Control Engineering Practice 91 (2019) 104098

Contents lists available at ScienceDirect

Control Engineering Practice

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

Nonlinear industrial soft sensor development based on semi-supervised probabilistic mixture of extreme learning machines

Weiming Shao [a], Zhiqiang Ge [a,*], Zhihuan Song [a,*], Kai Wang [b]

[a] State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China
[b] School of Automation, Central South University, Changsha 410083, China

ARTICLE INFO

Keywords:
Soft sensor
Semi-supervised learning
Extreme learning machine
Bayesian regularization
Variational Bayes expectation–maximization

ABSTRACT

Soft sensors play an important role in process industries for monitoring and control of key quality variables, and calibration of analyzers. Owing to the merits of fast learning speed and good generalization performance, extreme learning machines (ELMs) have been widely accepted to develop soft sensor models for nonlinear industrial processes. However, there still exist some challenges in developing high-accuracy ELM-based soft sensors. Specifically, ELMs with shallow networks seem to have inadequate representation capabilities for complex nonlinearities, while ELMs with deep networks have difficulties in determining the number of hidden layers and hidden nodes for each layer which readily results in overfitting. In addition, in soft sensor applications, labeled samples are usually limited due to technical or economical reasons, which adds obstacles to model training. To deal with these issues, we propose a semi-supervised probabilistic mixture of ELMs (referred to as the 'S^{2}PMELMs'). In the S^{2}PMELMs, localized ELMs are trained and combined, which are completed in a unified probabilistic way such that process nonlinearities and uncertainties can be accommodated. Moreover, based on the variational Bayes expectation–maximization algorithm, we develop a training algorithm for the S^{2}PMELMs, where unlabeled samples are able to be exploited and the regularization parameter for each ELM can be adaptively determined. The performance of the S^{2}PMELMs is evaluated through two real-world industrial processes, and the results demonstrate the advantages of the proposed method in contrast with several state-of-the-art relevant soft sensing approaches.

1. Introduction

In industrial plants large amounts of physical sensors are installed to deliver data for monitoring and automatic control of process variables. However, for a category of product quality-related variables (such as concentration, melt index, octane number, etc.) due to technical or economical limitations, there are no effective sensors to provide real-time measurements, which could deteriorate control performance, increase production cost, and raise obstacle in green and safe production (Shao, Ge and Song, 2019). With the advancements of data acquisition technologies and data analytics algorithms, data-driven models have established themselves as effective tools in dealing with this issue by online estimating those hard-to-measure variables (Ge, 2018; Ge, Song, Ding, & Huang, 2017; Jin, Wang, Huang, & Forbes, 2012). These models are often called 'soft sensors', taking secondary variables (i.e., easy-to-measure variables) as inputs and outputting delay-free estimations of quality variables. As collecting samples of quality variables is usually time-consuming or expensive due to off-line laboratory analysis or high investment cost, labeled samples are usually scarce. In contrast, gathering samples of secondary variables is much easier and cheaper, which implies that unlabeled samples are abundant. Therefore, developing soft sensor is essentially a semi-supervised task (Yan, Guo, Tian, & Gao, 2016).

Because of ever increasing demands from profits and environmental requirements, modern industrial processes are growing more and more complicated (Shao, Yao, Ge and Song, 2019). For example, in the penicillin fermentation process, the microorganisms have multiple growth phases, and the process mechanisms vary from phase to phase (Jin, Chen, Wang, Yang, & Wu, 2016). In addition, many industrial processes work with multiple conditions, due to demands of multiple product grades as well as variations in loads and feedstocks (Shao, Ge, & Song, 2018; Souza & Araújo, 2014). Therefore, the vast majority of industrial plants exhibit nonlinear behaviors, leading the quality variables to depend on the explanatory variables in nonlinear ways. For those nonlinear processes, traditional linear soft sensing algorithms such as principal component analysis (PCA), partial least squares (PLS) and independent component analysis (ICA) are inadequate. Artificial neural networks (ANNs) are one commonly adopted category of approaches to dealing with process nonlinearities. Particularly, deep learning-based ANNs using multiple hidden layers have shown strong

* Corresponding authors.
E-mail addresses: gezhiqiang@zju.edu.cn (Z. Ge), songzhihuan@zju.edu.cn (Z. Song).

https://doi.org/10.1016/j.conengprac.2019.07.016
Received 30 March 2019; Received in revised form 23 June 2019; Accepted 23 July 2019
Available online xxxx
0967-0661/

representation abilities and have found applications in many domains of computer science including speech recognition, image processing, natural language processing, computer vision, etc (Lecun, Bengio, & Hinton, 2015; Schmidhuber, 2015). Recently, for soft sensor modeling one can also find applications of deep learning-based ANNs, for example deep belief network (DBN) (Liu, Yang, Gao, & Yao, 2018; Shang, Yang, Huang, & Lyu, 2014), stacked auto-encoders (SAEs) (Yan, Tang, & Lin, 2017; Yuan, Huang, Wang, Yang, & Gui, 2018), deep Boltzman machine (DBM) (Graziani & Xibilia, 2017), multi-layer perception (MLP) (Gopakumar, Tiwari, & Rahman, 2018), etc.

While these ANNs are trained by the gradient-based back propagation (BP) algorithm, extreme learning machine (ELM), a newly emerged single-hidden layer feedforward network (SLFN) proposed by Huang, Zhu and Siew (2006), can avoid the BP algorithm or quadratic programming (required by the SVM, i.e., support vector machine). The stopping criterion and learning rate required by iterative learning are unnecessary for the ELM, neither (He, Geng, & Zhu, 2015). Meanwhile, the ELM has been proved to possess universal approximation capability for continuous functions, and studies have shown that the ELM could achieve better or comparable performance in contrast with SLFN-based methods such as the BP-based SLFN and SVM (Huang, Chen and Siew, 2006; Huang, Zhou, Ding, & Zhang, 2012; Liu, Gao, & Li, 2012). Therefore, ELM is framed for its extremely fast learning speed and good generalization performance. Since it was proposed, ELM has drawn increasing attentions for various learning tasks, such as regression, classification, data clustering, feature extraction, representational learning, etc (Huang, Huang, Song, & You, 2015), and has been extended to a variety of advanced variants including regularized ELM (Martínes Martínes, Escandell-Montero, Soria-Olivas, et al., 2011), Bayesian ELM (Chen, Yang, Wang, & Park, 2016; Soria-Olivas, Góez-Sanchis, Martín, Vila-Francés, et al., 2011), sparse ELM (Bai, Huang, Wang, et al., 2014), robust ELM (Zhang & Luo, 2015), semi-supervised and unsupervised ELMs (Huang, Song, Gupta, & Wu, 2014; Yi, Qiao, Zhou, et al., 2018), etc. Equipping the ELM with deep networks has also been realized, which is named 'hierarchical ELM (HELM)' (Kasun, Zhou, Huang, & Vong, 2013; Tang, Deng, & Huang, 2016) or 'deep ELM (DELM)' (Sun, Zhang, Zhang, & Hu, 2017). The HELM/DELM functions resembling the SAE (Hinton & Salakhutdinov, 2006), but could be significantly faster than the SAE trained by the BP algorithm in some applications (Tang et al., 2016).

Owing to these merits, ELM and its extensions have been widely adopted for soft sensing of nonlinear industrial processes. For example, Ghazvinei et al. explored the ELM in predicting the sugarcane growth for sustainable sugarcane production, which is the first time in this realm, and demonstrated that the ELM performed better than the traditional ANN (Ghazvinei et al., 2018). In addition, aiming at facilitating the selection of weight parameters that connect the hidden nodes and input nodes, Zhang et al. and Geng et al. proposed modified gravitational search algorithm (Zhang, Liu, & Zhang, 2016) and self-organizing-based ELMs, respectively (Geng, Dong, Chen, & Han, 2017). Hybrid models that integrate ELM with other methods have also been developed for soft sensor development. For instance, Ardabili et al. proposed a hybrid of ELM and response surface methodology (ELM–RSM) for prediction of ethyl ester and methyl ester, which is further used for optimization of the production yield. Their studies have verified the effectiveness of the hybrid model in enhancing the production yield (Ardabili et al., 2018). Also, taking the process dynamics into account, He et al. developed a dynamic ELM based on the PLS (He, Xu and Zhu, 2016). Moreover, considering the semi-supervised nature of soft sensing, and to take advantage of deep learning, Yao and Ge proposed a semi-supervised DELM-based soft sensing method by integrating DELM with manifold regularization (Yao & Ge, 2018). In He, Geng and Zhu (2016) and Peng, Zhou, Zhang, and Zheng (2017), soft sensor models based on multiple ELMs were constructed and aggregated using various ensemble learning strategies such as bagging and boosting.

Notwithstanding the improvements achieved by the above mentioned extended ELM-based soft sensing approaches, there are still some limitations associated with them. Firstly, the representation ability of a single ELM with shallow network seems to be inadequate when modeling complicated or strong nonlinearities. The ensemble ELMs could not substantially overcome such drawback as each member ELM is basically still constructed based on samples covering the entire operating area. Despite in theory the DELM has powerful representation ability, practically we found it could be nontrivial to determine proper network structure (i.e., the number of hidden layers and that of neurons for each layer) and other model parameters especially with insufficient labeled samples in the soft sensor application (which easily leads DELMs to suffer from overfitting). Secondly, to deal with the scarcity of labeled samples and to exploit abundant unlabeled samples, the most commonly used strategy for semi-supervised ELMs might be the manifold regularization, where the graph Laplacian matrix has to be constructed. Nevertheless, the graph Laplacian is lack of extrapolating ability with insufficient labeled samples and tends to cause biased solution (Kim, Steinke, & Hein, 2009), and these semi-supervised ELMs (Huang et al., 2014; Yao & Ge, 2018; Yi et al., 2018) could only exploit limited unlabeled data because the graph Laplacian matrix is memory-demanding and time-consuming with big data size.

To deal with these issues, in this paper, we propose a semi-supervised probabilistic mixture of ELMs (referred to as the 'S^2PMELMs'). Different from the global ELMs, the S^2PMELMs adopts the philosophy of 'divide and conquer' and constructs localized ELMs. By doing so, the complicated nonlinearities are modeled by multiple local ELMs, while the problems of using deep networks can be avoided. In addition, the S^2PMELMs employs the generative way under the Bayesian inference framework to realize semi-supervised learning, which does not needs manifold regularization and meanwhile could accommodate process uncertainties. Our novel contributions can be summarized as threefold: (1) a semi-supervised model structure that combines localized ELMs in a probabilistic way is proposed; (2) a variational Bayes expectation–maximization (VBEM)-based training algorithm is developed, which can complete the localization and train localized ELMs in a unified framework, and is also able to adaptively determine the important regularization parameter for each ELM; (3) performance of the proposed method is thoroughly evaluated using two real-world industrial processes, which could be informative and instructive for practitioners.

The remaining parts of this paper are outlined as follows. In Section 2 the basic ELM is briefly reviewed. Section 3 presents the proposed S^2PMELMs in detail including the model structure and training of S^2PMELMs and how to develop soft sensor using the S^2PMELMs. Section 4 evaluates the performance of the S^2PMELMs as well as those of several benchmarking methods using one industrial methanation process and one industrial primary reformer process. Finally, this paper is concluded in Section 5.

2. Extreme learning machine

By taking a single-output process as an example, ELM is illustrated in Fig. 1, which is composed of one input layer representing the d-dimensional input vector $\boldsymbol{x} \in \mathbb{R}^d$, one output layer representing the scalar output $y \in \mathbb{R}$, and one hidden layer connecting the input and output layers. From the input layer to the hidden layer, the original variable space is mapped into the m-dimensional feature space where $\boldsymbol{x}$ is transformed as $\boldsymbol{\psi}(\boldsymbol{x}) \in \mathbb{R}^m$

$$\boldsymbol{\psi}(\boldsymbol{x}) = \left[g(\widetilde{\boldsymbol{x}}^{\mathrm{T}}\boldsymbol{w}_1), g(\widetilde{\boldsymbol{x}}^{\mathrm{T}}\boldsymbol{w}_2), \ldots, g(\widetilde{\boldsymbol{x}}^{\mathrm{T}}\boldsymbol{w}_m)\right]^{\mathrm{T}} \tag{1}$$

where $\widetilde{\boldsymbol{x}} = [\boldsymbol{x}^{\mathrm{T}}, 1]^{\mathrm{T}}$, $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_m] \in \mathbb{R}^{(d+1)\times m}$ means the weight matrix which is randomly generated (Huang, Zhu et al., 2006), and $g(\cdot)$ means the activation function.

A popular activation function is the *sigmoid* function defined as

$$g(x) = \frac{1}{1+\exp(-x)} \tag{2}$$

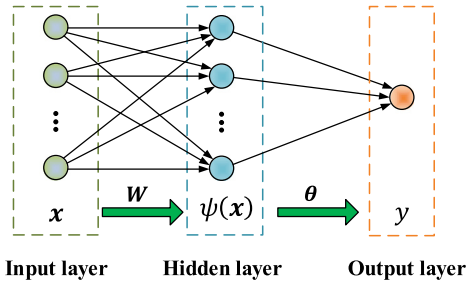


Fig. 1. Illustration of ELM.

In addition, $\theta \in \mathbb{R}^{m+1}$ means the regression coefficients obtained by solving the following optimization problem (Huang et al., 2014)

$$\min \quad \frac{C}{2}\theta^{\mathrm{T}}\theta + \frac{1}{2}\sum_{l=1}^{L}\left(y_l - \widetilde{\boldsymbol{\psi}}^{\mathrm{T}}(\boldsymbol{x}_l)\theta\right)^2, \tag{3}$$

where $\boldsymbol{x}_l$ and y_l denote the sampling values of the input and output variables of the lth labeled sample, $\widetilde{\boldsymbol{\psi}}(\boldsymbol{x}_l) = [\boldsymbol{\psi}^{\mathrm{T}}(\boldsymbol{x}_l), 1]^{\mathrm{T}}$, L means the number of labeled samples, and C represents the regularization/penalty parameter which is quite important for alleviating overfitting.

Solving (3) leads to

$$\theta = \begin{cases} \left(\widetilde{\boldsymbol{\Psi}}_L^{\mathrm{T}}\widetilde{\boldsymbol{\Psi}}_L + C\mathbf{I}_m\right)^{-1}\widetilde{\boldsymbol{\Psi}}_L^{\mathrm{T}}\boldsymbol{Y}_L, & m < L \\ \widetilde{\boldsymbol{\Psi}}_L^{\mathrm{T}}\left(\widetilde{\boldsymbol{\Psi}}_L\widetilde{\boldsymbol{\Psi}}_L^{\mathrm{T}} + C\mathbf{I}_L\right)^{-1}\boldsymbol{Y}_L, & \text{otherwise} \end{cases} \tag{4}$$

where $\mathbf{I}_L$ and $\mathbf{I}_m$ mean the unit matrices with order of L and m, respectively, $\boldsymbol{\Psi}_L = \left[\boldsymbol{\psi}(\boldsymbol{x}_1), \boldsymbol{\psi}(\boldsymbol{x}_2), \ldots, \boldsymbol{\psi}(\boldsymbol{x}_L)\right]^{\mathrm{T}}$, $\widetilde{\boldsymbol{\Psi}}_L = [\boldsymbol{\Psi}_L, \mathbf{1}_L]$, $\mathbf{1}_L = [1, 1, \ldots, 1]^{\mathrm{T}} \in \mathbb{R}^L$, and $\boldsymbol{Y}_L = \left[y_1, y_2, \ldots, y_L\right]^{\mathrm{T}}$. Note that if $m < L$ calculating $\left(\widetilde{\boldsymbol{\Psi}}_L^{\mathrm{T}}\widetilde{\boldsymbol{\Psi}}_L + C\mathbf{I}_m\right)^{-1}$ is cheaper; on the contrary, if $L < m$ calculating $\left(\widetilde{\boldsymbol{\Psi}}_L\widetilde{\boldsymbol{\Psi}}_L^{\mathrm{T}} + C\mathbf{I}_L\right)^{-1}$ is cheaper. Therefore, using which way to calculate θ depends on the values of L and m. In addition, by imposing a prior distribution on θ, the regularization can be automatically determined. Different covariances of the prior can lead to different results and performance. A simple covariance is the unit matrix multiplied by a positive value which is equivalent to the ridge regression, however it should be mentioned that the harmonic analysis is able to design more appropriate and more complex covariances if necessary (Zorzi & Chiuso, 2018).

Given the sample $\boldsymbol{x}^*$ of input variables, the ELM calculates the estimation ($\hat{y}^*$) of the corresponding true output y^* as

$$\hat{y}^* = \widetilde{\boldsymbol{\psi}}^{\mathrm{T}}(\boldsymbol{x}^*)\theta, \tag{5}$$

where $\widetilde{\boldsymbol{\psi}}(\boldsymbol{x}^*) = [\boldsymbol{\psi}^{\mathrm{T}}(\boldsymbol{x}^*), 1]^{\mathrm{T}}$.

3. Semi-supervised probabilistic mixture of extreme learning machines

The S^{2}PMELMs adopts the philosophy of 'divide and conquer', which constructs a group of localized ELMs. Each of these ELMs serves as an expert in a specific local area where all training samples are endowed with different weights representing the importances within the corresponding local area. Based on samples with different weights within each local area, a weighted and regularized local ELM is constructed. Note that for each local ELM, the regularization parameter (like 'C' in (4)) plays an important role in dealing with overfitting. However, manually determine proper regularization parameters (which is a part of training the S^{2}PMELMs) for each of the ELMs would be prohibitively difficult. In the S^{2}PMELMs, automatic regularization is realized by randomizing the regression coefficients that connect the hidden layer and the output layer. In addition, the localization

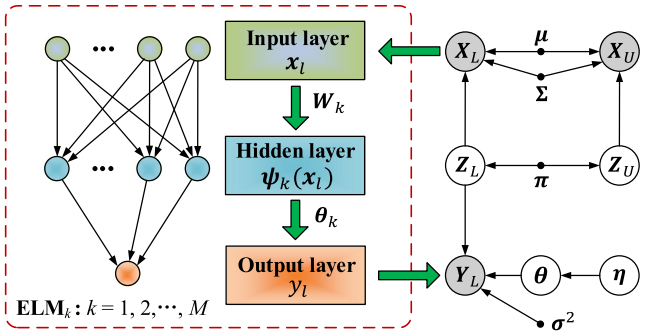


Fig. 2. Illustration of the S^{2}PMELMs using probabilistic graphical model.

(i.e., how to determine the weights of samples in each local area) and training of localized ELMs are completed in a unified framework, such that both the input and output information of labeled and unlabeled samples can be more effectively exploited. How to structure and train the S^{2}PMELMs, as well as how to develop soft sensor using the S^{2}PMELMs are detailed in the rest parts of this section.

3.1. Formulation of the S^{2}PMELMs

The S^{2}PMELMs employing M ELMs to localize the entire operating area is realized in the probabilistic way as illustrated in Fig. 2. For simplicity, we assume the M ELMs share the same number m of hidden neurons, thus the kth ELM is characterized by random weight matrix $\boldsymbol{W}_k = [\boldsymbol{w}_1^k, \boldsymbol{w}_2^k, \ldots, \boldsymbol{w}_m^k] \in \mathbb{R}^{(d+1)\times m}$ and regression coefficients $\theta_k \in \mathbb{R}^{m+1}$. For the kth ELM, the hidden layer maps the input vector $\boldsymbol{x}_l$ ($1 \le l \le L$) of the lth sample (labeled) as $\boldsymbol{\psi}_k(\boldsymbol{x}_l)$

$$\boldsymbol{\psi}_k(\boldsymbol{x}_l) = \left[g(\widetilde{\boldsymbol{x}}_l^{\mathrm{T}}\boldsymbol{w}_1^k), g(\widetilde{\boldsymbol{x}}_l^{\mathrm{T}}\boldsymbol{w}_2^k), \ldots, g(\widetilde{\boldsymbol{x}}_l^{\mathrm{T}}\boldsymbol{w}_m^k)\right]^{\mathrm{T}} \tag{6}$$

where $\widetilde{\boldsymbol{x}}_l = [\boldsymbol{x}_l^{\mathrm{T}}, 1]^{\mathrm{T}}$.

Other newly appeared notations in Fig. 2 are explained as follows:

- $\boldsymbol{X}_L = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_L]^{\mathrm{T}}$ means the input data matrix for labeled samples.
- $\boldsymbol{Z}_L = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_L]^{\mathrm{T}}$, where z_l represents the latent variable associated with labeled sample $\{\boldsymbol{x}_l, y_l\}$.
- $\boldsymbol{X}_U = [\boldsymbol{x}_{L+1}, \boldsymbol{x}_{L+2}, \ldots, \boldsymbol{x}_{L+U}]^{\mathrm{T}}$ means the input data matrix for unlabeled samples, where $\boldsymbol{x}_u$ is the uth ($L+1 \le u \le L+U$) sample (unlabeled) and U is the size of unlabeled dataset.
- $\boldsymbol{Z}_U = [\boldsymbol{z}_{L+1}, \boldsymbol{z}_{L+2}, \ldots, \boldsymbol{z}_{L+U}]^{\mathrm{T}}$, where z_u represents the latent variable associated with unlabeled sample $\boldsymbol{x}_u$.
- $\boldsymbol{\pi} = [\pi_1, \pi_2, \ldots, \pi_M]$, where π_k represents the prior probability for the kth ($1 \le k \le M$) ELM.
- $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_M]$, where $\boldsymbol{\mu}_k$ represents the mean of Gaussian distribution for the kth ELM.
- $\boldsymbol{\Sigma} = [\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \ldots, \boldsymbol{\Sigma}_M]$, where $\boldsymbol{\Sigma}_k$ represents the covariance of Gaussian distribution for the kth ELM.
- $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_M]$.
- $\boldsymbol{\eta} = [\eta_1, \eta_2, \ldots, \eta_M]$, where η_k represents the precision parameter for $\boldsymbol{\theta}_k$.
- $\boldsymbol{\sigma}^2 = [\sigma_1^2, \sigma_2^2, \ldots, \sigma_M^2]$, where σ_k^2 represents the variance of measurement noise for the kth ELM.

In the S^{2}PMELMs, we endow each observed sample with a discrete latent variable to indicate which localized ELM this observed sample is assigned to. Specifically, $\boldsymbol{z}_l = [z_{l1}, z_{l2}, \ldots, z_{lM}]^{\mathrm{T}}$, where $z_{lk} \in \{0, 1\}$ and $\sum_{k=1}^{M} z_{lk} = 1$. And $z_{lk} = 1$ means $\{\boldsymbol{x}_l, y_l\}$ is assigned to the kth ELM. Thereby, according to the physical meaning of $\boldsymbol{\pi}$, we have $p(z_{lk} = 1) = \pi_k$, or

$$p(\boldsymbol{z}_l) = \prod_{k=1}^{M}\pi_k^{z_{lk}}, \tag{7}$$

where $p(\cdot)$ means the probability density function (p.d.f.) of random variable(s). Similarly, a latent variable $\boldsymbol{z}_u = [z_{u1}, z_{u2}, \ldots, z_{um}]^{\mathrm{T}}$ is assigned to each unlabeled sample $\boldsymbol{x}_u$ $(L+1 \le u \le L+U)$, having the following distribution

$$p(\boldsymbol{z}_u) = \prod_{k=1}^{M} \pi_k^{z_{uk}}, \tag{8}$$

We assume the explanatory variables obey a mixture of Gaussian distributions, which is the basis of performing localization. Consequently, for each localized ELM, the process variables are considered to obey one Gaussian distribution. In other words, we have $p(\boldsymbol{x}_l|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, z_{lk} = 1) = \mathcal{N}(\boldsymbol{x}_l|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ and $p(\boldsymbol{x}_u|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, z_{uk} = 1) = \mathcal{N}(\boldsymbol{x}_u|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, or

$$p(\boldsymbol{x}_l|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{z}_l) = \prod_{k=1}^{M} \mathcal{N}(\boldsymbol{x}_l|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{lk}}, \tag{9}$$

$$p(\boldsymbol{x}_u|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{z}_u) = \prod_{k=1}^{M} \mathcal{N}(\boldsymbol{x}_u|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{uk}}, \tag{10}$$

where $\mathcal{N}(\cdot)$ represents the p.d.f. of the Gaussian distribution.

In addition, for the kth ELM, the measured output can be represented as

$$y_l = \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{\theta}_k + \zeta_k, \tag{11}$$

where ζ_k means the Gaussian measurement noise with zero mean and variance σ_k^2, $\widetilde{\boldsymbol{\psi}}_k(\boldsymbol{x}_l) = [\boldsymbol{\psi}_k^{\mathrm{T}}(\boldsymbol{x}_l), 1]^{\mathrm{T}}$.

Based on (11), the conditional p.d.f. of y_l can be obtained as

$$p(y_l|\boldsymbol{x}_l, \boldsymbol{\theta}_k, \sigma_k^2, z_{lk} = 1) = \mathcal{N}(y_l|\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{\theta}_k, \sigma_k^2), \tag{12}$$

or be re-formulated as

$$p(y_l|\boldsymbol{x}_l, \boldsymbol{\theta}_k, \sigma_k^2, \boldsymbol{z}_l) = \prod_{k=1}^{M} \mathcal{N}\left(y_l\middle|\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{\theta}_k, \sigma_k^2\right)^{z_{lk}}. \tag{13}$$

Moreover, $\boldsymbol{\theta}_k$ is treated as Gaussian random vector and its prior distribution is defined as

$$p(\boldsymbol{\theta}_k|\eta_k) = \mathcal{N}\left(\boldsymbol{\theta}_k|\boldsymbol{0}, \eta_k^{-1}\mathbf{I}_{m+1}\right), \tag{14}$$

where η_k is the precision parameter for $\boldsymbol{\theta}_k$ for which we select the conjugate prior

$$p(\eta_k) = \mathcal{G}\left(\eta_k|\lambda_0^1, \lambda_0^2\right), \tag{15}$$

where $\mathcal{G}(\cdot)$ represents the p.d.f. of the Gamma distribution, and λ_0^1 and λ_0^2 represent the shape parameter and scale parameter of the Gamma distribution, respectively.

It is worth pointing out that by using (14) the Bayesian regularization can be imposed on each $\boldsymbol{\theta}_k$, which is extremely helpful for anti-overfitting; while by using (15) the regularization parameters for each of the ELMs can be adaptively determined, otherwise it could be prohibitively difficult to manually configure these regularization parameters.

3.2. Training of the S²PMELMs

According to Fig. 2, the aim of training the S²PMELMs is to obtain the maximum likelihood estimates for parameters $\boldsymbol{\Omega} = \{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2\}$, and the posterior distributions over $\boldsymbol{\Upsilon} = \{\boldsymbol{Z}_L, \boldsymbol{Z}_U, \boldsymbol{\theta}, \boldsymbol{\eta}\}$ which are denoted as $p^*(\boldsymbol{Z}_L)$, $p^*(\boldsymbol{Z}_U)$, $p^*(\boldsymbol{\theta})$ and $p^*(\boldsymbol{\eta})$, respectively. To this end, we maximize the evidence lower bound $\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})$ based on the variational inference framework, where $\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})$ is defined as Bishop (2006)

$$\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega}) = \left\langle \ln p\left(\mathcal{D}, \boldsymbol{\Upsilon}\middle|\boldsymbol{\Omega}\right)\right\rangle - \left\langle \ln p^*\left(\boldsymbol{\Upsilon}\middle|\boldsymbol{\Omega}\right)\right\rangle, \tag{16}$$

where $\mathcal{D} = \{\boldsymbol{X}_L, \boldsymbol{Y}_L, \boldsymbol{X}_U\}$ represents the labeled and unlabeled training datasets, $\langle\cdot\rangle$ means the operation of calculating the expectation, $p\left(\mathcal{D}, \boldsymbol{\Upsilon}\middle|\boldsymbol{\Omega}\right)$ means the joint distribution over $\mathcal{D}$ and $\boldsymbol{\Upsilon}$ with fixed $\boldsymbol{\Omega}$, and $p^*\left(\boldsymbol{\Upsilon}\middle|\boldsymbol{\Omega}\right)$ represents the posterior distribution over $\boldsymbol{\Upsilon}$ (which can be any form of probability distribution) with fixed $\boldsymbol{\Omega}$.

For simplicity we consider that $p^*(\boldsymbol{\theta}_k, \eta_k)$ approximately factorizes as $p^*(\boldsymbol{\theta}_k, \eta_k) = p^*(\boldsymbol{\theta}_k)p^*(\eta_k)$, which leads to

$$p^*(\boldsymbol{\Upsilon}|\boldsymbol{\Omega}) = p^*(\boldsymbol{Z}_L)p^*(\boldsymbol{Z}_U)\prod_{k=1}^{M} p^*(\boldsymbol{\theta}_k)\prod_{k=1}^{M} p^*(\eta_k). \tag{17}$$

Based on (17), and by using the general rule for variational inference (Bishop, 2006), $p^*(\boldsymbol{Z}_L)$ can be obtained as

$$\begin{aligned}\ln p^*(\boldsymbol{Z}_L) &= \left\langle \ln p\left(\boldsymbol{Y}_L\middle|\boldsymbol{X}_L, \boldsymbol{Z}_L, \boldsymbol{\Theta}, \boldsymbol{\sigma}^2\right)\right\rangle \\ &\quad + \ln p\left(\boldsymbol{X}_L\middle|\boldsymbol{Z}_L, \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) + \ln p\left(\boldsymbol{Z}_L\middle|\boldsymbol{\pi}\right) + \mathscr{C}(\boldsymbol{Z}_L) \\ &= \sum_{l=1}^{L}\sum_{k=1}^{M} z_{lk}\ln\xi_{lk} + \mathscr{C}(\boldsymbol{Z}_L),\end{aligned} \tag{18}$$

where $\mathscr{C}(\cdot)$ means constant terms that are independent of corresponding variables, and we have assumed that the training samples are independently distributed and have defined

$$\ln\xi_{lk} = \left\langle \ln\mathcal{N}\left(y_l\middle|\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{\theta}_k, \sigma_k^2\right)\right\rangle + \ln\pi_k + \ln\mathcal{N}\left(\boldsymbol{x}_l\middle|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right). \tag{19}$$

It is found from (18) that the posterior distribution over $\boldsymbol{Z}_L$ is the multinomial distribution, that is,

$$p^*(\boldsymbol{Z}_L) \propto \prod_{l=1}^{L}\prod_{k=1}^{M} \xi_{lk}^{z_{lk}} = \prod_{l=1}^{L}\prod_{k=1}^{M} \beta_{lk}^{z_{lk}}, \tag{20}$$

where $\beta_{lk} = \xi_{lk} \Big/ \sum_{k=1}^{M}\xi_{lk}$.

Similarly, we have

$$\begin{aligned}\ln p^*(\boldsymbol{Z}_U) &= \ln p\left(\boldsymbol{X}_U\middle|\boldsymbol{Z}_U, \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) + \ln p\left(\boldsymbol{Z}_L\middle|\boldsymbol{\pi}\right) + \mathscr{C}(\boldsymbol{Z}_U) \\ &= \sum_{u=L+1}^{L+U}\sum_{k=1}^{M} z_{uk}\ln\xi_{uk} + \mathscr{C}(\boldsymbol{Z}_U),\end{aligned} \tag{21}$$

where

$$\ln\xi_{uk} = \ln\mathcal{N}\left(\boldsymbol{x}_u\middle|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) + \ln\pi_k. \tag{22}$$

Thereby, we can obtain the posterior distribution over $\boldsymbol{Z}_u$ as

$$p^*(\boldsymbol{Z}_U) \propto \prod_{u=L+1}^{L+U}\prod_{k=1}^{M} \xi_{uk}^{z_{uk}} = \prod_{l=L+1}^{L+U}\prod_{k=1}^{M} \beta_{uk}^{z_{uk}}, \tag{23}$$

where $\beta_{uk} = \xi_{uk} \Big/ \sum_{k=1}^{M}\xi_{uk}$.

As z_{lk} and z_{uk} are all binary, we have

$$\begin{aligned}\langle z_{lk}\rangle &= p^*(z_{lk} = 1) = \beta_{lk}, \\ \langle z_{uk}\rangle &= p^*(z_{uk} = 1) = \beta_{uk}.\end{aligned} \tag{24}$$

The posterior distribution over $\boldsymbol{\theta}_k$ can be calculated as

$$\begin{aligned}\ln p^*(\boldsymbol{\theta}_k) &= \left\langle \ln p\left(\boldsymbol{Y}_L|\boldsymbol{X}_L, \boldsymbol{Z}_L, \boldsymbol{\theta}_k, \sigma_k^2\right)\right\rangle + \left\langle \ln p(\boldsymbol{\theta}_k|\eta_k)\right\rangle + \mathscr{C}(\boldsymbol{\theta}_k) \\ &= -\frac{1}{2}\boldsymbol{\theta}_k^{\mathrm{T}}\left(\frac{1}{\sigma_k^2}\widetilde{\boldsymbol{\Psi}}_L^k\boldsymbol{\Lambda}_k\left(\widetilde{\boldsymbol{\Psi}}_L^k\right)^{\mathrm{T}} + \langle\eta_k\rangle\mathbf{I}_{m+1}\right)\boldsymbol{\theta}_k \\ &\quad + \frac{1}{\sigma_k^2}\boldsymbol{Y}_L^{\mathrm{T}}\boldsymbol{\Lambda}_k\left(\widetilde{\boldsymbol{\Psi}}_L^k\right)^{\mathrm{T}}\boldsymbol{\theta}_k + \mathscr{C}(\boldsymbol{\theta}_k),\end{aligned} \tag{25}$$

where $\boldsymbol{\Psi}_L^k = \left[\boldsymbol{\psi}_k(\boldsymbol{x}_1), \boldsymbol{\psi}_k(\boldsymbol{x}_2), \ldots, \boldsymbol{\psi}_k(\boldsymbol{x}_L)\right]^{\mathrm{T}}$, $\widetilde{\boldsymbol{\Psi}}_L^k = [\boldsymbol{\Psi}_L^k, \mathbf{1}_L]$, and $\boldsymbol{\Lambda}_k = diag\left(\beta_{1k}, \beta_{2k}, \ldots, \beta_{Lk}\right)$.

As can be seen from (25), $p^*(\boldsymbol{\theta}_k) = \mathcal{N}\left(\boldsymbol{\theta}_k\middle|\boldsymbol{g}_k, \boldsymbol{G}_k\right)$, where

$$\boldsymbol{G}_k = \left(\frac{1}{\sigma_k^2}\widetilde{\boldsymbol{\Psi}}_L^k\boldsymbol{\Lambda}_k\left(\widetilde{\boldsymbol{\Psi}}_L^k\right)^{\mathrm{T}} + \langle\eta_k\rangle\mathbf{I}_{m+1}\right)^{-1}, \tag{26}$$

$$\boldsymbol{g}_k = \frac{1}{\sigma_k^2}\boldsymbol{G}_k\left(\widetilde{\boldsymbol{\Psi}}_L^k\right)^{\mathrm{T}}\boldsymbol{\Lambda}_k\boldsymbol{Y}_L. \tag{27}$$

The posterior distribution over η_k can be calculated as

$$\begin{aligned}\ln p^*(\eta_k) &= \left\langle \ln \mathcal{N}\left(\boldsymbol{\theta}_k \middle| \mathbf{0}, \eta_k^{-1}\mathbf{I}_{m+1}\right)\right\rangle + \ln \mathcal{G}\left(\eta_k \middle| \lambda_0^1, \lambda_0^2\right) + \mathscr{C}\left(\eta_k\right) \\ &= \lambda_k^1 \ln \eta_k - \lambda_k^2 \eta_k + \mathscr{C}\left(\eta_k\right),\end{aligned} \tag{28}$$

where

$$\lambda_k^1 = \lambda_0^1 + \frac{m+1}{2}, \tag{29}$$

$$\lambda_k^2 = \lambda_0^2 + \frac{1}{2}\left(\boldsymbol{g}_k^{\mathrm{T}}\boldsymbol{g}_k + \mathrm{Tr}\left(\boldsymbol{G}_k\right)\right), \tag{30}$$

where $\mathrm{Tr}(\cdot)$ means the trace of a square matrix. It can be seen from (30) that $p^*\left(\eta_k\right) = \mathcal{G}\left(\eta_k \middle| \lambda_k^1, \lambda_k^2\right)$.

The maximum likelihood estimates for parameters $\boldsymbol{\Omega} = \{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\sigma}^2\}$ can be obtained by setting the derivatives of the lower bound $\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})$ with respect to each of them to 0.

Specifically, setting $\partial\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})/\partial\pi_k = 0$ and taking the constraint $\sum_{k=1}^{M}\pi_k = 1$ into consideration, we have

$$\pi_k = \frac{\sum_{l=1}^{L}\beta_{lk} + \sum_{u=L+1}^{L+U}\beta_{uk}}{L+U}. \tag{31}$$

Setting $\partial\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})/\partial\boldsymbol{\mu}_k = \mathbf{0}$ leads to

$$\boldsymbol{\mu}_k = \frac{\sum_{l=1}^{L}\beta_{lk}\boldsymbol{x}_l + \sum_{u=L+1}^{L+U}\beta_{uk}\boldsymbol{x}_u}{\sum_{l=1}^{L}\beta_{lk} + \sum_{u=L+1}^{L+U}\beta_{uk}}. \tag{32}$$

Setting $\partial\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})/\partial\boldsymbol{\Sigma}_k^{-1} = \mathbf{0}$ leads to

$$\boldsymbol{\Sigma}_k = \frac{\sum_{l=1}^{L}\beta_{lk}\overline{\boldsymbol{x}}_l^k\left(\overline{\boldsymbol{x}}_l^k\right)^{\mathrm{T}} + \sum_{u=L+1}^{L+U}\beta_{uk}\overline{\boldsymbol{x}}_u^k\left(\overline{\boldsymbol{x}}_u^k\right)^{\mathrm{T}}}{\sum_{l=1}^{L}\beta_{lk} + \sum_{u=L+1}^{L+U}\beta_{uk}}, \tag{33}$$

where $\overline{\boldsymbol{x}}_l^k = \boldsymbol{x}_l - \boldsymbol{\mu}_k$ and $\overline{\boldsymbol{x}}_u^k = \boldsymbol{x}_u - \boldsymbol{\mu}_k$.

Setting $\partial\mathscr{L}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})/\partial\sigma_k^{-2} = 0$ leads to

$$\sigma_k^2 = \frac{\sum_{l=1}^{L}\beta_{lk}\left(\left(y_l - \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}\left(\boldsymbol{x}_l\right)\boldsymbol{g}_k\right)^2 + \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}\left(\boldsymbol{x}_l\right)\boldsymbol{G}_k\widetilde{\boldsymbol{\psi}}_k\left(\boldsymbol{x}_l\right)\right)}{\sum_{l=1}^{L}\beta_{lk}}. \tag{34}$$

In addition, based on (25) and (28), the expectations required in (19) and (26) are calculated as

$$\begin{aligned}\left\langle \ln \mathcal{N}\left(y_l \middle| \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{\theta}_k, \sigma_k^2\right)\right\rangle &= -\frac{1}{2}\ln\sigma_k^2 - \frac{1}{2}\ln 2\pi \\ &\quad - \frac{1}{2\sigma_k^2}\left(\left(y_l - \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{g}_k\right)^2 + \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}_l)\boldsymbol{G}_k\widetilde{\boldsymbol{\psi}}_k(\boldsymbol{x}_l)\right)\end{aligned} \tag{35}$$

$$\left\langle \eta_k \right\rangle = \lambda_k^1/\lambda_k^2, \tag{36}$$

From the above derivation process, one can recognize that the formulas for parameter learning are coupled. Typically, we can first calculate $p^*(\boldsymbol{Z}_L)$ and $p^*(\boldsymbol{Z}_U)$ with other parameters fixed, and subsequently update the rest ones with $p^*(\boldsymbol{Z}_L)$ and $p^*(\boldsymbol{Z}_U)$ fixed. Such coordinate ascent strategy is referred to as the 'variational Bayes expectation–maximization (VBEM)' algorithm (Murphy, 2012), which is summarized in Algorithm 1 and illustrated in Fig. 3.

Specifically, at the beginning, one needs to select the model inputs including the number M of ELMs, the number m of hidden nodes for each ELM, the prior hyper-parameters λ_0^1 and λ_0^2, the maximum iteration times T, and the convergence threshold value ϖ. The parameters T and ϖ are used to detect whether the iteration process is converged, which can be easily determined. And λ_0^1 and λ_0^2 and the ratio λ_0^1/λ_0^2 between them can be set as small values so as to avoid undue regularization that penalizes the model complexity too heavily and results in significant bias (Shao, Yao et al., 2019). In addition, M and m are two integers with explicit physical meanings which should not be difficult to determine, as will be demonstrated shortly in Section 4.

Subsequently, in order to start the iteration, model parameters including the deterministic model parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \sigma_k^2\}_{k=1}^M$ and the posterior hyper-parameters $\{\boldsymbol{g}_k, \boldsymbol{G}_k, \lambda_k^1, \lambda_k^2\}_{k=1}^M$ for random model parameters need to be initialized. The initialized values for them can be randomly determined.

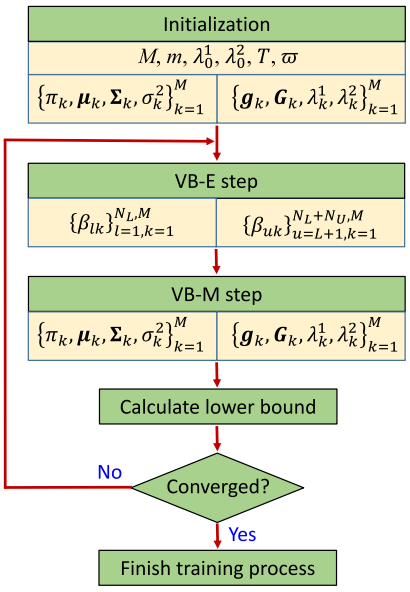


Fig. 3. Flowchart of training the S²PMELMs.

Afterwards, we enter the iteration process consisting of the variational Bayes expectation (VB-E) step and the variational Bayes maximization (VB-M) step. In the VB-E step, we calculate the probabilities of assigning the lth labeled sample and the uth unlabeled sample to the kth ELM, namely β_{lk} and β_{uk} where $k \in \{1, 2, \ldots, M\}$, $l \in \{1, 2, \ldots, N_L\}$ and $u \in \{L+1, L+2, \ldots, L+U\}$. Next, in the VB-M step, deterministic model parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \sigma_k^2\}_{k=1}^M$ and hyper-parameters $\{\boldsymbol{g}_k, \boldsymbol{G}_k, \lambda_k^1, \lambda_k^2\}_{k=1}^M$ are updated. After finishing the VB-M step, we calculate the lower bound, which is used for convergence detection. If the convergence condition is satisfied, then terminate the iteration process and output the required parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \sigma_k^2, \boldsymbol{g}_k, \boldsymbol{G}_k, \lambda_k^1, \lambda_k^2\}_{k=1}^M$; otherwise, return to the VB-E step. Repeat the above iteration steps will reach to the convergence which is guaranteed because the lower bound can be proved to monotonically increase and is bounded.

The convergence criterion can be set as

$$\left|\frac{\mathscr{L}^{(i)}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega}) - \mathscr{L}^{(i-1)}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})}{\mathscr{L}^{(i-1)}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})}\right| \times 100\% < \varpi \tag{37}$$

where $\mathscr{L}^{(i)}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})$ and $\mathscr{L}^{(i-1)}(\boldsymbol{\Upsilon}, \boldsymbol{\Omega})$ are the lower bounds calculated at the ith and $(i-1)$th iterations, respectively.

Remark. Detailed derivations for this subsection, and the calculation of the lower bound for convergence diagnosis are placed in the supplementary material.

3.3. Development of soft sensor using the S²PMELMs

In the soft sensor application, when a sample $\boldsymbol{x}^*$ of the secondary variables is available, one need to calculate the estimation $\hat{y}^*$ of the true value y^* of the quality variable, which can be realized using the proposed S²PMELMs.

Based on the Bayes rules, given $\boldsymbol{x}^*$, the posterior distribution over the associated latent assignment variable $\boldsymbol{z}^* = [z_1^*, z_2^*, \ldots, z_M^*]^{\mathrm{T}}$ can be calculated as

$$\begin{aligned}p(z_k^* = 1|\boldsymbol{x}^*) &= \frac{p(\boldsymbol{x}^*|z_k^* = 1)p(z_k^* = 1)}{\sum_{k=1}^{M}p(\boldsymbol{x}^*|z_k^* = 1)p(z_k^* = 1)} \\ &= \frac{\pi_k\mathcal{N}(\boldsymbol{x}^*|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k=1}^{M}\pi_k\mathcal{N}(\boldsymbol{x}^*|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \equiv \beta_k^*.\end{aligned} \tag{38}$$

Algorithm 1 Pseudo-codes for training the S^{2}PMELMs.

Input: the number M of ELMs, the number m of hidden nodes for each ELM, prior hyper-parameters λ_0^1 and λ_0^2, maximum iteration times T, and convergence threshold ϖ.
Output: the maximum likelihood estimates for $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\sigma}^2$, and the variational distributions $p^*(\boldsymbol{\theta})$ and $p^*(\boldsymbol{\eta})$ characterized by $\{\boldsymbol{g}_k, \boldsymbol{G}_k, \lambda_k^1, \lambda_k^2\}_{k=1}^M$.
Initialization: initialize $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\sigma}^2$, and $\{\boldsymbol{g}_k, \boldsymbol{G}_k, \lambda_k^1, \lambda_k^2\}_{k=1}^M$.
Set $i = 0$.
while $i \le T$ **do**
  $i = i + 1$.
  for $k = 1, 2, \cdots, M$; $l = 1, 2, \cdots, N_L$; $u = L+1, L+2, \cdots, L+U$ **do**
    Calculate β_{lk} using (20).
    Calculate β_{uk} using (23).
  end for
  for $k = 1, \cdots, M$ **do**
    Update π_k, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ and σ_k^2 using (31), (32), (33) and (34), respectively.
    Update $\boldsymbol{G}_k$ and $\boldsymbol{g}_k$ using (26) and (27), respectively.
    Update λ_k^1 and λ_k^2 using (29) and (30), respectively.
  end for
  if Convergence condition is satisfied **then**
    Break while.
  end if
end while

In addition, with $\boldsymbol{g}_k$ and $\boldsymbol{G}_k$, for the kth ELM the p.d.f. of y^* in association with $\boldsymbol{x}^*$ is obtained as

$$\begin{aligned} p\left(y^*\middle|\boldsymbol{x}^*, z_k^*=1\right) &= \int p\left(y^*\middle|\boldsymbol{x}^*, z_k^*=1, \boldsymbol{\theta}_k\right) p^*\left(\boldsymbol{\theta}_k\right) \mathrm{d}\boldsymbol{\theta}_k \\ &= \int \mathcal{N}\left(y^*\middle|\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{\theta}_k, \sigma_k^2\right) \mathcal{N}\left(\boldsymbol{\theta}_k\middle|\boldsymbol{g}_k, \boldsymbol{G}_k\right) \mathrm{d}\boldsymbol{\theta}_k \\ &= \mathcal{N}\left(y^*\middle|\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{g}_k, \sigma_k^2 + \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{G}_k\widetilde{\boldsymbol{\psi}}_k(\boldsymbol{x}^*)\right). \end{aligned} \tag{39}$$

where $\widetilde{\boldsymbol{\psi}}_k(\boldsymbol{x}^*) = [\boldsymbol{\psi}_k^{\mathrm{T}}(\boldsymbol{x}^*), 1]^{\mathrm{T}}$.

Based on (38) and (39), the predictive p.d.f. of y^* conditioned on $\boldsymbol{x}^*$ can be obtained as

$$\begin{aligned} p\left(y^*\middle|\boldsymbol{x}^*\right) &= \sum_{k=1}^M p\left(z_k^*=1\middle|\boldsymbol{x}^*\right) p\left(y^*\middle|\boldsymbol{x}^*, z_k^*=1\right) \\ &= \sum_{k=1}^M \beta_k^* \mathcal{N}\left(y^*\middle|\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{g}_k, \sigma_k^2 + \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{G}_k\widetilde{\boldsymbol{\psi}}_k(\boldsymbol{x}^*)\right). \end{aligned} \tag{40}$$

The expectation of the predictive distribution in (40) is used for making the estimation. Thus, according to the definition of expectation, the soft sensor's estimation $\widehat{y}^*$ of y^* is obtained as

$$\widehat{y}^* = \int y^* p\left(y^*\middle|\boldsymbol{x}^*\right) \mathrm{d}y^* = \sum_{k=1}^M \beta_k^* \widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{g}_k. \tag{41}$$

4. Case study

In this section, the performance of the S^{2}PMELMs is evaluated using two industrial processes. The performance of several relevant state-of-the-art soft sensing methods are also provided for comparison. The benchmarking methods include semisupervised ELM (S^{2}ELM) (Huang et al., 2014), semisupervised hierarchical ELM (S^{2}HELM) (Yao & Ge, 2018) which represents the deep ELM, ensemble ELMs based on Adaboost (AdaELM) (Peng et al., 2017), and a semisupervised deep learning method which integrates stacked de-noising autoencoders (DAE) with neural networks called 'SAE-NN' (Yan et al., 2017). For those ELM-based approaches, the *sigmoid* function defined in (2) is employed as the activation function, and the SAE-NN is trained by the BP algorithm aided by the stochastic gradient descent (SGD). The estimation accuracies for various methods are quantified by the root mean squares error (RMSE) defined as

$$\mathrm{RMSE} = \sqrt{\sum_{j=1}^N (y_j - \widehat{y}_j)^2 \Big/ N}, \tag{42}$$

where y_j and $\widehat{y}_j$ are the real and estimated labels for the jth testing sample, respectively, and N is the size of the testing dataset. Moreover, the computational efficiencies for various methods are compared using the physical computer time spent in model training (CPT$_{trn}$, in seconds) and in calculating estimations for testing samples (CPT$_{tst}$, in seconds). The configurations of the computer used for carrying out experiments are as follows. CPU: Core i7-6700 (3.4 GHz × 2), RAM: 16 GB, OS: Windows 7. Note that we use random initializations for all the investigated methods, and the median values of the RMSE, CPT$_{trn}$ and CPT$_{tst}$ from 100 independent runs are used as the average performance.

Table 1
Secondary variables for the methanation process.

Tags	Explanation
U_1	Flow rate of process gas at MF's exit
U_2	Pressure of process gas at MF's entrance
U_3	Pressure of process gas into downstream unit
U_4	Temperature of top bed of MF
U_5	Temperature of middle bed of MF
U_6	Temperature of bottom bed of MF
U_7	Temperature of process gas at MF's entrance
U_8	Temperature of process gas at MF's exit
U_9	Temperature of synthesis gas at exit of Separator 2
U_{10}	Liquid level of Separator 2

4.1. Application to an industrial methanation process

The methanation process schematically presented in Fig. 4 is a part of an ammonia synthesizing plant, which transforms carbon monoxide (CO) and carbon dioxide (CO_2) into methane (CH_4). Before entering into the methenation furnace (MF), the reactant gases are first heated by heat exchangers. In the furnace, the following reactions are set off

$$\begin{aligned} &CO + 3H_2 \longleftrightarrow CH_4 + H_2O + Q, \\ &CO_2 + 4H_2 \longleftrightarrow CH_4 + 2H_2O + Q. \end{aligned} \tag{43}$$

where the reactant temperature plays a crucial role. Thus three thermocouples (installed at the top, middle, bottom beds of the MF) are used for monitoring the reaction temperature. The outlet gases from the furnace are first cooled by heat exchanger and then fed into the separator for liquidation. Afterwards, the liquefied gases follow into the condenser, while the outlet gases from the separator, which mainly include CH_4, H_2 and residual CO and CO_2 will be fed into other units, where the CH_4 will be recycled as fuel gas and the H_2 will be fed into the NH_3 synthesizing column as the source reactant.

The most important quality variable is the content of residual CO and CO_2, which is required to be below the designed level. Conventionally, a mass spectrometer (Y) is installed to measure the content of CO and CO_2. However, it introduces exorbitant cost and frequently malfunctions. On the other hand, laboratory analysis causes hours of delay. Therefore, a soft sensor is required to estimate the content of CO and CO_2. The secondary variables are selected by professional prior knowledge of field engineers, and are explained in Table 1.

Datasets were collected from the database of the ammonia synthesizing plant, where the training set for model training consists of 1000 labeled and 10,000 unlabeled samples, while the testing dataset for generalization ability evaluation consists of 2000 labeled samples.

By *trial-and-error*, parameters of various methods were selected as follows.

- S^{2}ELM: the number of neurons was set as 8000, the penalty coefficient C was set as 20, the trade-off parameter λ was set 0.5, and the Gaussian function was selected for calculating the similarity matrix where the kernel parameter σ^2 was set as 0.1.

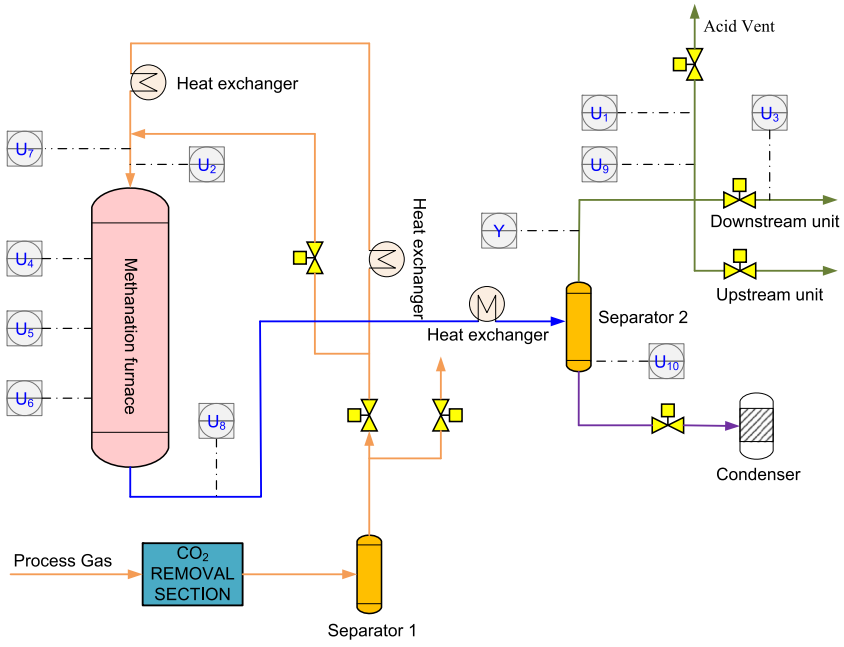


Fig. 4. Schematic diagram of the industrial methanation process.

- S^2HELM: the number of hidden layers was selected as 3, and the corresponding numbers of neurons were set as 300/1200/6000, $C = 0.3$, $\lambda = 0.1$, and the Gaussian function was selected for calculating the similarity matrix with $\sigma^2 = 0.05$.
- SAE-NN: the number of hidden layers was selected as 5, and the corresponding numbers of neurons were set as 10/8/8/8/8, the initial learning rate was set as 0.07, the momentum was set as 0.95, the mini-batch size was set as 200, the epochs was set as 1500 and the DAE epochs was set as 200.
- AdaELM: the number of neurons was set as 300, and the number of ELMs was set as 50, the threshold value was set as 0.1, and the power coefficient was selected as 1.
- S^2PMELMs: $M = 30$, $m = 100$, $\lambda_0^1 = \lambda_0^2 = 1$, $T = 300$, and $\varpi = 10^{-3}$.

The estimated contents of CO and CO_2 for the testing samples by various soft sensing approaches at their average performance are visually presented as Fig. 5, where the quality variable has been scaled. It is noticed that the estimations of the content of CO and CO_2 provided by various soft sensors can basically match the true values. However, intuitively the S^2PMELMs shows advantages in two respects. Firstly, the estimations by the S^2PMELMs-based soft sensor are more smooth. Secondly, the S^2PMELMs can better describe the whole operating area, in particular for the local area around the 1250th sample where the estimations by the S^2ELM, S^2HELM and AdaELM indicate significant biases. The scatter plot comparisons among the performance of the S^2PMELMs-based soft sensor and the others are shown in Fig. 6. In the scatter plot, the ideal case is that the scatters are located exactly on the diagonal line, meaning that the estimated values are precisely equal to the corresponding true values. In other words, the more closely the scatters are distributed along the diagonal line, the higher the estimation accuracy is. As can be recognized, compared with the scatters obtained by other four soft sensors, the scatters provided by the S^2PMELMs-based one basically lie more compactly and evenly at the two sides of the black diagonal line, which can qualitatively imply that the S^2PMELMs-based soft sensor achieves unbiased and better estimation performance. Moreover, the high-values of content of residual CO and CO_2 should draw our attention, because they may cause off-grade product. In this respect, the S^2PMELMs-based soft sensor outperforms the rest ones, because Fig. 6 indicates that for the large values (which are greater than 0.8) of content of CO and CO_2, the S^2PMELMs-based soft sensor's estimations are much more precise.

Table 2
Average performance of five soft sensors for the methanation process.

	avg. RMSE±std.	avg. CPT_{trn}	avg. CPT_{tst}
S^2ELM	0.0635 ± 0.00071	400.2	1.05
AdaELM	0.0622 ± 0.00035	1.1	1.55
S^2HELM	0.0600 ± 0.00064	425.9	2.46
SAE-NN	0.0569 ± 0.00292	17.0	1.88
S^2PMELMs	0.0510 ± 0.00219	170.1	0.082

Quantitative performance comparisons among various soft sensors are summarized in Table 2. One can recognize from the index CPT_{tst} that all five soft sensing approaches' computational efficiency for making online estimations seem not an issue. In particular, one can notice that the online computational efficiency for the S^2PMELMs-based soft sensor is much better. This is because for the S^2PMELMs the quantity of neurons is small and only quite simple algebraic manipulations of matrices and vectors are required as shown in (38)~(41). However, the index CPT_{trn} indicates that the S^2ELM and SH^2ELM seem to lose the ELM's advantage of being extremely fast. This is because for the S^2ELM and S^2HELM, the size of the constructed graph Laplacian matrix are quite big ($11,000 \times 11,000$), and in order to struggle to represent the complex methanation process, the numbers of neurons are also forced to be large (i.e., 8000 for the S^2ELM and 300/1200/6000 for the S^2HELM). As a consequence, the computations of inverting large size matrices were involved and the vast majority of CPU resources (about 99%) were occupied. In contrast, in the S^2PMELMs, the number of neurons for each localized ELM can be much smaller (only 100). Nevertheless, the iterative localization process involving the entire training set increases the time complexity of the S^2PMELMs, causing longer training time than the SAE-NN. This is because for the SAE-NN the numbers of neurons are further reduced and the SGD used only 10% percent of samples for gradient calculation. It is worth pointing out that, there is a feasible way of improving the training efficiency for the S^2PMELMs, i.e., to use the parallel computing strategy, where the whole computational task is partitioned and completed by a group of slave computing devices simultaneously (Shao, Yao et al., 2019).

Besides, as is observed from the index avg. RMSE in Table 2, compared with the S^2ELM that employs single shallow ELM, using either multiple shallow ELMs (by the AdaELM and S^2PMELMs) or deep

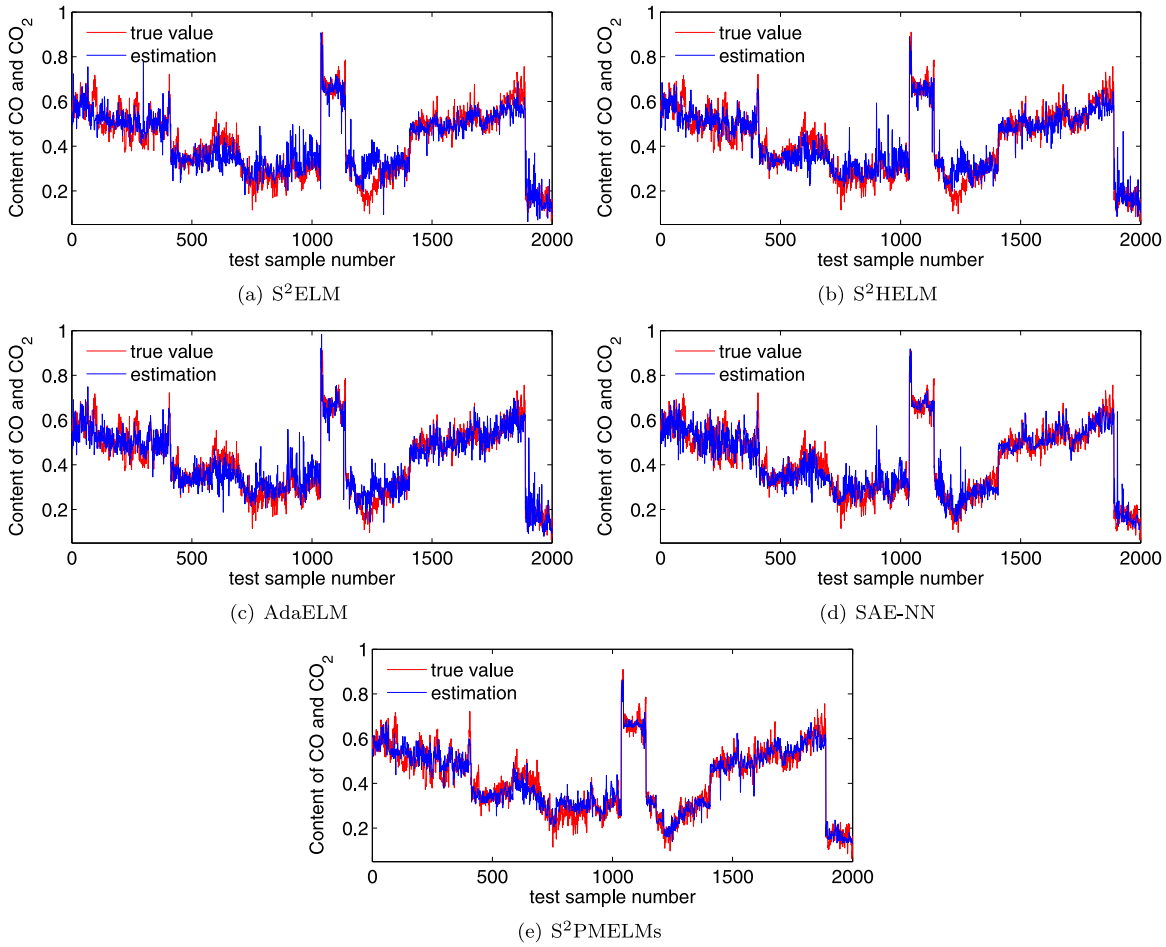


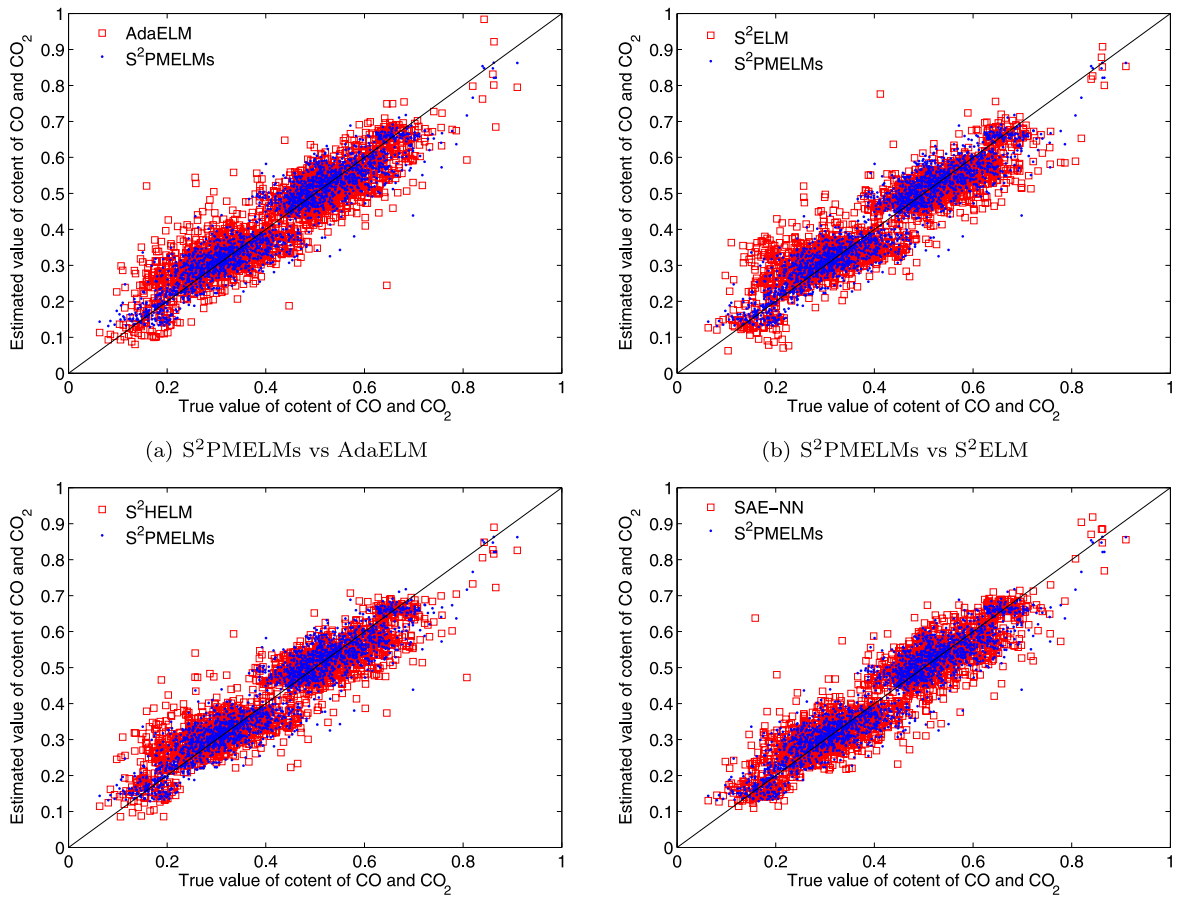
Fig. 5. Estimations of CO and CO_2 content by various soft sensors.

networks (by the S^{2}HELM and SAE-NN) can achieve performance enhancement. However, the improvement made by the AdaELM is slight, as it could not substantially overcome the drawback of a single shallow ELM, and it is incapable of exploiting unlabeled samples. Although the S^{2}HELM and SAE-NN, especially the SAE-NN, perform better than the AdaELM, it should be emphasized that determining the model structure (i.e., the number of hidden layers and the number of neurons for each hidden layer) and other parameters for them are quite intractable.

In contrast with the three ELM-based methods, the improvement achieved by the proposed S^{2}PMELMs can be regarded as significant. Specifically, compared with the S^{2}ELM, AdaELM and S^{2}HELM, the S^{2}PMELMs has reduced the average RMSE by around 20%, 18% and 15%, respectively. Even compared with the SAE-NN, the S^{2}PMELMs has gained predictive advantage by enhancing the estimation accuracy by around 10%. Meanwhile, for the S^{2}PMELMs, as λ_1^0 and λ_2^0 are easy to adjust (Shao, Yao et al., 2019), we only need to pay attention to two important parameters (i.e., M and m) which are both integers and thus not difficult to select. Therefore, training the S^{2}PMELMs is much easier than training the S^{2}HELM and SAE-NN. From these comparisons, we can basically verify the effectiveness of the S^{2}PMELMs in dealing with certain drawbacks of shallow and deep networks, i.e., the inadequate representation ability of a single shallow ELM for complex process and the difficulties in determining model structure and parameters for deep networks.

The reasons for which the S^{2}PMELMs can outperform the rest benchmarking methods in terms of the estimation accuracy are analyzed as follows. First, from (41) we see that the final estimation of the S^{2}PMELMs is the combination of those of M ELMs, where $\widetilde{\boldsymbol{\psi}}_k^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{g}_k$ and β_k^* are the estimation and contribution of the kth individual ELM, respectively. Note that β_k^* is adaptive according to (38), meaning that the contributions of ELMs for each query sample are adaptively calculated rather than being constant. Second, from (26) and (27) one can find that the regression coefficient vector $\boldsymbol{g}_k$ for the kth ELM is actually the solution of a weighted least-squares problem, where $\boldsymbol{\Lambda}_k$ contains the weight of each (labeled) training sample. In this way, the localization is realized, and each of the M ELMs just need to represent a local area, which can overcome the drawback of shallow network with limited representation ability with respect to the entire operating area, and does not suffer from the defect of the deep network (such as the SAE-NN) in determining the model structure, neither. Third, it is also recognized from (26) and (27) that each solution is regularized where $\langle\eta_k\rangle$ means the regularization parameter for the kth ELM. And according to (29), (30) and (36), $\langle\eta_k\rangle$ for $k \in \{1, 2, \ldots, M\}$ can be automatically determined, which is extremely important. This is because on the one hand, these regularizations play important role in dealing with overfitting, but improper regularization parameters lead to deteriorated performance. On the other hand, using manual work to properly select M regularization parameters is nontrivial, while in the S^{2}PMELMs this task can be well completed. In addition, the process of training the S^{2}PMELMs implies that both labeled and unlabeled sample are utilized, which however differs from the semisupervised learning ways in the SAE-NN and S^{2}HELM. In the SAE-NN and S^{2}HELM, the labeled and unlabeled samples are used separately while in the S^{2}PMELMs they are used jointly which is therefore more effective.

It is also noticed from Table 2 that, for the SAE-NN and S^{2}PMELMs the standard deviation (std.) of RMSE calculated from 100 runs are larger than those for the S^{2}ELM, AdaELM and S^{2}HELM. This is mainly because there exist multiple local maximums for the BP algorithm and the VBEM algorithm using random initializations.



(a) S²PMELMs vs AdaELM (b) S²PMELMs vs S²ELM

(c) S²PMELMs vs S²HELM (d) S²PMELMs vs SAE-NN

Fig. 6. Scatter plot comparisons among the performance of the S²PMELMs-based soft sensor and others.

Table 3
Results of the Wilcoxon test for the methanation process.

Hypothesis	*p*-value	Decision
$\mathcal{H}_0 : \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{S}^2\text{ELM}}$	5.81×10^{-22}	Reject $\mathcal{H}_0$
$\mathcal{H}_0 : \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{AdaELM}}$	8.88×10^{-26}	Reject $\mathcal{H}_0$
$\mathcal{H}_0 : \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{S}^2\text{HELM}}$	2.48×10^{-13}	Reject $\mathcal{H}_0$
$\mathcal{H}_0 : \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{SAE-NN}}$	1.17×10^{-12}	Reject $\mathcal{H}_0$

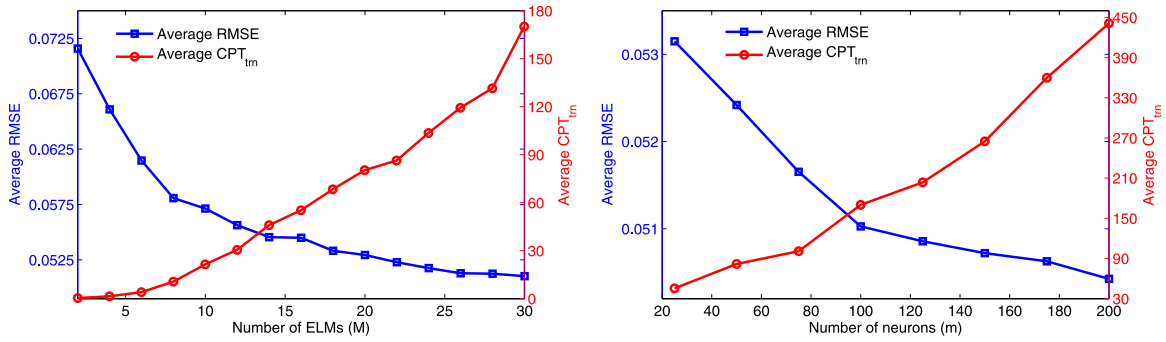
Table 4
Performance of the S²PMELMs with various sizes of unlabeled dataset.

U	avg. RMSE	avg. CPT_{trn}
0	0.0552	35.7
3 000	0.0546	96.1
5 000	0.0524	122.9
10 000	0.0510	170.1

To test if the performance of the S²PMELMs significantly differs from those of the benchmarking approaches, statistical testing has been conducted using the non-parametric Wilcoxon signed rank test (Rodríguez-Fdez, Canosa, Mucientes, & Bugarín, 2015). The Wilcoxon test is utilized to test if the medians of the squared estimated errors for two soft sensors (at the average accuracy level) are significantly different or not. Table 3 summarizes the results of the Wilcoxon test, where $\overline{E}^2_{\text{S}^2\text{ELM}}$, $\overline{E}^2_{\text{AdaELM}}$, $\overline{E}^2_{\text{S}^2\text{HELM}}$, $\overline{E}^2_{\text{SAE-NN}}$ and $\overline{E}^2_{\text{S}^2\text{PMELMs}}$ mean the medians of the squared estimated errors for the S²ELM, AdaELM, S²HELM, SAE-NN and S²PMELMs, respectively, and the significance level α was set as 5%. The Wilcoxon test returns a *p*-value that measures how likely the corresponding hypothesis could be accepted. If the *p*-value is less than α, the hypothesis should be rejected. As can be seen from Table 3, the *p*-values for the four hypotheses are extremely small, and as a result all hypotheses are rejected. That is, the comparisons among the S²PMELMs and other benchmarking approaches in the above paragraph have statistical significance.

Table 4 investigates the impact of the size of unlabeled dataset on the performance of the S²PMELMs for the methenation process. From Table 4 one can find that as the number U of unlabeled samples increases, the estimation accuracy of the S²PMELMs gets improved, which reveals that exploiting the unlabeled samples plays positive role in enhancing the estimation performance. However, the incorporation of unlabeled samples increases the training time, which is mainly due to the computations proportional to the size of unlabeled dataset according to (22) and (A.16) in the supplementary material.

Fig. 7 demonstrates the impacts of the two integral parameters (M and m) of the S²PMELMs on the estimation accuracy. As can be seen, with the increase of M or m, the estimation accuracy gets improved. This can be readily explained as follows. Larger M means more detailed localization, which can decrease the requirement of the representation ability for shallow ELM, while raising m could enhance the representation ability of shallow ELM. Meanwhile, by using (26), (27), (29) and (30), each localized ELM has been regularized and all regularized parameters are automatically determined (otherwise tremendous efforts would be required), which can effectively deal with overfitting. Consequently, for the S²PMELMs, selecting proper M and m is not a difficult task. At last, it is emphasized that, as M or m increases, the accuracy improvement seems to be less significant, while larger M or m leads to higher model complexity which results in



(a) Impact of M with $m = 100$ and $U = 10000$. (b) Impact of m with $M = 30$ and $U = 10000$.

Fig. 7. Impacts of M and m on the estimation accuracy.

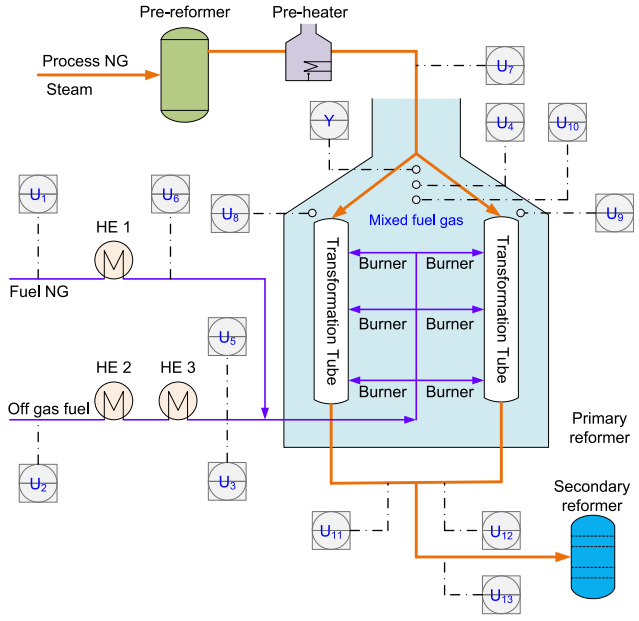


Fig. 8. Schematic diagram of the industrial primary reformer.

significantly longer training time. Therefore, the trade-off between the estimation accuracy and the training efficiency is suggested to be taken into account when determining M and m.

4.2. Application to an industrial primary reformer process

The primary reformer is also extracted from the ammonia synthesizing plant, which is schematically illustrated in Fig. 8. In the primary reformer process, the mixed gases consisting of natural gas (NG) and steam are first transformed in the pre-reformer where the temperature falls, and subsequently are fed into the primary reformer (PR) after being heated by the pre-heater. In the PR, a group of transformation tubes are installed where the following chemical reactions are produced:

$$\begin{aligned} &C_nH_{2n+2} + nH_2O \xleftrightarrow{\Delta} nCO + (2n+1)H_2, \\ &CH_4 + H_2O \xleftrightarrow{\Delta} CO + 3H_2, \\ &CO + H_2O \xleftrightarrow{\Delta} CO_2 + H_2, \end{aligned} \tag{44}$$

Mixed fuel gases (including fuel NG and offgas fuel) are burned at the burners in order to maintain the reaction level so as to obtain sufficient amount of H_2 which is the source material for the product NH_3. In the furnace the content of O_2 is used to adjust the reaction temperature, and a spectrometer (i.e., Y) is installed at the top of the furnace to measure the O_2 content. However, the spectrometer suffers from the same issues described in Section 4.1. Thus a soft sensor is desirable for online estimating the content of O_2.

Table 5
Secondary variables for the primary reformer process.

Tags	Explanation
U_1	Flow rate of fuel NG
U_2	Flow rate of fuel off gas
U_3	Pressure of fuel off gas at E3's exit
U_4	Pressure of furnace flue gas at PR's exit
U_5	Temperature of fuel off gas at E3's exit
U_6	Temperature of fuel NG at PH's exit
U_7	Temperature of process gas at PR's entrance
U_8	Temperature of furnace flue gas at PR's top left
U_9	Temperature of furnace flue gas at PR's top right
U_{10}	Temperature of mixed flue gas at PR's top
U_{11}	Temperature of transformed gas at PR' left exit
U_{12}	Temperature of transformed gas at PR' right exit
U_{13}	Temperature of transformed gas at PR' exit
Y	Oxygen content at the top of the furnace

Secondary variables (i.e., U_1~U_{13} in Fig. 8) for soft sensing the O_2 content are selected using the expert knowledge, which are explained in Table 5. Datasets were gathered from the database of the ammonia synthesizing plant. After data preprocessing, we use 1000 labeled samples and 6000 unlabeled samples as the training dataset, and around 3000 samples as the testing dataset.

Parameters of various soft sensing approaches were determined by *trial-and-error*, which are summarized as follows.

- S^{2}ELM: the number of neurons was set as 6000, the $C = 10$ was set as 20, $\lambda = 0.75$, and the Gaussian function was selected for calculating the similarity matrix where the kernel parameter $\sigma^2 = 0.08$.
- S^{2}HELM: the number of hidden layers was selected as 2, and the corresponding numbers of neurons were set as 400/6000, $C = 0.5$, $\lambda = 0.3$, and the Gaussian function was selected for calculating the similarity matrix with $\sigma^2 = 0.05$.
- SAE-NN: the number of hidden layers was selected as 4, and the corresponding numbers of neurons were set as 100/80/60/60, the initial learning rate was set as 0.07, the momentum was set as 0.95, the mini-batch size was set as 400, the epochs was set as 1500 and the DAE epochs was set as 200.
- AdaELM: the number of neurons was set as 500, and the number of ELMs was set as 50, the threshold value was set as 0.05, and the power coefficient was selected as 1.
- S^{2}PMELMs: $M = 30$, $m = 200$, $\lambda_0^1 = \lambda_0^2 = 1$, $T = 300$, and $\varpi = 10^{-3}$.

The estimations of contents of O_2 by various soft sensing methods are illustrated as Fig. 9, where the contents of O_2 have been scaled. Through Fig. 9 we can identify that the estimated values provided

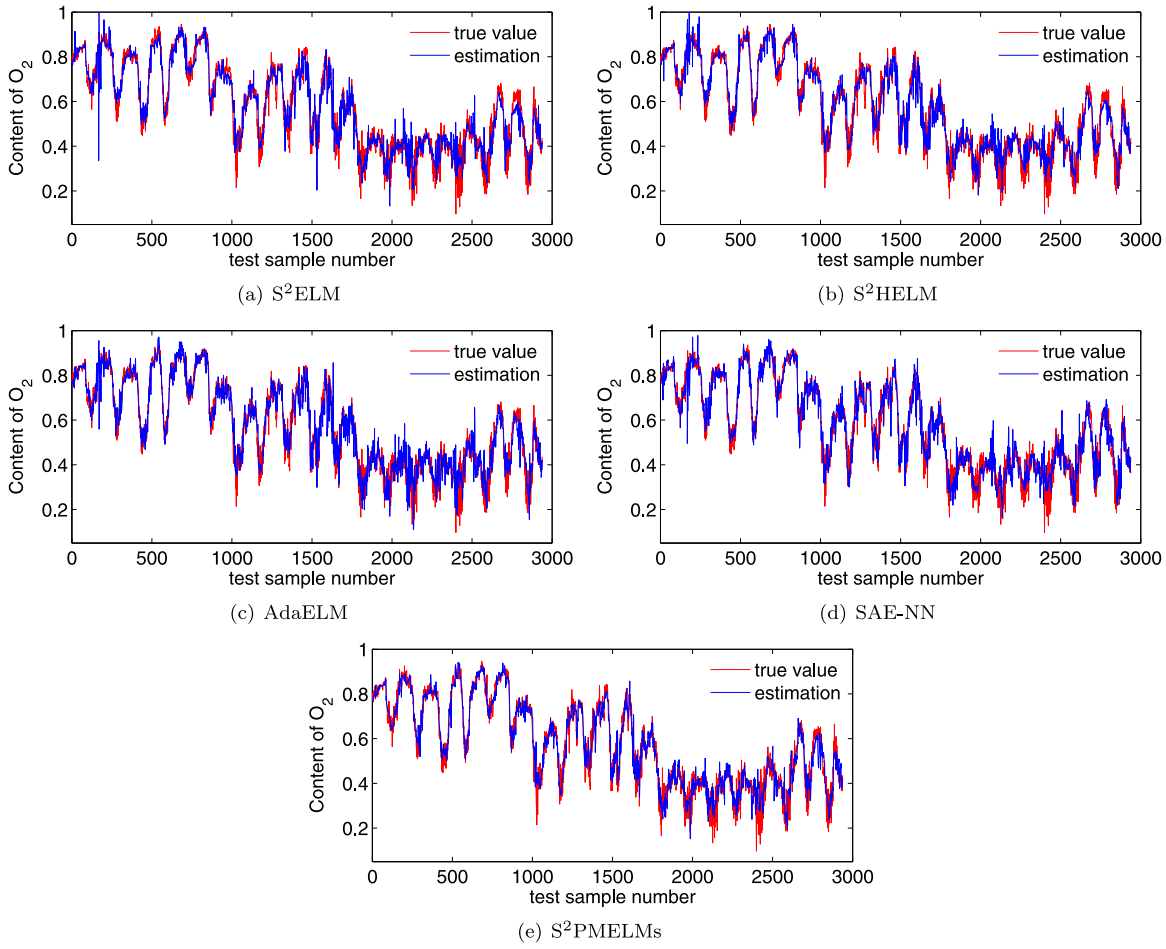


Fig. 9. Estimations of O_2 content by various soft sensors.

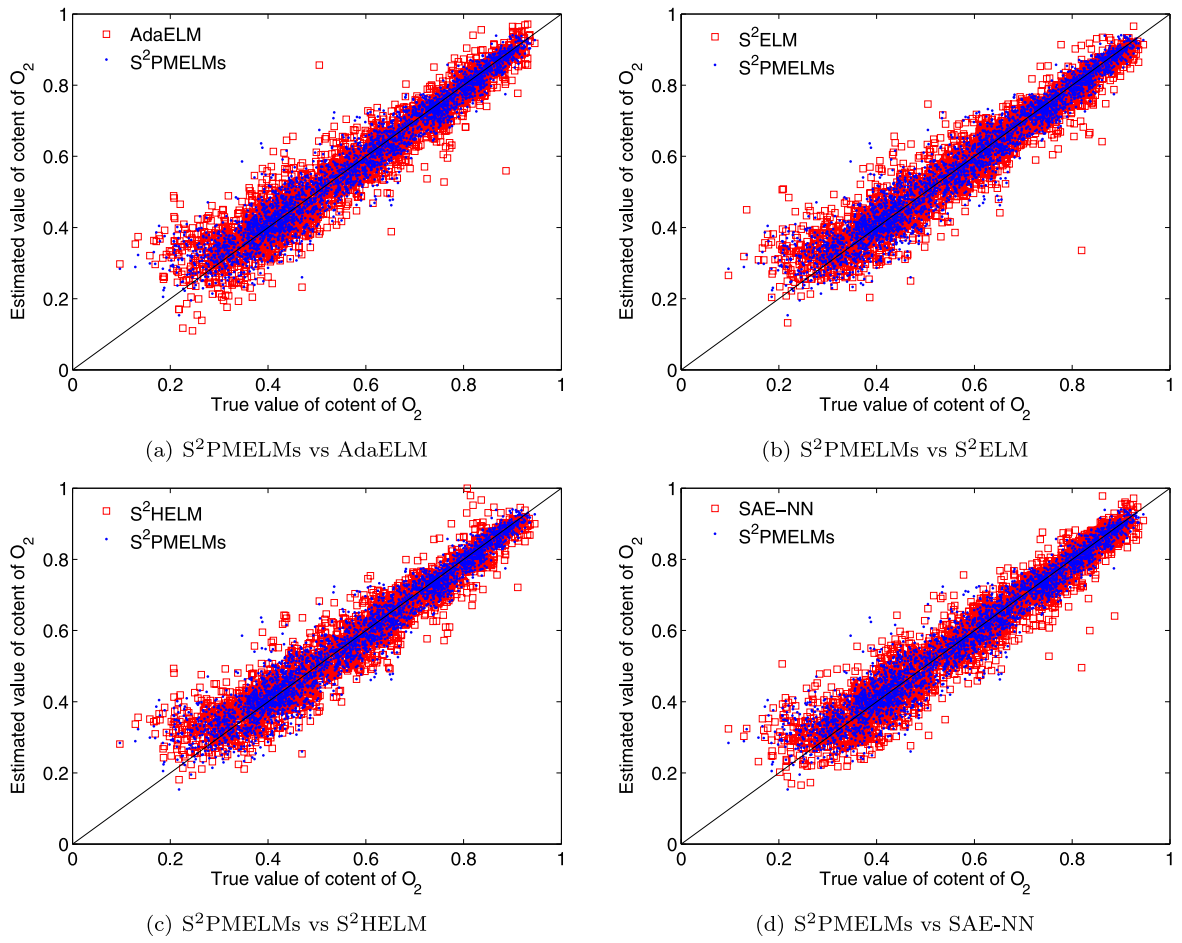
by the S^2HELM and S^2PMELMs match the true values better with better smoothness and smaller variance in comparison to those of the rest ones. Detailed comparisons between Fig. 9(b) and (e) can further reveal that: (1) the S^2PMELMs outperforms the S^2HELM in preventing remarkably deviated estimations; (2) the estimation performance of the S^2HELM seems inferior to that of the S^2PMELMs for operating area around the 180th sample, and that from 1500th to 2300th samples. Furthermore, Fig. 10 demonstrates that the scatters calculated by the S^2PMELMs are overall distributed more closely to the diagonal line, which can qualitatively confirm the advantage of the S^2PMELMs over the benchmarking methods. In particular, one can find from Fig. 10 that in the large-value areas (which should be also paid attention to as in those areas high energy consumptions usually occur) the S^2PMELMs-based soft sensor shows apparently superior performance.

Quantitative performance comparisons among the investigated soft sensing approaches are summarized in Table 6. As can be seen, in terms of the average RMSE, the S^2PMELMs-based soft sensor achieves the best performance, which can improve the estimation accuracy by 9.6%, 12.7%, 6.2% and 8.3% compared with those based on the S^2ELM, AdaELM, S^2HELM and SAE-NN, respectively. Note that in this case, the AdaELM that employs multiple non-localized ELMs is not able to gain improvement compared with the semi-supervised single ELMs (i.e., S^2ELM and S^2HELM). In addition, in terms of the computational efficiency for online calculating the estimations of O_2 content, Table 6 manifests the S^2PMELMs performs much better; however, Table 6 also reveals the computational deficiency of the S^2PMELMs from the perspective of off-line model training. It is noticed that compared with the previous case in Section 4.1, the S^2ELM and S^2HELM-based soft sensors require much shorter off-line training time. This can be explained as follows. In this case, the size of unlabeled dataset (6000) is much smaller than that in the case of the methanation process (10,000). Hence the sizes of the matrices which need to be inverted are much smaller in this primary reformer case, and meanwhile the CPU resources can be released; in addition, for the S^2HELM, only two hidden layers (deeper network causes significant performance deterioration) are used in this case. It is also noted that the average training time for the SAE-NN is longer than that in the previous case, which is mainly due to that in this case the numbers of neurons for hidden layers are much bigger than those in the previous case.

Table 6
Average performance of five soft sensors for the primary reformer process.

	avg. RMSE±std.	avg. CPT_{trn}	avg. CPT_{tst}
S^2ELM	0.0519 ± 0.00076	32.2	0.59
AdaELM	0.0537 ± 0.00049	1.76	2.7
S^2HELM	0.0498 ± 0.00061	39.7	0.29
SAE-NN	0.0508 ± 0.00307	35.8	3.2
S^2PMELMs	0.0469 ± 0.00194	156.7	0.076

Statistical testing results using the Wilcoxon test for the studied soft sensing approaches for this primary reformer process are tabulated in Table 7, where the significance level α was set as 0.05. As can be seen, for the four null hypotheses except the one for the S^2HELM and S^2PMELMs (i.e., $\mathcal{H}_0 : \overline{E}^2_{S^2\text{PMELMs}} = \overline{E}^2_{S^2\text{HELM}}$), the p-values are quite small; although the p-value for $\mathcal{H}_0 : \overline{E}^2_{S^2\text{PMELMs}} = \overline{E}^2_{S^2\text{HELM}}$ is much bigger, it is still below the significance level α. Therefore, the four null hypotheses are all rejected, meaning that the median of the squared estimation error obtained by the S^2PMELMs-based soft sensor (at the



(a) S²PMELMs vs AdaELM (b) S²PMELMs vs S²ELM

(c) S²PMELMs vs S²HELM (d) S²PMELMs vs SAE-NN

Fig. 10. Scatter plot comparisons among the performance of the S²PMELMs-based soft sensor and others.

Table 7
Results of the Wilcoxon test for the primary reformer process.

Hypothesis	p-value	Decision
$\mathcal{H}_0: \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{S}^2\text{ELM}}$	8.23×10^{-5}	Reject $\mathcal{H}_0$
$\mathcal{H}_0: \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{AdaELM}}$	8.01×10^{-13}	Reject $\mathcal{H}_0$
$\mathcal{H}_0: \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2_{\text{S}^2\text{HELM}}$	1.59×10^{-2}	Reject $\mathcal{H}_0$
$\mathcal{H}_0: \overline{E}^2_{\text{S}^2\text{PMELMs}} = \overline{E}^2$ SAE-NN	1.40×10^{-7}	Reject $\mathcal{H}_0$

average level) is statistically different from those obtained by the rest ones. Based on this, we can confirm that the advantage of the proposed method has statistical significance.

5. Conclusions

In this paper, we have proposed an ELM-based soft sensing approach referred to as the 'S²PMELMs'. In addition to being able to exploit both labeled and unlabeled samples, by using multiple localized shallow ELMs, the S²PMELMs is able to address certain issues of both shallow and deep networks, namely the inadequacy of the single shallow network in representing complex process and the difficulties of the deep network in determining model structure. The performance of the S²PMELMs has been evaluated by two industrial processes, and the comparisons among the S²PMELMs and several state-of-the-art related methods have verified the effectiveness of the proposed strategy (i.e., to model complex process by combining localized shallow networks) in improving the estimation accuracy, as well as its promising application foreground.

A drawback of the S²PMELMs has been found that although increasing the number of unlabeled samples and that of the component ELMs is helpful for improving the estimation performance, the computational efficiency for off-line model training may decrease a lot due to the increased complexity. We have pointed out a way to deal with this computational issue. Specifically, as the samples and component ELMs are treated to be independent with each other, the computations associated with training samples and those with ELMs can be divided into sub-tasks, each of which is allocated to one slave computing device. Those slave computing devices could complete the sub-tasks simultaneously, i.e., in the parallel way, such that the computational issue can be effectively dealt with. In other words, by using the parallel computing strategy, the S²PMELMs can be enabled to be suitable for large-scale problem which involves large size of dataset and requires large number of component ELMs. However, it should be emphasized this solution is at the cost of more computing resources.

In addition, the developed S²PMELMs in this paper is static, focusing on industrial processes with weak dynamic characteristics. For applications where modeling strong dynamics are required, the S²PMELMs can be extended to the dynamic version by using the moving average model structure that augments the input vector with information sampled at the previous sampling instants (Fortuna, Graziani, Rizzo, & Xibilia, 2007). However, this may introduce a numerical issue (i.e., the singularity of calculating $\{\boldsymbol{\Sigma}_k\}_{k=1}^M$) due to high dimensionality. To this end, $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^M$ can be further randomized where the Gaussian–Wishart distribution is selected as the joint prior distribution for $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k^{-1}$.

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.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (Grant No. 61703367), in part by the China Postdoctoral Science Foundations (Grant Nos. 2019T120516 and 2017M621929), and in part by the Key Project of National Natural Science Foundation of China (Grant No. 61833014).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.conengprac.2019.07.016.

References

Ardabili, S. F., Najafi, B., Alizamir, M., Mosavi, A., Shamshirband, S., & Rabczuk, T. (2018). Using SVR-RSM and ELM-RSM approaches for optimizing the production process of methyl and ethyl esters. *Energies*, *11*, 2889. http://dx.doi.org/10.3390/en11112889.

Bai, Z., Huang, G. B., Wang, D., et al. (2014). Sparse extreme learning machine for classification. *IEEE Transaction on Cybernetics*, *44*(10), 1858–1870.

Bishop, C. (2006). *Pattern Recognition and Machine Learning* (1st ed.). New York, USA: Springer-Verlag.

Chen, Y., Yang, J., Wang, C., & Park, D. S. (2016). Variational Bayesian extreme learning machine. *Neural Computing & Application*, *27*, 185–196.

Fortuna, L., Graziani, S., Rizzo, A., & Xibilia, M. G. (2007). *Soft sensors for monitoring and control of industrial processes*. London: Springer-Verlag.

Ge, Z. (2018). Process data analytics via probabilistic latent variable models: A tutorial review. *Industrial and Engineering Chemistry Research*, *57*(38), 12646–12661.

Ge, Z., Song, Z., Ding, S. X., & Huang, B. (2017). Data mining and analytics in the process industry: the role of machine learning. *IEEE Access*, *99*(5), 20590–20616.

Geng, Z., Dong, J., Chen, J., & Han, Y. (2017). A new self-organizing extreme learning machine soft sensor model and its applications in complicated chemical processes. *Engineering Application of Artificial Intelligence*, *62*, 38–50.

Ghazvinei, P. T., Darvishi, H. H., Mosavi, A., Yusof, K. b. W., Alizamir, M., Shamshirb, S., et al. (2018). Sugarcane growth prediction based on meteorological parameters using extreme learning machine and artificial neural network. *Engineering Applications of Computational Fluid Mechanics*, *12*(1), 738–749.

Gopakumar, V., Tiwari, S., & Rahman, I. (2018). A deep learning based data driven soft sensor for bioprocesses. *Biochemical Engineering Journal*, *136*, 28–39.

Graziani, S., & Xibilia, M. G. (2017). A deep learning based soft sensor for a sour water stripping plant. In *IEEE international instrumentation and measurement technology conference* (pp. 1–6). Torino, Italy.

He, Y., Geng, Z., & Zhu, Q. (2015). Data driven soft sensor development for complex chemical processes using extreme learning machine. *Chemical Engineering Research and Design*, *102*, 1–11.

He, Y., Geng, Z., & Zhu, Q. (2016). Soft sensor development for the key variables of complex chemical processes using a novel robust bagging nonlinear model integrating improved extreme learning machine with partial least square. *Chemometrics and Intelligent Laboratory Systems*, *151*, 78–88.

He, Y., Xu, Y., & Zhu, Q. (2016). Soft-sensing model development using PLSR-based dynamic extreme learning machine with an enhanced hidden layer. *Chemometrics and Intelligent Laboratory Systems*, *154*, 101–111.

Hinton, G., & Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Huang, G. B., Chen, L., & Siew, C. K. (2006). Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Networks*, *17*(4), 879–892.

Huang, G., Huang, G. B., Song, S., & You, K. (2015). Trends in extreme learning machines: A review. *Neural Networks*, *61*, 32–48.

Huang, G., Song, S., Gupta, J. N. D., & Wu, C. (2014). Semi-supervised and unsupervised extreme learning machines. *IEEE Transactions on Cybernetics*, *44*(12), 2405–2417.

Huang, G. B., Zhou, H., Ding, X., & Zhang, R. (2012). Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man & Cybernetics-Part B*, *42*(2), 513–529.

Huang, G. B., Zhu, Q., & Siew, C. (2006). Extreme learning machine: Theory and applications. *Neurocomputing*, *70*(1), 489–501.

Jin, H., Chen, X., Wang, L., Yang, K., & Wu, L. (2016). Dual learning-based online ensemble regression approach for adaptive soft sensor modeling of nonlinear time-varying processes. *Chemometrics and Intelligent Laboratory Systems*, *151*, 228–244.

Jin, X., Wang, S., Huang, B., & Forbes, F. (2012). Multiple model based LPV soft sensor development with irregular/missing process output measurement. *Control Engineering Practice*, *20*(2), 165–172.

Kasun, L., Zhou, H., Huang, G. B., & Vong, C. M. (2013). Representational learning with extreme learning machine for big data. *IEEE Intelligent Systems*, *28*(6), 31–34.

Kim, K. I., Steinke, F., & Hein, M. (2009). Semi-supervised regression using hessian energy with an application to semi-supervised dimensionality reduction. In *Advances in neural information processing systems* (pp. 979–987). Vancouver, Canada.

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*, 436–444.

Liu, X., Gao, C., & Li, P. (2012). A comparative analysis of support vector machines and extreme learning machines. *Neural Networks*, *33*, 58–66.

Liu, Y., Yang, C., Gao, Z., & Yao, Y. (2018). Ensemble deep kernel learning model for quality prediction in industrial polymerization processes. *Chemometrics and Intelligent Laboratory Systems*, *174*, 15–21.

Martínes Martínes, J. M., Escandell-Montero, P., Soria-Olivas, E., et al. (2011). Regularized extreme learning machine for regression problems. *Neurocomputing*, *74*(17), 3716–3721.

Murphy, K. (2012). *Machine learning: A probabilistic perspective*. Cambridge, USA: The MIT Press.

Peng, T., Zhou, J., Zhang, C., & Zheng, Y. (2017). Multi-step ahead wind speed forecasting using a hybrid model based on two-stage decomposition technique and AdaBoost-extreme learning machine. *Energy Conversion and Management*, *153*, 589–602.

Rodríguez-Fdez, I., Canosa, A., Mucientes, M., & Bugarń, A. (2015). STAC: a web platform for the comparison of algorithms using statistical tests. In *Proceedings of the 2015 IEEE international conference on fuzzy systems* (pp. 2–5). Istanbul, Turkey.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117.

Shang, C., Yang, F., Huang, D. X., & Lyu, W. (2014). Data-driven soft sensor development based on deep learning technique. *Journal of Process Control*, *24*, 223–233.

Shao, W., Ge, Z., & Song, Z. (2018). Soft-sensor development for processes with multiple operating modes based on semisupervised Gaussian mixture regression. *IEEE Transactions on Control Systems Technology*, http://dx.doi.org/10.1109/TCST.2018.2856845.

Shao, W., Ge, Z., & Song, Z. (2019). Semi-supervised mixture of latent factor analysis models with application to online key variable estimation. *Control Engineering Practice*, *84*, 32–47.

Shao, W., Yao, L., Ge, Z., & Song, Z. (2019). Parallel computing and SGD-based DPMM for soft sensor development with large-scale semisupervised data. *IEEE Transactions on Industrial Electronics*, *66*(8), 6362–6373.

Soria-Olivas, E., Góez-Sanchis, J., Martín, J. D., Vila-Francés, J., et al. (2011). BELM: Bayesian extreme learning machine. *IEEE Transactions on Neural Networks Learing Systems*, *22*(3), 505–509.

Souza, F., & Araújo, R. (2014). Mixture of partial least squares experts and application in prediction settings with multiple operating modes. *Chemometrics and Intelligent Laboratory Systems*, *130*(2), 192–202.

Sun, K., Zhang, J., Zhang, C., & Hu, J. (2017). Generalized extreme learning machine autoencoder and a new deep neural network. *Neurocomputing*, *230*, 374–381.

Tang, J., Deng, C., & Huang, G. B. (2016). Extreme learning machine for multilayer perception. *IEEE Transactions on Neural Networks Learing Systems*, *37*(4), 809–821.

Yan, W., Guo, P., Tian, Y., & Gao, J. (2016). A framework and modeling method of data-driven soft sensors based on semisupervised Gaussian Regression. *Industrial and Engineering Chemistry Research*, *55*(27), 7394–7401.

Yan, W., Tang, D., & Lin, Y. (2017). A data-driven soft sensor modeling method based on deep learning and its application'. *IEEE Transactions on Industrial Electronics*, *64*(5), 4237–4245.

Yao, L., & Ge, Z. (2018). Deep learning of semisupervised process data with hierarchical extreme learning machine and soft sensor application. *IEEE Transactions on Industrial Electronics*, *65*(2), 1490–1498.

Yi, Y., Qiao, S., Zhou, W., et al. (2018). Adaptive multiple graph regularized semi-supervised extreme learning machine. *Soft Computing*, *22*, 3545–3562.

Yuan, X., Huang, B., Wang, Y., Yang, C., & Gui, W. (2018). Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE. *IEEE Transactions on Industrial Electronics*, *14*(7), 3235–3243.

Zhang, M., Liu, X., & Zhang, Z. (2016). A soft sensor for industrial melt index prediction based on evolutionary extreme learning machine. *Chinese Journal of Chemical Engineering*, *24*, 1013–1019.

Zhang, K., & Luo, M. (2015). Outlier-robust extreme learning machine for regression problem. *Neurocomputing*, *151*, 1519–1527.

Zorzi, M., & Chiuso, A. (2018). The harmonic analysis of kernel functions. *Automatica*, *94*, 125–137.