Chemical Engineering Science 217 (2020) 115509

Contents lists available at ScienceDirect

Chemical Engineering Science

journal homepage: www.elsevier.com/locate/ces

A novel semi-supervised pre-training strategy for deep networks and its application for quality variable prediction in industrial processes

Xiaofeng Yuan, Chen Ou *, Yalin Wang *, Chunhua Yang, Weihua Gui

School of Automation, Central South University, Changsha, 410083 Hunan, PR China

HIGHLIGHTS

- A semi-supervised autoencoder (SS-AE) is first developed as the basic network to extract quality-related features.
- By hierarchically stacking multiple SS-AEs, a novel semi-supervised strategy is proposed for pretraining of deep networks.
- SS-SAE can learn deep hierarchical quality-relevant features from process data for quality prediction.
- High prediction performance and fast convergence ability of SS-SAE are validated on two refining industries.

ARTICLE INFO

Article history:
Received 6 May 2019
Received in revised form 25 December 2019
Accepted 24 January 2020
Available online 26 January 2020

Keywords:
Quality prediction
Soft sensor
Deep learning
Stacked autoencoder (SAE)
Semi-supervised SAE (SS-SAE)

ABSTRACT

Deep learning-based soft sensor has been a hot topic for quality variable prediction in modern industrial processes. Feature representation with deep learning is the key step to build an accurate and reliable soft sensor model from massive process data. To deal with the limited labeled data and abundant unlabeled data, a semi-supervised pre-training strategy is proposed for deep learning network in this paper, which is based on semi-supervised stacked autoencoder (SS-SAE). For traditional deep networks like SAE, the pre-training procedure is unsupervised and may discard important information in the labeled data. Different from them, SS-SAE automatically adjusts the training strategy according to the given data type. For unlabeled data, it learns the shape of the input distribution layer by layer. While for labeled data, it additionally learns quality-related features with the guidance of quality information. The proposed method is validated on two refining industries of a debutanizer column and a hydrocracking process. The results show that SS-SAE can utilize both labeled and unlabeled data to extract quality-relevant features for soft sensor modeling, which is superior to multi-layer neural network, traditional SAE and DBN.

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction

In modern industrial processes, various advanced monitoring, control and optimization technologies have been used to ensure the normal operation of industrial production, optimize the utilization efficiency of resources and alleviate the pressure of environmental pollution (Bosca and Fissore, 2011; Chen et al., 2018; Ge et al., 2013; Kano and Nakagawa, 2008; Liu et al., 2013). The development and implementation of these technologies often depend on the timely measurement of key product quality indicators. However, online measuring instruments for quality variables are often expensive and subjected to large time delay and severe environment. Hence, soft sensors have been developed to estimate the difficult-to-measure quality variables in real-time by establishing mathematical models with those easy-to-measure process variables (Shao et al., 2019; Shao and Tian, 2015; Yuan et al., 2019f). Soft sensors can be used to replace traditional hard sensors, serve as backup units, and calibrate the hard sensor. In the past decades, soft sensing technology has become a research hotspot in the field of process control (Kim et al., 2013; Yuan et al., 2018b). Generally, soft sensors can be divided into two types: first-principle models (FPM) (Dai et al., 2019; Huang et al., 2013) and data-driven models (Yuan et al., 2017a; Yuan et al., 2019d). For the first category, it seeks to construct accurate mathematical models by analyzing process physicochemical mechanisms. However, it is often expensive or even not available to obtain accurate first-principle models in modern complex industrial processes. In contrast, data-driven modeling technology does not require much process background knowledge (Kadlec et al., 2009; Zhou et al., 2014). With the wide applications of process sensor platform, intelligent instruments, distributed control system (DCS) and computer storage

* Corresponding authors.
E-mail addresses: 657943838@qq.com (C. Ou), ylwang@csu.edu.cn (Y. Wang).

https://doi.org/10.1016/j.ces.2020.115509
0009-2509/© 2020 Elsevier Ltd. All rights reserved.

technologies, massive process data can be collected and stored in modern process industry, which lays a solid foundation for data-driven soft sensor modeling (Qin et al., 2019; Shardt et al., 2015; Zhu et al., 2015). During the last decades, many data-driven soft sensor modeling methods have been proposed, such as principal component regression (PCR) (Yuan et al., 2017b), partial least squares regression (PLSR) (Ma et al., 2015b), artificial neural networks (ANNs) (Maiti et al., 2018) and support vector regression (SVR) (Kaneko and Funatsu, 2014), etc.

Typically, the data collected in modern industrial processes are characterized by high dimensionality, complex correlation and high redundancy, which may lead to deterioration of prediction performance of soft sensor models. Hence, it is an important step to extract useful information from massive raw data for soft sensor modeling, which promotes the development of feature extraction. Principal component analysis (PCA) (Yuan et al., 2014), partial least squares (PLS) (Sharmin et al., 2006) and artificial neural networks (ANNs) (Farizhandi et al., 2016) are traditional popular feature representation models. However, most of these models can be considered as shallow learning structures with only one or zero hidden layer. Many studies have shown that deep networks are more efficient than shallow ones in extracting hierarchical features for complex processes (Su et al., 2019; Yoshua, 2009; Yuan et al., 2019c). Recently, revolutionary deep learning techniques have been proposed for training of deep networks. For example, stacked autoencoder (SAE) (Yu et al., 2018) is a typical deep network by stacking a number of autoencoder (AE) units. Then, the deep learning technique uses the unsupervised layer-wise pre-training and supervised fine-tuning stages to train SAE. The pre-training procedure draws lessons from the multi-layer abstract function of the human brain. That is, extracting advanced features of data layer-by-layer. The layer-wise pre-training method also alleviates the gradient vanishing or exploding problem in model training of deep networks (Vincent et al., 2010).

Recently, unsupervised layer-wise pre-training techniques have also been used for deep learning-based process data-driven modeling like soft sensor, process monitoring and fault diagnosis. For example, deep belief network (DBN) was used for soft sensor modeling of a crude distillation unit to estimate the heavy diesel 95% cut point (Shang et al., 2014). A hierarchical extreme learning machine (HELM) was proposed to exploit additional information in unlabeled process samples for unsupervised pretraining (Yao and Ge, 2018). An unsupervised extended deep belief network (EDBN) was proposed to obtain useful information in the process data (Wang et al., 2019a).

However, the unsupervised pre-training only uses unlabeled data for feature learning, which cannot ensure the features are really relevant for regression or classification tasks. Hence, the subsequent prediction tasks may not benefit from unsupervised learning (Erhan et al., 2010). It is natural to carry out supervised pre-training with sample labels for deep networks. For example, a supervised LSTM network was proposed for feature learning of quality-related hidden dynamics in inferential modeling (Yuan et al., 2019b). A variable-wise weighted SAE was proposed for hierarchical output-related feature representation, in which a weighted objective function is designed for pre-training of each layer (Yuan et al., 2018a). Also, a supervised autoencoder was constructed to learn features for image classification, which seeks to represent face images of the same person similarly (Gao et al., 2015). In this model, image label information is used to construct similarity preservation term so that features of the same person should be the same or very similar. However, the output labels are very difficult and expensive to obtain for all data samples. There are very few studies on supervised pre-training since it is almost impossible to obtain fully labeled training data in practice. Especially in industrial processes, the input variables for soft sensor modeling are process variables such as flowrate, temperature, and pressure, which typically have very high sampling frequency. While for the quality output variables, they are often sampled with low sampling rates. Therefore, unlabeled samples from practical industrial processes are far more than labeled ones. However, most traditional deep networks are pre-trained in an unsupervised way with only unlabeled data, in which the feature learning may be blind to some degree. For supervised pre-training, it is usually unpractical in industrial processes since quality variables have much lower sampling frequency than the process input variables (Na et al., 2018; Shao et al., 2019). Thus, only a small number of labeled samples can be used for supervised pre-training, in which a large number of unlabeled data are not fully used.

To deal with the aforementioned problems, a deep learning modeling framework based on semi-supervised pre-training strategy is proposed for soft sensor development, which is based on semi-supervised stacked autoencoder (SS-SAE). In SS-SAE, all labeled and unlabeled samples can be fully utilized for model construction and pre-training. The whole SS-SAE modeling algorithm consists of two main stages: semi-supervised pre-training for feature extraction and supervised fine-tuning for quality variable prediction. The semi-supervised pre-training is achieved by stacking multiple semi-supervised autoencoders (SS-AE). In the basic SS-AE module, the output layer seeks to reconstruct its inputs and the quality output variables simultaneously. Both labeled and unlabeled data can be used to pre-train the SS-AE. The objective function consists of two parts for SS-AE. For the labeled training data, the objective function is to minimize the reconstruction error for both its input and quality output variables. While for unlabeled training data, the objective function only minimizes the reconstruction error for the input parts. Thus, the network parameters are determined by both labeled and unlabeled training samples. Then, multiple SS-AEs can be hierarchically stacked to construct deep semi-supervised stacked autoencoder (SS-SAE) for semi-supervised feature representation. After that, the supervised NN is constructed for quality prediction by adding a quality variable layer to the top hidden layer of SS-SAE. Therefore, the SS-SAE method can not only utilize the labeled samples, but also make use of additional unlabeled data. Moreover, the learned features are more relevant to the quality variables since the labeled information is used to guide the pre-training procedure. The effectiveness and flexibility of the proposed SS-SAE for soft sensor is demonstrated on two industrial refining processes, which are a debutanizer column and a hydrocracking process.

The rest of this paper is organized as follows. In Section 2, a description of the stacked autoencoder is briefly presented. Then, the proposed SS-SAE approach is described in detail in Section 3. In addition, the soft sensor modeling procedure based on the proposed SS-SAE is also introduced in this section. Section 4 presents the performance evaluation of the proposed method on a debutanizer column and an industrial hydrocracking process. Finally, Section 5 gives the conclusion of this paper.

2. Stacked autoencoder

2.1. Autoencoder

The traditional autoencoder is an unsupervised single hidden layer neural network, in which the output layer seeks to reconstruct its input variables. Fig. 1 gives the structure illustration of an AE. It can be divided into two parts: the encoder and the decoder. The encoder part maps the original input $\mathbf{x}$ into the hidden representation $\mathbf{h}$. Typically, the hidden representation can be obtained by nonlinear mapping function f_θ as

$$\mathbf{h} = f_\theta(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

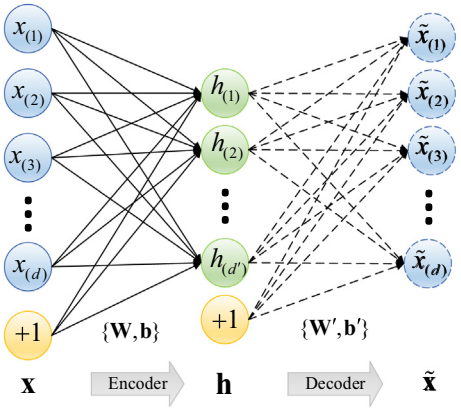

Fig. 1. Model structure of autoencoder.

where $\mathbf{W}$ is a$d' \times d$ weight matrix and $\mathbf{b} \in R^{d'}$ is the bias vector. s is the activation function. The parameter set of the encoder can be denoted as $\theta = \{\mathbf{W}, \mathbf{b}\}$.

In the decoder, the hidden representation $\mathbf{h}$ is mapped back to the reconstructed d-dimensional vector $\tilde{\mathbf{x}}$ by nonlinear mapping function $g_{\theta'}$ as

$$\tilde{\mathbf{x}} = g_{\theta'}(\mathbf{h}) = s(\mathbf{W}'\mathbf{h} + \mathbf{b}') \tag{2}$$

where $\mathbf{W}'$ is a $d \times d'$ weight matrix and $\mathbf{b}'$ is a bias vector of dimensionality d for the decoder. Also, the parameter set of the decoder can be denoted as$\theta' = \{\mathbf{W}', \mathbf{b}'\}$. Usually, the dimension of the hidden representation $\mathbf{h}$ is smaller than that of the input vector, and such an autoencoder acts as an effective compressor. In general, $\tilde{\mathbf{x}}$ is interpreted as an approximate reconstruction of $\mathbf{x}$. For this purpose, the loss function of an autoencoder is expressed as the reconstructed error

$$\underset{\theta,\theta'}{\operatorname{argmin}} \| g_{\theta'}(f_\theta(\mathbf{x})) - \mathbf{x} \|_2^2 \tag{3}$$

or equivalently

$$\underset{\theta,\theta'}{\operatorname{argmin}} \| \tilde{\mathbf{x}} - \mathbf{x} \|_2^2 \tag{4}$$

This is often a non-convex optimization problem. However, since the typical activation function s is smooth and continuously differentiable, it can be easily solved by gradient descent techniques.

Copying the input to the output sounds useless, but it doesn't care about the output of the decoder. Instead, it seeks to make the useful properties of feature representation$\mathbf{h}$by training the autoencoder to reconstruct the inputs. One way to obtain useful representation$\mathbf{h}$from an autoencoder is to limit the dimension of $\mathbf{h}$ to be less than $\mathbf{x}$, which is called an under-complete autoencoder (Vincent et al., 2010). Another approach is to encourage the model to learn other properties, including sparsity of the representation, smallness of the derivative of the representation, and missing inputs or robustness to noise, which is called a regularized autoencoder (Alain and Bengio, 2012).

2.2. Stacked autoencoder

In order to learn more complex and abstract features, the autoencoders can be used to build deep stacked autoencoders. SAE can be constructed by stacking multiple autoencoders in a hierarchical way, in which the raw input data is transmitted to the first level of the whole SAE network. The top level of the deep architecture obtains the final feature representation of the raw input data that can be used for complex prediction or classification tasks. This is often referred to as an unsupervised pre-training step. Fig. 2 shows the architecture of SAE.

3. Semi-supervised stacked autoencoder

As can be seen, traditional deep learning typically involves two steps: unsupervised layer-wise pre-training and supervised fine-tuning. Unsupervised layer-wise pre-training is used to obtain complex and abstract feature representations that can improve the performance of subsequent supervised learning (Hinton et al., 2006; Yuan et al., 2019e). In many recognition tasks, greedy layer-wise unsupervised pre-training can achieve significant improvements in classification performance. This observation began in 2006 with a renewed focus on deep neural networks (Bengio et al., 2007). In reference (Erhan et al., 2010), the authors explain why unsupervised pre-training is effective. That is, the learning algorithm can use the information learned in the unsupervised stage and perform better in the stage of supervised learning. The basic idea is that some features that are useful for unsupervised tasks may also be useful for supervising learning. However, this is not proved mathematically or theoretically. Hence, it is unknown which tasks can benefit from the form of unsupervised learning. Many aspects of this approach are highly dependent on the model used. For example, if one wants to add a linear classifier to the top hidden layer after pre-training, then the learned features must make the potential categories linearly separable. These properties usually occur naturally during unsupervised learning, but it does not always make sense. Reference (Ma et al., 2015a) studied the effects of pre-training on the prediction of chemical activity in machine learning models. As a result, on average, unsupervised pre-training has a slight negative impact. For soft sensor, it is very important to learn quality-related features for regression prediction. However, the unsupervised pretraining cannot ensure the learned features are relevant to the quality variable. On the contrary, there may be irrelevant information in the deep features. Supervised pre-training seems to be able to solve this problem (Gao et al., 2015; Yuan et al., 2019a), but labeled training data is so difficult to obtain. In most industrial processes, the amount of unlabeled data is much larger than the available labeled data. Hence, semi-supervised pre-training is designed to take advantage

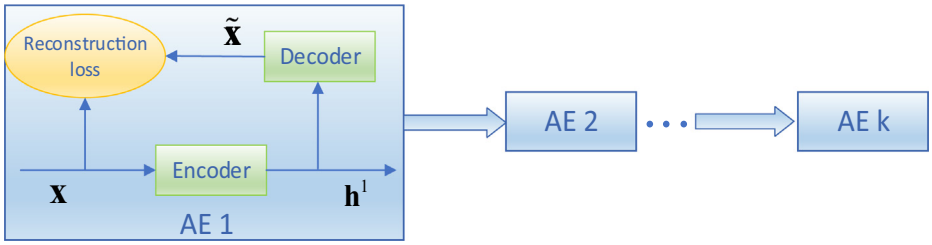

Fig. 2. Model structure of SAE.

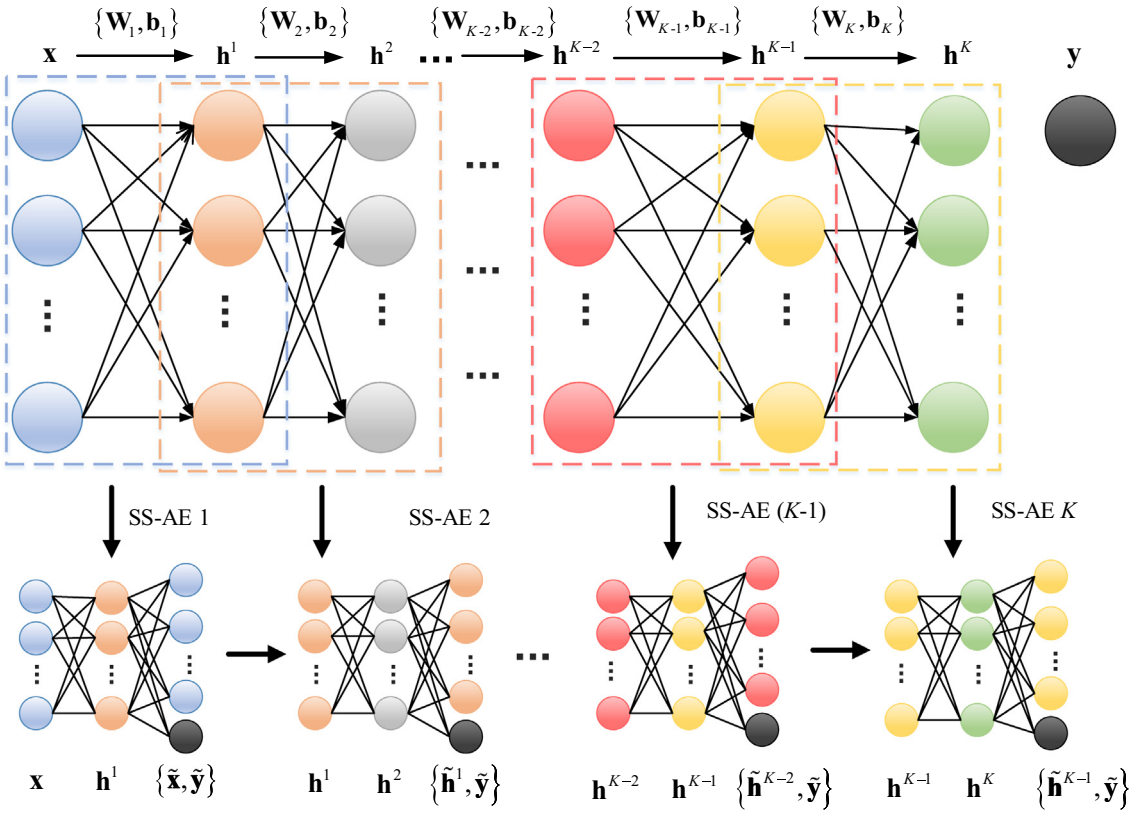

Fig. 3. The architecture of SS-SAE.

of all available samples in this paper. To this end, a new deep network structure is first designed with semi-supervised SAE.

3.1. Semi-supervised SAE

Motivated by above problems, it is recommended to exploited both labeled and unlabeled samples for pre-training of deep networks. In this way, a new semi-supervised stacked autoencoder is proposed for semi-supervised pre-training of deep networks, which can utilize all available process data to extract quality-related features for prediction. The architecture of SS-SAE is shown in Fig. 3, which is composed of multiple semi-supervised autoencoder (SS-AE) units.

Similar to the original autoencoder, the basic SS-AE consists of an encoder - which maps the original input to hidden representation and a decoder- which uses the hidden representation to reconstruct certain variables at the output layer. The main difference between the SS-AE and the original AE lies in that the reconstruction target is set as a combination of its input variables and the quality variables in SS-AE. In this way, SS-AE can be trained with both unlabeled data and labeled data. This is equivalent to the simultaneous execution of unsupervised and supervised learning, in which unsupervised learning preserves the information of the unlabeled data as much as possible, while supervised learning ensures the learning procedure can keep extraction of label-related information.

Assume $\mathbf{x}_u(u = 1, \ldots, N_u)$ are the unlabeled input samples. $\mathbf{x}_l(l = 1, \ldots, N_l)$ are the labeled input samples and $\mathbf{y}_l(l = 1, \ldots, N_l)$ represent the corresponding labeled quality samples. To construct the semi-supervised SAE, the pre-training procedure can be described as follows.

For the first SS-AE (SS-AE 1), the output layer seeks to simultaneously reconstruct the raw input variable $\mathbf{x}$ and predict the quality variable $\mathbf{y}$. As shown in Fig. 3, the reconstructed input and quality variable are denoted as $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$, respectively.

Thus, for the unlabeled training data $\mathbf{x}_u$, SS-AE 1 tries to reconstruct the input data as

$$\begin{aligned}\tilde{\mathbf{x}} &= g_{\theta'}(\mathbf{h}) = s(\mathbf{W}'_{1,x}\mathbf{h} + \mathbf{b}'_{1,x}) \\ &= s(\mathbf{W}'_{1,x}s(\mathbf{W}_1 x + \mathbf{b}_1) + \mathbf{b}'_{1,x})\end{aligned} \tag{5}$$

where $\mathbf{W}_1$ and $\mathbf{b}_1$ are the weight and bias terms of the encoder part, respectively. $\mathbf{W}'_{1,x}$ and $\mathbf{b}'_{1,x}$ are the weight and bias terms corresponding to the reconstructed inputs in the decoder, respectively.

For the labeled training data $(\mathbf{x}_l, \mathbf{y}_l)$, SS-AE 1 seeks to reconstruct the inputs and predict the outputs as

$$\begin{aligned}\begin{bmatrix}\tilde{\mathbf{x}} \\ \tilde{\mathbf{y}}\end{bmatrix} &= \left[s(\mathbf{W}'_{1,x}\mathbf{h} + \mathbf{b}'_{1,x})\right] = \begin{bmatrix}s(\mathbf{W}'_{1,x}\mathbf{h} + \mathbf{b}'_{1,x}) \\ s(\mathbf{W}'_{1,y}\mathbf{h} + \mathbf{b}'_{1,y})\end{bmatrix} \\ &= \begin{bmatrix}s(\mathbf{W}'_{1,x}s(\mathbf{W}_1 x + \mathbf{b}_1) + \mathbf{b}'_{1,x}) \\ s(\mathbf{W}'_{1,y}s(\mathbf{W}_1 x + \mathbf{b}_1) + \mathbf{b}'_{1,y})\end{bmatrix}\end{aligned} \tag{6}$$

where $\mathbf{W}'_{1,y}$ and $\mathbf{b}'_{1,y}$ are the weight and bias terms corresponding to the predicted outputs in the decoder, respectively. $\mathbf{W}'_1$ and $\mathbf{b}'_1$ are decoder weight and bias terms of the first SS-AE, respectively. Here, we have $\mathbf{W}'_1 = \begin{bmatrix}\mathbf{W}'_{1,x} \\ \mathbf{W}'_{1,y}\end{bmatrix}$ and $\mathbf{b}'_1 = \begin{bmatrix}\mathbf{b}'_{1,x} \\ \mathbf{b}'_{1,y}\end{bmatrix}$.

For SS-AE 1, it is pre-trained on all available labeled and unlabeled training data. For the lth labeled data, the output layer seeks to ensure that each reconstructed pair $\left(\tilde{\mathbf{x}}_l, \tilde{\mathbf{y}}_l\right)$ can keep similar to its real value $(\mathbf{x}_l, \mathbf{y}_l)$ as much as possible. While for the uth unlabeled training sample, the first SS-AE seeks the goal that $\tilde{\mathbf{x}}_u \approx \mathbf{x}_u$.

Hence, the reconstruction loss function of the first SS-AE is

$$\text{argmin}_{\theta,\theta',\eta}\frac{1}{2N_u}\sum_{u=1}^{N_u}\| g_{\theta'}(f_\theta(\mathbf{x}_u))-\mathbf{x}_u\|_2^2 + \frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\| g_{\theta'}(f_\theta(\mathbf{x}_l))-\mathbf{x}_l\|_2^2+\lambda\| g_\eta(f_\theta(\mathbf{x}_l))-\mathbf{y}_l\|_2^2\right) \quad (7)$$

where $\theta=\{\mathbf{W}_1,\mathbf{b}_1\}$, $\theta'=\{\mathbf{W'}_{1,x},\mathbf{b'}_{1,x}\}$ and $\eta=\{\mathbf{W'}_{1,y},\mathbf{b'}_{1,y}\}$. λ is the trade-off coefficient. The loss function is equivalent to

$$\text{argmin}_{\mathbf{W}_1,\mathbf{W'}_1,\mathbf{b}_1,\mathbf{b'}_1}\frac{1}{2N_u}\sum_{u=1}^{N_u}\| \tilde{\mathbf{x}}_u-\mathbf{x}_u\|_2^2 + \frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\| \tilde{\mathbf{x}}_l-\mathbf{x}_l\|_2^2+\lambda\| \tilde{\mathbf{y}}_l-\mathbf{y}_l\|_2^2\right) \quad (8)$$

Then, backpropagation (BP)-based gradient descent method is utilized to minimize the loss function by iteratively updating the parameter set $\{\mathbf{W}_1,\mathbf{W'}_1,\mathbf{b}_1,\mathbf{b'}_1\}$.

The properties of SS-SAE are as follows:

1) The first term in Eq. (8) is the reconstruction error item for input part of the unlabeled samples. It is expected that unsupervised pre-training can be helpful when the number of labeled samples is very small. Because unsupervised pre-training can extract part useful information from unlabeled data.
2) The rest two term in Eq. (8) are the input reconstruction error and quality prediction error for the labeled samples, respectively. Additional label information helps to improve the extraction capability of label-related feature over the basic autoencoder. It ensures that label-related features can be properly extracted from process variables with the guidance of quality information.
3) The trade-off parameter λ helps to balance the relative contributions of the last term of the labeled information in the optimization.

Deep networks typically have stronger feature extraction ability than shallow networks (Yoshua, 2009). Due to the layer-wise pre-training principle and the limited capacity of single-layer networks, label variable is added to each layer to ensure that the top-level feature representation is highly related to the predicted target variable. Hence, the basic SS-AE modules can be stacked hierarchically to construct the deep semi-supervised stacked autoencoder network. In SS-SAE, once a SS-AE has been pre-trained, the output of its hidden layer is used as the input for the next SS-AE, as shown in Fig. 3. Therefore, the proposed semi-supervised stacked autoencoder obtains a deep architecture through a layer-wise manner. Assume SS-AE (k-1) is constructed and pre-trained, its hidden feature data $\mathbf{h}_u^{k-1}(u=1,\ldots,N_u)$ and $\mathbf{h}_l^{k-1}(l=1,\ldots,N_l)$ can be calculated by forward propagation for the unlabeled and labeled samples, respectively. Then, the hidden layer feature $\mathbf{h}_u^{k-1}$ and $\mathbf{h}_l^{k-1}$ will be sent to the input layer of the next SS -AE (k) to obtain the high-level hidden feature $\mathbf{h}_u^k$ and $\mathbf{h}_l^k$. To pre-training SS-AE (k), the following reconstruction loss function is minimized for both labeled and unlabeled training samples

$$\text{argmin}_{\mathbf{W}_k,\mathbf{W'}_k,\mathbf{b}_k,\mathbf{b'}_k}\frac{1}{2N_u}\sum_{u=1}^{N_u}\| \tilde{\mathbf{h}}_u^{k-1}-\mathbf{h}_u^{k-1}\|_2^2 + \frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\| \tilde{\mathbf{h}}_l^{k-1}-\mathbf{h}_l^{k-1}\|_2^2+\lambda\| \tilde{\mathbf{y}}_l-\mathbf{y}_l\|_2^2\right) \quad (9)$$

where $\mathbf{W}_k$ and $\mathbf{W'}_k$ are the encoder and decoder weights of SS-AE (k), respectively; $\mathbf{b}_k$ and $\mathbf{b'}_k$ are the encoder and decoder bias of the first SS-AE (k), respectively. Once the whole network has been pre-trained, the output of the top layer will serve as final feature representation of the raw input data. Then, an additional output layer for quality variable is added to the top hidden feature layer for prediction. By performing forward propagation, the estimated output can be obtained as

$$\tilde{\mathbf{y}}_l=f_o\left(\mathbf{W}_o\mathbf{h}_l^K+\mathbf{b}_o\right) \quad (10)$$

where $\mathbf{W}_o$ and $\mathbf{b}_o$ are the weight matrix and bias vector of the additional output layer, respectively; f_o is the activation function of output layer. After forward propagation, the supervised fine-tuning procedure is performed by the BP-based gradient descent algorithm to minimize the errors between the estimated $\tilde{\mathbf{y}}_l$ and labeled output $\mathbf{y}_l$ as

$$J(\mathbf{W}_1,\mathbf{b}_1,\ldots,\mathbf{W}_K,\mathbf{b}_K,\mathbf{W}_o,\mathbf{b}_o)=\frac{1}{2N_l}\sum_{l=1}^{N_l}\| \tilde{\mathbf{y}}_l-\mathbf{y}_l\|^2 \quad (11)$$

3.2. SS-SAE based soft sensor

SS-SAE can be used to construct soft sensor model for quality prediction. The basic implementation procedure for the SS-SAE based soft sensor is summarized as follows.

Implementation procedure for SS-SAE based soft sensor

Input: unlabeled training samples $\{\mathbf{x}_u\}_{u=1\ldots N_u}$ and labeled training samples $\{\mathbf{x}_l,\mathbf{y}_l\}_{l=1\ldots N_l}$.

Output: $\mathbf{W}_i(i=1,\ldots,k,\ldots,K)$ and $\mathbf{b}_i(i=1,\ldots,k,\ldots,K)$**:** the weight matrix and bias vector connecting the $(i-1)$-th and i-th hidden layers, respectively; $\mathbf{W}_o$ and $\mathbf{b}_o$: the weight matrix and bias vector connecting the last hidden layer and the output layer, respectively; $\hat{\mathbf{y}}$: the predicted value of the quality variable.

Definitions:

$J^k(\theta)=\frac{1}{2N_u}\sum_{u=1}^{N_u}\| \tilde{\mathbf{h}}_u^{k-1}-\mathbf{h}_u^{k-1}\|_2^2+\frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\| \tilde{\mathbf{h}}_l^{k-1}-\mathbf{h}_l^{k-1}\|_2^2+\lambda\| \tilde{\mathbf{y}}_l^{k-1}-\mathbf{y}_l\|_2^2\right)$ is the objective function of the k-th SS-AE.

Step 1: Select secondary variables for soft sensor development based on process knowledge and operator experience.

Step 2: Collect and store data from the industrial processes. Determine the training, validation and testing datasets.

Step 3: Normalize the datasets before model construction.

Step 4: Set proper value of trade-off parameter of λ. Design ranges and candidates for hyper-parameters, like the learning rate, batch size and fine-tuning epoch, etc. For each combination of different hyper-parameters, carry out Step 5 and Step 6.

(*continued on next page*)

(*continued*)

Implementation procedure for SS-SAE based soft sensor

Step 5: Pre-train the semi-supervised stacked autoencoders as follow:

① Design and pre-train the first SS-AE 1 with unlabeled training data $\{\mathbf{x}_u\}_{u=1\ldots N_u}$ and labeled training data $\{\mathbf{x}_l,\mathbf{y}_l\}_{l=1\ldots N_l}$ by minimizing the following loss function:

$$J^1(\theta)=\frac{1}{2N_u}\sum_{u=1}^{N_u}\|\tilde{\mathbf{x}}_u-\mathbf{x}_u\|_2^2+\frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\|\tilde{\mathbf{x}}_l-\mathbf{x}_l\|_2^2+\lambda\|\tilde{\mathbf{y}}_l^0-\mathbf{y}_l\|_2^2\right)$$

After this, the pretrained weights and bias are obtained as $\{\mathbf{W_1},\mathbf{b}_1\}$ for the encoder part. Then, the first layer hidden features can be obtained as $\left\{\mathbf{h}_u^1,\mathbf{h}_l^1\right\}_{u=1,2,\ldots,N_u,\ l=1,2,\ldots,N_l}$, which are calculated by forward propagation from the input layer to the first hidden layer with data $\{\mathbf{x}_u,\mathbf{x}_l\}_{u=1,2,\ldots,N_u,\ l=1,2,\ldots,N_l}$.

② Pre-train the second SS-AE-2 with unlabeled training data $\left\{\mathbf{h}_u^1\right\}_{u=1\ldots N_u}$ and labeled training data $\left\{\mathbf{h}_l^1,\mathbf{y}_l\right\}_{l=1\ldots N_l}$ by minimizing

$$J^2(\theta)=\frac{1}{2N_u}\sum_{u=1}^{N_u}\|\tilde{\mathbf{h}}_u^1-\mathbf{h}_u^1\|_2^2+\frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\|\tilde{\mathbf{h}}_l^1-\mathbf{h}_l^1\|_2^2+\lambda\|\tilde{\mathbf{y}}_l^1-\mathbf{y}_l\|_2^2\right)$$

Then, the pretrained weights and bias are obtained as $\{\mathbf{W}_2,\mathbf{b}_2\}$ for the encoder part. Meanwhile, the second layer hidden features can be obtained as $\left\{\mathbf{h}_u^2,\mathbf{h}_l^2\right\}_{u=1,2,\ldots,N_u,\ l=1,2,\ldots,N_l}$, which are calculated by forward propagation from the first hidden layer to the second hidden layer with data $\left\{\mathbf{h}_u^1,\mathbf{h}_l^1\right\}_{u=1,2,\ldots,N_u,\ l=1,2,\ldots,N_l}$.

③ In a similar way, the following hidden layers can be progressively pretrained one by one. Here, assume that the (k-1)-th $_{(k=3,\ldots,K)}$ SS-AE is already pre-trained. For the k-th $_{(k=3,\ldots,K)}$ hidden layer, we can construct and pre-train SS-AE k by minimizing

$$J^k(\theta)=\frac{1}{2N_u}\sum_{u=1}^{N_u}\|\tilde{\mathbf{h}}_u^{k-1}-\mathbf{h}_u^{k-1}\|_2^2+\frac{1}{2N_l}\sum_{l=1}^{N_l}\left(\|\tilde{\mathbf{h}}_l^{k-1}-\mathbf{h}_l^{k-1}\|_2^2+\lambda\|\tilde{\mathbf{y}}_l^{k-1}-\mathbf{y}_l\|_2^2\right)$$

Then, the pretrained weights and bias are obtained as $\{\mathbf{W}_k,\mathbf{b}_k\}$ for the encoder part. Also, the k-th layer hidden features can be obtained as $\left\{\mathbf{h}_u^k,\mathbf{h}_l^k\right\}_{u=1,2,\ldots,N_u,\ l=1,2,\ldots,N_l}$ by forward propagation from the (k-1)th hidden layer to the kth hidden layer with data $\left\{\mathbf{h}_u^{k-1},\mathbf{h}_l^{k-1}\right\}_{u=1,2,\ldots,N_u,\ l=1,2,\ldots,N_l}$.

④ Once the last SS-AE K is obtained, the pre-training is finished for the deep SS-SAE network.

Step 6: After pre-training, the quality output layer is added to the top hidden layer of the SS-SAE for quality prediction. Utilize the corresponding pre-trained $\{\mathbf{W}_k,\mathbf{b}_k\}_{k=1,2,\ldots,K}$ to initialize the SS-SAE-based soft sensor network. Randomly initialize the output layer weight matrix $\mathbf{W}_o$ and bias $\mathbf{b}_o$. Then, fine-tune the whole network parameters with the labeled dataset $\{\mathbf{x}_l,\mathbf{y}_l\}_{l=1\ldots N_l}$ by carrying out forward propagation and back propagations iteratively.

Step 7: Determine the optimized network hyper-parameters on the validation dataset under different hyper-parameter combinations.

Step 8: Predict the quality variable $\hat{\mathbf{y}}$ for testing dataset.

For model evaluation, the root mean squared error (RMSE) is used to assess the prediction performance of soft sensor algorithms. The RMSE indicator is defined as follows:

$$RMSE=\sqrt{\sum_{i=1}^{N}(y_i-\hat{y}_i)^2/(N-1)} \tag{13}$$

where y_i and $\hat{y}_i$ are the labeled and predicted output values of the ith testing sample, respectively. Moreover, the original unsupervised SAE and DBN are also used to construct the soft sensor models for performance comparison.

4. Case study

4.1. Debutanizer column

The debutanizer column is an important part of the desulfurization and naphtha separation unit in the refining process. The flowchart of the debutanizer column is shown in Fig. 4. One of its main tasks is to reduce the butane (C4) content at the bottom of the debutanizer. This requires real-time measurement of the bottom butane content (Fortuna et al., 2005). However, the C4 content is measured by gas chromatograph equipped at the sequential deisopentanizer top, which is at a distance from the debutanizer column. In this way, there is a large measuring delay for C4 content. To solve this problem, soft sensor can be used to estimate C4 concentration at the bottom of the debutanizer in real time (Gupta et al., 2009; Jana et al., 2009). For this purpose, seven process variables are chosen for the construction of the soft sensor model, which are listed in Table 1.

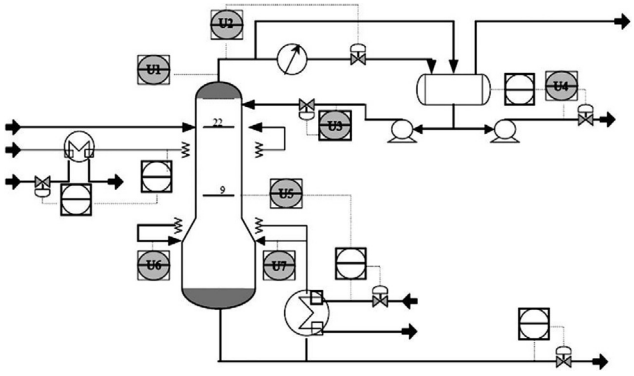

Fig. 4. Flowchart of the debutanizer column (Fortuna et al., 2005).

2390 labeled samples have been collected from this process. To build the soft sensor model, the first 1000 samples are used for training model, 600 samples are used as validation dataset for

Table 1
Descriptions of the 7 process variables in the debutanizer column.

Process variables	Descriptions
u_1	Top temperature
u_2	Top pressure
u_3	Reflux flow
u_4	Flow to next process
u_5	6th tray temperature
u_6	Bottom temperature A
u_7	Bottom temperature B

hyper-parameter selection and the remaining 790 are used for testing dataset. Moreover, process dynamics are taken into consideration before model construction. That is to say, the previous input and output variables are added to the original input variable to predict the quality variable at sampling instant k. The details of the choice of augmented variables can be found in reference (Fortuna et al., 2005). The augmented input variable for the SS-SAE network is designed as

$$\begin{bmatrix} u_1(k), u_2(k), u_3(k), u_4(k), u_5(k), u_5(k-1), \\ u_5(k-2), u_5(k-3), (u_6(k)+u_7(k))/2, \\ y(k-1), y(k-2), y(k-3), y(k-4) \end{bmatrix}^T \tag{14}$$

For fair comparison of different pre-training techniques, all the constructed networks are with the same structure, which can eliminate the performance difference caused by different network structures. Four different network structures are used for soft sensor modeling, which is shown in Table 2. The hyper-parameters for the four models are set as these in Table 3. To simulate the real scenarios, the label values are removed for 80% of the samples in the training dataset. Hence, only 20% of the samples have label values in the training set. Table 4 gives the RMSEs on the validation dataset for DBN, SAE and SS-SAE. It can be seen that the network with the No. 2 structure has the lowest RMSE for each method. In addition, the RMSE of SS-SAE is 0.0715, which is much lower than DBN and SAE. Therefore, No. 2 structure is selected for these models for further analysis.

Table 2
Four different network structures for comparison analysis.

No.	Structure
1	13-10-7-1
2	13-10-7-4-1
3	13-10-7-4-2-1
4	13-10-7-4-2-2-1

Table 3
Hyper-parameters for modeling.

Hyper-parameter	Value
Pre-training learning rate	0.01
Pre-training epochs	12
Fine-tuning learning rate	0.5
Fine-tuning epochs	200
λ	1.6

Table 4
Prediction results of four different network structures on validation set.

RMSE	No.1	No.2	No.3	No.4
DBN	0.1386	**0.1246**	0.1401	0.1464
SAE	0.1427	**0.1262**	0.1474	0.1473
SS-SAE	0.1042	**0.0715**	0.1261	0.1268

Table 5
Prediction accuracy using for DBN, SAE and SS-SAE (RMSE).

Algorithms	20%	40%	60%	80%	100%
DBN	0.1335	0.1036	0.0913	0.0857	0.0734
SAE	0.1380	0.1072	0.0940	0.0925	0.0794
SS-SAE	**0.0801**	**0.0677**	**0.0665**	**0.0645**	**0.0559**

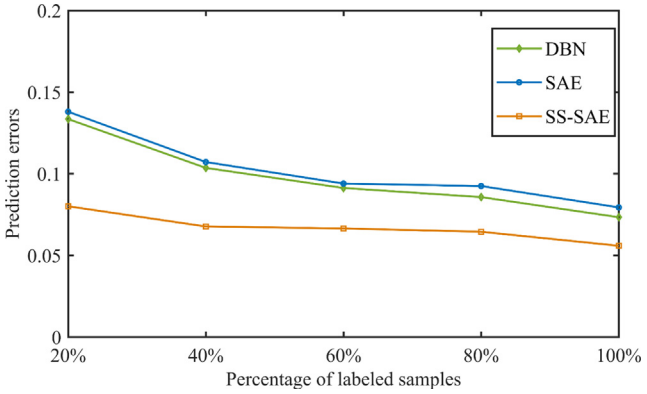

Fig. 5. The prediction errors with different ratio of labeled data.

Then, different percentages of labeled data are simulated on the training dataset to evaluate the prediction performance. Table 5 gives the RMSE values on testing set for DBN, SAE and SS-SAE with No. 2 structure. Fig. 5 shows an illustrative trend of the results. As can be seen, with the increase of labeled sample ratio, the prediction accuracy can get improved for DBN, SAE and SS-SAE. For DBN and SAE, the accuracy improvement mainly lies in the increase of labeled information for fine-tuning since the unsupervised pre-training does not involve the sample labels. However, this can benefit for both the pre-training and fine-tuning in SS-SAE. Since the label information is introduced during pre-training, the prediction RMSE of SS-SAE is smaller than the original SAE and DBN in each scenario with a labeled sample ratio. Moreover, the semi-supervised pre-training can make the model performance more stable than the other two. This is because in the pre-training phase, SAE and DBN use an unsupervised learning algorithm, which means that its feature extraction is almost objectless. In contrast, SS-SAE is a semi-supervised learning algorithm that can not only utilize a large number of unlabeled samples, but also use the label information to guide the pre-training process to extract useful features for fine-tuning.

To further evaluate the performance of the proposed algorithm, detailed prediction results is provided for the testing dataset in the scenario with only 20% labeled data for the training set. For performance comparison, the multi-layer neural network (NN) model with the same network topology is also built for soft sensor modeling. For some of the commonly used hyper-parameters of deep networks, a practical guide with recommendations are introduced

Table 6
Prediction results of the four methods in the scenario with 20% labeled data.

Algorithms	Training RMSE	Testing RMSE	Training time(s)
NN	0.0810	0.1477	**1.221**
DBN	0.0701	0.1335	2.789
SAE	0.0715	0.1380	1.489
SS-SAE	**0.0488**	**0.0808**	1.512

by Bengio in reference (Bengio, 2015). Since multi-layer neural network is a supervised learning algorithm without pre-training, it is only trained with labeled samples. The final prediction performance is provided in Table 6 for the four methods. As can be seen, SS-SAE based soft sensor has better generalization performance than traditional NN, DBN and SAE. The NN model has the worst performance since pre-training trick is not used for it. Compared to NN, DBN and SAE are able to obtain better prediction accuracy for the complex data due to the proper initial network parameters with pre-training. However, label information is not used for pre-training to learn quality-related features for DBN and SAE. In contrast, SS-SAE can further reduce the prediction RMSE by utilizing labeled information to overcome the limitations of unsupervised pre-training. As can be seen, DBN and SAE only improves the performance very little over NN model, whereas SS-SAE significantly enhances the prediction performance than NN. In terms of time efficiency, while NN spends the shortest time because NN without pre-training steps. As a probabilistic deep network, DBN spends the longest training time since it should use an additional contrastive divergence algorithm. Compared to SAE, SS-SAE only takes more time of 0.1 s to the training the whole network, which is mainly caused by the additional calculations with labeled data in the pretraining.

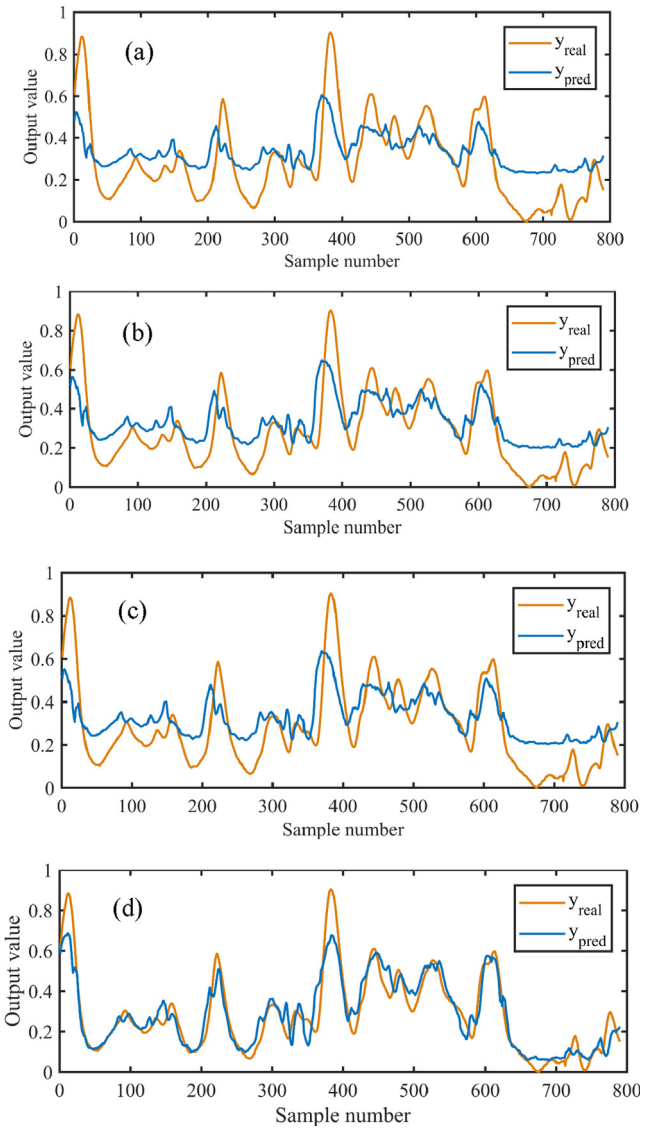

Fig. 6. Prediction curve of C4-concentration: (a). NN, (b). DBN, (c). SAE, (d). SS-SAE.

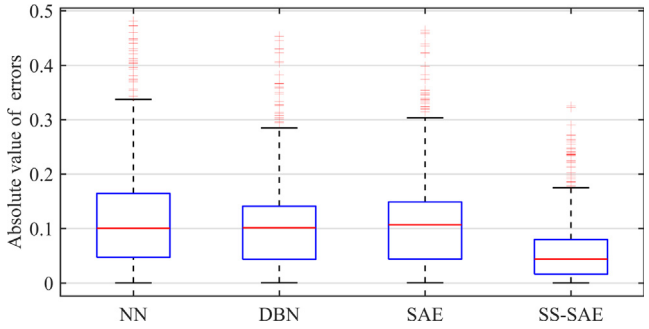

Fig. 7. The absolute values of prediction errors for NN, DBN, SAE and SS-SAE.

In the scenario with 20% labeled training data, the detailed prediction curves on the testing dataset are presented in Fig. 6 with NN, DBN, SAE and SS-SAE based soft sensor networks. As can be seen from Fig. 6 (a-c), it is difficult for NN, DBN and SAE to track the general trend of the actual quality values of the testing samples. There are large deviations between the predicted and labeled values. With the additional information of quality labels for pre-training, SS-SAE can track best with the actual quality values as shown in Fig. 6(d). In addition, the overall prediction error is very small. Thus, SS-SAE predictions are more stable without large fluctuations and show better robustness than the NN, DBN and SAE model.

Fig. 7 further shows the box-plot analysis for the absolute values of model residuals for NN, DBN, SAE and SS-SAE algorithm. The box plot can reflect the error distribution, the maximum, the median, the minimum, the 25th and 75th percentiles of model residuals. It can be seen that NN, DBN and SAE have wide error ranges than SS-SAE. Moreover, the proposed SS-SAE have a tighter error range close to zero. In addition, the fine-tuning procedure is also studied for the NN, DBN, SAE and SS-SAE based soft sensors in the scenario with 20% labeled training data. Fig. 8 illustrates the full-batch fine-tuning errors with the epoch for the four models. From Fig. 8, it can be seen that the SS-SAE converges with a faster speed at a smaller learning error than the other methods.

4.2. Hydrocracking process

The hydrocracking process is an important part of the petrochemical industry. Under the conditions of heating, high hydrogen pressure and catalyst, the heavy oil (vacuum wax oil) is cracked and converted into light and high-quality oils such as naphtha oil, gasoline, aviation kerosene (ATK), diesel oil, and so on. Hydrocracking process is characterized by wide range and strong adaptability of raw materials, as well as low carbon loss. Also, it has high flexibility of production, in which the product yield can be

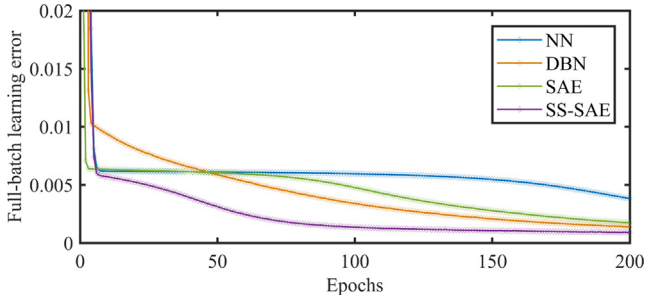

Fig. 8. Fine-tuning procedure of the four methods on the debutanizer column.

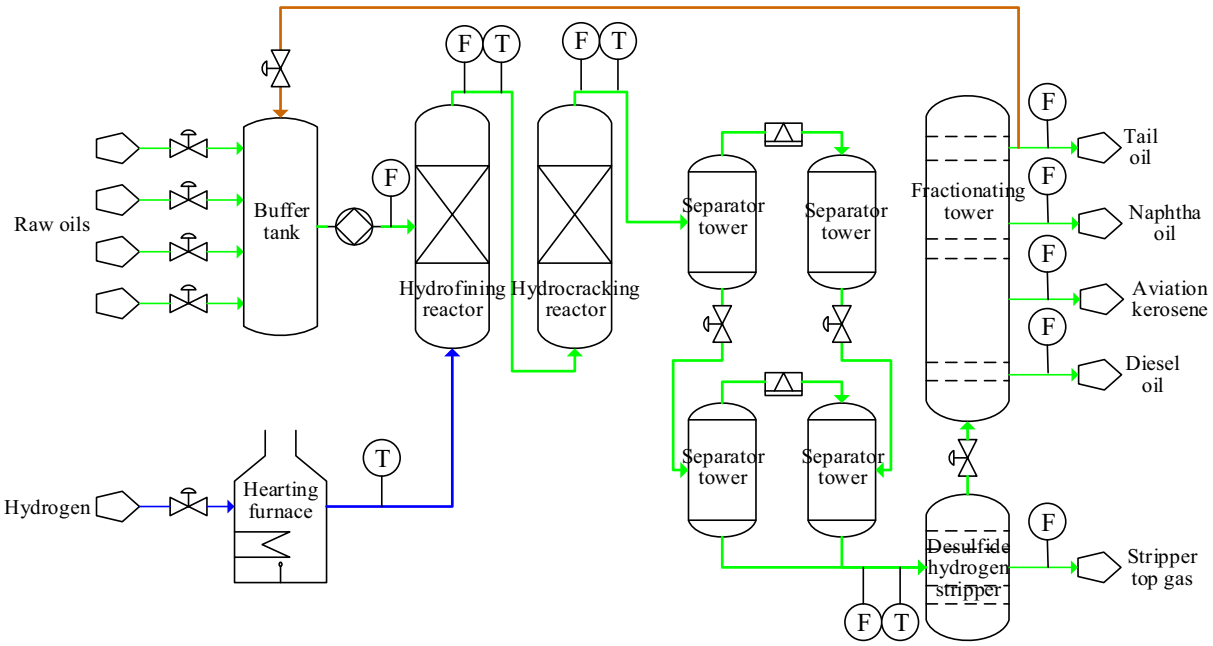

Fig. 9. The flowchart of the hydrocracking process (Yuan et al., 2018c).

controlled by different operating conditions. Moreover, the products are of good quality with less impurities such as sulphur, oxygen and nitrogen. It is one of the most important, reliable, flexible and effective processing methods for large refineries and petrochemical enterprises. Hence, the products of hydrocracking process are widely used in automobile, aviation, ships, heavy machinery, lubricating oil production, etc (Becker et al., 2017; Chen et al., 2017; Wang et al., 2018).

A simplified flowchart of this process is given in Fig. 9. It mainly consists of four parts: the hydrorefining, hydrocracking, high and low pressure separation, and fractionation system. In this process, the make-up hydrogen and recycled hydrogen are first fully mixed, compressed and preheated, which are then fed to the first reactor of the hydrorefining section. Also, the mixed raw oil is fed to it at the same time. Then, deep hydrodenitrogenation reaction is carried out in the first reactor. After that, the reactant stream is directly transmitted into the second reactor section for hydrocracking. Then, the flow at the outlet of the hydrocracking section enters the high and low pressure separator for gas/liquid separation. The hydrogen-rich gas separated from the top of the high pressure separator is used for recycling hydrogen, and the liquid distillate enters the low pressure separator for further gas/liquid separation. Finally, the liquid effluents from the low pressure separator enters the fractionation system to obtain different products (Sun et al., 2017; Wang et al., 2019b).

In this process, a large number of product quality variables should be monitored and measured for timely process control and optimization. However, it is often unpractical and unavailable to carry out online measurement for the quality variables. On one hand, most product attributes are obtained through laboratory test with large time delays. This unavoidably results in the fact that real-time control or optimization schemes cannot be carried out in an online way for the hydrocracking unit. On the other hand, a lot of variables, like pressure of hydrocracking reactor, total feed, and inlet temperature of hydrotreating reactor, can be measured and recorded in real-time with fast sampling frequency. Moreover, these routinely measured process variables are highly related to the quality variables with complex nonlinear relationships. This inspires the development of deep learning –based soft sensor modeling for the difficult-to-measure quality variables. Here, the final boiling point of the aviation kerosene is selected as the dependent variable to build the inferential soft sensor models.

The data were collected from a petrochemical industry in China. Among numerous routinely measured process variables, 43 of them are selected as input variables for soft sensor modeling, which are highly related to the quality variable. Detailed description of these secondary variables can be found in reference (Yuan et al., 2018c). Since the process secondary variables has faster sampling frequency than the quality variables, there are a larger number of unlabeled samples in the historical datasets than the labeled ones. In this way, the semi-supervised stacked autoencoder is applied to build the soft sensor model.

A total number of 1931 labeled samples were collected for the quality variable in this process with frequency of several hours per sample. For soft sensor modeling, the first 1200 samples are used as training dataset, 300 of them are used for model validation and the rest 431 are used for testing dataset. For technical protection, all the process variables and quality variable are normalized to range 0 to 1. To construct SS-SAE based soft sensor model, the hyper-parameters of the network should be first determined, which are chosen by grid search on the validation set. A proper network was obtained with structure 43–23–23–23–23–1, where there are four hidden layers in the deep network. The configuration of the SS-SAE model is shown in Table 7.

For the process secondary variables, they have a much faster sampling frequency than the quality variables. Hence, there are a lot of unlabeled samples apart from the labeled samples. Hence, it is very important to incorporate the unlabeled samples for

Table 7
Hyper-parameters of SS-SAE.

Hyperparameter	Value
Pre-training learning rate	0.01
Pre-training epochs	10
Fine-tuning learning rate	0.3
Fine-tuning epochs	2500
Hidden layers	4
λ	1.2

Table 8
Prediction results on the testing dataset and model training time for the four methods.

N_u	NN		DBN		SAE		SS-SAE	
	RMSE	Time(s)	RMSE	Time(s)	RMSE	Time(s)	RMSE	Time(s)
0	0.0947	41.8	0.0828	45.2	0.0843	**44.2**	**0.0683**	44.5
300	–	–	0.0816	45.6	0.0827	**44.4**	**0.0650**	44.8
600	–	–	0.0787	46.5	0.0814	**44.6**	**0.0586**	45.0
900	–	–	0.0779	46.9	0.0798	**44.7**	**0.0552**	45.4
1200	–	–	0.0763	47.8	0.0765	**46.6**	**0.0533**	46.8
1500	–	–	0.0769	48.9	0.0767	**47.2**	**0.0531**	47.8
1800	–	–	0.0761	52.4	0.0773	**49.6**	**0.0535**	49.9

semi-supervised modeling. To assess the effectiveness of the proposed SS-SAE, different number of unlabeled samples are added to the labeled training dataset for pre-training and fine-tuning. In this case, 300 additional unlabeled samples are sequentially added to the training dataset each time.

To compare the performance of SS-SAE based soft sensor model, NN, DBN and SAE are also adopted for soft sensor modeling in each case. As for NN, it is trained with only 1200 labeled samples because it is a supervised model without pre-training. Thus, detailed prediction results are given in Table 8 for the final boiling point of the aviation kerosene with the four models, where N_u denotes the number of unlabeled samples in the training dataset. As can be seen, the performance of DBN, basic SAE and the proposed SS-SAE can be improved as the number of unlabeled samples increases at the beginning. In addition, since NN cannot utilize the unlabeled samples without pre-training, its performance remains unchanged and is the worst among the four methods. Therefore, both the unsupervised pre-training of DBN and SAE and semi-supervised pre-training of SS-SAE with additional unlabeled samples can improve the prediction accuracy than the original supervised NN method. Moreover, the performance of SS-SAE is better than DBN and SAE in each scenario since DBN and SAE only uses the input data for unsupervised pre-training while SS-SAE utilizes additional quality information to guide the semi-supervised pre-training. However, the performance of DBN, SAE and SS-SAE has barely improved after 1,200 unlabeled samples are added to the training dataset. Hence, the useful information is limited when too much unlabeled samples are added. In terms of time efficiency, DBN training takes the longest time since it needs to adopt the additional CK algorithm. The training time of SS-SAE is longer than the original SAE, because the semi-supervised pre-training phase requires additional calculation for the label data.

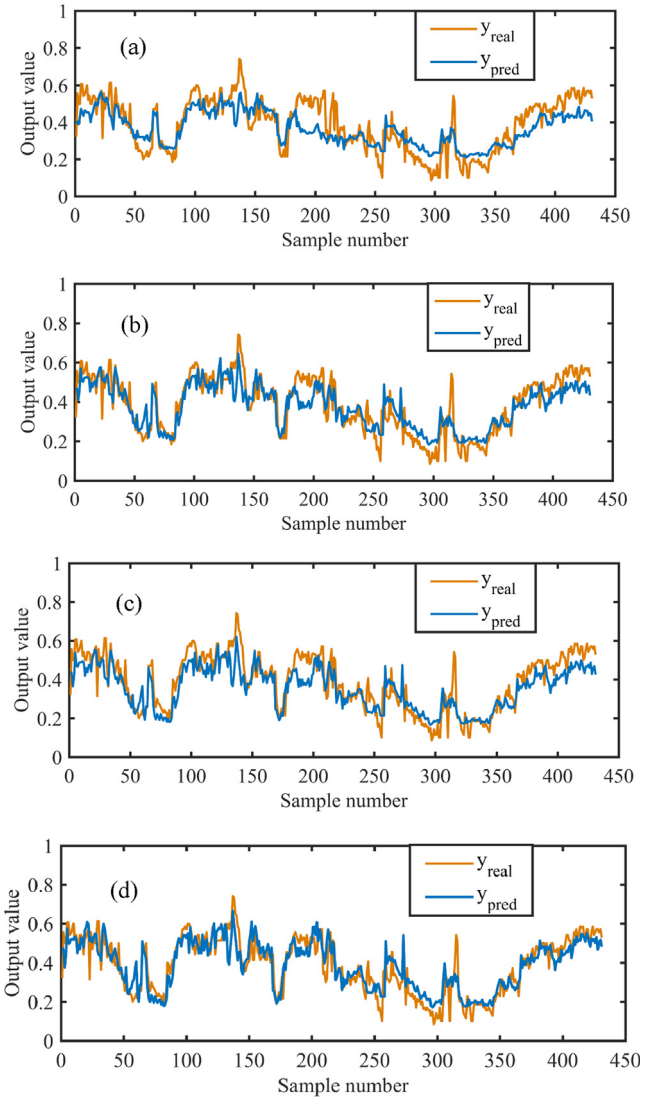

Fig. 10. Prediction performance on the hydrocracking process: (a). NN, (b). DBN, (c). SAE, (d). SS-SAE.

Intuitively, the detailed predictions on the testing dataset are shown in Fig. 10 for NN, DBN, SAE and SS-SAE in the case with 1200 unlabeled samples. For NN, the additional unlabeled data do not affect its performance since it can only use 1200 labeled samples for model training. As can be seen, the predicted output values of the NN model do not track well with the actual trend due to the lack of pre-training strategies and the inability to utilize

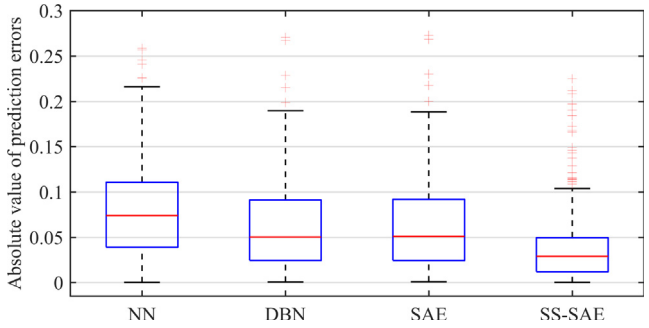

Fig. 11. The absolute values of prediction errors of NN, DBN, SAE and SS-SAE.

Table 9
RMSE of SS-SAE based soft sensor with different hidden layers.

Hidden layer structure	Training RMSE	Testing RMSE
SS-SAE(1layer)(23)	0.1136	0.1718
SS-SAE(2layer)(23/23)	0.0715	0.1469
SS-SAE(3layer)(23/23/23)	0.0459	0.0761
SS-SAE(4layer)(23/23/23/23)	**0.0383**	**0.0533**
SS-SAE(5layer)(23/23/23/23/23)	0.0598	0.0813
SS-SAE(6layer)(23/23/23/23/23/23)	0.0654	0.1045

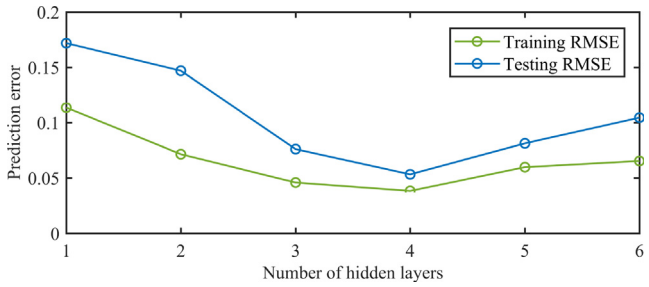

Fig. 12. RMSE of SS-SAE based soft sensor with different hidden layers.

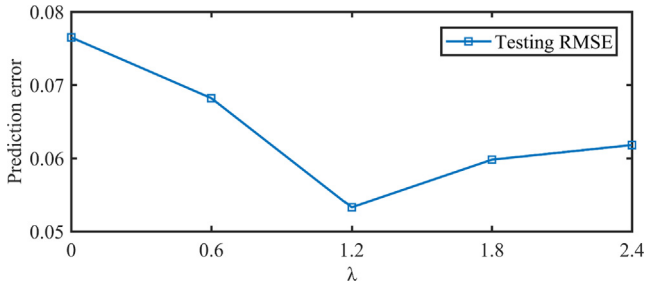

Fig. 13. The effect of different parameters λ in the SS-SAE.

unlabeled samples. When the pre-training techniques is utilized, the predicted values of DBN, SAE and SS-SAE are more consistent with the actual ones. Moreover, SS-SAE can further improve the prediction performance by using the labeled samples to construct the semi-supervised pre-training framework.

Fig. 11 shows the box plot on the absolute values of the prediction errors for NN, DBN, SAE and SS-SAE algorithm. It is easily seen that NN, DBN and SAE have wide error ranges than SS-SAE. Moreover, the proposed SS-SAE have a much tighter error range around zero.

Also, the performance of SS-SAE with different hidden layers is shown in Table 9 and Fig. 12. In this experiment, 1200 labeled samples and 1200 unlabeled samples were used for model pre-training. As can be seen, the performance of SS-SAE is not good when there is only one hidden layer. With the increase of the hidden layer number, the performance is improved and the RMSE has largely decreased. However, it will degenerate the prediction performance after 4 hidden layers. It can be seen that SS-SAE with 4 hidden layers has achieved the best performance.

Finally, the trade-off parameter λ in Eq. (5), is investigated for prediction performance, which is very important to adjust the importance of labeled information in the formulation of SS-SAE. Fig. 13 gives the prediction RMSE results with regard to different values of λ. As can be seen, the performance can be improved with the increase of λ at the beginning. When it reaches about 1.2, the prediction error is the smallest, which also demonstrates the importance of the supervised learning term. But if λ is too large, the weak learning ability for input features may decline as it strongly emphasizes on the label information.

5. Concluding remarks

In this paper, a new semi-supervised stacked autoencoder is proposed to pre-train deep network and extract output-related features for process soft sensor modeling, which can utilize both labeled and unlabeled samples for network pre-training effectively. In the basic SS-AE, quality label information is additionally used to design a new reconstructed variable vector at the output layer. Thus, the hidden features are learned to not only maintain the input data fidelity, but also to be quality-related. Then, multiple SS-AEs can be used to construct the deep SS-SAE network in a hierarchically stacking way. The combination of semi-supervised pre-training and supervised fine-tuning for prediction greatly improves the prediction performance than the basic SAE, which lies in that the proposed method can obtain extensive information from both unlabeled and labeled data samples. The predictive performance of the NN, DBN, SAE and SS-SAE methods are compared in two industrial refining processes, including a debutanizer column and a hydrocracking process. The results show the effectiveness and superiority of the proposed SS-SAE.

CRediT authorship contribution statement

Xiaofeng Yuan: Conceptualization, Methodology, Writing - review & editing. **Chen Ou:** Writing - original draft, Software, Validation. **Yalin Wang:** Writing - review & editing, Supervision. **Chunhua Yang:** Data curation, Investigation. **Weihua Gui:** Visualization, Investigation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This paper is supported in part by National Key Research and Development Program of China (2018YFB1701100), and in part by National Natural Science Foundation of China (NSFC) (U1911401, 61703440, 61590921), and in part by Innovation-driven plan in Central South University (2018CX011), and in part by the Fundamental Research Funds for the Central Universities of Central South University (2019zzts568), and in part by the Hunan Postgraduate Research Innovation Project (2019zzts148).

References

Alain, G., Bengio, Y., 2012. What regularized auto-encoders learn from the data generating distribution. J. Mach. Learn. Res. 15, 3563–3593.
Becker, P.J., Serrand, N., Celse, B., Guillaume, D., Dulot, H., 2017. A single events microkinetic model for hydrocracking of vacuum gas oil. Comput. Chem. Eng. 98, 70–79.
Bengio, Y., 2015. Practical recommendations for gradient-based training of deep architectures. Lect. Notes. Comput. Sc. 7700, 437–478.
Bengio, Y., Lamblin, P., Dan, P., Larochelle, H., 2007. Greedy layer-wise training of deep networks. Proc. Adv. Neural Inform. Process. Syst., 153–160
Bosca, S., Fissore, D., 2011. Design and validation of an innovative soft-sensor for pharmaceuticals freeze-drying monitoring. Chem. Eng. Sci. 66, 5127–5136.
Chen, G., Wang, Y., Xue, Y., Yuan, X., Zou, S., 2017. ELM-based Real-time Prediction for Hydrogenolysis Degree of Hydrofining Reaction in Hydrocracking Process. In: Proc. 36th. CCC, pp. 4488–4493.
Chen, N., Dai, J., Yuan, X., Gui, W., Ren, W., Koivo, H.N., 2018. Temperature prediction model for roller kiln by ALD-based double locally weighted kernel principal component regression. IEEE T. Instrum. Meas. 67, 2001–2010.
Dai, J., Chen, N., Yuan, X., Gui, W., Luo, L., 2019. Temperature prediction for roller kiln based on hybrid first-principle model and data-driven MW-DLWKPCR model. ISA T. https://doi.org/10.1016/j.isatra.2019.1008.1023.
Erhan, D., Bengio, Y., Courville, A.C., Manzagol, P.A., Bengio, S., 2010. Why does unsupervised pre-training help deep learning?. J. Mach. Learn. Res. 11, 625–660.
Farizhandi, A.A.K., Zhao, H., Lau, R., 2016. Modeling the change in particle size distribution in a gas-solid fluidized bed due to particle attrition using a hybrid artificial neural network-genetic algorithm approach. Chem. Eng. Sci. 155, 210–220.
Fortuna, L., Graziani, S., Xibilia, M.G., 2005. Soft sensors for product quality monitoring in debutanizer distillation columns. Control Eng. Pract. 13, 499–508.
Gao, S., Zhang, Y., Jia, K., Lu, J., Zhang, Y., 2015. Single sample face recognition via learning deep supervised auto-encoders. IEEE Trans. Inf. Forensics Secur 10, 2108–2118.
Ge, Z., Song, Z., Gao, F., 2013. Review of recent research on data-based process monitoring. Ind. Eng. Chem. Res. 52, 3543–3562.

Gupta, S., Ray, S., Samanta, A.N., 2009. Nonlinear control of debutanizer column using profile position observer. Comput. Chem. Eng. 33, 1202–1211.
Hinton, G.E., Osindero, S., Teh, Y.W., 2006. A fast learning algorithm for deep belief nets. Neural. Comput. 18, 1527–1554.
Huang, B., Qi, Y., Murshed, A.M., 2013. Dynamic Modelling and Predictive Control in Solid Oxide Fuel Cells: First Principle and Data-Based Approaches. John Wiley & Sons, NY, USA.
Jana, A.K., Samanta, A.N., Ganguly, S., 2009. Nonlinear state estimation and control of a refinery debutanizer column. Comput. Chem. Eng. 33, 1484–1490.
Kadlec, P., Gabrys, B., Strandt, S., 2009. Data-driven Soft Sensors in the process industry. Comput. Chem. Eng. 33, 795–814.
Kaneko, H., Funatsu, K., 2014. Application of online support vector regression for soft sensors. AIChE J. 60, 600–612.
Kano, M., Nakagawa, Y., 2008. Data-based process monitoring, process control, and quality improvement: recent developments and applications in steel industry. Comput. Chem. Eng. 32, 12–24.
Kim, S., Kano, M., Hasebe, S., Takinami, A., Seki, T., 2013. Long-term industrial applications of inferential control based on just-in-time soft-sensors: economical impact and challenges. Ind. Eng. Chem. Res. 52 (35), 12346–12356.
Liu, Y., Gao, Z., Chen, J., 2013. Development of soft-sensors for online quality prediction of sequential-reactor-multi-grade industrial processes. Chem. Eng. Sci. 102, 602–612.
Ma, J., Sheridan, R.P., Liaw, A., Dahl, G.E., Svetnik, V., 2015a. Deep neural nets as a method for quantitative structure–activity relationships. J. Chem. Informat. Model. 55, 263–274.
Ma, M., Khatibisepehr, S., Huang, B., 2015b. A Bayesian Framework for real-time identification of locally weighted partial least squares. AIChE J. 61, 518–529.
Maiti, S.B., Let, S., Bar, N., Das, S.K., 2018. Non-spherical solid-non-Newtonian liquid fluidization and ANN modelling: Minimum fluidization velocity. Chem. Eng. Sci. 176, 233–241.
Na, J., Jeon, K., Lee, W.B., 2018. Toxic gas release modeling for real-time analysis using variational autoencoder with convolutional neural networks. Chem. Eng. Sci. 181, 68–78.
Qin, Y., Zhao, C., Huang, B., 2019. A new soft-sensor algorithm with concurrent consideration of slowness and quality interpretation for dynamic chemical process. Chem. Eng. Sci. 199, 28–39.
Shang, C., Yang, F., Huang, D., Lyu, W., 2014. Data-driven soft sensor development based on deep learning technique. J. Process Contr. 24, 223–233.
Shao, W., Ge, Z., Song, Z., 2019. Quality variable prediction for chemical processes based on semisupervised Dirichlet process mixture of Gaussians. Chem. Eng. Sci. 193, 394–410.
Shao, W., Tian, X., 2015. Adaptive soft sensor for quality prediction of chemical processes based on selective ensemble of local partial least squares models. Chem. Eng. Res. Des. 95, 113–132.
Shardt, Y.A.W., Hao, H., Ding, S.X., 2015. A new soft-sensor-based process monitoring scheme incorporating infrequent KPI measurements. IEEE. T. Ind. Electron. 62, 3843–3851.
Sharmin, R., Sundararaj, U., Shah, S., Vande Griend, L., Sun, Y.-J., 2006. Inferential sensors for estimation of polymer quality parameters: Industrial application of a PLS-based soft sensor for a LDPE plant. Chem. Eng. Sci. 61, 6372–6384.
Su, H., Lian, C., Liu, J., Liu, H., 2019. Machine learning models for solvent effects on electric double layer capacitance. Chem. Eng. Sci. 202, 186–193.
Sun, K., Wang, Y., Li, L., Yuan, X., Chen, Q., 2017. Classification for Raw Oil Inlet Conditions of Hydrocracking Process Based on Ratio Cluster. In: Proc. 36th. CCC, pp. 4476–4481.
Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., 2010. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. 11, 3371–3408.
Wang, Y., Pan, Z., Yuan, X., Yang, C., Gui, W., 2019a. A novel deep learning based fault diagnosis approach for chemical process with extended deep belief network. ISA T. https://doi.org/10.1016/j.isatra.2019.1007.1001.
Wang, Y., Sun, K., Yuan, X., Yue, C., Ling, L., Koivo, H., 2018. AA novel sliding window PCA-IPF based steady-state detection framework and its industrial application. IEEE Access 6, 20995–21004.
Wang, Y., Wu, D., Yuan, X., 2019b. A two-layer ensemble learning framework for data-driven soft sensor of the diesel attributes in an industrial hydrocracking process. J. Chemometr. https://doi.org/10.1002/cem.3185, ee3185.
Yao, L., Ge, Z., 2018. Deep learning of semisupervised process data with hierarchical extreme learning machine and soft sensor application. IEEE. T. Ind. Electron. 65, 1490–1498.
Yoshua, B., 2009. Learning Deep Architectures for AI. Now Foundations Trends.
Yu, J., Hong, C., Rui, Y., Tao, D., 2018. Multi-Task autoencoder model for recovering human poses. IEEE. T. Ind. Electron. 65, 5060–5068.
Yuan, X., Ge, Z., Huang, B., Song, Z., 2017a. A probabilistic just-in-time learning framework for soft sensor development with missing data. IEEE. T. Contr. Syst. T. 25, 1124–1132.
Yuan, X., Ge, Z., Huang, B., Song, Z., Wang, Y., 2017b. Semisupervised JITL framework for nonlinear industrial soft sensing based on locally semisupervised weighted PCR. IEEE. T. Ind. Inf. 13, 532–541.
Yuan, X., Ge, Z., Song, Z., 2014. Locally weighted Kernel Principal component regression model for soft sensing of nonlinear time-variant processes. Ind. Eng. Chem. Res. 53, 13736–13749.
Yuan, X., Huang, B., Wang, Y., Yang, C., Gui, W., 2018a. Deep learning based feature representation and its application for soft sensor modeling with variable-wise weighted SAE. IEEE. T. Ind. Inf. 14, 3235–3243.
Yuan, X., Gu, Y., Wang, Y., Yang, C., Gui, W., 2019a. A deep supervised learning framework for data-driven soft sensor modeling of industrial processes. In: IEEE. Trans. Neural. Netw. Learn.Syst. DOI: http://doi.org/10.1109/TNNLS.2019.2957366.
Yuan, X., Li, L., Wang, Y., 2019b. Nonlinear dynamic soft sensor modeling with supervised long short-term memory network. In: IEEE. T. Ind. Inf., Doi: http://doi.org/10.1109/TII.2019.2902129.
Yuan, X., Li, L., Wang, Y., Yang, C., Gui, W., 2019c. Deep learning for quality prediction of nonlinear dynamic process with variable attention-based long short-term memory network. Can. J. Chem. Eng. DOI: http://doi.org/10.1002/cjce.23665.
Yuan, X., Ou, C., Wang, Y., Yang, C., Gui, W., 2019d. Deep quality-related feature extraction for soft sensing modeling: a deep learning approach with hybrid VW-SAE. Neurocomputing. https://doi.org/10.1016/j.neucom.2018.1011.1107.
Yuan, X., Ou, C., Wang, Y., Yang, C., Gui, W., 2019e. A layer-wise data augmentation strategy for deep learning networks and its soft sensor application in an industrial hydrocracking process. IEEE. Trans. Neural. Netw. Learn. Syst., 1–10
Yuan, X., Wang, Y., Yang, C., Ge, Z., Song, Z., Gui, W., 2018b. Weighted linear dynamic system for feature representation and soft sensor application in nonlinear dynamic industrial processes. IEEE. T. Ind. Electron. 65, 1508–1517.
Yuan, X., Zhou, J., Huang, B., Wang, Y., Yang, C., Gui, W., 2019f. Hierarchical quality-relevant feature representation for soft sensor modeling: a novel deep learning strategy. IEEE. T. Ind. Inf., Doi: http://doi.org/10.1109/TII.2019.2938890.
Yuan, X., Zhou, J., Wang, Y.L., Sun, M.X., Zhang, H.G., 2018c. A Comparative Study of Adaptive Soft Sensors for Quality Prediction in an Industrial Refining Hydrocracking Process. In: Proc. DDCLS'18, pp. 1064–1068.
Zhou, L., Chen, J., Song, Z., Ge, Z., Miao, A., 2014. Probabilistic latent variable regression model for process-quality monitoring. Chem. Eng. Sci. 116, 296–305.
Zhu, J., Ge, Z., Song, Z., 2015. Robust supervised probabilistic principal component analysis model for soft sensing of key process variables. Chem. Eng. Sci. 122, 573–584.