Journal of Process Control 78 (2019) 13–33

Contents lists available at ScienceDirect

Journal of Process Control

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

Scalable learning and probabilistic analytics of industrial big data based on parameter server: Framework, methods and applications

Le Yao, Zhiqiang Ge*

State Key Laboratory of Industrial Control Technology, Institute of Industrial Process Control, College of Control Science and Engineering, Zhejiang University, Hangzhou, 310027, PR China

ARTICLE INFO

Article history:
Received 5 August 2018
Received in revised form 24 February 2019
Accepted 25 March 2019
Available online 16 April 2019

Keywords:
Distributed parallel modeling
Stochastic Variational Inference
Parameter Server
Process monitoring
Quality prediction
Big data

ABSTRACT

With the ever increasing scale of industrial data, the computational burden for process modeling and analytics has becoming tremendous, particularly for large-scale processes. In this paper, a distributed parallel probabilistic learning framework based on scalable Parameter Server (PS) architecture is proposed for big process data. Under this framework, the traditional Variational Inference (VI) probabilistic model can be transformed to a scalable form through the Stochastic Variational Inference (SVI) algorithm, which can be further deployed on the PS architecture to make a distributed and parallel model. As an example, the traditional Variational Inference Mixture Factor Analysis (VIMFA) model is converted to the SVI-MFA model and deployed on the PS architecture for process big data modeling. Then it is utilized for process monitoring and quality prediction applications. A numerical case is first generated to validate the feasibility and efficiency of the SVI-MFA modeling algorithm, and then a TE benchmark process and a real Methanation Unit process demonstrate the effectiveness of the proposed framework for big process data modeling and analytics.

1. Introduction

Due to the progress of advanced Distributed Control Systems (DCS) and new measurement technologies, recording and collecting data from industrial processes become more easily in the past decades [1]. At the same time, theoretical and technical supports of data mining and learning technologies have also been provided for industrial processes [2–4], which all shows that data-driven methods are more flexible and can be easily implemented in more complicated industrial processes in contrast to traditional first-principle-based modeling methods. Through mining and analyzing the information from the process data, various types of data-based models can be constructed for industrial applications [5], e.g. process monitoring, mode clustering, soft sensing of key performance index (KPI).

From the perspective of data-driven modeling, various types of data-based models should be constructed to cope with different problems that exist in the process data. For instance, latent variable models such as PCA [6] and Factor Analysis (FA) [7] are able to handle the high-dimensional datasets, nonlinear modeling methods like Kernel PCA [8], Support Vector Machine (SVM) [9] and Gaussian Process Regression (GPR) [10] models are constructed to describe the nonlinear relationships, mixture models or adaptive models are efficient solutions in coping with the multi-mode process modeling [11]. Moreover, for the noisy industrial process data, probabilistic models have been widely paid attention by the researchers [12,13]. In the view of probability, inference and decision based on data analysis can be much more appropriate. Firstly, model parameters in a probabilistic form can be easily solved through an efficient Expectation Maximization (EM) algorithm or Variational Inference (VI) algorithm. Secondly, the missing data or the unlabeled data can be handled under the probabilistic framework for semi-supervised modeling. Thirdly, mixture models like Mixture Factor Analysis (MFA) [14], Mixture Probabilistic Principal Component Analysis (MPPCA) [6] and Gaussian Mixture Model (GMM) [15] can be easily built for multi-mode or nonlinear process modeling.

Since large amounts of process data have been collected through the advanced technologies, modern industries are stepping into the big data era [16], characterized by several V's, i.e. volume, velocity, variety, veracity, etc. The industrial process data can be called Big Data since it is hard to perform capture, analysis and visualization [17]. Besides, it is kind of big data with high volume and low-density value, which would help enterprises to sufficiently understand the process. Unfortunately, traditional methods of data storage, centralized data pre-processing and modeling are facing a

* Corresponding author.
E-mail address: gezhiqiang@zju.edu.cn (Z. Ge).

https://doi.org/10.1016/j.jprocont.2019.03.017
0959-1524/

predicament with low computation efficiency or out-of-memory. To provide a fast and low-cost solution, a parallel and distributed big data modeling platform that takes full advantage of the distributed computing resources and turns the heavy computation burden into parallelized small-scale processing is needed. Besides, traditional data-driven methods should be turned into a scalable form to handle the increasing data scale.

In recent years, big data modeling technologies and platforms are getting mature. As the most popular distributed and parallel computing system, MapReduce-based Hadoop provides a programming framework and file storage system for big data on a cluster of low-performance computer nodes to deal with big data applications that standalone machine cannot do [18–20]. Recently, Hadoop-based PCA [21] and GMM [22] algorithms have also been developed for monitoring with big process data. However, they conduct all the computing on local disks, which greatly restricts the speed of modeling, and most of the mainstream machine learning algorithms with iterations cannot be efficiently conducted on MapReduce-based Hadoop platform. To solve this problem, an efficient, scalable and distributed computing framework called Parameter Server (PS) has been proposed by researchers to handle large-scale of modeling with huge datasets [23]. The popular deep learning platforms like Tensorflow [24] and MXNet [25] were developed based on PS framework to realize the distributed and parallel computing on Graphic Processing Unit (GPU). PS is designed based on the Stochastic Gradient Descend (SGD) algorithm [26,27], which is the most commonly used optimization procedure for training machine learning or deep learning models [28]. However, distributed and parallel model based on PS framework for flexibly and efficiently processing of industrial big data has not been proposed for applications like monitoring and quality prediction. Therefore, the motivation of the work is to establish a scalable distributed and parallel probabilistic modeling framework under the Parameter Server for industrial big process data. In practice, mixture probabilistic models are more compatible for long-term big process data modeling, and the Variational Inference (VI) algorithm [29] can be used to determine the parameters and avoid overfitting while training models. To make the VI-based probabilistic model suitable for big datasets, the VI algorithm can be turned into the Stochastic Variational Inference (SVI) through SGD algorithm while maximizing the Evidence of Lower Bound (ELBO) [30].

In this paper, a distributed parallel probabilistic modeling framework is proposed for big process data based on the SVI algorithm and Parameter Server architecture, and applications of process monitoring and quality prediction are further developed in this framework. The traditional VI-based mixture probabilistic model can be transformed into a scalable form through the SVI algorithm under this framework, where only one or a mini-batch of samples are randomly selected to update parameters in each iteration according to the SGD algorithm. Then, the SVI-based probabilistic model is deployed on the Parameter Server. As an example, the traditional Variational Inference Mixture Factor Analysis (VIMFA) model is transformed into the SVI-MFA model and deployed on the PS framework for big process data modeling. The established SVI-MFA model is then utilized for process monitoring and quality prediction applications. It should be mentioned that the proposed distributed parallel modeling framework can be applied for many other process fields, like data pattern recognition, data visualization, optimization and etc. A numerical case, a TE benchmark process and a real Methanation Unit process with large-scale datasets will be utilized to validate the feasibility and effectiveness of the proposed modeling framework.

The remainder of this paper is organized as follows. In Section 2, the SVI algorithm and the architecture of PS is presented according to reference [30] and [23]. Then, the distributed and parallel probabilistic modeling framework based on the PS is proposed in Section 2.3, based on which the SVI-MFA model is proposed as an example in Section 3. In Section 4, the proposed SVI-MFA model is utilized for the applications of process monitoring and quality prediction. Next to that, a numerical example, a TE benchmark process and a real industrial process case are provided for performance evaluation. Finally, conclusions are made.

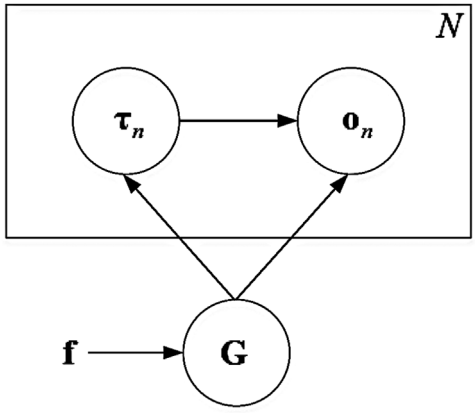


Fig. 1. Probabilistic graph of mixture model.

2. Distributed parallel probabilistic modeling framework

2.1. Stochastic variational inference for mixture probabilistic model

Before introducing the SVI-based mixture model, the mean-field Variational Inference (VI) algorithm for mixture probabilistic model should be revisited [29]. Generally, four kinds of variables are required in the mixture model: data samples $\mathbf{O} = \mathbf{o}_{1:N}$; global hidden variables $\mathbf{G}$; local hidden variables $\boldsymbol{\Gamma} = \boldsymbol{\tau}_{1:N}$, each of which is a collection of J variables $\boldsymbol{\Gamma}_n = \boldsymbol{\tau}_{n,1:J}$; hyper-parameters $\mathbf{f}$. N denotes the number of samples. The global variables represent the variables with overall meanings for the entire samples, such as the mixture proportion of the components, the parameters of the components and etc. The local variables are those variables related to each sample, such as the latent component label of each sample, the latent variables for each sample and etc. A probabilistic graph of mixture model is presented in Fig. 1.

According to Fig. 1, the joint distribution of the variables can be factorized into the following form:

$$p(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{G}|\mathbf{f}) = p(\mathbf{G}|\mathbf{f}) \prod_{n=1}^{N} p(\mathbf{o}_n, \boldsymbol{\tau}_n|\mathbf{G}) \tag{1}$$

The goal of learning parameters of the mixture probabilistic model is to infer the posterior distribution of hidden variables given observations, $p(\mathbf{G}, \boldsymbol{\Gamma}|\mathbf{O})$. For the convenience of calculation, it is assumed that the complete conditional distribution of hidden variables follows the exponential family distribution:

$$p(\mathbf{G}|\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{f}) = \mathbf{h}(\mathbf{G}) \exp\left\{\boldsymbol{\eta}_g(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{f})^T \mathbf{t}(\mathbf{G}) - \mathbf{a}_g\left(\boldsymbol{\eta}_g(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{f})\right)\right\} \tag{2}$$

$$p\left(\tau_{n,j}|\mathbf{o}_n, \boldsymbol{\tau}_{n,-j}, \mathbf{G}\right) = \mathbf{h}\left(\tau_{n,j}\right) \exp\left\{\boldsymbol{\eta}_\ell\left(\mathbf{o}_n, \boldsymbol{\tau}_{n,-j}, \mathbf{G}\right)^T \mathbf{t}\left(\tau_{n,j}\right) - \mathbf{a}_\ell\left(\boldsymbol{\eta}_\ell\left(\mathbf{o}_n, \boldsymbol{\tau}_{n,-j}, \mathbf{G}\right)\right)\right\} \tag{3}$$

where $\mathbf{h}(\bullet)$ and $\mathbf{a}(\bullet)$ are respectively the base measure and log-normalizer, $\boldsymbol{\eta}(\bullet)$ and $\mathbf{t}(\bullet)$ respectively denotes the natural parameter and sufficient statistics [30]. Base measure, log-normalizer, natural parameter and sufficient statistics are the elements in exponential family distributions [30], and the detailed

introductions can be seen in Appendix. The subscripts on the natural parameters and log-normalizers indicate complete conditionals for local or global hidden variables.

The VI-based algorithm is utilized to approximate posterior distributions through variational distributions $q(\mathbf{G}, \boldsymbol{\Gamma})$. And the variational distributions $q(\mathbf{G}, \boldsymbol{\Gamma})$ are obtained through maximizing the Evidence of Lower Bound (ELBO) of mixture model:

$$ELBO: L(q) \triangleq E_q[\log p(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{G})] - E_q[\log q(\boldsymbol{\Gamma}, \mathbf{G})] \tag{4}$$

where $E_q(\bullet)$ represents the expectation under the $q(\bullet)$ distribution. In mean-field theory, each hidden variable is independent and governed by its own parameter:

$$q(\boldsymbol{\Gamma}, \mathbf{G}) = q\left(\mathbf{G}|\boldsymbol{\delta}_{\mathbf{G}}\right) \prod_{n=1}^{N} \prod_{j=1}^{J} q\left(\tau_{n,j}|\boldsymbol{\delta}_{\tau_{n,j}}\right) \tag{5}$$

where global hyper-parameter $\boldsymbol{\delta}_{\mathbf{G}}$ governs the global hidden variables and local parameter $\boldsymbol{\delta}_{\tau_{n,j}}$ governs the local hidden variables for the nth data sample. According to reference [30], the variational distributions $q\left(\mathbf{G}|\boldsymbol{\delta}_{\mathbf{G}}\right)$ and $q\left(\tau_{n,j}|\boldsymbol{\delta}_{\tau_{n,j}}\right)$ are also set to be the same exponential family as Eqs. (2) and (3), expressed as follows:

$$q\left(\mathbf{G}|\boldsymbol{\delta}_{\mathbf{G}}\right) = \mathbf{h}(\mathbf{G}) \exp\left\{\boldsymbol{\delta}_{\mathbf{G}}^T \mathbf{t}(\mathbf{G}) - \mathbf{a}_g\left(\boldsymbol{\delta}_{\mathbf{G}}\right)\right\} \tag{6}$$

$$q\left(\tau_{n,j}|\boldsymbol{\delta}_{\tau_{n,j}}\right) = \mathbf{h}\left(\tau_{n,j}\right) \exp\left(\boldsymbol{\delta}_{\tau_{n,j}}^T \mathbf{t}\left(\tau_{n,j}\right) - \mathbf{a}_\ell\left(\boldsymbol{\delta}_{\tau_{n,j}}\right)\right) \tag{7}$$

where the hyper-parameters $\boldsymbol{\delta}_{\mathbf{G}}$ and $\boldsymbol{\delta}_{\tau_{n,j}}$ are the natural parameters of the variational distributions.

To obtain hyper-parameters of variational distributions, the gradient is taken to Eq. (4) for $\boldsymbol{\delta}_{\mathbf{G}}$:

$$\nabla_{\boldsymbol{\delta}_{\mathbf{G}}} L = \nabla^2_{\boldsymbol{\delta}_{\mathbf{G}}} \mathbf{a}_g\left(\boldsymbol{\delta}_{\mathbf{G}}\right)\left(E_q\left[\boldsymbol{\eta}_g(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{f})\right] - \boldsymbol{\delta}_{\mathbf{G}}\right) \tag{8}$$

and set the gradient to zero, the updating form for global natural parameters can be obtained as follows:

$$\boldsymbol{\delta}_{\mathbf{G}} = E_q\left[\boldsymbol{\eta}_g(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{f})\right] = \mathbf{f} + E_{q(\boldsymbol{\tau}|\boldsymbol{\delta}_{\boldsymbol{\tau}})}\left(\sum_{n=1}^{N} \mathbf{t}(\mathbf{o}_n, \boldsymbol{\tau}_n)\right) \tag{9}$$

where $\mathbf{t}(\mathbf{o}_n, \boldsymbol{\tau}_n)$ is the sufficient statistics of all local variables and observations. For local variational natural parameters, the gradient form is the same as the global one:

$$\nabla_{\delta_{\tau_{n,j}}} L = \nabla^2_{\delta_{\tau_{n,j}}} \mathbf{a}_\ell\left(\delta_{\tau_{n,j}}\right)\left(E_q\left[\boldsymbol{\eta}_\ell\left(\mathbf{o}_n, \boldsymbol{\tau}_{n,-j}, \mathbf{G}\right)\right] - \delta_{\tau_{n,j}}\right) \tag{10}$$

Then the local variational natural parameters can be obtained:

$$\delta_{\tau_{n,j}} = E_q\left[\boldsymbol{\eta}_\ell\left(\mathbf{o}_n, \boldsymbol{\tau}_{n,-j}, \mathbf{G}\right)\right] \tag{11}$$

The global and local variational natural parameters iterates until convergence.

In Eq. (9), the updating of global natural parameters are depended on all the local variational parameters, and the updating of local parameter (Eq. (11)) only depends on the global variables and the current observation. However, since the local updating should be conducted on each observation, it is inefficiency for big data. In this part, the SVI algorithm is revisited to introduce an efficient algorithm for probabilistic modeling with big process data.

SVI algorithm is the stochastic optimizing version of traditional VI algorithm, which has been efficiently used in many machine learning cases with large-scale datasets. The Robbins-Monro algorithm [31] is utilized to optimize the ELBO in a gradient-descending form. Besides, SVI uses the natural gradients [30] of the Riemannian metric to replace the standard gradients of the Euclidean metric. The natural gradient is computed by pre-multiplying the gradient by the inverse of Riemannian metric $\Psi\left(\boldsymbol{\delta}_{\mathbf{G}}\right)^{-1}$:

$$\hat{\nabla}_{\boldsymbol{\delta}_{\mathbf{G}}} L \triangleq \Psi\left(\boldsymbol{\delta}_{\mathbf{G}}\right)^{-1} \nabla_{\boldsymbol{\delta}_{\mathbf{G}}} L \tag{12}$$

where $\Psi\left(\boldsymbol{\delta}_{\mathbf{G}}\right)$ is the Fisher information matrix of $q\left(\boldsymbol{\delta}_{\mathbf{G}}\right)$ that equals the second derivative of the log-normalizer:

$$\Psi\left(\boldsymbol{\delta}_{\mathbf{G}}\right) = \nabla^2_{\boldsymbol{\delta}_{\mathbf{G}}} \mathbf{a}_g\left(\boldsymbol{\delta}_{\mathbf{G}}\right) \tag{13}$$

Then the gradient of Eqs. (8) and (10) are transformed to the following simple forms:

$$\hat{\nabla}_{\boldsymbol{\delta}_{\mathbf{G}}} L = E_q\left[\boldsymbol{\eta}_g(\mathbf{O}, \boldsymbol{\Gamma}, \mathbf{f})\right] - \boldsymbol{\delta}_{\mathbf{G}} \tag{14}$$

$$\hat{\nabla}_{\delta_{\tau_{n,j}}} L = E_q\left[\boldsymbol{\eta}_\ell\left(\mathbf{o}_n, \boldsymbol{\tau}_{n,-j}, \mathbf{G}\right)\right] - \delta_{\tau_{n,j}} \tag{15}$$

SVI algorithm optimizes ELBO by subsampling the data to form noisy estimates of the natural gradient. It considers a variable that chooses an index of the data uniformly at random, $I \sim Unif(1, \ldots, N)$. Define the sub-sampled ELBO to be $L_I(q)$, then the natural gradient of $L_I(q)$ for $\boldsymbol{\delta}_{\mathbf{G}}$ is a noisy but unbiased estimate of the natural gradient of $L(q)$ for $\boldsymbol{\delta}_{\mathbf{G}}$. It is supposed that the ith data is observed N times, then the natural gradient of Eq. (14) can be derived as follows:

$$\hat{\nabla}_{\boldsymbol{\delta}_{\mathbf{G}}} L_I = E_q\left[\boldsymbol{\eta}_g\left(\mathbf{o}_i^{(N)}, \boldsymbol{\tau}_i^{(N)}, \mathbf{f}\right)\right] - \boldsymbol{\delta}_{\mathbf{G}} \tag{16}$$

where $\left\{\mathbf{o}_i^{(N)}, \boldsymbol{\tau}_i^{(N)}\right\}$ is a dataset formed by N replicas of observation $\mathbf{o}_n$ and hidden variables $\boldsymbol{\tau}_N$. Finally, Eq. (16) is used in a Robbins-Monro algorithm [31] to update the variational natural parameters:

$$\begin{aligned} \boldsymbol{\delta}_G^{(T)} &= \boldsymbol{\delta}_G^{(T-1)} + \rho^{(T)} \bullet \hat{\nabla}_{\boldsymbol{\delta}_{\mathbf{G}}} L_I = \boldsymbol{\delta}_G^{(T-1)} + \rho^{(T)}\left(\hat{\boldsymbol{\delta}}_{\mathbf{G}}^{(T)} - \boldsymbol{\delta}_G^{(T-1)}\right) \\ &= \left(1 - \rho^{(T)}\right)\boldsymbol{\delta}_G^{(T-1)} + \rho^{(T)}\hat{\boldsymbol{\delta}}_{\mathbf{G}}^{(T)} \end{aligned} \tag{17}$$

where $\hat{\boldsymbol{\delta}}_{\mathbf{G}}^{(T)}$ denotes an intermediate parameter at the Tth iteration, defined as follows:

$$\hat{\boldsymbol{\delta}}_{\mathbf{G}}^{(T)} \triangleq E_q\left[\boldsymbol{\eta}_g\left(\mathbf{o}_i^{(N)}, \boldsymbol{\tau}_i^{(N)}, \mathbf{f}\right)\right] = \mathbf{f} + N \bullet E_{\delta_{\tau_I}}\left[\left(\mathbf{t}(\mathbf{o}_n, \boldsymbol{\tau}_n), 1\right)\right] \tag{18}$$

Besides, $\rho^{(T)} = (T+1)^{-\kappa}$ is the learning rate at iteration T, and the forgetting rate $\kappa \in (0.5, 1]$.

2.2. Architecture of Parameter Server

Parameter Server is a distributed and parallel computing framework built on stochastic optimization algorithms for machine learning with big data [23]. SGD algorithm is perhaps the most commonly used optimization procedure for training machine learning or deep learning models, which is derived from the basic Gradient Descent (GD) algorithm that takes all the training samples in the computing of gradients. However, SGD merely randomly takes a batch of samples to calculate the gradient of parameters in each updating procedures, which can also obtain the converged parameters with the same accuracy as GD algorithm. It should be mentioned that the time complexity of SGD-based algorithm is much smaller than GD-based algorithm. In SGD algorithm, model parameters are set to initial values, and then they are updated along the steepest descending direction of gradients iteratively. The gradients are calculated through maximizing the objective function with the randomly selected batch of dataset and the current parameters. When the objective function reaches the maximum, the parameters obtain converged values. The updating formula of parameters for the basic SGD algorithm is given as follows:

$$\boldsymbol{\theta}^{(T)} = \boldsymbol{\theta}^{(T-1)} + \rho^{(T)} \nabla \boldsymbol{\theta}^{(T-1)} \tag{19}$$

where $\boldsymbol{\theta}^{(T)}$ represents the updated parameters at the Tth iteration, $\nabla \boldsymbol{\theta}^{(T-1)}$ is the gradient calculated by the parameters at the $T-1$th iteration and randomly selected data samples. Besides, $\rho^{(T)}$ denotes the learning rate that controls the speed of convergence.

Parameter Server skillfully takes advantage of the updating formulation of Eq. (19) to deploy the modeling procedures on a

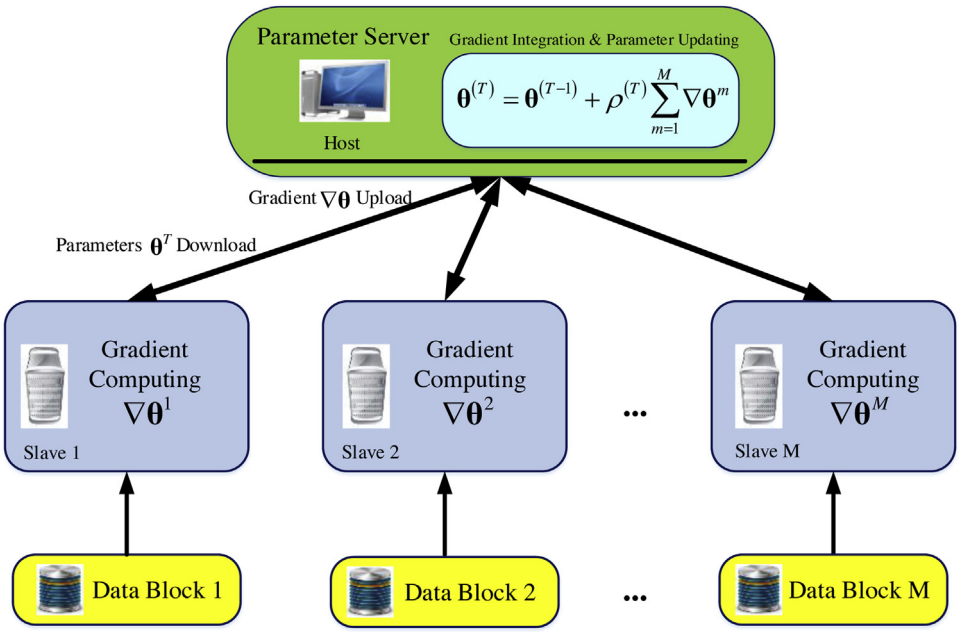


Fig. 2. Architecture of Parameter Server framework.

distributed and parallel form. As an illustration, the architecture of Parameter Server framework is displayed in Fig. 2. It consists of one server node and several slave nodes. The server node is in charge of the unified scheduling of the SGD-based algorithm. The slave nodes are responsible to execute the computation burden of gradients for the dataset in a distributed manner. In each iteration, the randomly selected training data batch is divided into several data blocks on the slave nodes and the gradients of parameters are separately calculated with these data blocks. Then the gradients are uploaded and integrated at the server node to update the parameters. In the next iteration, the parameters will be downloaded to slave nodes to calculate new gradients. The upload and download operations conduct until parameters converge.

While implementing the SGD-based algorithm on the Parameter Server, three execution parameters should be determined: batch-size, iteration number and epoch number. Batch-size represents the number of randomly selected samples in each iteration. Then the iteration number equals the number of batches. Besides, an epoch is defined as all batches finished iterations once. Commonly, a number of epochs are conducted for the training algorithms to traverse the whole dataset several times for a stable result. Totally, the global iteration number equals epoch number multiplies the iteration number. It can be seen from Fig. 2 that SGD-based algorithms can be effectively conducted on the parameter framework to obtain a stable result. This parallel and distributed modeling scheme takes full advantages of the distributed computing resources and turns the heavy computation burden into parallelized small-scale processing, which shows a great potential for modeling with big data.

2.3. Distributed parallel probabilistic modeling framework based on PS

In order to develop the distributed parallel probabilistic modeling framework, the SVI-based probabilistic model should be deployed on the Parameter Server architecture. An architecture of distributed and parallel probabilistic modeling framework is displayed in Fig. 3. While training the SVI-based probabilistic model on the PS platform, the mini-batch SVI algorithm is utilized. The randomly selected mini-batch is firstly distributed to M local blocks, which are then transferred to M local slave computing nodes. According to Section 2.1, the variational parameters are divided into two parts: the local variational parameters τ and the global variational parameters G. The global variational parameters are initialized on the Host (PS) and also downloaded to local slave nodes to calculate the local variational parameters for each sample. For SVI-based model, the intermediate global variational parameters $\hat{\boldsymbol{\delta}}_G$ can be calculated by the natural gradients on the local slave nodes. Then $\hat{\boldsymbol{\delta}}_G$ is uploaded to the host for the global gradient integration and parameters updating. Thus far, once iteration is completed. In the next iteration, the updated global parameters are downloaded to local slave nodes once again. Also, a new batch of randomly selected samples are transferred to the slave nodes for the local computing of variational parameters and intermediate parameters. The iterating cycle conducts until the iteration number ends.

3. Distributed and parallel MFA model on parameter server

In this section, the VI-MFA is taken as an example to be transformed to the SVI-MFA, and then deployed on the Parameter Server.

3.1. Variational inference mixture factor analysis

The basic Factor Analysis (FA) model describes a $M \times N$ data matrix $\mathbf{X}$ as a linear function of a $D \times N$ dimensional matrix of latent factors $\mathbf{t}$ plus some noise $\boldsymbol{\Phi}$:

$$\mathbf{X} = \mathbf{W}t + \boldsymbol{\mu} + \boldsymbol{\Phi} \tag{20}$$

where $\mathbf{W} \in R^{M\times D}$ is the loading matrix and $\boldsymbol{\mu}$ is the mean value for data. The latent factors are assumed to be standard-normal distributed:

$$p(\mathbf{t}) = \prod_{n=1}^{N} N(\mathbf{t}_n|0, \mathbf{I}_D) \tag{21}$$

$\boldsymbol{\Phi}$ is a normally distributed noise matrix with zero mean and diagonal covariance matrix $diag(\boldsymbol{\tau})^{-1}$, where $\boldsymbol{\tau} \in R^{M\times 1}$ is the precision

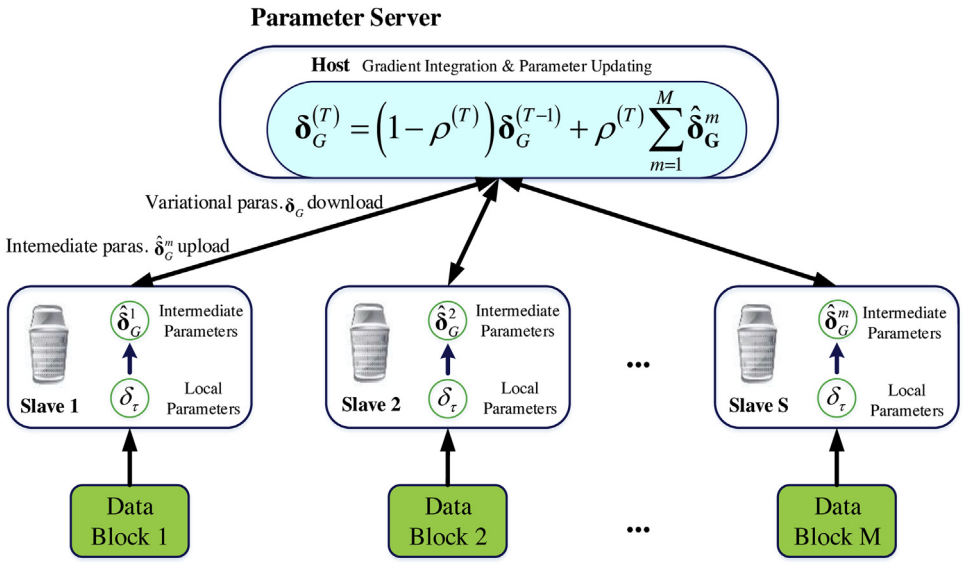


Fig. 3. Architecture of distributed and parallel probabilistic modeling framework.

vector of **X** with different values. Given the latent factors **t**, **X** is hence normally distributed as follows:

$$p(\mathbf{X}|\mathbf{W},\mathbf{t},\boldsymbol{\mu},\boldsymbol{\tau}) = \prod_{n=1}^{N}\prod_{m=1}^{M} N\left(\mathbf{X}_n^m|\mathbf{W}_m\mathbf{t}_n+\boldsymbol{\mu}_m,\boldsymbol{\tau}_m^{-1}\right) \tag{22}$$

In the mixture factor analysis (MFA) model, **X** belongs to one of K independent factor analyzers with different factor loading matrices $\mathbf{W}=\left\{\mathbf{W}^k\right\}_{k=1}^K$ and mean vectors $\boldsymbol{\mu}=\left\{\boldsymbol{\mu}^k\right\}_{k=1}^K$:

$$\mathbf{X} = \mathbf{W}^k\mathbf{t}+\boldsymbol{\mu}^k+\boldsymbol{\Phi} \tag{23}$$

Latent indicator $\mathbf{C}=\left\{\mathbf{C}_n\right\}_{n=1}^N$ is introduced to represent which analyzer $\mathbf{X}_n$ belongs to. Then the data from a MFA model can be described in a mixture distribution as follows:

$$p(\mathbf{X}_n) = \sum_{k=1}^{K}\boldsymbol{\pi}^k p(\mathbf{X}_n|\mathbf{C}_n=k) = \sum_{k=1}^{K}\boldsymbol{\pi}^k p\left(\mathbf{X}_n|\mathbf{W}^k\mathbf{t}_n^k+\boldsymbol{\mu}^k, diag(\boldsymbol{\tau})^{-1}\right) \tag{24}$$

where $\boldsymbol{\pi}^k$ is the mixture proportions for analyzers.

The model parameters $\boldsymbol{\Theta}=\left\{\boldsymbol{\pi},\mathbf{W},\boldsymbol{\mu},\boldsymbol{\tau}\right\}$ can be estimated by Maximum Likelihood Estimation (MLE) or Bayesian Inference (BI). MLE via EM algorithm is fast but subject to over-fitting, since it does not take into account model complexity. BI prevents overfitting by imposing priors on the model parameters, then approximated inference of them can be obtained through a Variational Inference (VI) algorithm.

While conducting the VI algorithm for MFA model, the parameters are seen as random variables and proper priors are selected for them. The prior over mixture proportion $\boldsymbol{\pi}$ is a symmetric Dirichlet distribution with hyper-parameter $\boldsymbol{\gamma}$:

$$p(\boldsymbol{\pi}) = Dir(\boldsymbol{\pi}|\boldsymbol{\gamma}) \tag{25}$$

Accordingly, the prior for latent indicators **C** is set the Multinomial distribution:

$$p(\mathbf{C}) = \prod_{n=1}^{N} Multi(\mathbf{C}_n|\boldsymbol{\pi}) \tag{26}$$

Each factor loading matrix $\mathbf{W}^k$ factorizes by row m, which are normally distributed with mean zero and diagonal matrix $\left[diag\left(\boldsymbol{\alpha}^k\right)\right]^{-1}$. Thus, the prior for loading matrices can be given as follows:

$$p(\mathbf{W}) = \prod_{k=1}^{K}\prod_{m=1}^{M} N\left(\mathbf{W}_m^k|0,\left[diag\left(\boldsymbol{\alpha}^k\right)\right]^{-1}\right) \tag{27}$$

For the precision vector $\boldsymbol{\alpha}$, Gamma distribution with hyper-parameters **a** and **b** are set as the prior:

$$p(\boldsymbol{\alpha}) = \prod_{k=1}^{K}\prod_{d=1}^{D} Ga\left(\boldsymbol{\alpha}_d^k|\mathbf{a}_d^k,\mathbf{b}_d^k\right) \tag{28}$$

The mean vector $\boldsymbol{\mu}^k$ is normally distributed for each row with zero mean and covariance $\left(\boldsymbol{\beta}^k\right)^{-1}$:

$$p(\boldsymbol{\mu}) = \prod_{k=1}^{K}\prod_{m=1}^{M} N\left(\boldsymbol{\mu}_m^k|0,\left(\boldsymbol{\beta}^k\right)^{-1}\right) \tag{29}$$

Besides, for the precision vector $\boldsymbol{\tau}$, prior of Gamma distribution with hyper-parameters **c** and **d** is given as follows:

$$p(\boldsymbol{\tau}) = \prod_{m=1}^{M} Ga(\boldsymbol{\tau}_m|\mathbf{c}_m,\mathbf{d}_m) \tag{30}$$

Totally, random variables are defined as $\mathbf{T}=\left\{\boldsymbol{\pi},\mathbf{W},\boldsymbol{\alpha},\boldsymbol{\mu},\boldsymbol{\tau},\mathbf{t},\mathbf{C}\right\}$, and the probabilistic graph of VI-MFA model is displayed in Fig. 4.

In VI algorithm, the posterior distributions of random variables are approximated by the variational distributions. $q(\mathbf{T}|\boldsymbol{\lambda})$., where $\boldsymbol{\lambda}$ represents the variational hyper-parameters. It should be mentioned that the variational inference follows the same distribution as the priors. The mean-field approximating theory is utilized to derive tractable solution of posterior distributions as follows:

$$\begin{aligned} &q(\mathbf{T}|\boldsymbol{\lambda}) = q\left(\mathbf{W}|\boldsymbol{\lambda}_{\boldsymbol{\xi}},\boldsymbol{\lambda}_{\mathbf{V}}\right) q(\boldsymbol{\alpha}|\boldsymbol{\lambda}_{\mathbf{a}},\boldsymbol{\lambda}_{\mathbf{b}}) q\left(\boldsymbol{\pi}|\boldsymbol{\lambda}_{\boldsymbol{\gamma}}\right) \\ &q\left(\boldsymbol{\mu}|\boldsymbol{\lambda}_{\boldsymbol{\mu}},\boldsymbol{\lambda}_{\boldsymbol{\beta}}\right) q(\boldsymbol{\tau}|\boldsymbol{\lambda}_{\mathbf{c}},\boldsymbol{\lambda}_{\mathbf{d}}) \\ &q(\mathbf{t}|\mathbf{C},\boldsymbol{\lambda}_{\boldsymbol{v}},\boldsymbol{\lambda}_{\boldsymbol{\Sigma}}) q(\mathbf{C}|\boldsymbol{\lambda}_{\mathbf{C}}) \end{aligned} \tag{31}$$

Through maximizing the Evidence of Lower Bound (ELBO) of VI algorithm, the q distributions approximately equal the posteriors and these hyper-parameters can be obtained. The ELBO is firstly

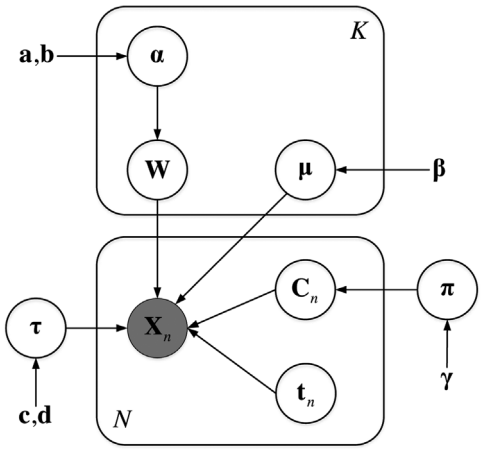


Fig. 4. Probabilistic graph of VI-MFA.

derived as follows:

$$
\begin{aligned}
&L(\boldsymbol{\lambda}) = E_{q(\mathbf{T}|\boldsymbol{\lambda})} \ln p(\mathbf{T}, \mathbf{X}) - E_{q(\mathbf{T}|\boldsymbol{\lambda})} \ln q(\mathbf{T}) \\
&= \sum_{n=1}^{N} E_{q(\mathbf{T}|\boldsymbol{\lambda})} \left\{ \begin{array}{l} \ln p(\mathbf{t}_n|0, \mathbf{I}_D) + \sum_{m=1}^{M}\sum_{k=1}^{K} I(\mathbf{C}_n = k) \ln p\left(\mathbf{X}_n^m | \mathbf{W}_m^k \mathbf{t}_n^k + \boldsymbol{\mu}_m^k, \tau_m^{-1}\right) \\ + \ln p(\mathbf{C}_n|\boldsymbol{\pi}) - \sum_{n=1}^{K} I(\mathbf{C}_n = k) \ln q\left(\mathbf{t}_n^k | \boldsymbol{\lambda}_{\boldsymbol{\nu}_n}^k, \boldsymbol{\lambda}_{\boldsymbol{\Sigma}_n}^k\right) - \ln q(\mathbf{C}_n|\boldsymbol{\lambda}_{\mathbf{C}_n}) \end{array} \right\} \\
&+ \sum_{k=1}^{K} E_{q(\mathbf{T}|\boldsymbol{\lambda})} \left\{ \begin{array}{l} + \sum_{m=1}^{M} \ln p\left(\mathbf{W}_m^k|0, \left[diag\left(\boldsymbol{\alpha}^k\right)\right]^{-1}\right) + \ln p\left(\boldsymbol{\pi}^k|\boldsymbol{\gamma}^k\right) \\ + \sum_{d=1}^{D} \ln p\left(\boldsymbol{\alpha}_d^k|\mathbf{a}_d^k, \mathbf{b}_d^k\right) + \sum_{m=1}^{M} \ln p\left(\boldsymbol{\mu}_m^k|0, \left(\boldsymbol{\beta}^k\right)^{-1}\right) \\ - \sum_{m=1}^{M} \ln q\left(\boldsymbol{\mu}_m^k|\boldsymbol{\lambda}_{\boldsymbol{\mu}_m}^k, \boldsymbol{\lambda}_{\boldsymbol{\beta}_m}^k\right) - \sum_{d=1}^{D} \ln q\left(\boldsymbol{\alpha}_d^k|\boldsymbol{\lambda}_{\mathbf{a}_d}^k, \boldsymbol{\lambda}_{\mathbf{b}_d}^k\right) \\ - \sum_{m=1}^{M} \ln q\left(\mathbf{W}_m^k|\boldsymbol{\lambda}_{\boldsymbol{\xi}_m}^k, \boldsymbol{\lambda}_{\mathbf{V}_m}^k\right) - \sum_{k=1}^{K} \ln q\left(\boldsymbol{\pi}^k|\boldsymbol{\lambda}_{\boldsymbol{\gamma}}^k\right) \end{array} \right\} \\
&+ \sum_{m=1}^{M} E_{q(\mathbf{T}|\boldsymbol{\lambda})} \left\{ \ln p(\boldsymbol{\tau}_m|\mathbf{c}_m, \mathbf{d}_m) - \ln q(\boldsymbol{\tau}_m|\boldsymbol{\lambda}_{\mathbf{c}_m}, \boldsymbol{\lambda}_{\mathbf{d}_m}) \right\}
\end{aligned}
\tag{32}
$$

Then, maximizing the first term of Eq. (32), the variational parameters of $q\left(\mathbf{C}_n|\boldsymbol{\lambda}_{\mathbf{C}_n}\right)$ for each sample can be calculated as follows:

$$
\begin{aligned}
&\ln \boldsymbol{\lambda}_{\mathbf{C}_n^k} = \Psi\left(\boldsymbol{\lambda}_{\boldsymbol{\gamma}}^k\right) - \Psi\left(\sum_{j=1}^{K} \boldsymbol{\lambda}_{\boldsymbol{\gamma}}^j\right) + \frac{1}{2} \ln\left|\boldsymbol{\lambda}_{\boldsymbol{\Sigma}_n}^k\right| + \frac{1}{2} \\
&+ \sum_{m=1}^{M} \left\{ \begin{array}{l} \langle \boldsymbol{\tau}_m \rangle \left(\left\langle \mathbf{W}_m^k \right\rangle \left\langle \mathbf{t}_n^k \right\rangle + \left\langle \boldsymbol{\mu}_m^k \right\rangle \right) \mathbf{X}_n^m \\ -\frac{1}{2} \langle \boldsymbol{\tau}_m \rangle \left[\begin{array}{l} Tr\left(\left\langle \left(\mathbf{W}_m^k\right)^T \left(\mathbf{W}_m^k\right) \right\rangle \left\langle \mathbf{t}_n^k \left(\mathbf{t}_n^k\right)^T \right\rangle \right) \\ +2 \left\langle \boldsymbol{\mu}_m^k \right\rangle \left\langle \mathbf{W}_m^k \right\rangle \left\langle \mathbf{t}_n^k \right\rangle + \left\langle \left(\boldsymbol{\mu}_m^k\right)^2 \right\rangle \end{array} \right] \\ -\frac{1}{2} \langle \tau_m \rangle \left(\mathbf{X}_n^m\right)^2 + \frac{1}{2} \left\langle \ln \boldsymbol{\tau}_m \right\rangle \end{array} \right\}
\end{aligned}
\tag{33}
$$

where $k = 1, \ldots, K$, $\langle \bullet \rangle$ represent the expectations of corresponding parameters or sufficient statistics under the variational distribution. For categorical distribution, the variational hyper-parameters $\boldsymbol{\lambda}_{\mathbf{C}_n}$ should be standardized to the area $[0, 1]$ through a *softmax* function:

$$
\begin{aligned}
\boldsymbol{\lambda}_{\mathbf{C}_n^k} &= softmax\left(\left[\ln \boldsymbol{\lambda}_{\mathbf{C}_n^1}, \ldots, \ln \boldsymbol{\lambda}_{\mathbf{C}_n^K} \right]^T \right) \\
&= \left[e^{\ln \lambda_{C_n^1}} \left(\sum_{k=1}^{K} e^{\ln \lambda_{C_n^k}} \right)^{-1}, \ldots, e^{\ln \lambda_{C_n^K}} \left(\sum_{k=1}^{K} e^{\ln \lambda_{C_n^k}} \right)^{-1} \right]^T
\end{aligned}
\tag{34}
$$

The rest hyper-parameters are obtained through maximizing the second and the third term of Eq. (32) with the utilizing the results of (34), given as follows:

$$
\begin{bmatrix} \boldsymbol{\lambda}_{\boldsymbol{\Sigma}_n}^k \\ \boldsymbol{\lambda}_{\boldsymbol{\nu}_n}^k \end{bmatrix} = \begin{bmatrix} \left(\mathbf{I}_D + \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \left(\mathbf{W}_m^k\right)^T \right\rangle \right)^{-1} \\ \boldsymbol{\lambda}_{\boldsymbol{\Sigma}_n}^k \bullet \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \right\rangle^T \left(\mathbf{X}_n^m - \left\langle \boldsymbol{\mu}_m^k \right\rangle \right) \end{bmatrix}
\tag{35}
$$

$$
\begin{bmatrix} \boldsymbol{\lambda}_{\mathbf{V}_m}^k \\ \boldsymbol{\lambda}_{\boldsymbol{\xi}_m}^k \end{bmatrix} = \begin{bmatrix} \left(diag\left(\left\langle \boldsymbol{\alpha}^k \right\rangle\right) + \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \left\langle \mathbf{t}_n^k \left(\mathbf{t}_n^k\right)^T \right\rangle \right)^{-1} \\ \boldsymbol{\lambda}_{\mathbf{V}_m}^k \bullet \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \left\langle \mathbf{t}_n^k \right\rangle \left(\mathbf{X}_n^m - \left\langle \boldsymbol{\mu}_m^k \right\rangle \right) \end{bmatrix}
\tag{36}
$$

$$
\begin{bmatrix} \boldsymbol{\lambda}_{\boldsymbol{\beta}_m}^k \\ \boldsymbol{\lambda}_{\boldsymbol{\mu}_m}^k \end{bmatrix} = \begin{bmatrix} \left(\boldsymbol{\beta}^k + \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \right)^{-1} \\ \boldsymbol{\lambda}_{\boldsymbol{\beta}_m}^k \bullet \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \left(\mathbf{X}_n^m - \left\langle \mathbf{W}_m^k \right\rangle \left\langle \mathbf{t}_n^k \right\rangle \right) \end{bmatrix}
\tag{37}
$$

$$
\begin{bmatrix} \boldsymbol{\lambda}_{\mathbf{c}_m} \\ \boldsymbol{\lambda}_{\mathbf{d}_m} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_m + \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left\langle I(\mathbf{C}_n = k) \right\rangle \\ \mathbf{d}_m + \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left\langle I(\mathbf{C}_n = k) \right\rangle \left\langle \left(\mathbf{X}_n^m - \mathbf{W}_m^k \mathbf{t}_n^k - \boldsymbol{\mu}_m^k \right)^2 \right\rangle \end{bmatrix}
\tag{38}
$$

$$
\begin{bmatrix} \boldsymbol{\lambda}_{\mathbf{a}_d}^k \\ \boldsymbol{\lambda}_{\mathbf{b}_d}^k \end{bmatrix} = \begin{bmatrix} \mathbf{a}_d^k + \frac{1}{2} M \\ \mathbf{b}_d^k + \frac{1}{2} \sum_{m=1}^{M} \left\langle \left(\mathbf{W}_{m,d}^k \right)^2 \right\rangle \end{bmatrix}
\tag{39}
$$

$$
\boldsymbol{\lambda}_{\boldsymbol{\gamma}}^k = \boldsymbol{\gamma}_k + \sum_{n=1}^{N} \left\langle I(\mathbf{C}_n = k) \right\rangle
\tag{40}
$$

where the expectations of the parameters and sufficient statistics are provided as follows:

$$
\left\{ \begin{array}{l} \left\langle I(\mathbf{C}_n = k) \right\rangle = \boldsymbol{\lambda}_{\mathbf{C}_n^k}, \langle \boldsymbol{\pi} \rangle = \boldsymbol{\lambda}_{\boldsymbol{\gamma}}, \left\langle \boldsymbol{\alpha}_d^k \right\rangle = \boldsymbol{\lambda}_{\mathbf{a}_d}^k / \boldsymbol{\lambda}_{\mathbf{b}_d}^k, \\ \left\langle \mathbf{W}_m^k \right\rangle = \boldsymbol{\lambda}_{\boldsymbol{\xi}_m}^k, \left\langle \left(\mathbf{W}_m^k\right)^T \left(\mathbf{W}_m^k\right) \right\rangle = \boldsymbol{\lambda}_{\mathbf{V}_m}^k + \left(\boldsymbol{\lambda}_{\boldsymbol{\xi}_m}^k \right)^2, \\ \left\langle \mathbf{t}_n^k \right\rangle = \boldsymbol{\lambda}_{\boldsymbol{\nu}_n}^k, \left\langle \mathbf{t}_n^k \left(\mathbf{t}_n^k\right)^T \right\rangle = \boldsymbol{\lambda}_{\boldsymbol{\Sigma}_n}^k + \left(\boldsymbol{\lambda}_{\boldsymbol{\nu}_n}^k \right)^2, \\ \left\langle \boldsymbol{\mu}_m^k \right\rangle = \boldsymbol{\lambda}_{\boldsymbol{\mu}_m}^k, \left\langle \left(\boldsymbol{\mu}_m^k\right)^2 \right\rangle = \boldsymbol{\lambda}_{\boldsymbol{\beta}_m}^k + \left(\boldsymbol{\lambda}_{\boldsymbol{\mu}_m}^k \right)^2, \\ \langle \boldsymbol{\tau}_m \rangle = \boldsymbol{\lambda}_{\mathbf{c}_m} / \boldsymbol{\lambda}_{\mathbf{d}_m}, \left\langle \ln \boldsymbol{\tau}_m \right\rangle = \Psi(\boldsymbol{\lambda}_{\mathbf{c}_m}) - \ln \boldsymbol{\lambda}_{\mathbf{d}_m}, \end{array} \right.
\tag{41}
$$

3.2. Stochastic variational inference mixture factor analysis

In this section, the VI-MFA model is transformed to a scalable form via the SVI algorithm [30]. In SVI algorithm, the probabilistic density functions of random variables are written in the form of exponential family distribution, where the variational hyper-parameters should be turned into natural parameters $\boldsymbol{\delta}$ through the following equations:

$$\begin{bmatrix} \boldsymbol{\lambda}^k_{\mathbf{V}_m} \\ \boldsymbol{\lambda}^k_{\boldsymbol{\xi}_m} \end{bmatrix} = \begin{bmatrix} \left(-2\boldsymbol{\delta}^k_{\mathbf{V}_m}\right)^{-1} \\ -\boldsymbol{\delta}^k_{\boldsymbol{\xi}_m} / \left(2\boldsymbol{\delta}^k_{\mathbf{V}_m}\right) \end{bmatrix} \tag{42}$$

$$\begin{bmatrix} \boldsymbol{\lambda}_{\mathbf{c}_m} \\ \boldsymbol{\lambda}_{\mathbf{d}_m} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\delta}_{\mathbf{c}_m} + 1 \\ -\boldsymbol{\delta}_{\mathbf{d}_m} \end{bmatrix} \tag{43}$$

$$\begin{bmatrix} \boldsymbol{\lambda}^k_{\boldsymbol{\beta}_m} \\ \boldsymbol{\lambda}^k_{\boldsymbol{\mu}_m} \end{bmatrix} = \begin{bmatrix} \left(-2\boldsymbol{\delta}^k_{\boldsymbol{\beta}_m}\right)^{-1} \\ -\boldsymbol{\delta}^k_{\boldsymbol{\mu}_m} / \left(2\boldsymbol{\delta}^k_{\boldsymbol{\beta}_m}\right) \end{bmatrix} \tag{44}$$

$$\begin{bmatrix} \boldsymbol{\lambda}^k_{\mathbf{a}_d} \\ \boldsymbol{\lambda}^k_{\mathbf{b}_d} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\delta}^k_{\mathbf{a}_d} + 1 \\ -\boldsymbol{\delta}^k_{\mathbf{b}_d} \end{bmatrix} \tag{45}$$

$$\boldsymbol{\lambda}^k_{\boldsymbol{\gamma}} = \boldsymbol{\delta}^k_{\boldsymbol{\gamma}} + 1 \tag{46}$$

According to SVI algorithm, only one sample is selected while maximizing the ELBO in each iteration, and it is assumed to be sampled N times. Then the ELBO becomes:

$$\begin{aligned} L\left(\boldsymbol{\delta}\right) &= E_{q(\mathbf{T}|\boldsymbol{\delta})} \ln p(\mathbf{T}, \mathbf{X}) - E_{q(\mathbf{T}|\boldsymbol{\delta})} \ln q(\mathbf{T}) \\ &= N \bullet E_{q(\mathbf{T}|\boldsymbol{\delta})} \left\{ \begin{array}{l} \ln p(\mathbf{t}_n|0, \mathbf{I}_D) + \sum_{m=1}^{M} \sum_{k=1}^{K} I(\mathbf{C}_n = k) \ln p\left(\mathbf{X}^m_n | \mathbf{W}^k_m \mathbf{t}^k_n + \boldsymbol{\mu}^k_m, \boldsymbol{\tau}^{-1}_m\right) \\ + \ln p(\mathbf{C}_n|\boldsymbol{\pi}) - \sum_{n=1}^{K} I(\mathbf{C}_n = k) \ln q\left(\mathbf{t}^k_n | \boldsymbol{\lambda}^k_{\boldsymbol{\nu}_n}, \boldsymbol{\lambda}^k_{\boldsymbol{\Sigma}_n}\right) - \ln q(\mathbf{C}_n|\boldsymbol{\lambda}_{\mathbf{C}_n}) \end{array} \right\} \\ &+ \sum_{k=1}^{K} E_{q(\mathbf{T}|\boldsymbol{\delta})} \left\{ \begin{array}{l} + \sum_{m=1}^{M} \ln p\left(\mathbf{W}^k_m|0, \left[diag\left(\boldsymbol{\alpha}^k\right)\right]^{-1}\right) + \ln p\left(\boldsymbol{\pi}^k|\boldsymbol{\gamma}^k\right) \\ + \sum_{d=1}^{D} \ln p\left(\boldsymbol{\alpha}^k_d | \mathbf{a}^k_d, \mathbf{b}^k_d\right) + \sum_{m=1}^{M} \ln p\left(\boldsymbol{\mu}^k_m|0, \left(\boldsymbol{\beta}^k\right)^{-1}\right) \\ - \sum_{m=1}^{M} \ln q\left(\boldsymbol{\mu}^k_m | \boldsymbol{\delta}^k_{\boldsymbol{\mu}_m}, \boldsymbol{\delta}^k_{\boldsymbol{\beta}_m}\right) - \sum_{d=1}^{D} \ln q\left(\boldsymbol{\alpha}^k_d | \boldsymbol{\delta}^k_{\mathbf{a}_d}, \boldsymbol{\delta}^k_{\mathbf{b}_d}\right) \\ - \sum_{m=1}^{M} \ln q\left(\mathbf{W}^k_m | \boldsymbol{\delta}^k_{\boldsymbol{\xi}_m}, \boldsymbol{\delta}^k_{\mathbf{V}_m}\right) - \sum_{k=1}^{K} \ln q\left(\boldsymbol{\pi}^k | \boldsymbol{\delta}^k_{\boldsymbol{\gamma}}\right) \end{array} \right\} \\ &+ \sum_{m=1}^{M} E_{q(\mathbf{T}|\boldsymbol{\delta})} \left\{ \ln p(\boldsymbol{\tau}_m|\mathbf{c}_m, \mathbf{d}_m) - \ln q\left(\boldsymbol{\tau}_m | \boldsymbol{\delta}_{\mathbf{c}_m}, \boldsymbol{\delta}_{\mathbf{d}_m}\right) \right\} \end{aligned} \tag{47}$$

The latent indicator $\mathbf{C}$ and latent factors $\mathbf{t}_n$ are taken as the local variables since they are related to each local sample, and the variational hyper-parameter of $q(\mathbf{C})$ and $q(\mathbf{t}_n)$ are also calculated by Eqs. (33)–(35). Then the rest variables are regarded as the global variables and calculated by the stochastic optimization algorithm. According to Eqs. (16) and (18), the natural gradient is utilized to calculate the intermediate natural parameters for global variables as follows:

$$\begin{bmatrix} \hat{\boldsymbol{\delta}}^k_{\mathbf{V}_m} \\ \hat{\boldsymbol{\delta}}^k_{\boldsymbol{\xi}_m} \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}\left(diag\left(\left\langle \boldsymbol{\alpha}^k \right\rangle\right) + \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \left\langle \mathbf{t}^k_n \left(\mathbf{t}^k_n\right)^T \right\rangle\right) \\ \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \left\langle \mathbf{t}^k_n \right\rangle \left(\mathbf{X}^m_n - \left\langle \boldsymbol{\mu}^k_m \right\rangle\right) \end{bmatrix} \tag{48}$$

$$\begin{bmatrix} \hat{\boldsymbol{\delta}}_{\mathbf{c}_m} \\ \hat{\boldsymbol{\delta}}_{\mathbf{d}_m} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_m - 1 + \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} \left\langle I(\mathbf{C}_n = k) \right\rangle \\ -\mathbf{d}_m - \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} \left\langle I(\mathbf{C}_n = k) \right\rangle \left\langle \left(\mathbf{X}^m_n - \mathbf{W}^k_m \mathbf{t}^k_n - \boldsymbol{\mu}^k_m\right)^2 \right\rangle \end{bmatrix} \tag{49}$$

$$\begin{bmatrix} \hat{\boldsymbol{\delta}}^k_{\boldsymbol{\beta}_m} \\ \hat{\boldsymbol{\delta}}^k_{\boldsymbol{\mu}_m} \end{bmatrix} = \begin{bmatrix} -\frac{1}{2}\left(\boldsymbol{\beta}^k + \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} \left\langle I(\mathbf{C}_n = k) \right\rangle\right) \\ \langle \boldsymbol{\tau}_m \rangle \sum_{n=1}^{N} I(\mathbf{C}_n = k) \left(\mathbf{X}^m_n - \left\langle \mathbf{W}^k_m \right\rangle \left\langle \mathbf{t}^k_n \right\rangle\right) \end{bmatrix} \tag{50}$$

$$\begin{bmatrix} \hat{\boldsymbol{\delta}}^k_{\mathbf{a}_d} \\ \hat{\boldsymbol{\delta}}^k_{\mathbf{b}_d} \end{bmatrix} = \begin{bmatrix} \mathbf{a}^k_d - 1 + \frac{1}{2}M \\ -\mathbf{b}^k_d - \frac{1}{2}\sum_{m=1}^{M} \left\langle \left(\mathbf{W}^k_{m,d}\right)^2 \right\rangle \end{bmatrix} \tag{51}$$

$$\hat{\boldsymbol{\delta}}^k_{\boldsymbol{\gamma}} = \boldsymbol{\gamma}_k - 1 + \sum_{n=1}^{N} \left\langle I(\mathbf{C}_n = k) \right\rangle \tag{52}$$

Finally, the Robbins-Monro average of the global natural parameters at the Tth updating can be written as follows:

$$\boldsymbol{\delta}^{(T)}_* = \left(1 - \rho^{(T)}\right) \boldsymbol{\delta}^{(T-1)}_* + \rho^{(T)} \hat{\boldsymbol{\delta}}^{(T)}_* \tag{53}$$

where $*$ represents the global variables. It should be mentioned that the learning rate ρ is set to slowly decay in an exponential curve [32] as follows:

$$\rho^{(T)} = \rho_{\min} + (\rho_{\max} - \rho_{\min}) * \frac{2\exp\left(-\varepsilon_1 \left(t/T\right)^{\varepsilon_2}\right)}{1 + \exp\left(-\varepsilon_1 \left(t/T\right)^{\varepsilon_2}\right)} \tag{54}$$

where $\rho_{\max} = 0.98$, $\rho_{\min} = 0.01$, $\varepsilon_1 = 75$, $\varepsilon_2 = 5$. This kind of learning rate can effectively avoid the local minimum in the iterations [31]. After all the iterations, Eqs. (42)–(46) will be used to calculate the variational hyper-parameters.

3.3. Deployment of SVI-MFA on parameter server

According to the architecture of the proposed distributed and parallel modeling framework, the SVI-MFA model can be easily deployed on the Parameter Server. Besides, in order to improve the stability of SVI algorithm, a mini-batch of samples can be randomly selected at each iteration. For SVI-MFA, S samples $\mathbf{X}_{1:S}$ is randomly selected at each iteration to compute the local variational parameters $\boldsymbol{\lambda}_{\mathbf{C}_{1:S}}$, $\boldsymbol{\lambda}_{\boldsymbol{\nu}}$ and $\boldsymbol{\lambda}_{\boldsymbol{\Sigma}}$ through Eqs. (33)–(35), then the intermediate global parameters $\hat{\boldsymbol{\delta}}^S_*$ are computed. Finally, they are averaged to update the global natural parameters:

$$\boldsymbol{\delta}^{(T)} = \left(1 - \rho^{(T)}\right) \boldsymbol{\delta}^{(T-1)} + \frac{\rho^{(T)}}{S} \sum_{S} \hat{\boldsymbol{\delta}}^S \tag{55}$$

The mini-batch SVI can effectively amortize any computational expenses associate with updating the global parameters across more data points. Besides, mini-batches can help SVI algorithm take less steps to find the optimal value since sampling only one data sample may lead to a poor optimum. Finally, the proposed SVI-MFA model can be deployed on the Parameter Server framework to conduct distributed and parallel computing for big data modeling as Fig. 5.

Furthermore, the full procedures of training SVI-MFA model on the Parameter Server are listed in the following Algorithm 1.

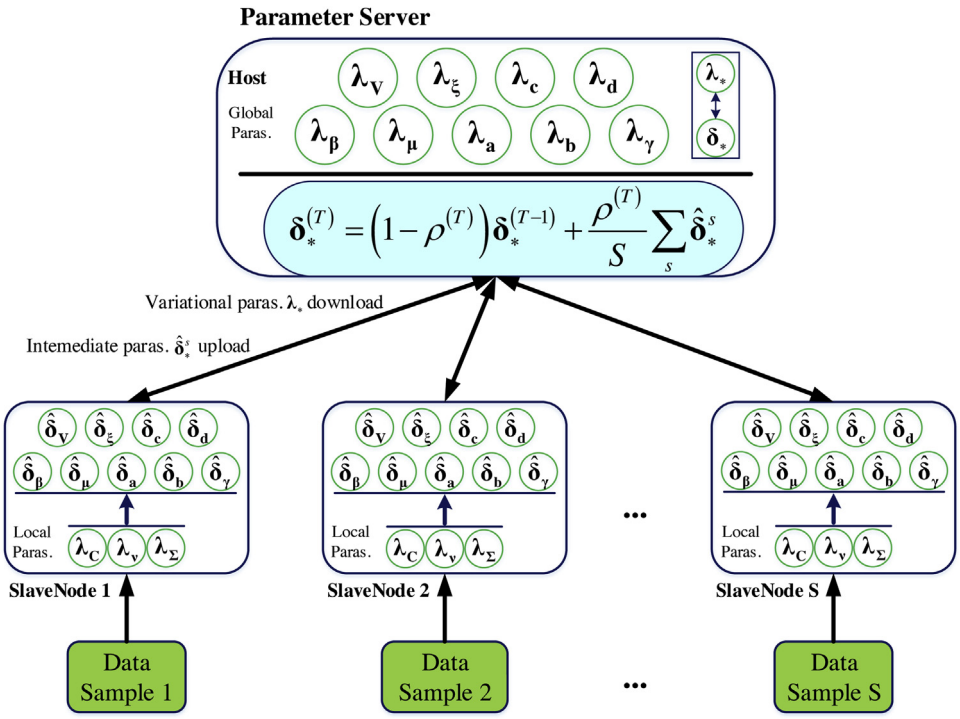


Fig. 5. Structure of Parameter Server with SVI-MFA.

Algorithm 1 Full Algorithm Procedures of SVI-MFA

1: Initialize variational parameters λ_* on slave nodes.
2: **While** (convergence criterion is not meet) **do:**
3: Set the learning rate $\rho^{(T)}$.
4: Randomly select a mini-batch of sample $\mathbf{X}_{1:S}$ from the huge dataset and distribute to local slave nodes.
5: Local: Compute local paras. λ_C, λ_v and λ_Σ by (33)-(35) on each local slave node.
6: Compute intermediate natural paras. by (48)-(52) on each local slave node.
7: Upload intermediate global natural paras. to the host.
8: Global: Update global natural paras by (48)-(52) on the host.
9: Transform natural paras. to variational paras. by (42)-(46) on the host.
10: Download variational paras. λ_* to each local slave node.
11: **End while.**
12: Store global variational parameters.

4. Applications

In this section, two common applications of data-driven models, process monitoring and quality prediction, are presented based on SVI-MFA. The proposed SVI-MFA model is utilized to build monitoring statistics and predictor as an example. However, it should be mentioned that the proposed distributed parallel modeling framework can be further utilized for many other process applications, like data-based optimization, data pattern recognition, data visualization, etc.

4.1. Process monitoring

After the SVI-MFA model is trained through the distributed and parallel modeling platform, we can use it for process monitoring. Similar to the traditional PCA method, two monitoring statistics T^2 and SPE can be developed. Compared to the traditional PCA method, the latent factors in each local factor analyzer can be represented by the their expectations given in Eq. (35). Intuitively, the mean value of latent factors can be written as follows:

$$\hat{\mathbf{t}}^k = \left\langle \mathbf{t}^k \right\rangle = \left(\mathbf{I}_D + \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \left(\mathbf{W}_m^k \right)^T \right\rangle \right)^{-1} \bullet \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \right\rangle^T \left(\mathbf{X}_n^m - \left\langle \boldsymbol{\mu}_m^k \right\rangle \right) \tag{56}$$

Also, the covariance of $\hat{\mathbf{t}}^k$ can be given as follow:

$$var\left(\hat{\mathbf{t}}^k\right) = \left(\mathbf{I}_D + \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \left(\mathbf{W}_m^k \right)^T \right\rangle \right)^{-1} \tag{57}$$

Hence, the Hotelling T^2 statistics can be constructed for the kth local factor analyzer as follow [14]:

$$T_k^2 = \left(\hat{\mathbf{t}}^k\right)^T \left(var\left(\hat{\mathbf{t}}^k\right) \right)^{-1} \hat{\mathbf{t}}^k \tag{58}$$

The expectation of the residual vector in kth local factor analyzer and the corresponding SPE statistics can be given as follows:

$$\hat{\mathbf{e}}_k = \mathbf{X} - \left\langle \boldsymbol{\mu}^k \right\rangle - \left\langle \mathbf{W}^k \right\rangle \hat{\mathbf{t}}^k \tag{59}$$

$$SPE_k = \left(\hat{\mathbf{e}}_k\right)^T \left(diag\left(\langle \boldsymbol{\tau} \rangle \right) \right) \hat{\mathbf{e}}_k \tag{60}$$

The control limits of the T^2 and SPE statistics can be both determined by the χ^2 distribution:

$$\begin{aligned} T_k^2 &\le T_{\lim}^2 = \chi_\varphi^2(D) \\ SPE_k^2 &\le SPE_{\lim}^2 = \chi_\varphi^2(M) \end{aligned} \tag{61}$$

where D is the number of retained factors in kth local factor analyzer, M is the number of process variables, and φ is the significance level of both monitoring statistics [33].

For monitoring a new data sample $\mathbf{X}_{new}$, it is tedious to monitor all the local monitoring statistics that constructed in local factor

analyzers. Therefore, the monitoring performance and implementation efficiency can be further improved if the results in different local factor analyzer subspace can be integrated. Firstly, the two monitoring statistics of $\mathbf{X}_{new}$ are calculated in kth local factor analyzer subspace as follows:

$$\begin{aligned}\hat{\mathbf{t}}_{new}^{k} &= \left\langle \mathbf{t}_{new}^{k} \right\rangle = \left(\mathbf{I}_D + \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \left(\mathbf{W}_m^k \right)^T \right\rangle \right)^{-1} \\ &\bullet \sum_{m=1}^{M} \langle \boldsymbol{\tau}_m \rangle \left\langle \mathbf{W}_m^k \right\rangle^T \left(\mathbf{X}_{new}^m - \left\langle \boldsymbol{\mu}_m^k \right\rangle \right) \\ T_{k,new}^2 &= \left(\hat{\mathbf{t}}_{new}^{k} \right)^T \left(var\left(\hat{\mathbf{t}}_{new}^{k} \right) \right)^{-1} \hat{\mathbf{t}}_{new}^{k}\end{aligned} \tag{62}$$

$$\begin{aligned}\hat{\mathbf{e}}_{k,new} &= \mathbf{X}_{new} - \left\langle \boldsymbol{\mu}^k \right\rangle - \left\langle \mathbf{W}^k \right\rangle \hat{\mathbf{t}}_{new}^{k} \\ SPE_{k,new} &= \left(\hat{\mathbf{e}}_{k,new} \right)^T \left(diag\left(\langle \boldsymbol{\tau} \rangle \right) \right) \hat{\mathbf{e}}_{k,new}\end{aligned} \tag{63}$$

Then the Bayesian method is utilized to transform the monitoring statistics into fault probabilities as follows:

$$P_{T^2}^k(F|\mathbf{X}_{new}) = \frac{P_{T^2}^k(\mathbf{X}_{new}|F) P_{T^2}^k(F)}{P_{T^2}^k(\mathbf{X}_{new}|N) P_{T^2}^k(N) + P_{T^2}^k(\mathbf{X}_{new}|F) P_{T^2}^k(F)} \tag{64}$$

$$P_{SPE}^k(F|\mathbf{X}_{new}) = \frac{P_{SPE}^k(\mathbf{X}_{new}|F) P_{SPE}^k(F)}{P_{SPE}^k(\mathbf{X}_{new}|N) P_{SPE}^k(N) + P_{SPE}^k(\mathbf{X}_{new}|F) P_{SPE}^k(F)} \tag{65}$$

where $P_{T^2}^k(N)$, $P_{T^2}^k(F)$ and $P_{SPE}^k(N)$, $P_{SPE}^k(F)$ are the priors of process being normal and faulty for the two statistics of the kth local factor analyzer. They are determined as follows:

$$\begin{aligned}P_{T^2}^k(N) &= P_{SPE}^k(N) = 1 - \varphi \\ P_{T^2}^k(F) &= P_{SPE}^k(F) = \varphi\end{aligned} \tag{66}$$

Besides, four conditional probabilities should be given, which are defined as follows:

$$P_{T^2}^k(\mathbf{X}_{new}|N) = \exp\left\{ -\frac{T_{k,new}^2}{T_{k,lim}^2} \right\}, P_{SPE}^k(\mathbf{X}_{new}|N) = \exp\left\{ -\frac{SPE_{k,new}}{SPE_{k,lim}} \right\} \tag{67}$$

$$P_{T^2}^k(\mathbf{X}_{new}|F) = \exp\left\{ -\frac{T_{k,lim}^2}{T_{k,new}^2} \right\}, P_{SPE}^k(\mathbf{X}_{new}|F) = \exp\left\{ -\frac{SPE_{k,lim}}{SPE_{k,new}} \right\} \tag{68}$$

where $T_{k,lim}^2$ and $SPE_{k,lim}$ are corresponding control limits in the kth local subspace. After the monitoring statistics are obtained, we can easily integrate them through the posterior probabilities $g_k(\mathbf{X}_{new})$ that the new sample belongs to, which can be calculated through Eqs. (33) and (34). Finally, the two integrated monitoring statistics can be constructed as follows:

$$IT_{new}^2 = \sum\nolimits_{k=1}^{K} g_k(\mathbf{X}_{new}) P_{T^2}^k(F|\mathbf{X}_{new}) \tag{69}$$

$$ISPE_{new} = \sum\nolimits_{k=1}^{K} g_k(\mathbf{X}_{new}) P_{SPE}^k(F|\mathbf{X}_{new}) \tag{70}$$

Therefore, the process behavior can be judged through checking if the two integrated monitoring statistics have exceeded their control limits, which are both given by the significance level φ.

4.2. Quality prediction

In real industrial processes, many key performance indices (KPIs) are regarded as quality variables that cannot be measured in a low-cost or convenient way. Predictive models can be built based on the historical data to provide accurate predictions for those quality variables [34]. In order to take advantage of the large scale of process data, the proposed distributed and parallel modeling framework can be utilized. After the modeling, the model parameters of MFA are obtained. In order to provide predictions, the unsupervised MFA model should be transformed to the supervised MFA model. Thus, the whole dataset $\mathbf{X}_n$ should be partitioned into input and output parts as follows:

$$\mathbf{X}_n = \begin{bmatrix} \mathbf{x}_n \\ \mathbf{y}_n \end{bmatrix} \tag{71}$$

where $\mathbf{x}_n$ is the input and $\mathbf{y}_n$ denotes the output. Accordingly, the parameters of MFA models are also divided into two parts:

$$\mathbf{W}^k = \begin{bmatrix} \mathbf{W}_{\mathbf{x}}^k \\ \mathbf{W}_{\mathbf{y}}^k \end{bmatrix}, \boldsymbol{\mu}^k = \begin{bmatrix} \boldsymbol{\mu}_{\mathbf{x}}^k \\ \boldsymbol{\mu}_{\mathbf{y}}^k \end{bmatrix}, \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_{\mathbf{x}} \\ \boldsymbol{\tau}_{\mathbf{y}} \end{bmatrix} \tag{72}$$

When a new testing sample $\mathbf{x}_{new}$ comes, the corresponding latent factor can be estimated as follows:

$$\begin{aligned}\hat{\mathbf{t}}_{new}^{k} &= \left\langle \mathbf{t}_{new}^{k} \right\rangle = \left(\mathbf{I}_D + \sum_{m=1}^{M} \left\langle \boldsymbol{\tau}_{\mathbf{x},m} \right\rangle \left\langle \mathbf{W}_{\mathbf{x},m}^k \left(\mathbf{W}_{\mathbf{x},m}^k \right)^T \right\rangle \right)^{-1} \\ &\bullet \sum_{m=1}^{M} \left\langle \boldsymbol{\tau}_{\mathbf{x},m} \right\rangle \left\langle \mathbf{W}_{\mathbf{x},m}^k \right\rangle^T \left(\mathbf{x}_n^m - \left\langle \boldsymbol{\mu}_{\mathbf{x},m}^k \right\rangle \right)\end{aligned} \tag{73}$$

Here, m represents the dimension of the input variables. Then the output prediction $\hat{\mathbf{y}}_n$ can be given by a weighted sum of predictions of K local analyzers as follows:

$$\hat{\mathbf{y}}_{new} = \sum_{k=1}^{K} g_k(\mathbf{x}_{new}) \left[\left\langle \mathbf{W}_{\mathbf{y}}^k \right\rangle \hat{\mathbf{t}}_{new}^{k} + \left\langle \boldsymbol{\mu}_y^k \right\rangle \right] \tag{74}$$

where $g_k(\mathbf{x}_{new})$ is the weight for input vector $\mathbf{x}_{new}$, and it represents the probability that $\mathbf{x}_{new}$ generated from the kth analyzer.

To evaluate the prediction performance of the designed predictive model, the Root Mean Square Error (RMSE) is calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{Nt} \left| \mathbf{y}_i - \hat{\mathbf{y}}_{new,i} \right|^2}{Nt}} \tag{75}$$

where Nt represents the length of testing dataset. $\mathbf{y}_{new,i}$ and $\hat{\mathbf{y}}_{new,i}$ represent the actual and predicted value of testing outputs, respectively.

5. Case studies

In this section, a numerical example will be presented to demonstrate the feasibility of the proposed SVI-MFA method for modeling with big data, then the TE benchmark process will be utilized to verify the effectiveness of SVI-MFA model for process monitoring and quality prediction, and finally the SVI-MFA-based process monitoring and quality prediction will be conducted on a real industrial case of Methanation Furnace Unit. Before conducting the case studies, a computing cluster for Parameter Server framework is established with 6 computing nodes, and the configurations are given in Table 1. The Parallel Computing Toolbox in MathWorks MATLAB 2013b is utilized to build the Parameter Server framework.

Table 1
Configuration of six nodes in the computing cluster.

IP of node	Role in cluster	CPU	Memory
192.168.0.10	Parameter Server	Intel Core i5-4590, 3.30GHz	4 GB
192.168.0.16	Slave node	Intel Core i5-4590, 3.30GHz	4 GB
192.168.0.18	Slave node	Intel Core i5-4590, 3.30GHz	4 GB
192.168.0.25	Slave node	Intel Core i5-4590, 3.30GHz	4 GB
192.168.0.28	Slave node	Intel Core i5-4590, 3.30GHz	4 GB
192.168.0.30	Slave node	Intel Core i5-4590, 3.30GHz	4 GB

To demonstrate the effectiveness of the proposed SVI-MFA algorithm, execution parameters for the training process should be firstly determined. The batch-size of SVI-MFA equals 1% of the data-size, then the iteration number equals 100 to completely use all the data samples. Besides, the epoch of iterations is set as 3, then the total iteration number for SVI-MFA equals 300. The number of iterations for VI-MFA is also set as 300 for a fair comparison.

5.1. Validation of SVI-MFA on a numerical example

The numerical dataset contains three different modes, each mode consists of five variables, which are generated from three latent factors. The equations used for generating dataset are given as follows:

$$\begin{aligned}\mathbf{X}^1 &= \mathbf{W}^1\mathbf{t}^1 + \boldsymbol{\mu}^1 + \boldsymbol{\Phi}\\ \mathbf{X}^2 &= \mathbf{W}^2\mathbf{t}^2 + \boldsymbol{\mu}^2 + \boldsymbol{\Phi}\\ \mathbf{X}^3 &= \mathbf{W}^3\mathbf{t}^3 + \boldsymbol{\mu}^3 + \boldsymbol{\Phi}\end{aligned} \tag{76}$$

where $\mathbf{W}^1$, $\mathbf{W}^2$ and $\mathbf{W}^3$ are three 5×3 loading matrices, which follow Gaussian distribution with zero mean and unity variance, and the latent factors $\mathbf{t}^1$, $\mathbf{t}^2$ and $\mathbf{t}^3$ are generated from three standard normal distributions. The mean value of the three modes are $\boldsymbol{\mu}_1 = [3; 5; 9; 12; 4]$, $\boldsymbol{\mu}_2 = [5; 4; 8; 11; 5]$ and $\boldsymbol{\mu}_3 = [2; 7; 9; 10; 6]$. The noise level in the three modes are assumed to be the same, and it follows the Gaussian distribution $\boldsymbol{\Phi} \sim N\left(0, diag(\boldsymbol{\tau})^{-1}\right)$, $\boldsymbol{\tau} = [6.67; 6.25; 4.17; 2.08; 2.78]$. Besides, the combining weights of the three modes for the dataset are set as $\boldsymbol{\pi} = [0.3, 0.3, 0.4]$. In this example, a large dataset with $N = 10^6$ samples are generated to validate the feasibility of the proposed SVI-MFA algorithm.

Firstly, the capability of VI-MFA and SVI-MFA in determining the number of local analyzers is tested. According to the "birth and death" principle [29], it is possible to start the training with a large number of local models and let them die, the finally survived models will be the selected. Assigning initial number of local analyzers as 5 for the two algorithms, the mixture proportions $\boldsymbol{\pi}$ are obtained after iterations. The low values of $\boldsymbol{\pi}$ are discarded to obtain a fine pruned number of local analyzers. Finally, the determined number of local analyzers is 3 for both of the two models and the three modes of the data are displayed in Fig. 6. The loading matrices of different local analyzers can also be obtained, from which the number of latent factors for each local analyzer can be determined through the number of retained columns. The Hinton Graphs [29] of factor loading matrices of the two models are displayed in Figs. 7 and 8, respectively. From them, it can be seen that the number of retained columns (latent factors) is 3 for all the local analyzers, which is the same as the setting of the example.

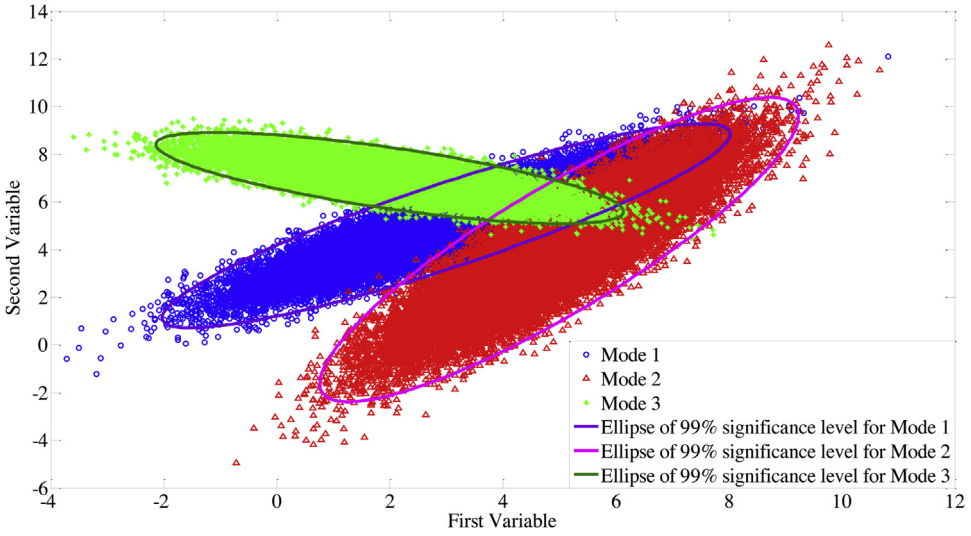
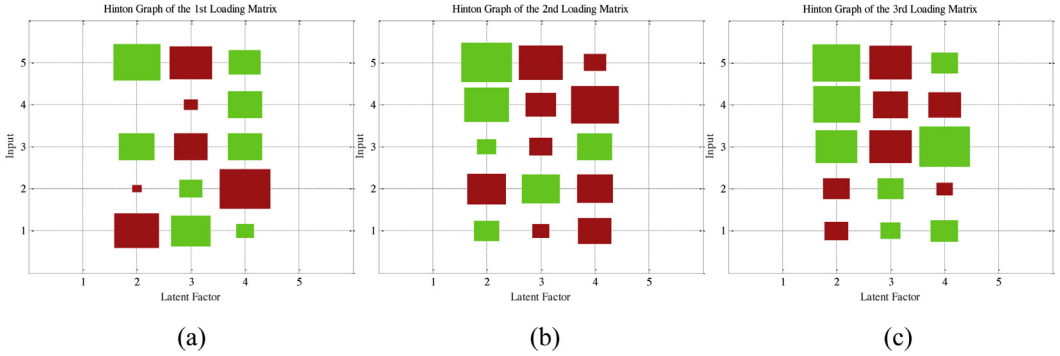


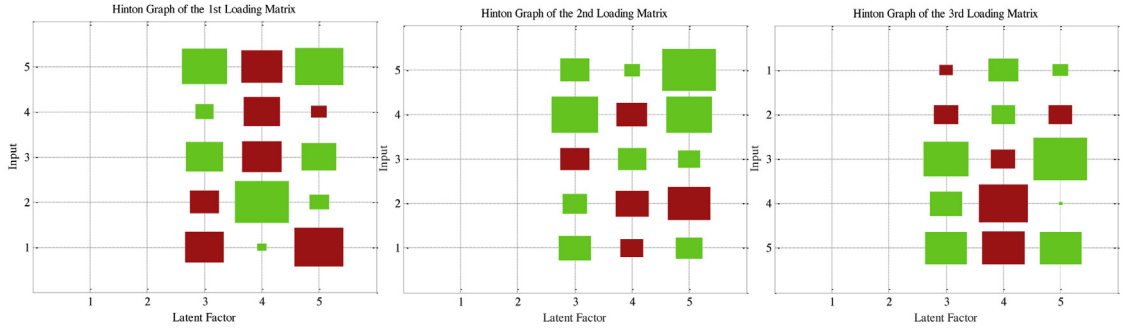
Fig. 6. Scatter plots of the data patterns with 10^5 samples.



(a) Hinton Graph of identified $\mathbf{W}^1$ by VI-MFA; (b) Hinton Graph of identified $\mathbf{W}^2$ by VI-MFA; (c) Hinton Graph of identified $\mathbf{W}^3$ by VI-MFA

Fig. 7. Hinton Graphs of the loading matrices identified by VI-MFA.

Fig 7. Hinton Graphs of the loading matrices identified by VI-MFA



(a) Hinton Graph of identified $\mathbf{W}^1$ by SVI-MFA; (b) Hinton Graph of identified $\mathbf{W}^2$ by SVI-MFA; (c) Hinton Graph of identified $\mathbf{W}^3$ by SVI-MFA

Fig. 8. Hinton Graphs of the loading matrices identified by SVI-MFA.

Table 2
Training Performance of VI-MFA and SVI-MFA for Numerical Example.

N	Model	π	μ	τ	Training Time (s)
10^5	VI	[0.2978, 0.2998, 0.4024]	[2.58, 5.14, 9.14, 11.92, 4.37; 4.78, 4.19, 8.01, 10.78, 4.72; 2.02, 7.08, 8.87, 9.95, 5.98]	[6.54, 6.11, 4.07, 2.03, 2.74]	27133.4
	SVI	[0.2979, 0.2997, 0.4024]	[2.56, 5.17, 9.18, 11.91, 4.38; 4.87, 4.09, 8.02, 10.68, 4.71; 2.01, 7.05, 8.83, 9.93, 5.85]	[6.55, 6.10, 4.12, 2.04, 2.72]	**60.7**
3×10^5	VI	[0.3015, 0.2993, 0.3992]	[2.55, 5.20, 9.09, 11.79, 4.18; 4.95, 3.98, 8.02, 10.91, 4.96; 2.01, 6.99, 9.01, 9.99, 6.03]	[6.67, 6.13, 4.12, 2.05, 2.81]	97578.8
	SVI	[0.3017, 0.2991, 0.3991]	[2.54, 5.30, 9.11, 11.84, 4.31; 4.96, 3.98, 8.03, 10.90, 4.95; 2.00, 6.99, 9.02, 9.99, 6.01]	[6.68, 6.12, 4.11, 2.05, 2.83]	**172.1**
5×10^5	VI	[0.3025, 0.2989, 0.3985]	[2.62, 5.18, 9.12, 11.95, 4.18; 4.95, 3.96, 8.05, 10.93, 4.99; 2.00, 7.01, 8.96, 9.95, 5.95]	[6.65, 6.14, 4.15, 2.06, 2.76]	148501.1
	SVI	[0.3045, 0.2969, 0.3985]	[2.60, 5.27, 9.10, 11.85, 4.28; 4.96, 3.97, 8.03, 10.91, 4.98; 2.01, 7.02, 8.95, 9.94, 5.94]	[6.64, 6.13, 4.17, 2.07, 2.77]	**290.6**
7×10^5	VI	[0.3005, 0.2993, 0.4002]	[2.66, 5.18, 9.09, 11.98, 4.13; 4.96, 3.97, 8.04, 10.94, 4.99; 2.00, 7.02, 8.98, 10.01, 5.98]	[6.65, 6.22, 4.14, 2.08, 2.79]	204355.2
	SVI	[0.3025, 0.2973, 0.4002]	[2.65, 5.20, 9.10, 11.88, 4.23; 4.95, 3.94, 8.06, 10.91, 4.99; 2.00, 7.01, 8.95, 10.00, 5.96]	[6.64, 6.23, 4.13, 2.07, 2.79]	**408.4**
10^6	VI	[0.3021, 0.2978, 0.4001]	[2.81, 5.07, 9.04, 11.96, 4.10; 4.98, 3.99, 8.01, 10.99, 5.01; 2.01, 7.01, 8.99, 10.01, 6.03]	[6.67, 6.23, 4.19, 2.08, 2.77]	304879.7
	SVI	[0.3011, 0.2988, 0.4001]	[2.85, 5.05, 9.07, 11.94, 4.13; 4.98, 3.99, 8.02, 10.97, 5.02; 2.00, 7.00, 8.96, 10.01, 6.01]	[6.66, 6.25, 4.18, 2.07, 2.78]	**582.3**

Then, the training processes are conducted on VI-MFA and SVI-MFA with different large scales of samples to demonstrate the superiority of SVI-based algorithms. Detailed training results of parameters and training time are listed in Table 2. It can be seen that the identified results of parameters are approximating the real values for the two models as the data-size increases. It means that the larger scale of samples can provide more useful information in modeling. Even the batch-size equals 1% of the data-size while training the SVI-MFA, the identified parameters are almost the same accurate as the VI-MFA, which incorporates all the data samples into the training process. When the data-size is large enough (N=10^6), the identified results of SVI-MFA are mostly equal to the real values. Moreover, it should be emphasized that the training time of SVI-MFA is much shorter than the VI-MFA. The cause of the difference is that SVI-MFA selects only a batch of data samples to conduct the updating iterations of the parameters and deploys the computing of local parameters on a distributed and parallel framework, while VI-MFA uses all the samples in each of the iteration. Therefore, it can be concluded that SVI-MFA provides a much more efficient modeling process when facing a large training dataset.

5.2. Experiments on TE benchmark process

As an extraction of real industrial chemical production, Tennessee Eastman (TE) benchmark process has recently become a benchmark for the evaluation of MSPM methods [35]. The process contains 41measurement variables and 12 manipulated variables.

Table 3
Selected variables for TE process.

No.	Variables	No.	Variables	No.	Variables
1	A feed	12	Product separator level	23	D feed flow rate
2	D feed	13	Product separator pressure	24	E feed flow rate
3	E feed	14	Product separator underflow	25	A feed flow rate
4	Total feed	15	Stripper level	26	Total feed flow rate
5	Recycle flow	16	Stripper pressure	27	Purge valve
6	Reactor feed rate	17	Stripper underflow	28	Separator pot liquid flow rate
7	Reactor pressure	18	Stripper temperature	29	Stripper liquid product flow rate
8	Reactor level	19	Stripper steam flow	30	Reactor cooling water flow
9	Reactor temperature	20	Compressor work	31	Condenser cooling water flow
10	Purge rate	21	Reactor cooling water outlet temperature		
11	Product separator temperature	22	Separator cooling water outlet temperature		

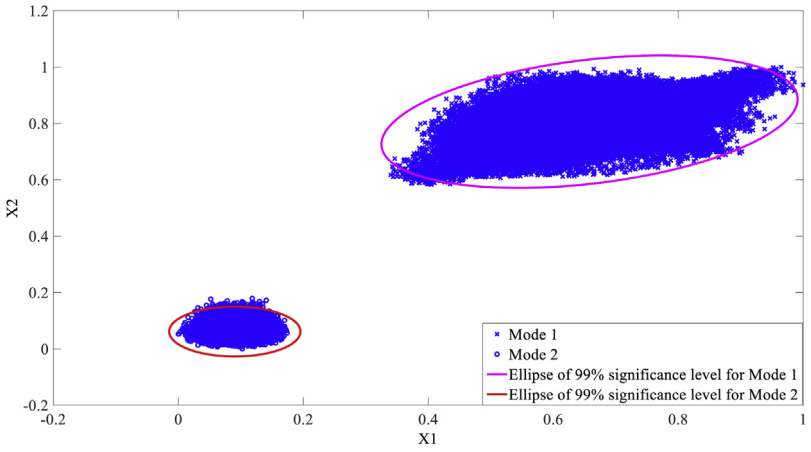


Fig. 9. Scatter plots of the normalized TE data patterns with $6^{*}10^{5}$ samples.

Since some variables are constants or quality-related variables, 31 variables are selected in this work and listed in Table 3.

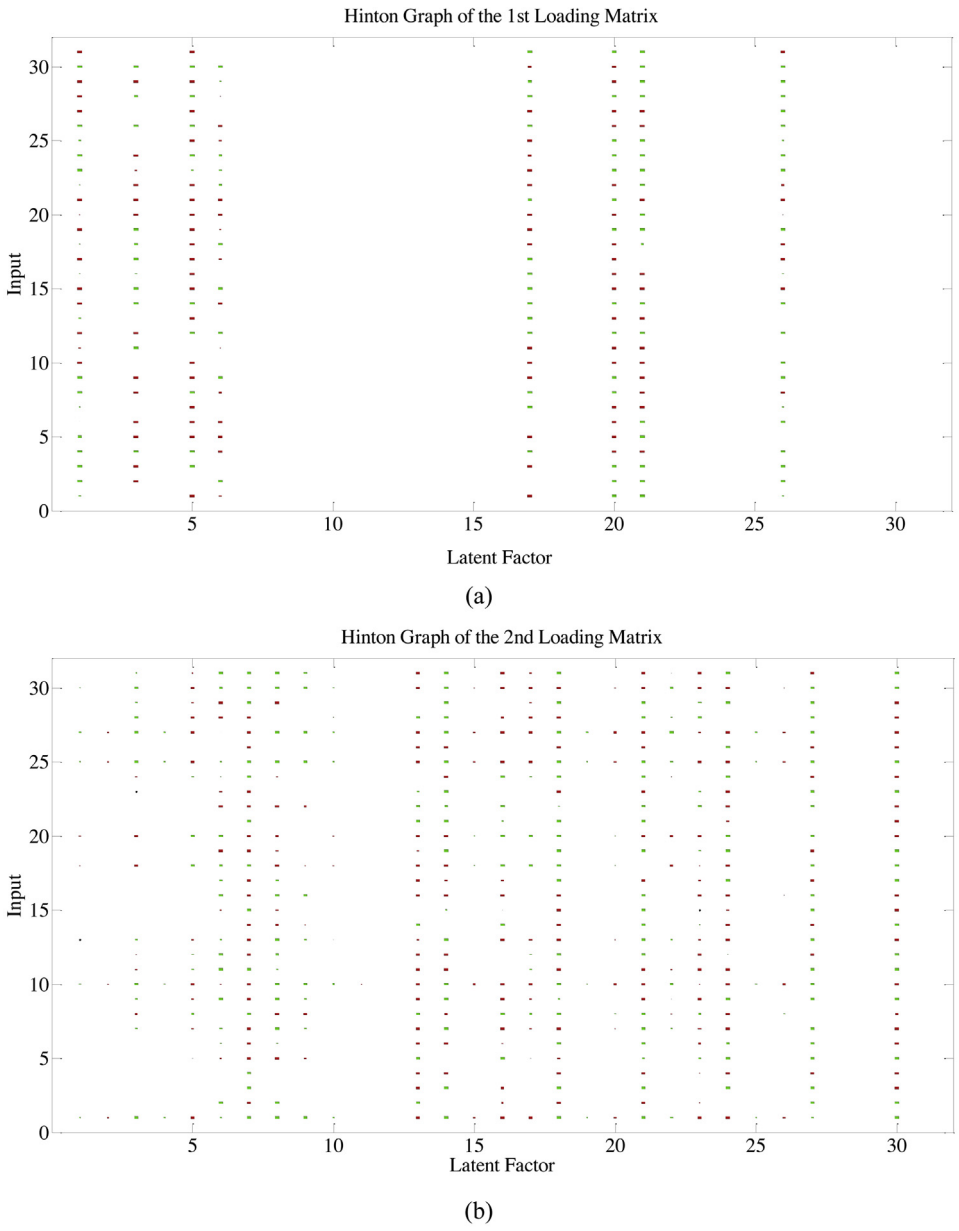
For model validation, the revised TE plant simulator is used for data generation [36]. The two steady-state operating modes, Mode 1 and Mode 3, are assumed to be presented with equal probabilities during normal process operations. Then, data samples from 3000 running hours are collected under each operation condition at a sampling interval of 36 s. Therefore, the entire dataset contains 0.6 million samples, which would be a heavy computing burden for traditional modeling algorithms. As an alternative solution, the modeling is conducted on the distributed and parallel framework. In this case, the initial number of local factors is set as 5 for the SVI-MFA model. Once the modeling procedure is finished, the pattern of each local model is displayed in Fig. 9. It can be seen that the distributed and parallel SVI-MFA model can capture the operating conditions from the huge dataset. Besides, the mixture proportions $\boldsymbol{\pi}$ are estimated as [0.4978, 0.5022], which is consistent with the settings. In this sense, it can be demonstrated that big data modeling can be successfully conducted with a distributed and parallel modeling framework. Besides, the loading matrices of the two local models can also be obtained. The Hinton Graphs of factor loading matrices of the two models are displayed in Fig. 10(a) and (b), respectively. From them, it can be seen that the number of retained latent factors are 8 and 18 for the two local analyzers.

To explore the performance of the proposed SVI-MFA model for monitoring, two typical faults, IDV10 from mode 1 and IDV27 from mode 3 is investigated in detail, the explicit fault descriptions can be found in [36]. Note that 1000 samples are collected for each fault, and fault signals are introduced at the 300th sample. The significance level is set as 0.01 in this case. The integrated monitoring results for both faults are shown in Figs. 11(a) and 12 (a), respectively. At the same time, the monitoring results of the local models for the two faults are given in Figs. 11(b), (c) and 12 (b), (c), respectively. From these figures, it can be seen that the SVI-MFA model can be effectively used for fault detection, and the integrated monitoring statistics can more sensitively detect faults than the local factor analyzers.

Then the proposed SVI-MFA model is utilized for quality prediction in TE process. In stream 11, contents of four productions of TE process are measured with noises every 11 min. A data-driven predictive model can be built here to provide real-time estimation for the contents. The number of training samples is 0.5 million with two process modes. Then, for testing, 1000 samples are selected for each product. The predicted results of the testing samples are displayed in Fig. 13. Besides, the prediction RMSEs are calculated and displayed in the sub-figures. Then, the training and testing operations are conducted on Variational Inference Gaussian Mixture Regression (VI-GMR), VI-MFA and SVI-MFA with different scales of big data to demonstrate the superiority of SVI-based algorithms. Detailed testing RMSE and training time are listed in Table 4. It can be seen that the testing RMSE are decreasing for the three models as the data-size increases. It means that the larger scale of samples can provide more useful information in modeling. Even the batch-size equals 1% of the data-size while training the SVI-MFA, the testing RMSE can be smaller than the VI-GMR and VI-MFA, which incorporate all the data samples into the training process. Moreover, it can be seen that the training time of SVI-MFA is much shorter than the VI-GMR and the VI-MFA. It demonstrates that the SVI-based algorithm is much more efficient than the traditional VI-based algorithms while modeling large-scale TE process data.

5.3. Application on the methanation furnace unit

The Methanation Furnace Unit is a significant production unit that derived from a real Ammonia Synthesis process. One of the production materials, hydrogen, is generated from the methane



(a) Hinton Graph of the 1st loading matrix by SVI-MFA; (b) Hinton Graph of the 2nd loading matrix by SVI-MFA

Fig. 10. Hinton Graphs of the loading matrices obtained by SVI-MFA.

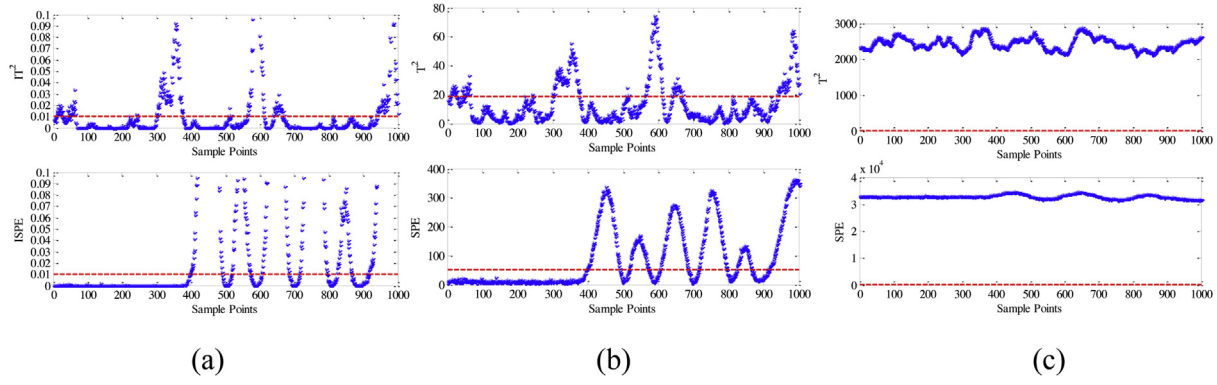
decarburization unit of the process. However, the carbon element still exists in the process gas in the form of CO and CO_2. The main function of the Methanation Furnace Unit is to transform the CO and CO_2 to the Methane, which would be transferred back for recycling. While the process gas flows through the Methanation Furnace Unit, the chemical reactions taken place are described as follows:

$$\begin{aligned} CO + 3H_2 &\rightleftharpoons CH_4 + H_2O + Q \\ CO_2 + 4H_2 &\rightleftharpoons CH_4 + 2H_2O + Q \end{aligned} \tag{77}$$

According to the process design, a brief flowchart of Methanation Furnace Unit with all the 11 process instruments is shown in Fig. 14, and the descriptions of the process variables and the devices in the flowchart are displayed in Tables 5 and 6, respectively.

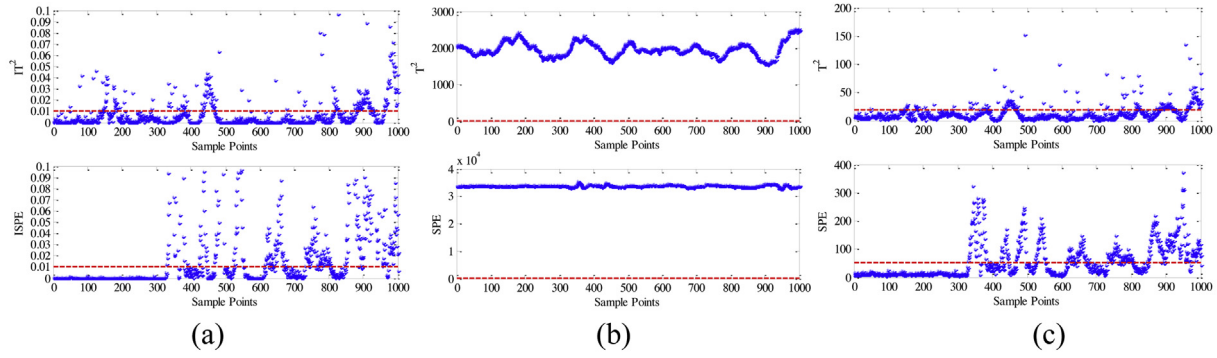
In this unit, the goal is to furthest reduce the content of CO and CO_2 in the process gas and to stably operate the process in an energy-saving style. Thus, the first and foremost procedure is to measure the content of residual CO and CO_2 at the outlet of the unit as a critical quality variable. Due to the variation of original material at the inlet of the whole process, the production mode of this unit commonly changes. At the same time, the operations should be adjusted according to different process modes. Thus, monitoring the process and detect the variations timely is extremely important, which can not only effectively prevent accidents, but remind the operators to switch the production operations as well. However, in real process, the content of residual CO and CO_2 is measured through an expensive mass spectrometer. Therefore, a quality predictive model can be constructed for the estimation of CO and CO_2 content and then the model can be utilized for the process monitoring. Since the process dataset is huge, the proposed distributed and parallel SVI-MFA model can be established here.

In order to build the process model for monitoring and quality prediction, a big dataset with 2.1×10^5 samples are collected from the real process, which would be challenging for the traditional modeling algorithms. Besides, due to the changing of process materials, serval operating modes exist in the dataset. Thus, the proposed



(a) Integrated monitoring results of IDV 10 by SVI-MFA; (b) Monitoring results of IDV 10 by local model 1; (c) Monitoring results of IDV 10 by local model 2

Fig. 11. Monitoring results of IDV 10 from mode 1 by SVI-MFA.



(a) Integrated monitoring results of IDV 27 by SVI-MFA; (b) Monitoring results of IDV 27 by local model 1; (c) Monitoring results of IDV 27 by local model 2

Fig. 12. Monitoring results of IDV 27 from mode 3 by SVI-MFA.

Table 4
Modeling Performance of VI-GMR, VI-MFA and SVI-MFA for TE Benchmark Process.

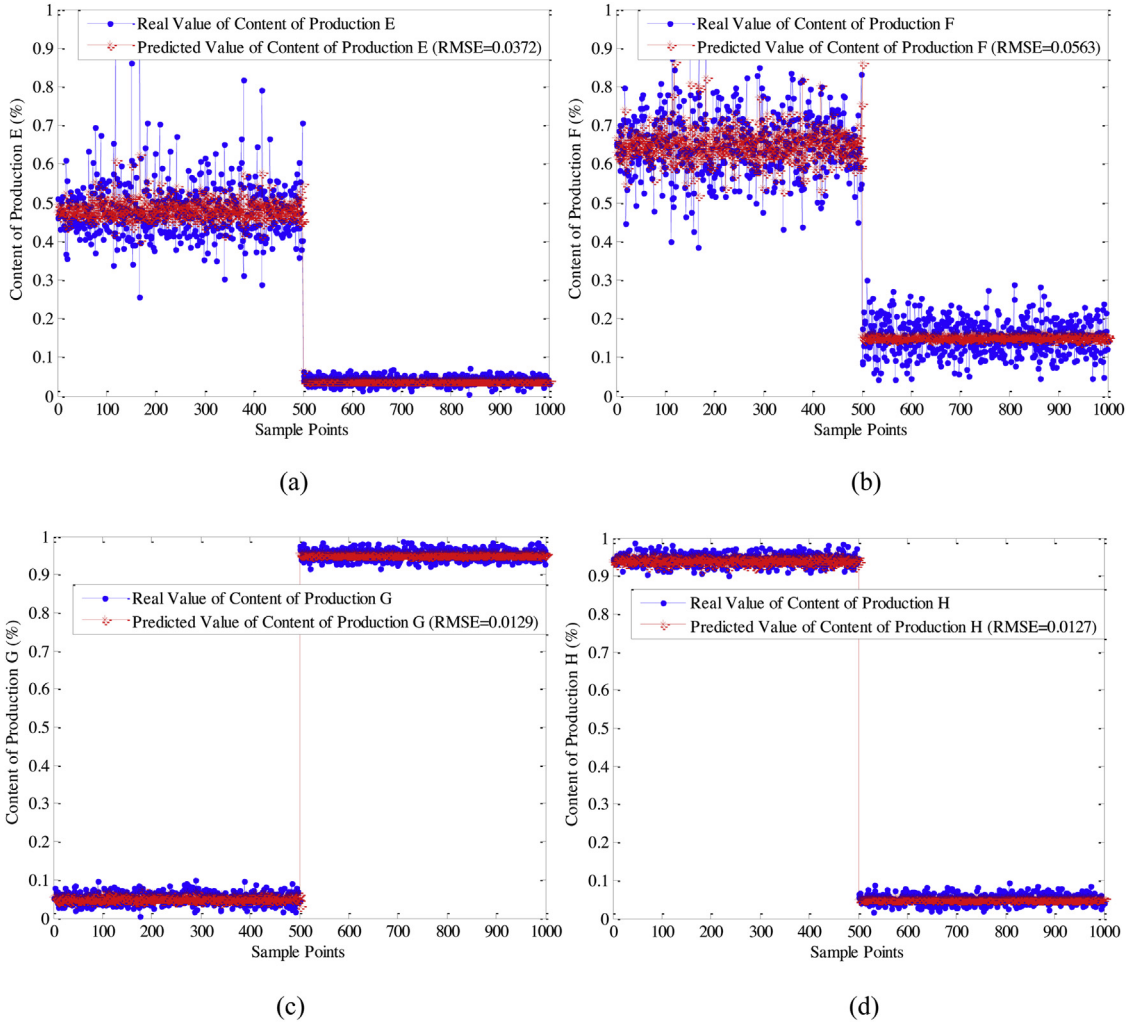
N	Model	Testing RMSE for Production G	Training Time (s)
	VI-GMR	0.0147	1878.5
	VI-MFA	0.0144	2490.3
	SVI-MFA	0.0142	**89.58**
	VI-GMR	0.0143	$> 10^4$
	VI-MFA	0.0137	$> 10^4$
	SVI-MFA	0.0137	**717.36**
	VI-GMR	0.0140	$> 10^4$
	VI-MFA	0.0136	$> 10^5$
	SVI-MFA	0.0135	**2639.8**
	VI-GMR	0.0134	$> 10^5$
	VI-MFA	0.0129	$> 3*10^5$
	SVI-MFA	0.0129	**4677.19**

Table 5
Descriptions of process instruments in Methanation Furnace Unit.

Tags	Descriptions
U1	Flow rate of process gas at MF's exit
U2	Pressure of process gas at MF's entrance
U3	Pressure of process gas into F3's entrance
U4	Temperature of upper bed at MF
U5	Temperature of middle bed at MF
U6	Temperature of lower bed at MF
U7	Temperature of process gas at MF's exit
U8	Temperature of process gas at MF's exit
U9	Temperature of synthesis gas at F2's exit
U10	Liquid level of F2
Y	Content of CO and CO2 at MF's exit

SVI-MFA model on the distributed and parallel framework can be constructed here for the modeling.

In this case, the initial number of modes is set from 3 to 10 to automatically determine the final number of modes according to the updated mixture proportions. When initial is set large enough, the final valid K will shrink to a fixed value. Detailed determination results of SVI-MFA are presented in Fig. 15. It can be seen that when the initial increases, the fitting RMSE declined and finally stay at a certain value. Meanwhile, the final will also vary in a relatively small range. According to the curves, $K = 6$ is the best choice for modeling since it is the most commonly appeared final value with a relatively small calculation burden. For the six local factor analyzers, the loading matrices are displayed through the Hinton Graphs, given in Fig. 16.



(a) Predicted results of production E by SVI-MFA; (b) Predicted results of production F by SVI-MFA; (c) Predicted results of production G by SVI-MFA; (d) Predicted results of production H by SVI-MFA

Fig. 13. Predicted results of the four quality variables in TE process by SVI-MFA.

Table 6
Descriptions of the devices in the flowchart.

Tags	Descriptions	Tags	Descriptions
Methanation Furnace	Convert CO & CO2 to CH4	E3	Heat exchanger
E1	Heat exchanger	F1	Separator
E2	Heat exchanger	F2	Separator

Since the content of residual CO and CO_2 at the outlet of the unit is the critical quality variable for the production, the proposed SVI-MFA model is constructed here for the prediction. With the large training dataset, a predictive model is established. And a testing dataset is then randomly selected from the process to test the prediction performance. Predicted results of the three compared methods (VI-GMR, VI-MFA and SVI-MFA) for testing samples are displayed in Fig 17 , Fig 18 and Fig 19 . It can be seen that the SVI-MFA provides the best prediction performance. Similarly, the training and testing operations are also implemented on VI-GMR, VI-MFA and SVI-MFA with different scales of big data. Detailed testing RMSE and training time are listed in Table 7, where the testing RMSE are decreasing for the three models as the data-size increases. It can also be seen that the testing RMSEs of the SVI-MFA are smaller than the VI-GMR and VI-MFA. Besides, the training time of SVI-MFA is much shorter than the VI-GMR and the VI-MFA.

Table 7
Modeling Performance of VI-GMR, VI-MFA and SVI-MFA for Methanation Furnace Unit.

N	Model	Testing RMSE for CO & CO2 content	Training Time (s)
	VI-GMR	0.0851	1654.5
	VI-MFA	0.0844	1997.7
	SVI-MFA	0.0846	**80.67**
	VI-GMR	0.0834	$> 10^4$
	VI-MFA	0.0826	$> 10^4$
	SVI-MFA	0.0823	**454.17**
	VI-GMR	0.0825	$> 10^4$
	VI-MFA	0.0810	$> 10^4$
	SVI-MFA	0.0806	**848.7**
	VI-GMR	0.0798	$> 10^5$
	VI-MFA	0.0761	$> 10^5$
	SVI-MFA	0.0754	**1656.9**

To evaluate the process monitoring performance of the proposed distributed and parallel SVI-MFA model, three kinds of abnormal operating conditions are used, in which 1000 samples are collected in each mode. It should be mentioned that the fault occurs

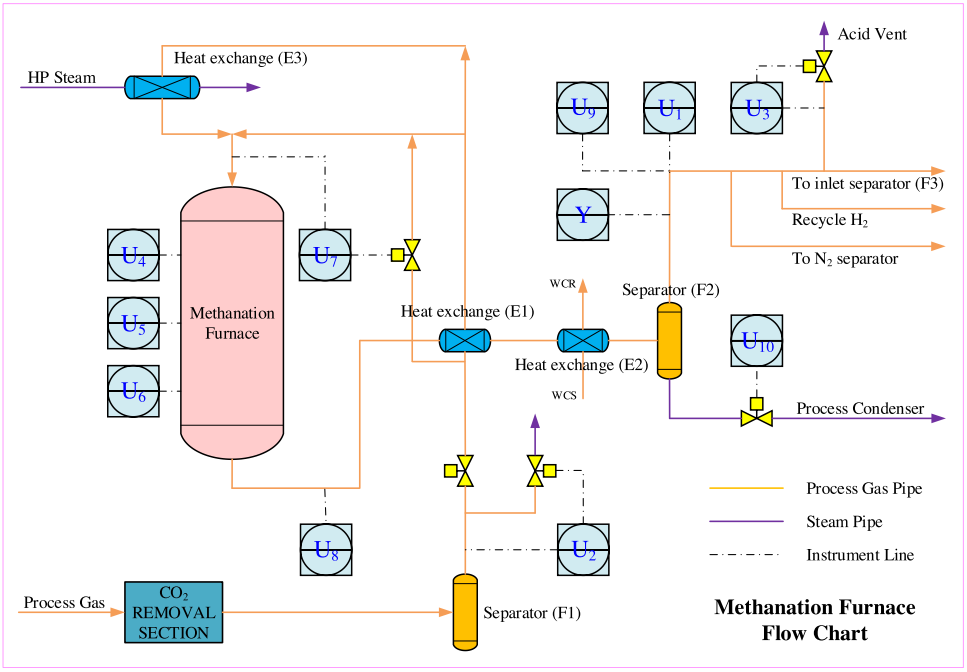


Fig. 14. Flowchart of the Methanation Furnace Unit.

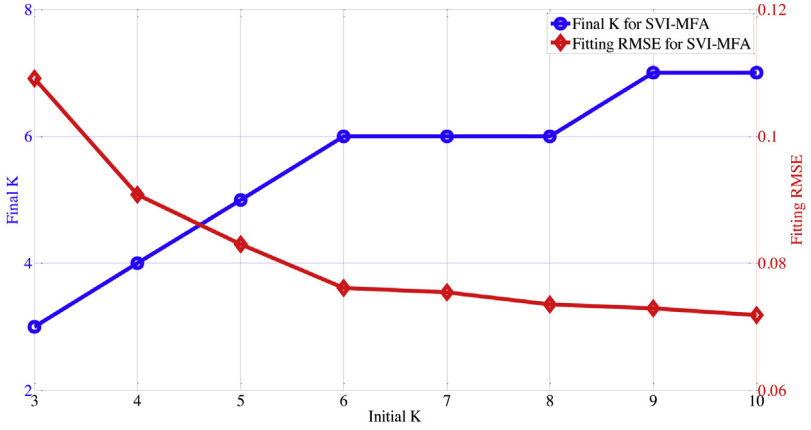


Fig. 15. Determining the number of local models for real industrial case.

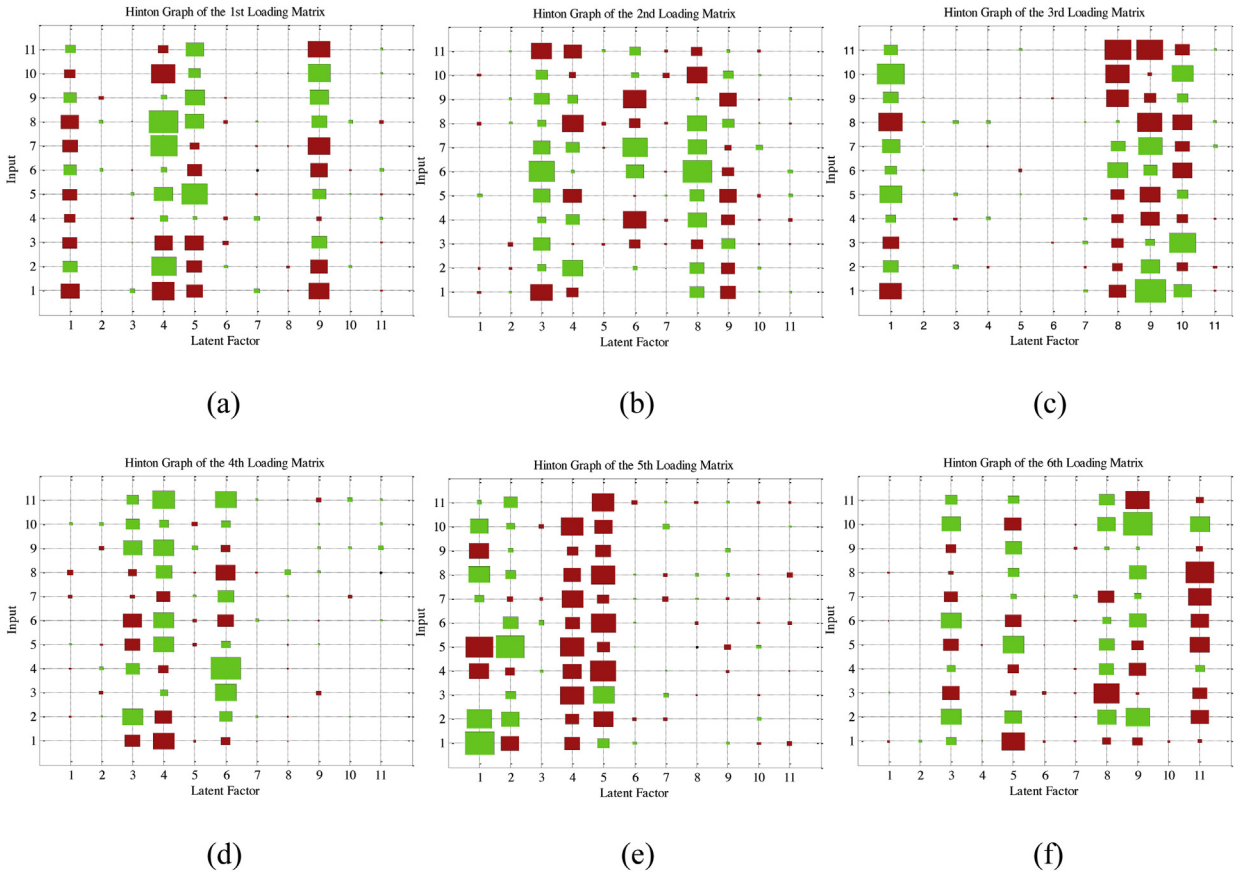
after 500 samples in each mode. The significance level is also set as 0.01. Finally, the detailed process monitoring results for integrated and local models are shown in Figs. 20 and 21, respectively.

6. Conclusions and future works

In this paper, a distributed parallel probabilistic learning framework is proposed based on the scalable Parameter Server architecture for big process data. The traditional VI-based mixture probabilistic models are changed into a scalable form through the SVI algorithm. Then the SVI-based mixture probabilistic models are easily deployed on the Parameter Server for a distributed and parallel computing. The big process data is divided into mini-batches and transferred to the iteration successively for the updating of parameters, which shows more efficient than the traditional VI-based algorithm. Then the VI-MFA model is taken as an example in the paper to be transformed to the SVI-MFA model and deployed on the PS framework. Then the model is utilized for process monitoring and quality prediction applications. The numerical example has demonstrated the feasibility and efficiency of the proposed SVI-MFA model for modeling with a huge dataset. And then the TE benchmark process and the Methanation Furnace Unit process have demonstrated the effectiveness of the proposed distributed parallel probabilistic modeling framework for big process data in the two industrial applications.

As real industrial processes commonly exhibit strong dynamics, the order (or sequence) of the measurements play a significant role in capturing process behavior. It seems that the stochastic optimization algorithm may break the dynamic feature between samples. However, it should be mentioned that the proposed framework can also be suitable for dynamic probabilistic models [35,37] through a serialization operation. Serialization is the most commonly used method to maintain the self-correlation feature between samples. For most time sequence models, while using stochastic optimization methods to solve model parameters, the serialization operation is conducted on the data samples [38].

In order not to disturb the time sequence of the samples and extract the dynamic feature, we can conduct a serialization oper-



(a) Hinton Graph of the 1st loading matrix by SVI-MFA; (b) Hinton Graph of the 2nd loading matrix by SVI-MFA; (c) Hinton Graph of the 3rd loading matrix by SVI-MFA; (d) Hinton Graph of the 4th loading matrix by SVI-MFA; (e) Hinton Graph of the 5th loading matrix by SVI-MFA; (f) Hinton Graph of the 6th loading matrix by SVI-MFA

Fig. 16. Hinton Graphs of the loading matrices obtained by SVI-MFA.

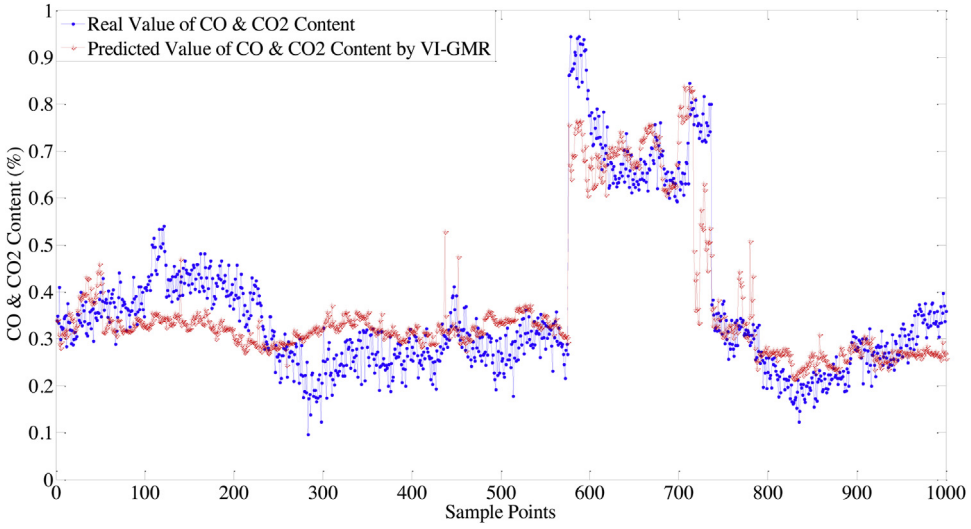


Fig. 17. Predicted results of CO & CO_2 content by VI-GMR.

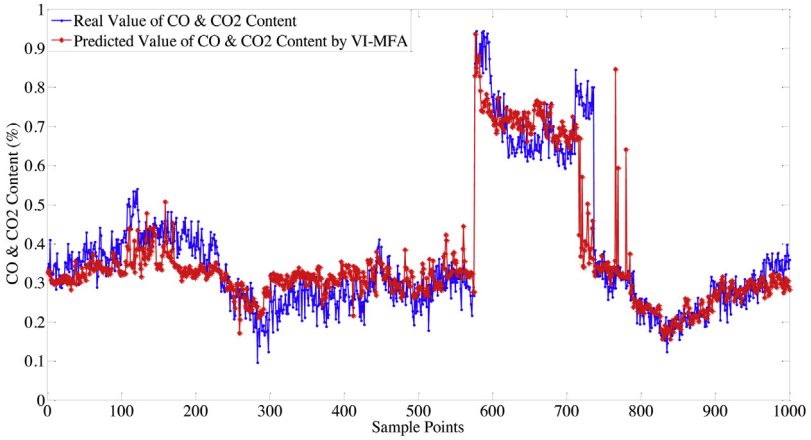


Fig. 18. Predicted results of CO & CO_2 content by VI-MFA.

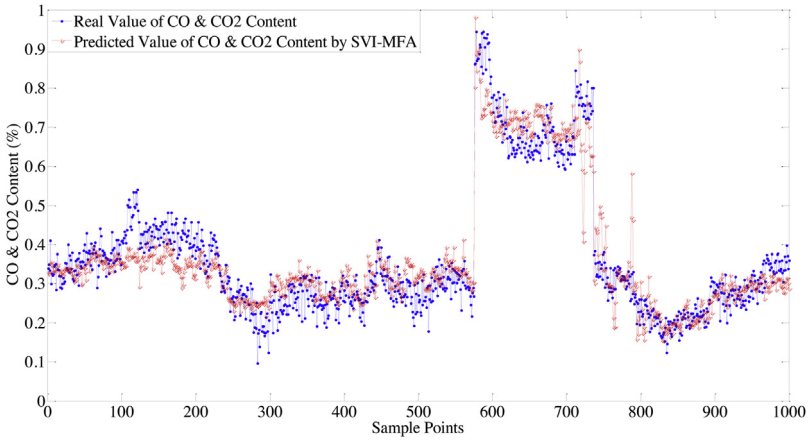


Fig. 19. Predicted results of CO & CO_2 content by SVI-MFA.

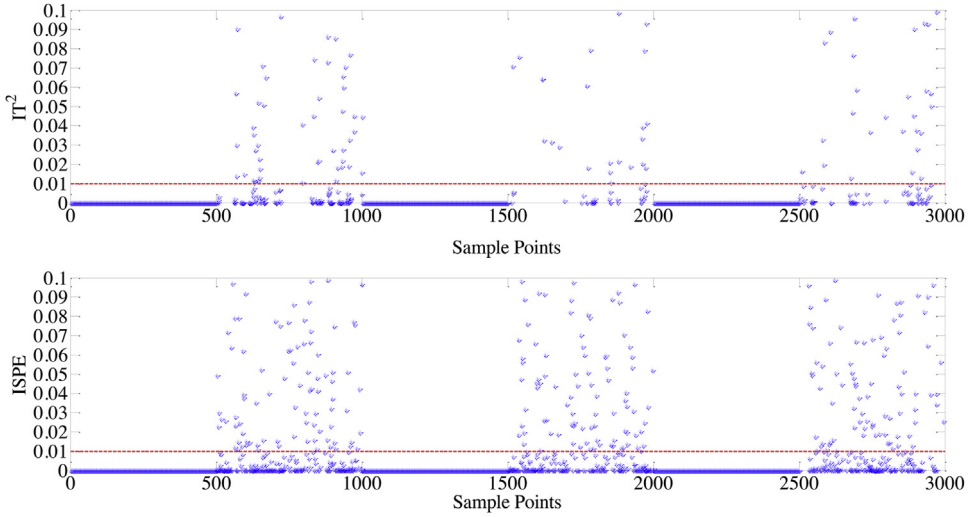
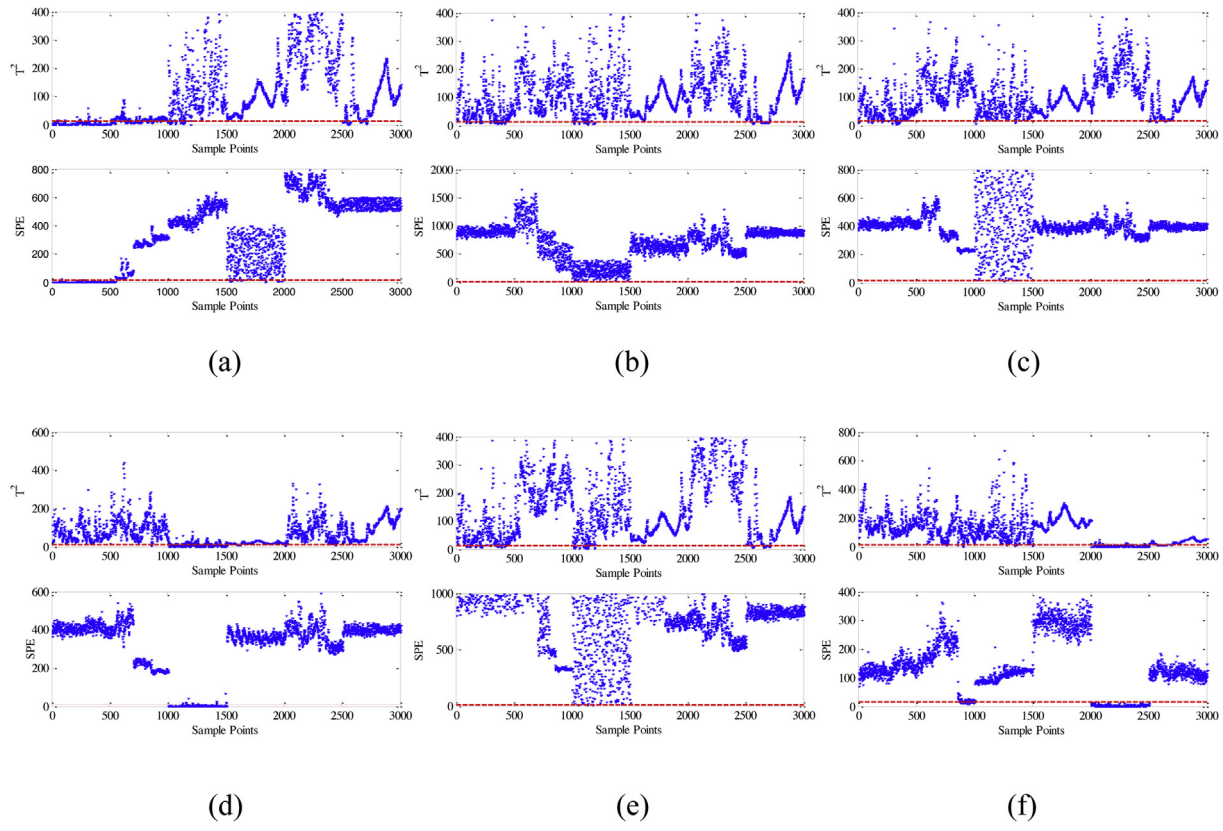


Fig. 20. Integrated monitoring results of Methanation Furnace in three operation modes by SVI-MFA.

ation on the dataset before training the model, which is formed as follows: Assuming the original dataset is represented by $\mathbf{D} = \{\mathbf{x}_i\}$, $i = 1, 2, ..., N$, where $\mathbf{x}_i$ denotes process variable and N represents the total number of training samples. Then a time window whose width equals the time order length t is utilized to scan the original dataset $\boldsymbol{D}$. As a result, the serialized dataset $\mathbf{D^S} = \left\{\mathbf{x}_j^{\mathbf{S}}\right\}$, where $\mathbf{x}_j^{\mathbf{S}} = \{\mathbf{x}_{j-t+1}, \mathbf{x}_{j-t+2}, ..., \mathbf{x}_j\}$, $j = t, t+1, ..., N$ is obtained. A flowchart of the serialization operation with $t = 3$ for original dataset is depicted in Fig. 22. It can be seen from the figure that the original dataset $\boldsymbol{D}$ is transformed into a serialized dataset $\boldsymbol{D^s}$ without breaking up the time sequence between samples. In other words, even if the sequence of the $\boldsymbol{D^s}$ is disrupted, the dynamic fea-



(a) Monitoring results by local model 1; (b) Monitoring results by local model 2; (c) Monitoring results by local model 3; (d) Monitoring results by local model 4; (e) Monitoring results by local model 5; (f) Monitoring results by local model 6

Fig. 21. Monitoring results of the local models by SVI-MFA in Methanation Furnace Unit.

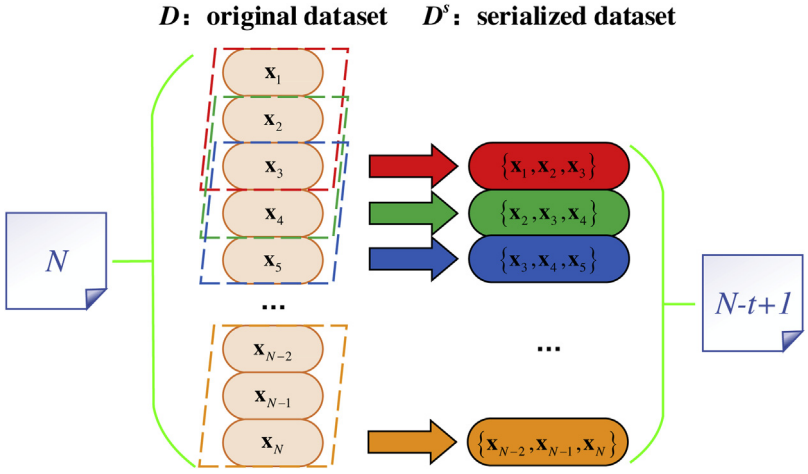


Fig. 22. The flowchart of serialization operation with $t = 3$.

ture contained in the original sequence samples will still be kept. It also can be seen that the length of dataset is reduced from N to N-t+1. The loss of data length can be ignored in a large scale of dataset.

Since the motivation of the paper is to propose a distributed and parallel probabilistic modeling framework, the basic SVI-MFA model is taken as a basic example. In future work, one may take more features of the process data into consideration such as dynamics and nonlinearity, thus more specific and detailed models on dynamics and nonlinearity can be developed under this framework to cope with big process data [39–41]. In addition, more potential applications can be made under this distributed parallel modeling framework, except for the monitoring and quality prediction applications made in the current paper.

Declaration of interests

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.

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) (61833014, 61722310), the Natural Science Foundation of Zhejiang Province (LR18F030001), and the Fundamental Research Funds for the Central Universities under grant 2018XZZX002-09.

Appendix A

Exponential family distributions

The probability density function (pdf) of exponential family distributions can be expressed in the following exponential function form:

$$p\left(\mathbf{x}|\theta\right) = h(\mathbf{x})\exp\left\{\boldsymbol{\eta}\left(\theta\right)^T\mathbf{t}(\mathbf{x}) - \mathbf{a}(\boldsymbol{\eta})\right\} \tag{78}$$

Here the elements, $\mathbf{h}(\bullet)$ and $\mathbf{a}(\bullet)$ are the base measure and log-normalizer, $\boldsymbol{\eta}(\bullet)$ and $\mathbf{t}(\bullet)$ respectively denotes the natural parameter and sufficient statistics. The exponential family distributions that used in the paper are listed as follows:

(1) Dirichlet distribution:

$$p(\boldsymbol{\pi}|\boldsymbol{\alpha}) = \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma(\alpha_k)}\prod_{k=1}^{K}\pi_k^{\alpha_k-1} \tag{79}$$

Base measure: $h(\boldsymbol{\pi}) = 1$;
Natural parameters: $\boldsymbol{\eta}(\boldsymbol{\alpha}) = [\alpha_1 - 1, \alpha_2 - 1, ..., \alpha_K - 1]^T$;
Sufficient statistics: $\mathbf{t}(\boldsymbol{\pi}) = [\ln\pi_1, \ln\pi_2, ..., \ln\pi_K]^T$;
Log-normalizer: $\mathbf{a}(\boldsymbol{\eta}) = \sum_{i=1}^{K}\ln\Gamma(\alpha_i) - \ln\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)$, where $\Gamma(\bullet)$ represents the gamma function.

(2) Categorical distribution:

$$p(\mathbf{c}|\boldsymbol{\pi}) = [\pi_1, \pi_2, ..., \pi_K][I(c=1), I(c=2), ..., I(c=K)]^T \tag{80}$$

Base measure: $h(\mathbf{c}) = 1$;
Natural parameters: $\boldsymbol{\eta}(\boldsymbol{\pi}) = [\ln\pi_1, \ln\pi_2, ..., \ln\pi_K]^T$;
Sufficient statistics: $\mathbf{t}(\mathbf{c}) = [I(c=1), I(c=2), ..., I(c=K)]^T$;
Log-normalizer: $\mathbf{a}(\boldsymbol{\eta}) = 0$.

(3) Normal distribution:

$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \left((2\pi)^D\left|\Sigma\right|\right)^{-\frac{1}{2}}e^{-\frac{(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}{2}} \tag{81}$$

Base measure: $h(\mathbf{c}) = (2\pi)^{-\frac{D}{2}}$;
Natural parameters: $\boldsymbol{\eta}(\boldsymbol{\mu}\,\boldsymbol{\Sigma}) = \left[\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} - \frac{1}{2}\boldsymbol{\Sigma}^{-1}\right]^T$;
Sufficient statistics: $\mathbf{t}(\mathbf{x}) = \left[\mathbf{x}, \mathbf{x}^T\mathbf{x}\right]^T$;
Log-normalizer: $\mathbf{a}(\boldsymbol{\eta}) = \frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \frac{1}{2}\ln\left|\boldsymbol{\Sigma}\right|$.

(4) Gamma distribution:

$$p\left(\boldsymbol{\tau}|\beta, \alpha\right) = \frac{\beta^\alpha}{\Gamma(\alpha)}\boldsymbol{\tau}^{\alpha-1}e^{-\beta\mathbf{x}}, \boldsymbol{\tau} > 0 \tag{82}$$

Base measure: $h(\boldsymbol{\tau}) = 1$;
Natural parameters: $\boldsymbol{\eta}\left(\beta, \alpha\right) = \left[\alpha - 1, -\beta\right]^T$;
Sufficient statistics: $\mathbf{t}(\boldsymbol{\tau}) = [\ln\boldsymbol{\tau}, \boldsymbol{\tau}]^T$;
Log-normalizer: $\mathbf{a}(\boldsymbol{\eta}) = \ln\Gamma(\alpha) - \alpha\ln\beta$.

References

[1] Z. Ge, Z. Song, S.X. Ding, B. Huang, Data mining and analytics in the process industry: the role of machine learning, IEEE Access 5 (2017) 20590–20616.
[2] S. Yin, X. Li, H. Gao, O. Kaynak, Data-based techniques focused on modern industry: an overview, IEEE Trans. Ind. Electron. 62 (1) (2015) 657–667.
[3] Z. Ge, Process data analytics via probabilistic latent variable models: a tutorial review, Ind. Eng. Chem. Res. 57 (2018) 12646–12661.
[4] Z. Ge, J. Chen, Plant-wide industrial process monitoring: a distributed modeling framework, IEEE Trans. Industr. Inform. 12 (2016) 310–321.
[5] Y. Liu, C. Yang, Z. Gao, Y. Yao, Ensemble deep kernel learning with application to quality prediction in industrial polymerization processes, Chemom. Intell. Lab. Syst. 174 (2018) 15–21.
[6] Q. Liu, S. Qin, T. Chai, Decentralized fault diagnosis of continuous annealing processes based on multilevel PCA, IEEE Trans. Autom. Sci. Eng. 10 (3) (2013) 687–698.
[7] L. Yao, Z. Ge, Moving window adaptive soft sensor for state shifting process based on weighted supervised latent factor analysis, Control Eng. Pract. 61 (2017) 72–80.
[8] X. Yuan, Z. Ge, Z. Song, Locally weighted kernel principal component regression model for soft sensing of nonlinear time-variant processes, Ind. Eng. Chem. Res. 53 (35) (2014) 13736–13749.
[9] K. Fujiwara, M. Kano, S. Hasebe, et al., Soft sensor development using correlation based just in time modeling, AIChE J. 55 (7) (2009) 1754–1765.
[10] R. Grbić, D. Slišković, P. Kadlec, Adaptive soft sensor for online prediction and process monitoring based on a mixture of Gaussian process models, Comput. Chem. Eng. 58 (2013) 84–97.
[11] L. Yao, Z. Ge, Locally weighted prediction methods for latent factor analysis with supervised and semi-supervised process data, IEEE Trans. Autom. Sci. Eng. 14 (1) (2017) 126–138.
[12] X. Yuan, Z. Ge, B. Huang, et al., A probabilistic just-in-time learning framework for soft sensor development with missing data, IEEE Trans. Control. Syst. Technol. 25 (3) (2017) 1124–1132.
[13] Z. Ge, X. Chen, Dynamic probabilistic latent variable model for process data modeling and regression application, IEEE Trans. Control. Syst. Technol. 27 (2019) 323–331.
[14] Z. Ge, Z. Song, Maximum-likelihood mixture factor analysis model and its application for process monitoring, Chemom. Intell. Lab. Syst. 102 (1) (2010) 53–61.
[15] J. Zhu, Z. Ge, Z. Song, Variational Bayesian Gaussian mixture regression for soft sensing key variables in non-Gaussian industrial processes, IEEE Trans. Control. Syst. Technol. 25 (3) (2017) 1092–1099.
[16] S.J. Qin, Process data analytics in the era of big data, Aiche J. 60 (2014) 3092–3100.
[17] J. Zhu, Z. Ge, Z. Song, F. Gao, Review and big data perspectives on robust data mining approaches for industrial process modeling with outliers and missing data, Annual Reviews in Control 46 (2018) 107–133.
[18] M. Gaggero, S. Leo, S. Manca, F. Santoni, O. Schiaratura, G. Zanetti, Parallelizing bioinformatics applications with MapReduce, Cloud Comput. Appl. (2008) 22–23.
[19] J. Lin, C. Dyer, Data-intensive text processing with MapReduce, Synth. Lect. Hum. Lang. Technol. 3 (1) (2010) 1–177.
[20] L. Yao, Z. Ge, Big data quality prediction in the process industry: a distributed parallel modeling framework, J. Process Control 68 (2018) 1–13.
[21] J. Zhu, Z. Ge, Z. Song, Distributed parallel PCA for modeling and monitoring of large-scale plant-wide processes with big data, IEEE Trans. Industr. Inform. 13 (4) (2017) 1877–1885.
[22] J. Zhu, Y. Yao, D. Li, F. Gao, Monitoring big process data of industrial plants with multiple operating modes based on Hadoop, J. Taiwan Inst. Chem. Eng. 91 (2018) 10–21.
[23] M. Li, Scaling distributed machine learning with the parameter server, International Conference on Big Data Science and Computing (2014) 3.
[24] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, A. Ng, Large scale distributed deep networks, in: International Conference on Neural Information Processing Systems, 2012, pp. 1223–1231.
[25] M. Li, D.G. Andersen, A.J. Smola, K. Yu, Communication efficient distributed machine learning with the parameter server, in: Advances in Neural Information Processing Systems (NIPS), 2014, pp. 19–27.
[26] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (3) (1951) 400–407.
[27] L. Bottou, Large-scale machine learning with stochastic gradient descent, Proceedings of COMPSTAT'2010 (2010) 177–186.
[28] Y. Liu, Y. Fan, J. Chen, Flame images for oxygen content prediction of combustion systems using DBN, Energy Fuels 31 (8) (2017) 8776–8783.
[29] Z. Ghahramani, M. Beal, Variational inference for Bayesian mixtures of factor analyzers, Adv. Neural Inf. Process. Syst. (2000) 449–455.
[30] M.D. Hoffman, D.M. Blei, C. Wang, J. Paisley, Stochastic variational inference, J. Mach. Learn. Res. 14 (1) (2013) 1303–1347.
[31] W. Shao, L. Yao, Z. Ge, Z. Song, Parallel computing and SGD based DPMM for soft sensor development with large-scale semisupervised data, IEEE Trans. Ind. Electron. 66 (2019) 6362–6373, http://dx.doi.org/10.1109/TIE.2018.2874589.

[32] Y. Zhang, X. Tian, P. Ren, An adaptive bilateral filter based framework for image denosing, Neurocomputing 140 (2014) 299–316.
[33] L. Zhou, J. Zheng, Z. Ge, Z. Song, S. Shan, Multimode process monitoring based on switching autoregressive dynamic latent variable model, IEEE Trans. Ind. Electron. 65 (2018) 8184–8194, http://dx.doi.org/10.1109/TIE.2018.2803727.
[34] L. Yao, Z. Ge, Deep learning of semi-supervised process data with hierarchical extreme learning machine and soft Sensor application, IEEE Trans. Ind. Electron. 65 (2018) 1490–1498.
[35] L. Zhou, G. Li, Z. Song, et al., Autoregressive dynamic latent variable models for process monitoring, IEEE Trans. Control. Syst. Technol. 25 (1) (2017) 366–373.
[36] A. Bathelt, N.L. Ricker, M. Jelali, Revision of the Tennessee Eastman process model, IFAC Papers-online 48 (8) (2015) 309–314.
[37] Q. Wen, Z. Ge, Z. Song, Data-based linear Gaussian state-space model for dynamic process monitoring, AIChE J. 58 (12) (2012) 3763–3776.
[38] Q. Sun, Z. Ge, Probabilistic sequential network for deep learning of complex process data and Soft sensor application, IEEE Trans. Industr. Inform. (2018), http://dx.doi.org/10.1109/TII.2018.2869899.
[39] L. Yao, Z. Ge, Scalable semi-supervised GMM for big data quality prediction in multimode processes, IEEE Trans. Industr. Electron. 66 (5) (2019) 3681–3692.
[40] L. Yao, Z. Ge, Distributed parallel deep learning of hierarchical extreme learning machine for multimode quality prediction with big process data, Eng. App. Artif. Intell. 81 (2019) 450–465.
[41] X. Zhang, Z. Ge, Local parameter optimization of LSSVM for industrial soft sensing with big data and cloud implementation, IEEE Trans. Industr. Inform. (2019), http://dx.doi.org/10.1109/TII.2019.2900479.