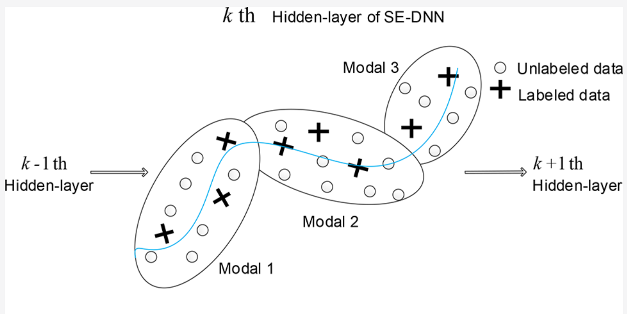
I&EC research
Industrial & Engineering Chemistry Research
pubs.acs.org/IECR
Article

Soft Sensor Modeling Method Based on Semisupervised Deep Learning and Its Application to Wastewater Treatment Plant

Weiwu Yan,* Renchao Xu, Kundong Wang,* Tang Di, and Zhang Jiang

Cite This: *Ind. Eng. Chem. Res.* 2020, 59, 4589−4601
Read Online
ACCESS | Metrics & More | Article Recommendations

ABSTRACT: This paper proposes a semisupervised deep neural regression network with embedding manifold (SSE-DNN) for soft sensor modeling that integrates manifold embedding into deep neural regression networks. Manifold embedding is imposed on the hidden layer of the deep neural regression network to form a semisupervised deep neural regression network. Manifold embedding exploits the local neighbor relationship among industrial data and utilizes unlabeled data effectively to improve the performance of the deep neural regression model. The SSE-DNN model exploits the global information and local manifold among industrial large data simultaneously and implements implicitly multimodal models of industrial process. The soft sensor model based on the SSE-DNN is applied to the estimation of total Kjeldahl nitrogen (TKN) in a long-term complicated wastewater treatment process. The experimental results demonstrate that the SSE-DNN model has a better performance than other soft sensors and provides an effective method for soft sensor modeling of complex industrial processes.



Feature distribution of the SSE-DNN’ s hidden layer.

1. INTRODUCTION

In modern industrial processes, some key process variables are difficult or impossible to measure online owing to economic or technical limitations. A soft sensor is a virtual sensing technique that estimates the key variables based on other available process variables to provide feasible and economical alternatives to costly or impractical physical measurement sensors. A soft sensor is a technology that combines data processing, mathematical modeling, and software techniques. Over the past two decades, soft sensors have attracted increasing attention and been considered a valuable alternative to the traditional means of acquiring critical process parameters, internal state estimation, process monitoring, and fault detection.[1−4]

The core issue of the soft sensor is the soft sensor model, which generates a virtual measurement to replace a real sensor measurement. From the perspective of modeling methods, soft sensor models can be divided into three categories: first-principles models, data-driven models, and mixed models of the first-principles model and the data-driven model. The first-principles model is not often available because of the complexity of the industrial process mechanisms and it’s computationally time consuming. Data-driven models are the most popular models for soft sensors. Currently, data-driven soft sensor modeling methods include Kalman filters, artificial neural networks (ANNs), partial least squares regression (PLS), support vector regression (SVR), Gaussian process regression (GPR), kernel ridge regression (KRR), Bayesian learning, ensemble learning, semisupervised learning, deep learning networks, and hybrid methods.[2−12] Although data-driven soft sensors have shown significant progress in theoretical studies and have been implemented successfully in real applications in industrial processes, there are still some important unresolved issues in soft sensor modeling, e.g., the generalization of the model, unlabeled data, and large industrial samples.

In soft sensor modeling of industrial processes, one can find a wealth of available unlabeled examples easily, while collecting labeled examples is often costly and time-consuming. Therefore, it is significant to develop algorithms that are able to utilize the information on unlabeled data to improve the performance of soft sensors. Currently, the issue of unlabeled data and different-rate process data is a popular topic in the soft sensor modeling field.[7,9,10,14,41] Semisupervised learning provides an effective way to make use of unlabeled data for a training model—typically a small amount of labeled data with a large amount of unlabeled data.[7] The commonly used semisupervised algorithms include transductive support vector machines, the semisupervised Bayesian method, and the semisupervised Gaussian process.[16] Semisupervised algorithms

Received: September 12, 2019
Revised: January 30, 2020
Accepted: February 7, 2020
Published: February 7, 2020

have been used to treat unlabeled data in soft sensor modeling.[7,14,15]

With the wide utilization of the distributed control system in industrial processes, large amounts of data have been recorded. The recorded industrial large data provide a feasible and effective way to improve the accuracy and generalization of the soft sensor model. However, the most widely used modeling methods for soft sensors are shallow learning methods that have shallow network architectures with one hidden layer and are unsuitable for large data modeling.[9] For complicated processes, it is not easy for shallow learning methods to obtain stable and reliable prediction results. Compared with conventional shallow learning modeling methods, deep neural networks with many nonlinearity layers could represent highly nonlinear and highly varying functions. Deep learning networks are especially suitable for large samples, which can greatly improve the model's performance. Deep learning networks have been successfully applied to pattern recognition and classification.[13] Considering that soft sensor modeling is a typical regression problem, a deep neural regression network provides a promising method for soft sensor modeling.[9,10,14]

Semisupervised regression and deep learning increasingly focus on the soft sensor area. It is necessary to combine semisupervised algorithms with deep neural networks to exploit large industrial data sets and unlabeled data for soft sensor modeling. Semisupervised deep learning is a variant of deep learning techniques that makes use of unlabeled data for better training the deep learning network. Semisupervised deep learning methods for soft sensor modeling include a deep learning network with unsupervised pretraining and a semisupervised deep learning network. Shang developed a soft sensor based on deep belief networks (DBNs) for the estimation of the heavy diesel 95% cut point of a crude distillation unit,[14] in which DBN utilized unlabeled data to pretrain parameters of deep learning networks. Yan et al. applied a semisupervised deep neural network (DNN) with stacked autoencoder (SAE) pretraining to estimate the oxygen content in flue gases in ultrasuperficial units.[9] Yuan proposed a semisupervised deep learning soft sensor with a variablewise weighted SAE for a debutanizer column process.[10] The semisupervised deep learning with pretraining initialization is a two-stage training method of deep learning networks, in which a greedy restricted Boltzmann machine (RBM) or autoencoder (AE) is used for parameter initialization. Subsequently, fine-tuning is used to train the deep learning network. Recently, sophisticated parameter initialization methods have been proposed for large-scale deep learning networks. In one-stage semisupervised deep learning networks,[17] unlabeled data are involved in the cost function optimization instead of only weight initialization, which may lead to an optimal model. This paper discusses a one-stage semisupervised deep neural regression network with embedding manifold in the settings of large and unlabeled data.

The wastewater treatment system is a very important type of facility in environmental problems. Some key process variables of the wastewater treatment process are difficult to measure online, such as total Kjeldahl nitrogen (TKN). These variables play an important role in controlling and optimizing the effluent process. Conventional data-driven methods such as PLS, SVR, and feed-forward neural networks (FNNs) have been applied to soft sensor modeling in wastewater treatment systems.[18−20] Most research has focused on the soft sensor modeling in wastewater treatment plants (WWTPs) under the condition of stable operation; however, there is little work on soft sensor modeling for long-term complicated WWTPs. Deep learning provides an effective method for the soft sensor modeling of the long-term complicated WWTPs. This paper discusses soft sensor modeling that integrates manifold embedding into the deep neural regression network (SSE-DNN). Then, the SSE-DNN based soft sensor is applied to the soft sensor modeling of a long-term complicated WWTP.

The remainder of the paper is organized as follows. Section 2 introduces the preliminary knowledge. Section 3 proposes the manifold embedding deep neural network for semisupervised soft sensor modeling. A case study on estimating TKN in a wastewater treatment plant is provided in section 4. Finally, conclusions are given in section 5.

2. PRELIMINARY KNOWLEDGE

The semisupervised deep regression network in this paper derives from manifold learning and a deep neural network. Deep neural networks and manifold learning are reviewed in this section, respectively.

2.1. Deep Neural Networks. Deep learning is a set of algorithms in machine learning that attempts to model high-level abstractions in data by using multiple nonlinear transformations. In deep learning algorithms, a layer-by-layer feature transform is employed to obtain essential and abstract feature representation to improve the prediction accuracy in supervised learning.[21] The commonly used building blocks in deep learning networks are restricted Boltzmann machines (RBMs),[22] autoencoder variants,[23] sparse coding variants,[24] and so on. Based on these structures, deep learning architectures include deep neural networks (DNNs),[25] deep belief networks (DBNs),[28] convolutional neural networks (CNNs),[27] recurrent neural networks (RNNs),[26] deep Boltzmann machines (DBMs),[29] and generative adversarial networks (GANs).[30]

Deep neural networks (DNNs) are feed-forward neural networks with deep structures and multiple stacks of nonlinear layers. With multiple nonlinear layers, a DNN can implement an extremely intricate function.[13] A simple DNN architecture is shown in Figure 1.

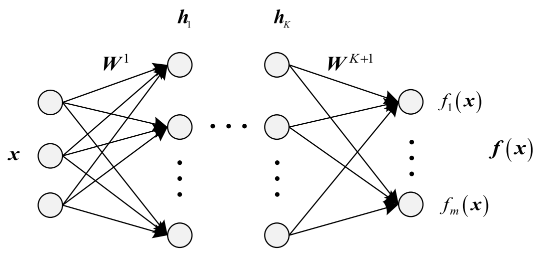


Figure 1. Architecture of a deep neural network.

For a DNN containing K hidden layers, the output of each hidden layer is defined as follows:

$$h_i^k(\boldsymbol{x}) = S\left(\sum_{j=1}^{n_{k-1}} w_j^{k,i} h_j^{k-1}(\boldsymbol{x}) + b^{k,i}\right) \tag{1}$$

where h_i^k represents the ith output of the kth hidden layer with n_k hidden units $(1 \leq k \leq K)$. $\boldsymbol{x}$ is the input of the DNN, and n is the dimension of x. $\boldsymbol{w}$ denotes the weights of the network, b

is the bias of the network, and S is the activation function such as sigmoid, ReLu (rectified linear units), and Tanh.

The output of the DNN is

$$f_i(\boldsymbol{x}) = \sum_{j=1}^{n_k} w_j^{k+1,i} h_j^k(\boldsymbol{x}) + b^{k+1,i}, \qquad i = 1, ..., m \tag{2}$$

where $f_i(\boldsymbol{x})$ is the ith dimension of the output, $f(\boldsymbol{x}) = (f_i(\boldsymbol{x}), ..., f_m(\boldsymbol{x}))$ is the output function of the network, and m denotes the dimension of the output.

The cost function of the DNN is written as

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_i(\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{f}(\boldsymbol{x}_i), \boldsymbol{y}_i, \theta) \tag{3}$$

where $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}^N$ denotes the data set, $\mathcal{L}(\bullet, \bullet)$ represents the loss function of the network parametrized by θ, and $\mathcal{L}_i(\theta)$ is the loss function of the ith training sample.

2.2. Manifold Learning. Seung, Tenenbaum, and Roweis et al. expounded upon the concept and development trend of manifold learning from the perspectives of visual memory and perception.[31−33] The aim of manifold learning is to obtain a low-dimensional manifold structure of the high-dimensional data space.

Manifold learning can be classified into three categories: surface projection methods, topology model generation methods, and manifold embedding methods. Manifold embedding methods are the most commonly used manifold learning algorithms and are often combined with other machine learning algorithms. Commonly used manifold learning algorithms include Laplacian eigenmaps (LEs), locally linear embedding (LLE), and isometric mapping (ISOMAP). An ISOMAP is a global nonlinear embedding algorithm that defines the geodesic distance along the manifold as a measure of similarity. However, it has a large computational complexity. LLE is a locally embedded manifold learning algorithm that is a combination of a series of local principal component analyses (PCAs). However, LLE performs poorly when attempting to process the data that are distributed over a closed sphere, ellipsoid, or cylinder. LE is also a local embedding algorithm that aims to study the low-dimensional manifold structure while best preserving the local neighborhood relationship of data. LE is closely related to LLE but is robust to outliers and noise. LE captures local information by constructing an adjacency graph and is capable of representing the intrinsic characteristics of mapping from the data space to the manifold space.

Manifold embedding algorithms can be described as a general optimization problem: Given a data set $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N$, find an embedding $\boldsymbol{g}$ of each point $\boldsymbol{x}_i$ by minimizing

$$\frac{1}{N}\sum_{i,j=1}^{N} l(\boldsymbol{g}(\boldsymbol{x}_i), \boldsymbol{g}(\boldsymbol{x}_j), w_{ij}) \tag{4}$$

where $\boldsymbol{g}(\boldsymbol{x}) \in \mathbb{R}^m$ is the embedded function. $l(\bullet,\bullet)$ is the embedding loss function between pairs of samples. $\boldsymbol{W}$ denotes the weight matrix, and w_{ij} measures the similarity between data $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

Assuming that the original data are $\boldsymbol{X} = \{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^n\}_{i=1}^N$, where N is the number of the data set and n is the original data dimension, LE expects that the correlated points will be as close as possible in the manifold space. Therefore, the objective function of LE is as follows:

$$\min \sum_{i,j=1}^{N} w_{ij} \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2 \tag{5}$$

where w_{ij} is the element of the symmetric matrix $\boldsymbol{W}$ that reflects the neighborhood relations. Define the graph Laplacian matrix as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D}$ is the diagonal matrix with diagonal elements $d_{ii} = \sum_j w_{ij}$. A k-nearest-neighbors (kNN) graph is usually used for the graph construction owing to its simplicity and effectiveness. In a kNN graph, the affinity matrix $\boldsymbol{W}$ is calculated as $w_{ij} = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$.

3. DEEP NEURAL NETWORK WITH MANIFOLD EMBEDDING FOR SEMISUPERVISED SOFT SENSOR MODELING

Soft sensor modeling is essentially semisupervised regression rather than widely used supervised regression.[7] Semisupervised regression provides a more suitable framework than supervised regression does for building a data-driven soft sensor model. Therefore, it is necessary to develop semisupervised algorithms to improve the performance of a soft sensor.

The graph-based manifold embedding algorithm is a typical example of semisupervised learning. Manifold regularization provides a framework in which manifold embedding acts as a regularization term to combine with other machine learning methods.[34] Under the manifold regularization framework, this paper discusses a semisupervised soft sensor modeling method that integrates the manifold embedding into a deep regression network. It is called a semisupervised manifold embedding deep neural network (SSE-DNN) and combines LE with DNNs to form a semisupervised deep neural network algorithm.

3.1. Manifold Regularization Framework. Define $\boldsymbol{X} = \{\boldsymbol{x}_i \in R^n, i = 1, ..., l + u\}$ as the set of training samples, which consists of l labeled samples with the label of each sample being $y_i \in R$ and u unlabeled samples. Denote $\boldsymbol{X} = \boldsymbol{X}_{\mathrm{L}} \cup \boldsymbol{X}_{\mathrm{U}}$, where $\boldsymbol{X}_{\mathrm{L}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_l\}$ is the set of labeled samples drawn according to the joint distribution $P(\boldsymbol{X},\boldsymbol{Y})$ defined on $\boldsymbol{X} \times \boldsymbol{R}$, and $\boldsymbol{X}_{\mathrm{U}} = \{\boldsymbol{x}_{l+1}, \boldsymbol{x}_{l+2}, ..., \boldsymbol{x}_{l+u}\}$ is the set of unlabeled samples drawn according to the marginal distribution $P(\boldsymbol{X})$.

For supervised regression such as SVR, the goal is to create a mapping $\boldsymbol{Y} = \boldsymbol{f}(\boldsymbol{X})$ from labeled input−output pairs $(\boldsymbol{X}, \boldsymbol{Y})$. For a Mercer kernel K: $\boldsymbol{X} \times \boldsymbol{X} \to \mathbb{R}$, there is an associated reproducing kernel Hilbert space (RKHS) H_{K} of functions $\boldsymbol{X} \to \mathbb{R}$ with the corresponding norm $\|\bullet\|_K$.[35] Given a set of labeled examples, the supervised regression algorithms estimate an unknown function by minimizing

$$\underset{F \in H_{\mathrm{K}}}{\arg\min} \frac{1}{N_{\mathrm{L}}} \sum_{i=1}^{N_{\mathrm{L}}} V(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{f}) + \gamma_A \|\boldsymbol{f}\|_K^2 \tag{6}$$

where $V(\bullet)$ is a loss function. Penalizing the RKHS norm $\|\boldsymbol{f}\|_K^2$ imposes smoothness conditions on possible solutions.

For semisupervised regression, both labeled samples and large amounts of unlabeled samples are utilized to create the mapping $\boldsymbol{Y} = \boldsymbol{f}(\boldsymbol{X}) = \boldsymbol{f}(\boldsymbol{X}_{\mathrm{L}}, \boldsymbol{X}_{\mathrm{U}})$. According to Bayesian theory, $P(\boldsymbol{Y}|\boldsymbol{X})$ and $P(\boldsymbol{X})$ are closely related, so $P(\boldsymbol{X}_{\mathrm{U}})$ may be used to improve $P(\boldsymbol{Y}|\boldsymbol{X})$.Unlabeled data $\boldsymbol{X}_{\mathrm{U}}$ can be usefully processed to learn natural invariances to develop better regression tasks.

Manifold regularization exploits the geometry of the marginal distribution $P(\boldsymbol{X})$ by making an assumption about the connection between the marginal and the conditional distributions.[35,36] Assume that if two points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \boldsymbol{X}$ are close in the intrinsic geometry of $P(X)$, then $P(y|x_1)$ and $P(y|x_2)$ are similar. In other words, the conditional distribution $P(y|x)$ varies smoothly along the geodesics in the intrinsic geometry of $P(X)$.

Belkin et al. proposed a Laplacian manifold regularization framework based on the classical RKHS combined with LEs. To ensure that the solution is smooth with respect to both the ambient space and the marginal distribution $P(X)$ (intrinsic space), an additional regularizer $\|f\|_I^2$, named the intrinsic regularizer or manifold regularizer, is integrated into the objective function. The manifold regularization framework is expressed as an optimization problem:[35]

$$\underset{F \in H_K}{\arg\min} \frac{1}{N_L} \sum_{i=1}^{N_L} V(x_i, y_i, f) + \gamma_A \|f\|_K^2 + \gamma_I \|f\|_I^2 \tag{7}$$

In most applications, the marginal $P(X)$ is not available. Therefore, the manifold regularizer is usually approximated by the graph Laplacian matrix associated with X. Assume the support M of $P(X)$ is a compact submanifold, the approximation of the manifold regularizer $\|f\|_I^2$ is

$$\|f\|_I^2 = \sum_{i,j=1}^{N_L+N_U} w_{ij} \left\|g(x_i) - g(x_j)\right\|^2 \tag{8}$$

The concept of the semisupervised manifold embedding method is to learn the inner geometric manifold structure of the data using the unlabeled samples and to learn the labeled information on the entire manifold from a small amount of labeled information. Thus, given a set of l labeled examples and a set of u unlabeled examples, the objective function of semisupervised manifold regularization is written as the following optimization problem:[35]

$$\underset{F \in H_K}{\arg\min} \frac{1}{N_L} \sum_{i=1}^{N_L} V(x_i, y_i, f) + \gamma_A \|f\|_K^2 + \frac{\gamma_I}{(l+u)^2} \sum_{i,j=1}^{N_L+N_U} w_{ij} \left\|g(x_i) - g(x_j)\right\|^2 \tag{9}$$

where V is an arbitrary loss function, H_K is the RKHS, and K is the RBF kernel. γ_A and γ_I are regularization parameters. $\{w_{ij}\}_{i,j=1}^{N_L+N_U}$ measures the similarity between samples. The above optimization objective is composed of three terms: the first term is the loss function of supervised learning, the second term is the RKHS norm as a stabilizer in the framework, and the third term is the unsupervised Laplacian manifold regularization.

3.2. Semisupervised Manifold Embedding DNNs.

In soft sensor applications, since variables of industrial processes have strong coupling and collinearity, the intrinsic degree of freedom of the industrial system may be lower than the dimension of the observed variables in the data space. The distribution of industrial data can be seen as a low-dimensional manifold in a high-dimensional representation space. According to the Laplacian manifold regularization framework,[17,35] manifold embedding algorithms can be applied to deep neural networks naturally to form semisupervised deep regression models for soft sensor modeling, in which the deep neural regression network aims to represent highly nonlinear functions and provide a global regression model on the data set. Meanwhile, manifold embedding exploits the local intrinsic structure among the data to improve the soft sensor model with a wealth of unlabeled examples.

The manifold embedding is introduced into the DNNs by imposing the manifold embedding regularization onto the objective function of the deep regression network to optimize jointly. The deep neural regression network and embedding manifold are decoupled in one formulation to build a global and local regression model. Compared with the manifold regularization framework, the cost of supervise learning is replaced by that of a deep regression network in the SSE-DNN model. The general objective function of the SSE-DNN model is written as follows:

$$\sum_{i=1}^{N_L} \mathcal{L}(f(x_i), y_i) + \lambda \sum_{i,j=1}^{N_L+N_U} l(g(x_i), g(x_j), w_{ij}) \tag{10}$$

where the first term is the loss function of the deep regression network, taking the place of the loss function of the regularization algorithms, and the second term is the manifold regularization term. The parameter λ is the regularization parameter that controls the trade-off between the supervised learning loss and the manifold regularization loss.

The manifold regularization term is imposed on different positions of the deep learning regression network to form different semisupervised deep learning models. There are two implementations of semisupervised regularization in deep architectures, which are detailed as follows:

First, the manifold embedding regularization is added to the output of the entire network. The SSE-DNN structure with manifold embedding is shown in Figure 2a. The objective

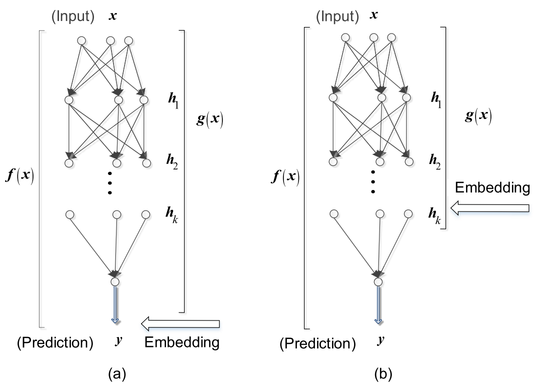


Figure 2. Two modes of embedding in deep neural networks.

function in this scenario is equivalent to eq 10. In this network structure, $g(x)$ and $f(x)$ have the same formula. The manifold embedding regularization is equivalent to the constraint of the network to some extent, thus ensuring that $g(x)$ is continuous on the input x.

Second, the manifold embedding regularization is imposed on the kth hidden layer of the network. The overall objective function is

$$\sum_{i=1}^{N_L} \mathcal{L}(f(x_i), y_i) + \lambda \sum_{i,j=1}^{N_L+N_U} l(g^k(x_i), g^k(x_j), w_{ij}) \tag{11}$$

where $g^k(\bullet)$ denotes the manifold embedding on the kth hidden layer. The regularization term on the hidden layer can strengthen the generalization ability of the network due to its improved feature extraction capability. The SSE-DNN structure with manifold embedding applied to the kth layer of the network is shown in Figure 2b.

Because the second method is flexible in structure, easy to implement, and effective in optimization, it is chosen for the soft sensor modeling. For deep regression networks, the mean square error loss is generally selected as the supervised objective function, i.e.

$$\mathcal{L}(f(\boldsymbol{x}_i), \boldsymbol{y}_i) = \|f(\boldsymbol{x}_i) - \boldsymbol{y}_i\|^2$$

Based on the Laplacian manifold regularization framework, LE is added to the kth hidden layer of the network. The overall objective function is

$$\sum_{i=1}^{N_L} \|f(\boldsymbol{x}_i) - \boldsymbol{y}_i\|^2 + \lambda \sum_{i,j=1}^{N_L+N_U} w_{ij} \left\|g(\boldsymbol{x}_i) - g(\boldsymbol{x}_j)\right\|^2 \tag{12}$$

The radial basis function (RBF) is used to measure the distance between the data in the original input space by mapping the similarity of x_i and x_j to $d_{ij} \in [0, 1]$. In addition, w_{ij} can be obtained according to d_{ij}.

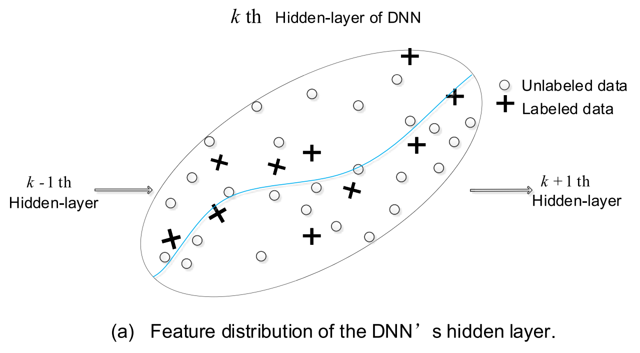
$$d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{e}^{-\|x_i - x_j\|^2/\gamma} \tag{13}$$

$$w_{ij} = \begin{cases} d_{ij} & \boldsymbol{x}_i, \boldsymbol{x}_j \text{ are neighbors} \\ 0 & \boldsymbol{x}_i, \boldsymbol{x}_j \text{ are not neighbors} \end{cases} \tag{14}$$

where $\gamma = 2\sigma^2$ and σ is the radius of the RBF.

3.3. Illustrations of the Feature Distribution of the SSE-DNN. The SSE-DNN is a modified version of the DNN, in which embedding manifold is employed to exploit the local manifold structure. The schematic diagrams in Figure 3 are adopted to illustrate the variation of hidden feature distribution from DNN to the SSE-DNN. Figure 3a exhibits the feature distribution of the DNN's hidden layer, and Figure 3b exhibits the feature distribution of the SSE-DNN's hidden layer. Blue lines represent the curve fitting using DNN and SSE-DNN. The DNN mainly builds a global model based on Euclidean distance in data sets. In the SSE-DNN, embedding manifold captures the local manifold structures through exploiting neighbor relationships among input data, especially the large number of unlabeled data that influence feature distribution of modeling data, and decrease the influence of noise and outliers on the model to some extent. The local manifold structures of the process data can be considered as different modals of industrial processes. Therefore, the SSE-DNN builds the local models and implements implicitly multimodal models. Meanwhile, the local manifold structure decreases the influence of outlier on the model; it is useful for improving robustness of the model.

3.4. Algorithm and Training Strategy. The SSE-DNN model can be obtained by solving the objective function in eq 12. According to different strategies for optimizing the regularization loss, the training methods can be divided into the joint training method and the separate training method. In the joint training method, supervised loss and regularization loss are united to take the derivative of the parameter. For the separate training method, supervised learning loss and regularization loss are trained and optimized in separate steps, with the network updated in the iteration. Two loss terms can take different training methods, and their training times can each be adjusted. Considering that the optimization objective of SSE-DNN is the combination of two convex functions, the convergence of the alternative optimization for training SSE-DNN can be guaranteed. The separate training method is used to solve the objective function of SSE-DNN. Table 1 depicts an overview of the learning algorithm with the separate training method.


k th Hidden-layer of DNN
Unlabeled data
Labeled data
k - 1 th Hidden-layer
k + 1 th Hidden-layer
(a) Feature distribution of the DNN's hidden layer.

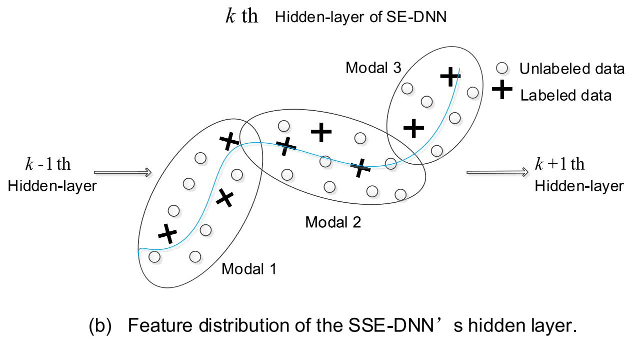


Figure 3. Diagram of hidden feature distribution for illustrating the SSE-DNN.

4. EXPERIMENT

A case study on estimating key variables in a wastewater treatment process is discussed in this section. The benchmark simulation model no. 2 (BSM2) is considered to evaluate the effectiveness of the soft sensor modeling method based on the SSE-DNN. Different soft sensor modeling methods are compared on different sizes of data sets.

4.1. Wastewater Treatment Plants. Wastewater treatment plants (WWTPs) are very important examples of an environmental problem that aim to reduce the quantity of organic or hazardous chemicals that flow into sewage. WWTPs are large nonlinear systems that are subject to large perturbations in influent flow rate and pollutant load as well as uncertainties concerning the composition of the incoming wastewater. Some key process variables of wastewater treatment processes are difficult to measure online, such as total Kjeldahl nitrogen (TKN). These variables play an important role in controlling and optimizing the effluent process. To verify the effectiveness of soft sensors based on the SSE-DNN, the benchmark simulation model of WWTPs is considered as a case study.

The Benchmark Simulation Model No. 1 (BSM1) is a simulation environment of wastewater treatment processes.[37] The layout of the BSM1 plant is shown in Figure 4. BSM1 is composed of a five-compartment activated sludge reactor, i.e., two anoxic tanks and three aerobic tanks. Usually, the input file in BSM1 consists of 14 days of sampled data with a sampling interval of 15 min. However, several limitations remain in

Table 1. Algorithm of the Semisupervised Manifold Embedding Deep Neural Network

```
Algorithm : Semi-supervised Manifold Embedding Deep Neural Network

Inputs: Labeled dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i) | i = 1, \ldots, N_L\}$
        Unlabeled dataset $\{\boldsymbol{x}_i | i = 1, \ldots, N_U\}$
Parameters: Regression network: $\boldsymbol{f}(\boldsymbol{x})$
            Embedding network: $\boldsymbol{g}(\boldsymbol{x})$
            Nearest neighbors: $K$
            Distance measure function of original space: $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\gamma}}$
Step 1: Compute the distance matrix of samples $\{\boldsymbol{x}_i | i = 1, \ldots, N_L + N_U\}$ according to $d_{ij}$. Sort
        distance by $K$ to determine the nearest neighbor of samples. If $x_i$ and $x_j$ are not near
        neighbors, then $w_{ij} = 0$.
Step 2: Initialize weights of deep network using parameters initialization.
Step 3: Solve the overall objective function $\sum_{i=1}^{N_L} \|\boldsymbol{f}(\boldsymbol{x}_i) \text{-} \boldsymbol{y}_i\|^2 + \lambda \sum_{i,j=1}^{N_L+N_U} w_{ij} \|\boldsymbol{g}(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_j)\|^2$.
        Train and optimize $\mathscr{L}(\boldsymbol{f}(\boldsymbol{x}_i), \boldsymbol{y}_i)$ and $\lambda l(\boldsymbol{g}(\boldsymbol{x}_i), \boldsymbol{g}(\boldsymbol{x}_j), w_{ij})$.
        Repeat
            For t steps do
                Sample from labeled dataset $(\boldsymbol{x}_i, \boldsymbol{y}_i)$.
                Optimize and update $\sum_{i=1}^{N_L} \|\boldsymbol{f}(\boldsymbol{x}_i) \text{-} \boldsymbol{y}_i\|^2$ by gradient.
            End for
            Choose nearest neighbor pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ in $\{\boldsymbol{x}_i | i = 1, \ldots, N_L + N_U\}$ randomly.
            Optimize and update $\lambda \sum_{i,j=1}^{N_L+N_U} w_{ij} \|\boldsymbol{g}(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_j)\|^2$ by gradient.
        Until stopping criteria are met.
Step 4: Predict key variables using SSE-DNN-based soft sensor.
```

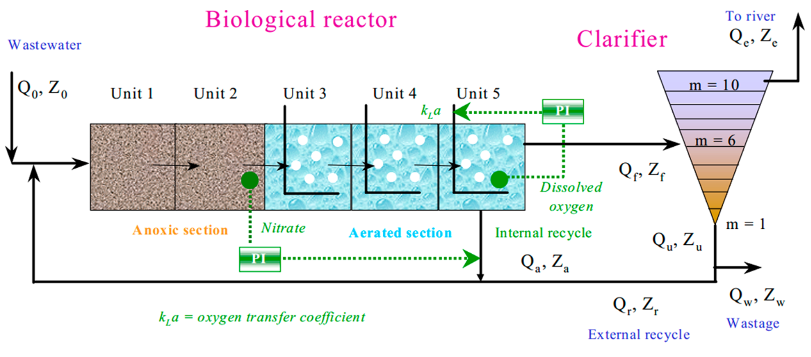


Figure 4. General overview of the BSM1 plant. Reprinted with permission from ref 37. Copyright 2008 Lund University.

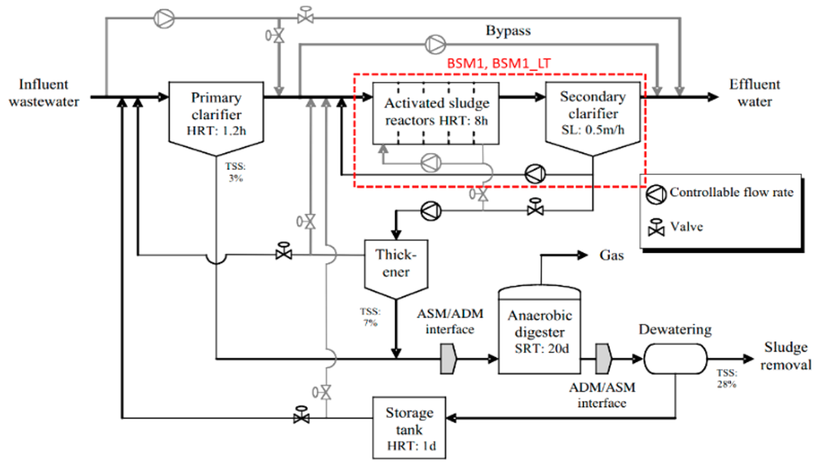


Figure 5. General overview of the BSM2 plant. Reprinted with permission from ref 39. Copyright 2007 IWA Publishing.

BSM1: the BSM1 is restricted to the activated sludge process; the sensors in the model are ideally far from the actual situation; the perturbation data are imperfect; the simulation time is too short; the simulation data can only reflect the characteristics of wastewater within 4 weeks, which is insufficient for a term test. Many studies aiming at the soft sensor data analysis of 14 days of data under different weather conditions have achieved very good accuracy. However, there is little research on using soft-sensing modeling for a long-term complicated WWTP, which is close to a real application situation. For instance, some key variables of a long-term complicated WWTPs fluctuate sharply, which makes the prediction more difficult.

Gernaey et al.[38] proposed the long-term benchmark simulation model No. 1 (BSM1_LT) to extend BSM1 for evaluating process monitoring methods. Based on the BSM1_LT, Jeppsson et al.[39] proposed the benchmark simulation model no. 2 (BSM2) for evaluation of plantwide control strategies. The input files of BSM1_LT and BSM2 are not determined from predefined data files but are designed according to the model. BSM1_LT and BSM2 can reflect the long-term and complex characteristics of the system. The layout of the BSM2 plant is depicted in Figure 5.

4.2. Experiments and Results.

The output data of BSM2, whose input file is the dynamic standard input of BSM1_LT, are used for simulation. Seventeen process variables in the output of BSM2 are chosen as secondary variables for TKN prediction, which are listed in Table 2.

There are 609 days of data from BSM2, and they are partitioned into 245 days as the setting cycle and 364 days as the evaluation cycle. In the evaluation cycle, the first 184 days of data, with 17 664 total samples, are used for training and the remaining 180 days of data, 17 280 samples, are used as the test set. All variable values are scaled to [0, 1]:

$$\boldsymbol{x}^* = \frac{x - \min(\boldsymbol{x})}{\max(\boldsymbol{x}) - \min(\boldsymbol{x})} \tag{15}$$

In the simulation, the SSE-DNN-based soft sensor is compared to the PLS-, KRR-, GPR-, SVR-, DNN-, and SAE-DNN-based soft sensors using different numbers of labeled samples. SVR is a state-of-the-art method for soft sensor modeling that often is used as a baseline for soft sensor modeling. One-fifth of the training data are selected as the validation set for fine-tuning the hyperparameters.

Table 2. List of Process Variables as Secondary Variables for TKN Prediction

variable definition	Notation
soluble inert organic matter	S_I
readily biodegradable substrate	S_S
particulate inert organic matter	X_I
slowly biodegradable substrate	X_S
active heterotrophic biomass	X_{BH}
active autotrophic biomass	X_{BA_1} and X_{BA_2}
particulate products arising from biomass decay	X_P
oxygen	S_O
alkalinity	S_{ALK}
NH_4^+ + NH_3 nitrogen	S_{NH}
nitrate and nitrite nitrogen	S_{NO_2}, S_{NO}, and S_{N_2O}
nitrogen	S_{N_2}
flow	Q
temperature	T

The structure of the DNN-based soft sensors is selected from a group of manually designed network configurations, including the number of hidden layers, the number of units in each hidden layer, and the activation. Considering the prediction performance and overfitting on the validation set, the structure of the SAE-DNN-, SSE-DNN-, and DNN-based soft sensors is finally chosen to be 17/40/35/25/15/5/1, and the activation is ReLu. Another hyperparameter of the SSE-DNN, the nearest neighbor *K*, is chosen to be 6 by similar experiments. Lecun uniform is used as parameter initialization for both the DNN and SSE-DNN. The SAE-DNN, SSE-DNN, and DNN models have the same experimental conditions: the optimizer is Adam, the learning rate is 0.003 at the beginning, and the weight decay technique that reduces the learning rate over time is used. The Gaussian kernel is used in the SVR, and the hyperparameters of the SVR are chosen through the validation set. The candidate sets of the initial learning rate for pretraining and the momentum value are {0.01, 0.1, 1, 10, 30, 100, 500} and {0.01, 0.03, 0.1, 0.3, 1, 3, 10}, respectively. Similar to SVR, RBF kernel is used in KRR, and the candidate sets of parameters alpha and gamma are {0.003, 0.01, 0.1, 10, 30, 100} and {0.01, 0.03, 0.1, 0.3, 1, 3, 10}, respectively. The number of principal components of PLS is selected using 5-fold cross validation. RBFs are also used in GPR where the

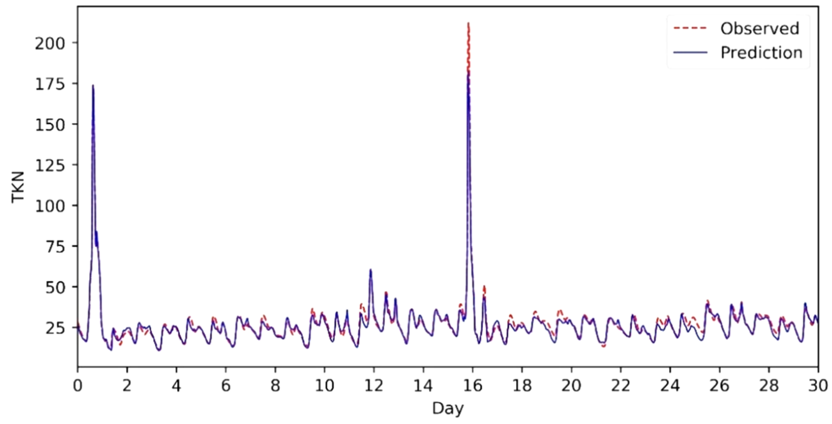


Figure 6. Prediction curves of TKN by the soft sensor based on SSE-DNN.

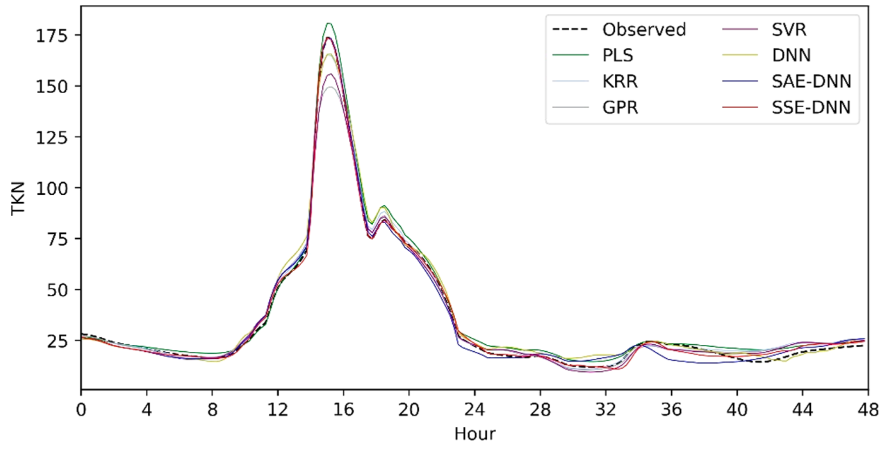


Figure 7. Prediction curves of TKN by different soft sensors on a sharp spike data.

noise $\varepsilon \sim N(0{,}0.15)$ and kernel parameters are selected by GPML Toolbox.[40]

The PLS-, KRR-, GPR-, SVR-, DNN-, SAE-DNN-, and SSE-DNN-based soft sensors are trained using different sample sizes (1000, 2000, and 5000) to illustrate the model performance. The three sets of labeled samples are selected from the training set randomly. To investigate the effect of unlabeled data on prediction accuracy, an extra 15 000 unlabeled data points are chosen from the training set. Considering that spike data account for only a small proportion of the data set, 15 000 unlabeled data are employed in order to get more spike data for the training model. For the PLS-, KRR-, GPR-, SVR-, and DNN-based soft sensors, only labeled samples are used for training, while the SAE-DNN- and SSE-DNN-based soft sensors are trained using the labeled samples and 15 000 unlabeled samples.

The mean square error (MSE), mean absolute percentage error (MAPE), and relative variance tracking performance (RVTP) in the regression tasks are used as evaluation indexes to measure the global error and relative error of prediction results as well as the performance trend. When RVTP is closer to 1, the soft sensor has a more favorable trend. MSE, MAPE, and RVTP are calculated as follows:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_{o,i} - y_{p,i})^2 \tag{16}$$

$$\mathrm{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_{o,i} - y_{p,i}}{y_{o,i}}\right| \tag{17}$$

$$\mathrm{RVTP} = 1 - \frac{\sum_{i=1}^{N}(y_{o,i} - y_{p,i})^2}{\sum_{i=1}^{N}(y_{o,i} - \bar{y}_o)^2} \tag{18}$$

where $y_{o,i}$ and $y_{p,i}$ represent the ith observation and ith prediction, respectively. $\bar{y}_o$ denotes the mean value of all observations.

BSM2 simulates the long-term characteristics of WWTPs and defines dynamic disturbances in which the long-term output data vary in a complicated fashion. The output in BSM2 fluctuates sharply every half-month, which makes the regression task more difficult. The prediction curves of TKN by the SSE-DNN-based soft sensor trained with 1000 labeled samples as well as the 15 000 unlabeled samples are shown in Figure 6 (only 30 days of prediction are shown). From Figure 6, it is seen that there are a large number of data under the condition of stable operation and a small number of spike data in the data set of the BSM2 system. The prediction curves in Figure 6 demonstrate that the SSE-DNN-based soft sensors perform well in following the varying trend of TKN. To clearly show the detailed performance and comparisons of different soft sensors (SVR, PLS, KRR, GPR, DNN, SAE-DNN, and SSE-DNN), the prediction curves of TKN by the different soft sensors at a spike are shown in Figure 7 and the prediction

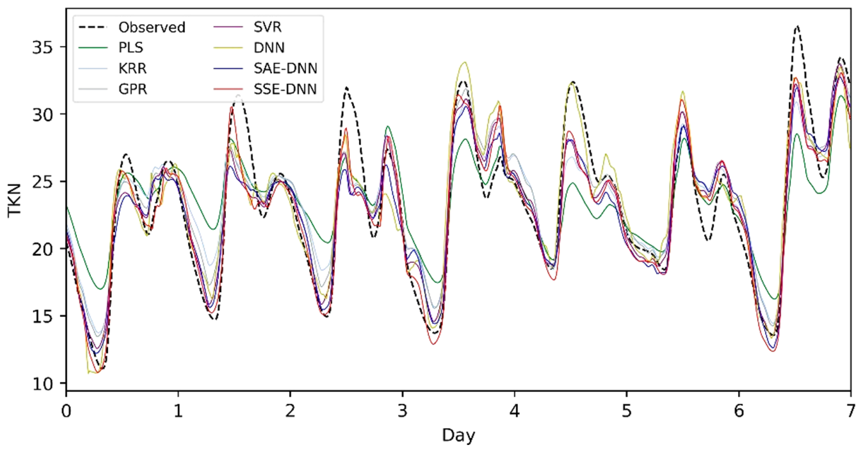


Figure 8. Prediction curves of TKN by different soft sensors during the stable operation period.

Table 3. Prediction Results of TKN by Different Soft Sensors with Different Labeled Samples

		MSE		MAPE		RVTP	
methods	labeled samples	train	test	train	test	train	test
PLS	5000	11.2462	10.0310	0.3215	0.2425	0.9311	0.9252
	2000	11.2525	9.8464	0.3185	0.2377	0.9362	0.9266
	1000	11.5181	9.7798	0.3125	0.2382	0.9388	0.9271
KRR	5000	5.8135	5.7405	0.1557	0.1480	0.9644	0.9572
	2000	5.9140	6.0697	0.1567	0.1511	0.9665	0.9547
	1000	7.1217	7.6116	0.1781	0.1671	0.9621	0.9432
GPR	5000	5.7792	5.7042	0.1529	0.1454	0.9646	0.9575
	2000	5.9027	6.0345	0.1553	0.1494	0.9665	0.9550
	1000	5.9238	6.9698	0.1514	0.1454	0.9685	0.9480
SVR	5000	4.9154	5.0839	0.1142	0.1266	0.9699	0.9621
	2000	6.4197	6.7263	0.1491	0.1494	0.9636	0.9498
	1000	6.5707	7.1399	0.1434	0.1483	0.9651	0.9468
DNN	5000	3.9665	4.6243	0.1200	0.1248	0.9757	0.9655
	2000	5.1528	6.0205	0.1385	0.1406	0.9708	0.9551
	1000	5.4232	6.9608	0.1317	0.1534	0.9712	0.9481
SAE-DNN	5000	3.7834	4.5136	0.1000	0.1158	0.9677	0.9586
	2000	4.7892	6.0139	0.1112	0.1343	0.9726	0.9534
	1000	5.3153	6.8000	0.1312	0.1451	0.9741	0.9493
SSE-DNN	5000	3.5618	4.0333	0.1061	0.1131	0.9782	0.9699
	2000	4.5277	5.1973	0.1226	0.1316	0.9743	0.9612
	1000	4.9210	6.4232	0.1339	0.1362	0.9738	0.9521

curves of TKN by the different soft sensors during the stable operation period are shown in Figure 8.

From Figures 7 and 8, it is seen that the SSE-DNN-based soft sensor exhibits better comprehensive performance than the other soft sensors, especially at spikes. The SSE-DNN model successfully captures local structure such as the spikes of long-term complicated WWTPs.

To verify the effectiveness of the SSE-DNN-based soft sensors, the prediction results of TKN by the different soft sensors trained with 1000, 2000, and 5000 labeled samples are reported in Table 3. The experimental results show that the soft sensor based on the SSE-DNN outperforms the other soft sensor modeling methods on MSE, MAPE, and RVTP measures. The SSE-DNN model decreases prediction errors by about 13% compared with the DNN model and by 21% compared with the SVR model. Compared with the DNN-based soft sensor, the SSE-DNN-based soft sensor improves performance by exploiting local manifold structure in the data set and incorporating information of the unlabeled data with the labeled data. A few spike data represent local structures in the data set; thus the manifold embedding of the SSE-DNN model could capture local structure in the data set well and exhibit excellent performance at spikes. The SSE-DNN model and DNN model seem to perform similarly during the period of stable operation. Since the number of spike data is small compared with the number of stable operation data, the whole error difference between the DNN and SSE-DNN on all data sets is not as large as on spike data in terms of MSE, MAPE, and RVTP. The SAE-DNN-based soft sensor exhibited slightly better performance than the DNN-based soft sensor but poor performance compared with the SSE-DNN-based soft sensor. The SAE-DNN model is a two-stage semisupervised DNN model that consists of two independent networks, i.e., an unsupervised network for weight initialization and a supervised DNN for regression model. A stacked autoencoder (SAE) initialization is used for weight initialization in the pretraining of the SAE-DNN model. In contrast, the SSE-DNN model is a concise end-to-end network model, in which unlabeled samples are involved in the cost function optimization of the SSE-DNN model instead of weight initialization. For the complex data set with spike data of the BSM2 system, weight initialization does not improve the performance of the DNN model too much. In fact, the batch normalization technology has reduced the sensitivity of different parameter initializations to network initialization in real applications. Therefore, the SSE-DNN model is especially suitable for the soft sensor modeling of long-term complicated WWTPs.

To compare efficiencies of different models, computational time experiments of TKN prediction are conducted on different sizes of data sets. All experiments are carried out on an Intel Xeon E5-2640 CPU and an NVidia Titan XP GPU. Table 4 shows training time and testing times of different modeling methods on different sizes of data sets. It should be noted that the SSE-DNN-based soft sensors are trained using both labeled samples and unlabeled samples. It can be seen that it takes more computational time to train the DNN model, the SAE-DNN model, and the SSE-DNN model than many other models. This is mainly because there are more parameters to be tuned for multiple layers of the deep learning network. The linear modeling methods such as PLS have the highest efficiency for both the training model and prediction and are free of the size of the training data due to their simple model structures. The kernel based models such as KRR, GPR, and SVR models are highly efficient on small data sets but become inefficient with increasing of the data size. However, once the model is well trained, prediction of the deep learning network model is highly efficient and free of the size of training data.

Table 4. Training Time and Testing Time of TKN Prediction by Different Modeling Methods

methods	labeled samples	training time (s)	prediction time (s)
PLS	5000	2.8	0.002
	2000	2.5	0.002
	1000	2.4	0.002
KRR	5000	24.1	0.273
	2000	10.1	0.098
	1000	5.5	0.068
GPR	5000	52.4	0.723
	2000	14.5	0.394
	1000	3.3	0.167
SVR	5000	38.8	1.354
	2000	6.2	0.578
	1000	3.7	0.287
DNN	5000	445.6	0.330
	2000	295.5	0.330
	1000	186.4	0.330
SAE-DNN	5000	495.1	0.330
	2000	360.5	0.330
	1000	229.2	0.330
SSE-DNN	5000	495.1	0.350
	2000	481.2	0.350
	1000	476.7	0.350

To verify the influence of the manifold embedding position on the performance of the SSE-DNN-based soft sensors, experiments in which the manifold regularization term is projected onto different hidden layers are conducted under the same conditions: the same training set (2000 labeled samples and 15 000 unlabeled samples) and test set and the same training method are used. The corresponding experimental results are shown in Table 5. Li (i = 1, 2, 3, 4, 5) indicates that the manifold is imposed on the different layers of the deep learning network.

Table 5. Evaluation Results of TKN by the SSE-DNN-Based Soft Sensors with Manifold Embedding on Different Hidden Layers

	MSE		MAPE		RVTP	
hidden layer	train	test	train	test	train	test
L1	5.1261	5.7615	0.1294	0.1353	0.9709	0.9570
L2	5.0219	5.7229	0.1283	0.1359	0.9715	0.9573
L3	4.6842	5.4714	0.1242	0.1326	0.9734	0.9592
L4	4.5248	5.2986	0.1245	0.1323	0.9744	0.9605
L5	4.5277	5.1973	0.1226	0.1316	0.9743	0.9612

From Table 5, the experimental results demonstrate that the performance of the SSE-DNN-based soft sensor is related to the layer number of the manifold embedding. When the manifold embedding is closer to the output of the SSE-DNN model, better prediction performance is obtained. When the manifold embedding is on the last hidden layer, the SSE-DNN-based soft sensor has the best performance. When training a deep learning network, because gradient-based learning methods are used to optimize the multilevel hierarchy of the DNN model, the gradients decay from the input layer to the output layer. Therefore, when a manifold embedding is close to the output layer (back-end hidden layer) of the DNN model, it will have more effect on the DNN model. If manifold embedding is added to the closed input layer (fore-end hidden layer) of the DNN model, the effect of manifold embedding on the DNN model will decay. That is, a manifold embedding on the back-end hidden layer has more influence on and makes a greater contribution to the DNN model. Therefore, it is recommended to impose the manifold embedding onto the back-end hidden layer of a deep learning network instead of the fore-end hidden layer in the SSE-DNN model.

To further investigate the influence of the manifold embedding on the performance of the DNN model, experiments of the DNN model with different numbers of layers, seven layers and five layers respectively, are carried out on different sizes of data sets. The corresponding evaluation results of TKN by the DNN and SSE-DNN-based soft sensors with different layers are shown in Table 6. It can be found that the SSE-DNN model with five layers even has a slightly better performance than the DNN model with seven layers, while the DNN model with seven layers has a better performance than the DNN model with five layers. That is, manifold embedding decreases the depth and complexity of the deep neural network to some extent. In the experiment, with the number of layers of the DNN-based soft sensor increasing, the DNN model is somewhat sensitive to training methods and hyperparameters. It is not easy to reach a stable convergent state and avoid overfitting, especially when the number of hidden layers in the network grows. By contrast, the SSE-DNN-based soft sensor is insensitive to training methods and hyperparameters. The SSE-DNN-based soft sensor can overcome the overfitting problem of the DNN-based soft sensor to some extent.

4.3. Visualization Analysis. To further evaluate the effect of manifold embedding, features extracted from the hidden layer of the DNN- and SSE-DNN-based soft sensor are

Table 6. Evaluation Results of TKN by DNN and SSE-DNN-Based Soft Sensors

methods	labeled samples	MSE train	MSE test	MAPE train	MAPE test	RVTP train	RVTP test
DNN (7 layers: 17−40−35−25−15−5−1)	5000	3.9665	4.6243	0.1200	0.1248	0.9757	0.9655
	2000	5.1528	6.0205	0.1385	0.1406	0.9708	0.9551
	1000	5.4232	6.9608	0.1317	0.1534	0.9738	0.9521
DNN (5 layers: 17−40−35−15−1)	5000	4.6804	5.3718	0.1166	0.1299	0.9713	0.9599
	2000	5.2908	6.2448	0.1340	0.1421	0.9700	0.9534
	1000	5.5728	7.0723	0.1404	0.1503	0.9704	0.9473
SSE-DNN (5 layers: 17−40−35−15−1)	5000	4.0470	4.5791	0.1044	0.1172	0.9752	0.9658
	2000	5.1661	5.9784	0.1263	0.1370	0.9707	0.9554
	1000	5.5914	6.8395	0.1447	0.1433	0.9703	0.9490

selected to visualize change information. The feature representation of the randomly selected hidden layer is mapped into a two-dimensional space by PCA, and the dimension is then extended by the corresponding output (TKN) to build a three-dimensional visualization to show the input−output mapping relationship of the soft sensor model intuitively and qualitatively. Figure 9 shows the fourth hidden layer visualization of the DNN model with 2000 labeled samples, and Figure 10 shows the fourth hidden layer visualization of the SSE-DNN model with 2000 labeled samples and 15 000 unlabeled samples. Dots in Figures 9 and 10 represent the distribution of the hidden features after dimension reduction. It can be seen in Figure 9 that the input and output of the DNN model do not show a good linear relationship in the three-dimensional space and that there are some outliers and divergence. However, when a manifold is embedded into the DNN model, the input and output of the SSE-DNN model show a good linear and smooth relationship in Figure 10, to some extent, which is useful for improving the soft sensor model. Thus, fine-tuning the SSE-DNN model improves the essential structure of the manifold on the data set by manifold embedding.

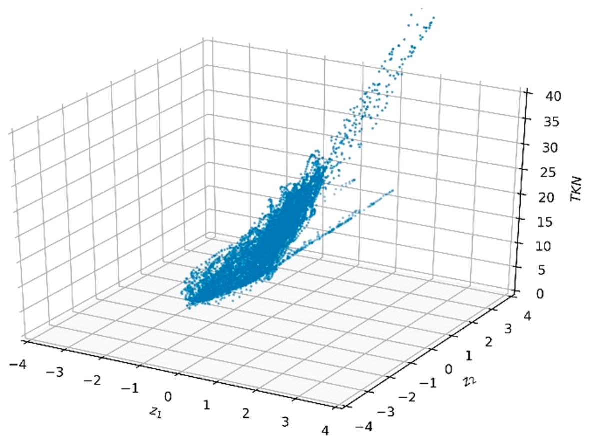


Figure 9. Visualization of the DNN model with 2000 labeled samples.

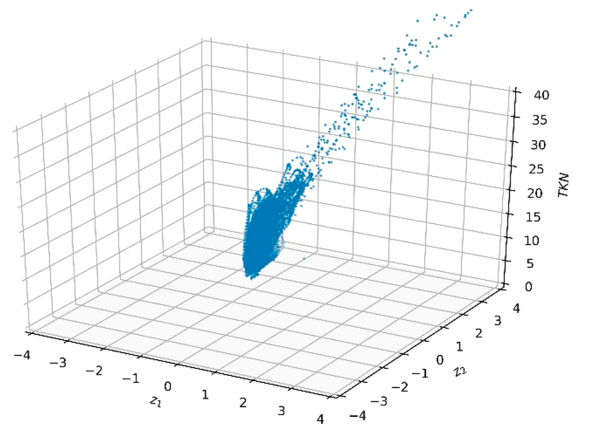


Figure 10. Visualization of the SSE-DNN model with 2000 labeled samples and 15 000 unlabeled samples.

5. CONCLUSION

This paper proposes a semisupervised soft sensor modeling method based on a deep neural network that integrates embedding manifold into a deep neural regression network. Manifold embedding aims to exploit the local neighbor relationship among industrial data and utilizes rich information among unlabeled data. The local structure of embedding manifold in SSE-DNN improves the robustness of the soft sensor to noise and outliers. The SSE-DNN can be also considered as a DNN model with multimodal models that is especially suitable for the soft sensor modeling of complex industrial processes. Soft sensor based on the SSE-DNN is applied to the estimation of TKN in a long-term complicated wastewater treatment process. The experimental results demonstrate that the SSE-DNN-based soft sensor improved the performance and generalization of DNN-based soft sensors. Manifold embedding in the SSE-DNN model successfully captures local structure such as the spikes of the long-term complicated WWTPs. The SSE-DNN model is an effective method for soft sensor modeling of complex industrial processes.

■ AUTHOR INFORMATION

Corresponding Authors

Weiwu Yan − *Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China; Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai 200240, China;* orcid.org/0000-0002-9976-3248; Email: yanwwsjtu@sjtu.edu.cn

Kundong Wang − *Department of Instrument Science and Technology, Shanghai Jiao Tong University, Shanghai 200240, China*; Email: kdwang@sjtu.edu.cn

Authors

Renchao Xu − *Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China*

Tang Di − *Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China*

Zhang Jiang − *Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China*

Complete contact information is available at:
https://pubs.acs.org/10.1021/acs.iecr.9b05087

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This research is partially funded by the China National R&D Key Research Program (2019YFB1705702) and supported by the Natural Science Foundation of China (Grant 60974119 and Grant 61533013).

■ REFERENCES

(1) Fortuna, L.; Graziani, S.; Rizzo, A.; Xibilia, M. G. *Soft Sensors for Monitoring and Control of Industrial Processes*; Springer-Verlag: London, 2007.

(2) Kadlec, P.; Gabrys, B.; Strandt, S. Data-driven soft sensors in the process industry. *Comput. Chem. Eng.* **2009**, *33* (4), 795−814.

(3) Hu, X.; Cao, D.; Egardt, B. Condition monitoring in advanced battery management systems: moving horizon estimation using a reduced electrochemical model. *IEEE/ASME Trans. Mechatronics* **2018**, *23* (1), 167−178.

(4) Grbić, R.; Slišković, D.; Kadlec, P. Adaptive soft sensor for online prediction and process monitoring based on a mixture of Gaussian process models. *Comput. Chem. Eng.* **2013**, *58* (11), 84−97.

(5) Yan, W.; Shao, H.; Wang, X. Soft sensing modeling based on support vector machine and Bayesian model selection. *Comput. Chem. Eng.* **2004**, *28* (8), 1489−1498.

(6) Bidar, B.; Sadeghi, J.; Shahraki, F.; Khalilipour, M. M. Data-driven soft sensor approach for online quality prediction using state dependent parameter models. *Chemom. Intell. Lab. Syst.* **2017**, *162*, 130−141.

(7) Yan, W.; Guo, P.; Tian, Y.; Gao, J. A Framework and Modeling Method of Data-Driven Soft Sensors Based on Semisupervised Gaussian Regression. *Ind. Eng. Chem. Res.* **2016**, *55* (27), 7394−7401.

(8) Ma, Y.; Huang, B. Bayesian learning for dynamic feature extraction with application in soft sensing. *IEEE Trans. Ind. Electron.* **2017**, *64* (9), 7171−7180.

(9) Yan, W.; Tang, D.; Lin, Y. A data-driven soft sensor modeling method based on deep learning and its application. *IEEE Trans. Ind. Electron.* **2017**, *64* (5), 4237−4245.

(10) Yuan, X.; Huang, B.; Wang, Y.; Yang, C.; Gui, W. Deep Learning-Based Feature Representation and Its Application for Soft Sensor Modeling With Variable-Wise Weighted SAE. *IEEE Trans. Ind. Electron.* **2018**, *14* (7), 3235−3243.

(11) Hu, X.; Li, S.; Yang, Y. Advanced machine learning approach for lithium-ion battery state estimation in electric vehicles. *IEEE Trans. Transp. Electrif.* **2016**, *2* (2), 140−149.

(12) Bakirov, R.; Gabrys, B.; Fay, D. Multiple adaptive mechanisms for data-driven soft sensors. *Comput. Chem. Eng.* **2017**, *96* (4), 42−54.

(13) LeCun, Y.; Bengio, Y.; Hinton, G. E. Deep learning. *Nature* **2015**, *521* (7553), 436−444.

(14) Shang, C.; Yang, F.; Huang, D.; Lyu, W. Data-driven soft sensor development based on deep learning technique. *J. Process Control* **2014**, *24* (3), 223−233.

(15) Ge, Z.; Huang, B.; Song, Z. Nonlinear semisupervised principal component regression for soft sensor modeling and its mixture form. *J. Chemom.* **2014**, *28* (11), 793−804.

(16) Zhu, X.; Goldberg, A. B. Introduction to semi-supervised learning. *Synth. Lect. Artif. Intell. Mach. Learn.* **2009**, *3* (1), 1−130.

(17) Weston, J.; Ratle, F.; Mobahi, H.; Collobert, R. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*; Montavon, G., Orr, G. B., Müller, K. R., Eds.; Lecture Notes in Computer Science 7700; Springer: 2012; pp 639−655.

(18) Haimi, H.; Mulas, M.; Corona, F.; Vahala, R. Data-derived soft-sensors for biological wastewater treatment plants: An overview. *Environ. Model. Softw.* **2013**, *47* (47), 88−107.

(19) Luttmann, R.; Bracewell, D. G.; Cornelissen, G.; Gernaey, K. V.; Glassey, J.; Hass, V. C.; Kaiser, C.; Preusse, C.; Striedner, G.; Mandenius, C. F. Soft sensors in bioprocessing: a status report and recommendations. *Biotechnol. J.* **2012**, *7* (8), 1040−1048.

(20) Fernandez de Canete, J.; Del Saz-Orozco, P.; Baratti, R.; Mulas, M.; Ruano, A.; Garcia-Cerezo, A. Soft-sensing estimation of plant effluent concentrations in a biological wastewater treatment plant using an optimal neural network. *Expert Syst. Appl.* **2016**, *63* (63), 8−19.

(21) Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems 19 (NIPS 2006)*; Schölkopf, B., Platt, J. C., Hoffman, T., Eds.; Neural Information Processing Systems Foundation, Inc.: 2007; pp 153−160.

(22) Le Roux, N.; Bengio, Y. Representational power of restricted Boltzmann machines and deep belief networks. *Neural Comput.* **2008**, *20* (6), 1631−1649.

(23) Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P. A. Extracting and composing robust features with denoising autoencoders. In *ICML '08: Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland*; Association for Computing Machinery: 2008; pp 1096−1103.

(24) Bengio, Y.; Yao, L.; Alain, G.; Vincent, P. Generalized denoising auto-encoders as generative models. *Advances in Neural Information Processing Systems 26 (NIPS 2013)*; Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. Q., Eds.; Neural Information Processing Systems Foundation, Inc.: 2013; pp 899−907.

(25) Hinton, G. E.; Deng, L.; Yu, D.; Dahl, G. E.; Mohamed, A.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T. N.; Kingsbury, B. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **2012**, *29* (6), 82−97.

(26) Li, X.; Wu, X. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*; IEEE: 2015; pp 4520−4524.

(27) Krizhevsky, A.; Sutskever, I.; Hinton, G. E.Imagenet classification with deep convolutional neural networks*Advances in Neural Information Processing Systems 25 (NIPS 2012)*; Pereira, F., Burges, C. J. C., Bottou, L., Weinberger, K. Q., Eds.; Neural Information Processing Systems Foundation, Inc.: 2012; pp 1097−1105.

(28) Hinton, G. E. Deep belief networks. *Scholarpedia* **2009**, *4* (6), 5947.

(29) Srivastava, N.; Salakhutdinov, R. R. Multimodal learning with deep boltzmann machines. *Advances in Neural Information Processing Systems 25 (NIPS 2012)*; Pereira, F., Burges, C. J. C., Bottou, L., Weinberger, K. Q., Eds.; Neural Information Processing Systems Foundation, Inc.: 2012; pp 2222−2230.

(30) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., Weinberger, K. Q., Eds.; Neural Information Processing Systems Foundation, Inc.: 2014; pp 2672−2680.

(31) Seung, H. S.; Lee, D. D. The manifold ways of perception. *Science* **2000**, *290* (5500), 2268−2269.

(32) Tenenbaum, J. B.; De Silva, V.; Langford, J. A global geometric framework for nonlinear dimensionality reduction. *Science* **2000**, *290* (5500), 2319−2323.

(33) Roweis, S. T.; Saul, L. K. Nonlinear dimensionality reduction by locally linear embedding. *Science* **2000**, *290* (5500), 2323−2326.

(34) Luo, J.; Chen, H.; Tang, Y. Analysis of Graph-Based Semi-supervised Regression. In *Proceedings of International Conference on Fuzzy Systems & Knowledge Discovery*; IEEE: 2008; pp 111−115.

(35) Belkin, M.; Niyogi, P. Semi-supervised learning on Riemannian manifolds. *Mach. Learn.* **2004**, *56* (1), 209−239.

(36) Chen, L.; Tsang, I. W.; Xu, D. Laplacian embedded regression for scalable manifold regularization. *IEEE Trans. Neural Networks Learn. Syst.* **2012**, *23* (6), 902−915.

(37) Alex, J.; Benedetti, L.; Copp, J.; Gernaey, K. V.; Jeppsson, U.; Nopens, I.; Pons, M. N.; Rieger, L.; Rosen, C.; Steyer, J. P.; Vanrolleghem, P.; Winkler, S.; IWA Taskgroup on Benchmarking of Control Strategies for WWTPs. *Benchmark Simulation Model No. 1 (BSM1)*; Department of Industrial Electrical Engineering and Automation, Lund University: 2008.

(38) Gernaey, K. V.; Vrecko, D.; Rosen, C.; Jeppsson, U. BSM1 versus BSM1_LT: Is the control strategy performance ranking maintained? In *Proceedings of the 7th International IWA Symposium on Systems Analysis and Integrated Assessment in Water Management, Washington, DC (WATERMATEX 2007)*; International Water Association: 2007.

(39) Jeppsson, U.; Pons, M.-N.; Nopens, I.; Alex, J.; Copp, J. B.; Gernaey, K. V.; Rosén, C.; Steyer, J. P.; Vanrolleghem, P. A. Benchmark simulation model no 2: general protocol and exploratory case studies. *Water Sci. Technol.* **2007**, *56* (8), 67−78.

(40) Rasmussen, C. E.; Nickisch, H. Gaussian processes for machine learning (GPML) toolbox. *J. Mach. Learn. Res.* **2010**, *11* (6), 3011−3015.

(41) Shang, C.; Huang, B.; Yang, F.; Huang, D. Probabilistic slow feature analysis-based representation learning from massive process data for soft sensor modeling. *AIChE J.* **2015**, *61* (12), 4126−4139.