, no. 4, pp. 499-508, 2005.
- [43] L. Fortuna, S. Graziani, A. Rizzo, and M. G. Xibilia, *Soft sensors for monitoring and control of industrial processes*: Springer Science & Business Media, 2007.
- [44] Y. Bengio, "Practical Recommendations for Gradient-Based Training of Deep Architectures," *Neural networks: Tricks of the trade*, Berlin, Heidelberg: Springer, 2012.

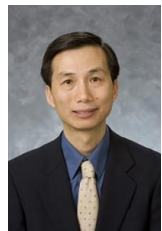


**Xiaofeng Yuan(M'17)** received the B.Eng. and Ph.D. degrees from the Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2011 and 2016, respectively.

He was a visiting scholar with the Department of Chemical and Materials Engineering, University of Alberta, Edmonton, AB, Canada, from November 2014 to May 2015. He is currently an Associate Professor with the School of Automation, Central south University. His research interests include deep learning and artificial intelligence, machine learning and pattern recognition, industrial process soft sensor modeling, process data analysis, etc.



**Jiao Zhou** is a postgraduate student at the School of Automation, Central South University, Changsha, China. She received her B.Eng. degree in School of Electrical Engineering from Nanhua University, Hengyang, China, in 2017. Her research interests include machine learning, soft sensor modeling, process data mining, etc.



**Biao Huang (M'97-SM'11-F'17)** received the B.Sc. and M.Sc. degrees from Beijing University of Aeronautics and Astronautics, Beijing, China, in 1983 and 1986, respectively, and the Ph.D. degree from the University of Alberta, Edmonton, AB, Canada, in 1997.

He is currently a Professor in the Department of Chemical and Materials Engineering, the University of Alberta. His research interests are in the areas of process control, state estimation, soft sensors, system identification, control performance assessment, fuel cell and biomedical modeling. Dr. Huang is a Editor-in-Chief of Control Engineering Practice, an Associate Editor of Journal of Process Control and Canadian Journal of Chemical Engineering. He is a Fellow of IEEE, a Fellow of the Canadian Academy of Engineering.



**Yalin Wang(M'17)** received the B.Eng. and Ph.D. degrees from the Department of Control Science and Engineering, Central South University, Changsha, China, in 1995 and 2001, respectively.

Since 2003, she has been with the School of Information Science and Engineering, Central south University, where she was at first an Associate Professor and is currently a Professor. Her research interests include the modeling, optimization and control for complex industrial processes, intelligent control, and process simulation.



**Chunhua Yang(M'09)** received the M.Eng. degree in automatic control engineering and the Ph.D. degree in control science and engineering from Central South University, Changsha, China, in 1988 and 2002, respectively. She was with the Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium, from 1999 to 2001. She is currently a Full Professor with Central South University. Her current research interests include modeling and optimal control of complex industrial process, intelligent control system, and fault-tolerant computing of real-time systems.



**Weihua Gui(M'09)** received the B.Eng. and M.Eng. degrees in control science and engineering from Central South University, Changsha, China, in 1976 and 1981, respectively.

From 1986 to 1988, he was a Visiting Scholar at the University Duisburg-Essen, Duisburg, Germany. He has been a Full Professor with the School of Information Science and Engineering, Central South University, since 1991. His main research interests include the modeling and optimal control of complex industrial processes, distributed robust control, and fault diagnoses.