Developing Soft-Sensor Models Using Latent Dynamic Variational Autoencoders

Yi Shan Lee, Sai Kit Ooi, Dave Tanny, Junghui Chen*

Department of Chemical Engineering, Chung-Yuan Christian University, Tao-Yuan, Taiwan, R.O.C
*(*e-mail:* jason@wavenet.cycu.edu.tw)

Abstract: Quality variables, which are usually measured offline, play important roles in describing process behaviors. However, online data obtained from soft sensors are significant as they provide accurate and immediate information. The reliability of online soft sensors is questionable due to changes in sensors, equipment, raw material availability, and operation conditions. In addition, chemical plants have dynamic properties and complex correlations amidst a large number of process variables. This causes most of the predictions obtained from steady-state soft sensors to be inaccurate in representing the particular chemical process. In this paper, the latent dynamic variational autoencoder is proposed to provide an estimation model and supervise soft-sensors. The input data are encoded in the latent space to remove underlying noises and disturbances in the data. Afterward, the dynamical properties are learned in the latent space through the bi-directional recurrent neural network, whose output (latent variable) is used to reconstruct back the input data. A simulation case study is conducted to show the effectiveness of the proposed method.

Keywords: Dynamic Nonlinear Process, Soft-Sensor Prediction, Supervised, Variational Autoencoder.

1. INTRODUCTION

To obtain accurate and immediate information in a continuous process, soft-sensors are usually used to approximate the difficult-to-measure quality variables. However, the lifetime of soft-sensors is limited because the sensor, equipment, feedstock availability, or operating conditions may change from time to time. Although data-driven soft-sensor models can be updated based on the recently available training data, process dynamics, outliers and nonlinearity affect the prediction performance significantly. Therefore, maintaining the reliability of the soft-sensor is crucial and demanding in manufacturing nowadays.

Using a moving window mimicking the concept of autoregressive moving average exogenous time series model is a common method for data-driven models to learn dynamic behaviors of the system. The moving window simply extends the data to the matrix form to include the data at each specific time point within a certain time range. This allows principal component analysis (PCA) and partial least squares (PLS) to be extended to dynamic principal component analysis (DPCA) (Ku et al., 1995) and dynamic partial least squares (DPLS) (Kano et al., 1998) respectively, but the usage of DPCA and DPLS is limited to linear processes. The nonlinear dynamic models, such as dynamic kernel principal component analysis (DKPCA) (Jia et al., 2010) and dynamic kernel partial least squares (DKPLS) (Jia & Zhang, 2016), are developed using kernel tricks to represent nonlinear dynamic processes. However, applications of DKPCA and DKPLS are limited as huge historical data in chemical processes cause a large computational load.

Alternatively, the recursive neural network (RNN) is developed to learn the dynamic characteristics of processes (Fig. 1). The dynamic properties are learned relying on the cell states as inputs along the input sequence are fed into the RNN nodes. RNN allows a deep nonlinear representation of the dynamic system. As chemical plants exhibit stochastic nature because of noises and disturbances, the RNN structure can be extended from the deterministic characteristic to the probabilistic form.

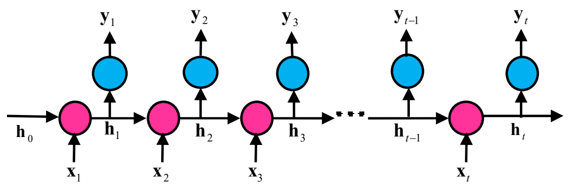


Fig. 1. Recurrent Neural Network (RNN) structure.

Recent advances in artificial intelligence have shown growth in variations of variational autoencoder (VAE) model extended to learn the dynamic properties of the system, particularly in robotic motion, music, and video in computer science. The popular extension of VAE typically includes RNN in the encoder and decoder of VAE (Chen et al., 2019; Chung et al., 2015; Habibie et al., 2017) while other extension includes dynamic formulation approaches in the VAE model, such as Kalman filters (Krishnan et al., 2015; Karl et al., 2017; Fraccaro et al., 2017), dynamic movement primitives (Fraccaro et al., 2017), and the Lyapunov function (Agand et al., 2017). However, the model is developed in an unsupervised manner and the model parameters are constrained to be linear.

In this paper, a latent dynamic variational autoencoder (LDVAE) is proposed. The proposed method uses Kalman filters to model the nonlinear dynamic properties through a bi-directional RNN (bi-RNN) framework. The input data are projected onto lower dimensions for noise and disturbance removal. This can significantly reduce the computational load while learning the dynamic properties in the latent space. Moreover, the bi-RNN allows the data to move in a cyclical loop to prevent over-fitting.

The remainder of this paper is organized as follows. The next section presents the methodology of the proposed LDVAE. In Section 3, a numerical case is presented first following by an industrial ammonia synthesis case. Finally, the conclusions presented in the last section summarize the results and merits of the proposed method.

2. METHODOLOGY

2.1 Problem formulation

Given a set of time-sequential data with a total of T data which consists of process data $\mathbf{X}=\{\mathbf{x}_t \in \mathbb{R}^m, t=1,2,\cdots,T\}$ and quality data $\mathbf{Y}=\{\mathbf{y}_t \in \mathbb{R}^n, t=1,2,\cdots,T\}$, the dynamic properties of $\mathbf{X}$ and $\mathbf{Y}$ can be represented by a stochastic continuous nonlinear latent variable (LV) model in a first-order Markov manner:

$$\mathbf{z}_t = f(\mathbf{z}_{t-1},\mathbf{a}_t)+\mathbf{w}_t \qquad \mathbf{a}_t = h(\mathbf{x}_t,\mathbf{y}_t)+\mathbf{m}_t \tag{1}$$

$$\mathbf{x}_t = g_x(\mathbf{z}_t)+\mathbf{v}_t \qquad \mathbf{y}_t = g_y(\mathbf{z}_t)+\mathbf{n}_t \tag{2}$$

$\mathbf{z}_t$ is known as the state variable in dynamic systems. To avoid misunderstanding, $\mathbf{z}_t$ is referred to as LV from now on while $\mathbf{a}_t$ is known as the temporal correlation existing in the changes of the process and quality variables. The nonlinear dynamic relations are assumed to exist in LV $(\mathbf{z}_t)$, which depends on the previous latent variable $(\mathbf{z}_{t-1})$ and the current temporal changes $(\mathbf{a}_t)$. $\mathbf{z}_t$ and $\mathbf{a}_t$ can be approximated by the functions $f(\mathbf{z}_{t-1},\mathbf{a}_t)$ and $h(\mathbf{x}_t,\mathbf{y}_t)$ in Eq.(1), respectively. This approximation closely associates the latent variables with the process and quality variables, so $\mathbf{z}_t$ can represent the temporal changes/dynamic properties in the $\mathbf{x}_t$ and $\mathbf{y}_t$ at each time point. With the corresponding LV, the process and the quality variables are formulated by the unknown nonlinear functions $g_x(\mathbf{z}_t)$ and $g_y(\mathbf{z}_t)$. $\mathbf{m}_t$ and $\mathbf{w}_t$ are the noises associated with the dynamic changes in $\mathbf{a}_t$ and $\mathbf{z}_t$ respectively while $\mathbf{v}_t$ and $\mathbf{n}_t$ represent the noise of $\mathbf{x}_t$ and $\mathbf{y}_t$ respectively. All the noises are assumed to follow zero-mean Gaussian distributions:

$$p(\mathbf{m}_t)\sim N(0,\mathbf{I}),\ p(\mathbf{w}_t)\sim N(0,\mathbf{I}) \tag{3}$$

$$p(\mathbf{v}_t)\sim N(0,\mathbf{\Gamma}),\ p(\mathbf{n}_t)\sim N(0,\Upsilon) \tag{4}$$

where the covariance matrices of $\mathbf{m}_t$ and $\mathbf{w}_t$ can be assumed to be an identity matrix. ~~which does not affect the model representation~~ Meanwhile, the covariance matrices of $\mathbf{v}_t$ and $\mathbf{n}_t$ are $\mathbf{\Gamma}$ and Υ. With the conditional probability representation, Eq.(1) & (2) can also be re-written as:

$$\begin{aligned} p(\mathbf{z}_t\,|\,\mathbf{z}_{t-1}) &\sim N(f(\mathbf{z}_{t-1}),\mathbf{I}) \\ p(\mathbf{a}_t\,|\,\mathbf{x}_t,\mathbf{y}_t) &\sim N(h(\mathbf{x}_t,\mathbf{y}_t),\mathbf{I}) \end{aligned} \tag{5}$$

$$\begin{aligned} p(\mathbf{x}_t\,|\,\mathbf{z}_t) &\sim N(g_x(\mathbf{z}_t),\mathbf{\Gamma}) \\ p(\mathbf{y}_t\,|\,\mathbf{z}_t) &\sim N(g_y(\mathbf{z}_t),\Upsilon) \end{aligned} \tag{6}$$

$p(\mathbf{z}_t\,|\,\mathbf{z}_{t-1})$ is known as the transition distribution showing the changes of the latent variable from the previous time point till the current time point while $p(\mathbf{x}_t\,|\,\mathbf{z}_t)$ and $p(\mathbf{y}_t\,|\,\mathbf{z}_t)$ are known as emission distributions for process and quality variables. As $p(\mathbf{z}_t\,|\,\mathbf{z}_{t-1})$ is modeled at the previous time point, an initial latent variable distribution is assumed to be

$$p(\mathbf{z}_0)\sim N(\boldsymbol{\mu}_0,\mathbf{\Lambda}_0) \tag{7}$$

For process modeling, all the parameters $\{\boldsymbol{\mu}_0,\mathbf{\Lambda}_0,h(\mathbf{x}_t,\mathbf{y}_t),f(\mathbf{z}_{t-1},\mathbf{a}_t),g_x(\mathbf{z}_t),g_y(\mathbf{z}_t),\mathbf{\Gamma},\Upsilon\}$ can be estimated by the EM algorithm (E-Step and M-Step). These two steps are repeated until the parameters converge and the condition of the loss function has been achieved.

2.2 Latent dynamic variational autoencoder (LDVAE)

The whole sequential dataset of process and quality variables are divided into K data samples of serial vectors (moving window), with τ sequential data in each data window respectively as $\mathbf{X}_k=[\mathbf{x}_{t-\tau+1}\ \cdots\ \mathbf{x}_{t-1}\ \mathbf{x}_t]$ and $\mathbf{Y}_k=[\mathbf{y}_{t-\tau+1}\ \cdots\ \mathbf{y}_{t-1}\ \mathbf{y}_t]$. To train the LDVAE model, the objective function is defined to maximize the marginal distribution of the process and quality data within the sequences for each moving window.

$$J=\max\sum_{k=1}^{K}\log p(\mathbf{X}_k,\mathbf{Y}_k) \tag{8}$$

To clarify this point, only one moving window with τ data is used instead of using the whole sequence of the process and quality data. The window index k is removed in the following derivations. To describe the way to infer latent variables from the process and quality variables, the posterior distributions are defined as follows:

$$q_\phi(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})=\prod_{t=1}^{\tau}q_\phi(\mathbf{z}_t\,|\,\mathbf{z}_{t-1},\mathbf{a}_t)\,q_\phi(\mathbf{a}_t\,|\,\mathbf{x}_t,\mathbf{y}_t)\,q_\phi(\mathbf{z}_0) \tag{9}$$

$$p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})=\prod_{t=1}^{\tau}p_{\boldsymbol{\theta}}(\mathbf{z}_t\,|\,\mathbf{z}_{t-1},\mathbf{a}_t)\,p_{\boldsymbol{\theta}}(\mathbf{a}_t\,|\,\mathbf{x}_t,\mathbf{y}_t)\,p_{\boldsymbol{\theta}}(\mathbf{z}_0) \tag{10}$$

From the definition of Kullback-Leibler (KL) divergence (Eq.(11)), KL is used to compare the similarity between $q_\phi(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})$ and the true posterior $p_\theta(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})$.

$$\begin{aligned} &KL\left(q_\phi(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})\,\|\,p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})\right)= \\ &E_{q_\phi(\mathbf{Z},\mathbf{A}|\mathbf{X},\mathbf{Y})}\left[\log q_\phi(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})-\log p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})\right] \end{aligned} \tag{11}$$

Through the Bayes' theorem, the true posterior $p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})$ can be represented by

$$p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\,|\,\mathbf{X},\mathbf{Y})=\frac{\prod_{t=1}^{\tau}p_{\boldsymbol{\theta}}(\mathbf{x}_t\,|\,\mathbf{z}_t)\,p_{\boldsymbol{\theta}}(\mathbf{y}_t\,|\,\mathbf{z}_t)\,p_{\boldsymbol{\theta}}(\mathbf{z}_t\,|\,\mathbf{z}_{t-1},\mathbf{a}_t)\,p_{\boldsymbol{\theta}}(\mathbf{a}_t)\,p_{\boldsymbol{\theta}}(\mathbf{z}_0)}{\prod_{t=1}^{\tau}p_{\boldsymbol{\theta}}(\mathbf{x}_t,\mathbf{y}_t)} \tag{12}$$

The process variable $\mathbf{X}$ and quality variable $\mathbf{Y}$ are functions of the latent variable $\mathbf{Z}$. The dynamic correlations are learned in the latent space within each time sequence. By substituting

Eq.(9) and (12) into Eq.(11), the KL divergence between the two posterior distributions can be rewritten as

$$KL\left(q_\phi(\mathbf{Z},\mathbf{A}\mid\mathbf{X},\mathbf{Y})\,\|\,p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\mid\mathbf{X},\mathbf{Y})\right)=E_{q_\phi(\mathbf{Z},\mathbf{A}|\mathbf{X},\mathbf{Y})}$$
$$\left[\log\prod_{t=1}^{\tau}p_{\boldsymbol{\theta}}(\mathbf{x}_t,\mathbf{y}_t)-\log\prod_{t=1}^{\tau}p_\theta(\mathbf{y}_t\mid\mathbf{z}_t)-\log\prod_{t=1}^{\tau}p_\theta(\mathbf{x}_t\mid\mathbf{z}_t)\right]$$
$$+KL\left[q_\phi(\mathbf{z}_0)\,\|\,p_\theta(\mathbf{z}_0)\right] \tag{13}$$
$$+\sum_{t=1}^{\tau}KL\left[q_\phi(\mathbf{z}_t\mid\mathbf{z}_{t-1},\mathbf{a}_t)q_\phi(\mathbf{a}_t\mid\mathbf{x}_t,\mathbf{y}_t)\,\|\,p_\theta(\mathbf{z}_t\mid\mathbf{z}_{t-1},\mathbf{a}_t)p_\theta(\mathbf{a}_t)\right]$$

As $p_{\boldsymbol{\theta}}(\mathbf{x}_t,\mathbf{y}_t)$ is independent of the latent variable $\mathbf{z}_t$ and $\mathbf{a}_t$, it can be taken out from the expectation of $E_{q_\phi(\mathbf{Z},\mathbf{A}|\mathbf{X},\mathbf{Y})}$. Assume $KL[q_\phi(\mathbf{Z},\mathbf{A}\mid\mathbf{X},\mathbf{Y})\,\|\,p_{\boldsymbol{\theta}}(\mathbf{Z},\mathbf{A}\mid\mathbf{X},\mathbf{Y})]$ is equal to zero, which makes the posterior distribution similar to the true posterior. The variational lower bound term L can be maximized instead. It would yield the same definition of maximizing the marginal distribution:

$$L=\sum_{t=1}^{\tau}\log E_{q_\phi(\mathbf{Z},\mathbf{A}|\mathbf{X},\mathbf{Y})}\prod_{t=1}^{\tau}p_{\boldsymbol{\theta}}(\mathbf{x}_t,\mathbf{y}_t)$$
$$=\sum_{t=1}^{\tau}E_{q_\phi(\mathbf{Z},\mathbf{A}|\mathbf{X},\mathbf{Y})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}_t\mid\mathbf{z}_t)\right]+\sum_{t=1}^{\tau}E_{q_\phi(\mathbf{Z},\mathbf{A}|\mathbf{X},\mathbf{Y})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{y}_t\mid\mathbf{z}_t)\right] \tag{14}$$
$$-\sum_{t=1}^{\tau}KL\left(q_\phi(\mathbf{z}_t\mid\mathbf{z}_{t-1},\mathbf{a}_t)q_\phi(\mathbf{a}_t\mid\mathbf{x}_t,\mathbf{y}_t)\,\|\,p_{\boldsymbol{\theta}}(\mathbf{z}_t\mid\mathbf{z}_{t-1},\mathbf{a}_t)p_{\boldsymbol{\theta}}(\mathbf{a}_t)\right)$$
$$-KL\left(q_\phi(\mathbf{z}_0)\,\|\,p_{\boldsymbol{\theta}}(\mathbf{z}_0)\right)$$

Based on Eq.(14), the LDVAE structure is shown in Fig. 2. The formulation of the loss function in Eq.(14) only accounts for single-window data. To take the whole dataset sequence into account, the formulation needs to be extended for the whole K window data.

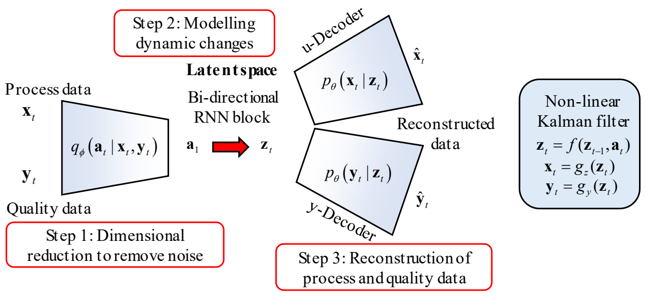

Step 2: Modelling dynamic changes
Latent space
Bi-directional RNN block
Process data
$\mathbf{x}_t$
$q_\phi(\mathbf{a}_t\mid\mathbf{x}_t,\mathbf{y}_t)$
$\mathbf{y}_t$
Quality data
$\mathbf{a}_t$
$\mathbf{z}_t$
u-Decoder
$p_\theta(\mathbf{x}_t\mid\mathbf{z}_t)$
$\hat{\mathbf{x}}_t$
Reconstructed data
$p_\theta(\mathbf{y}_t\mid\mathbf{z}_t)$
$\hat{\mathbf{y}}_t$
y-Decoder
Non-linear Kalman filter
$\mathbf{z}_t=f(\mathbf{z}_{t-1},\mathbf{a}_t)$
$\mathbf{x}_t=g_z(\mathbf{z}_t)$
$\mathbf{y}_t=g_y(\mathbf{z}_t)$
Step 1: Dimensional reduction to remove noise
Step 3: Reconstruction of process and quality data

Fig. 2. Model structure of the latent dynamic variational autoencoder

2.3 *The procedure of training LDVAE*

First, each data window consists of process and quality data sequences and $\mathbf{a}_t$ is the window data projected onto the latent space by the encoder $q_\phi(\mathbf{a}_t\mid\mathbf{x}_t,\mathbf{y}_t)$. The aims of encoding the observation data into the latent space are to eliminate noises and disturbances in the data and reduce the computational load required for modeling the dynamic properties of the process. The encoded variable $\mathbf{a}_t$ is taken as an input into the bi-RNN block. In RNN, the forward RNN ($\mathbf{h}_t$ cell state) is used to produce the filtered distributions $q_f(\mathbf{z}_t)$. It represents the relation from the past time point to the current time point. The backward RNN ($\tilde{\mathbf{h}}_t$ cell state) is used to produce the smoothed distributions $q_s(\mathbf{z}_t)$, which represent the relationship between the future time point and the current time point. The forward and backward RNN cell states are given by Eq.(15) with the same function ψ and their relations are shown in Fig. 3.

$$\mathbf{h}_t=\psi\left(\mathbf{h}_{t-1},\mathbf{a}_t\right)\qquad\tilde{\mathbf{h}}_{t-1}=\psi\left(\tilde{\mathbf{h}}_t,\mathbf{a}_{1:t-1}\right) \tag{15}$$

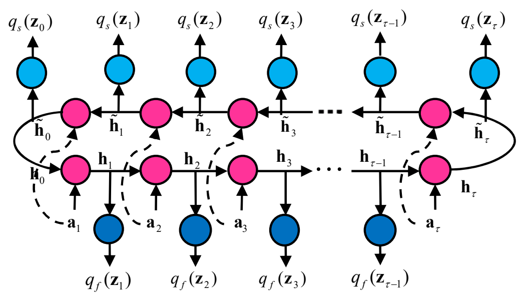


Fig. 3. Bi-RNN network structure.

As $\mathbf{h}_{t-1}$ is the accumulation of information of $\mathbf{a}_{1:t-1}$, the backward cell state in Eq.(15) can be expressed as follows:

$$\tilde{\mathbf{h}}_{t-1}=\psi\left(\tilde{\mathbf{h}}_t,\mathbf{h}_{t-1}\right) \tag{16}$$

The forward RNN network in the bi-RNN (a.k.a Kalman filtered distribution) learns the dynamic properties from the initial time point till the final time point. In the meantime, the hidden cell states in the RNN network are the inputs to produce the filtered latent variable. The final hidden cell state acts as the initial hidden cell state for backward RNN recursion. The backward RNN network in the bi-RNN (a.k.a Kalman smoothed distribution) adjusts the dynamic changes to be more robust and smoother. Like the forward RNN network, each hidden cell state is the input to produce the smoothed latent variable. The final hidden state of the backward RNN would also act as the initial forward hidden cell state for the forward RNN network. This cyclical movement not only allows the forward RNN networks to represent the dynamical changes from the past to the current time points but also dynamical changes from the future to the current time points. Using bi-RNN in the latent space can significantly reduce more computational load than the conventional RNN. Another benefit of using bi-RNN is that the filtering and smoothing action of the Kalman filter can be directly applied without specifying the state parameters although the past research (Karl et al., 2017; Fraccaro et al., 2017) states that parameters need to be specified beforehand. The trained bi-RNN is suitable for off-line or on-line modeling.

During off-line training, the data are divided into past and future data to train the filtered $q_f(\mathbf{z}_t)$ and smoothed distributions $q_s(\mathbf{z}_t)$. As the end of the backward RNN is needed to be connected to the start of the forward RNN, the initial cell state $\tilde{\mathbf{h}}_0$ of the forward RNN (the final cell state of the backward RNN) has to be assigned. With the initial value $\tilde{\mathbf{h}}_0$, the calculation of the filtered distributions is carried out following by the smoothed distributions in the backward RNN cell state. Assume that the initial prior latent distribution ($\mathbf{z}_0$) at the time frame 0 is $\mathbf{z}_0\sim N(0,\mathbf{I})$, the posterior latent variable distribution $p(\mathbf{z}_t\mid\mathbf{z}_{t-1}^s,\mathbf{a}_{t:\tau})$ is given by the smoothed transition distribution of $q_s(\mathbf{z}_t^s\mid\mathbf{z}_{t-1}^s)$. Hence, the smooth transition distribution $q_s(\mathbf{z}_t^s\mid\mathbf{z}_{t-1}^s)$ can be represented by the

Gaussian distribution with the mean and covariance of each cell state in the backward RNN:

$$q_s\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}^s\right) \sim N\left(\tilde{\mathbf{v}}\left(\tilde{\mathbf{h}}_t\right), \tilde{\boldsymbol{\Phi}}\left(\tilde{\mathbf{h}}_t\right)\right) \tag{17}$$

The posterior latent variable distribution is approximated by the latent variable from the previous smoothed distribution and the previous cell state:

$$p\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}^s, \mathbf{a}_{t:T}\right) \sim N\left(\tilde{\mathbf{v}}\left(\mathbf{z}_{t-1}^s, \tilde{\mathbf{h}}_{t-1}\right), \tilde{\boldsymbol{\Phi}}\left(\mathbf{z}_{t-1}^s, \tilde{\mathbf{h}}_{t-1}\right)\right) \tag{18}$$

After the smooth posterior distribution is constructed, sampling is performed and the sampled latent variable would be reconstructed back as the process data and quality data at the time point *t*. To allow backpropagation of the neural networks, the reparameterization trick is done with $p(\boldsymbol{\varepsilon}) \sim N(0, \mathbf{I})$:

$$\begin{aligned} \mathbf{z}_t &= \mathbf{O}_t\left(\tilde{\mathbf{h}}_t\right)^{-0.5} \boldsymbol{\varepsilon} + \boldsymbol{\chi}_t\left(\tilde{\mathbf{h}}_t\right) \\ \mathbf{a}_t &= \Lambda^{0.5} \boldsymbol{\varepsilon}^{(m)} + \boldsymbol{\mu} \end{aligned} \tag{19}$$

The neural network output of the smoothed distribution consists of mean and diagonal elements of the covariance matrix. The mean output is given by the linear activation function while the covariance matrix is given by the softplus function $\zeta(x) = \ln(1+e^x)$, which keeps the value to be positive. Similarly, these activation functions are kept the same in the filter distribution (forward RNN), the encoder, the decoder for process and quality variables, and the prediction network.

During online learning, as future data cannot be gotten in advance, only the filtered distribution is used and the target latent variable would be updated by the filtered distribution. The smoothed transition distribution is replaced by the filtered transition distribution:

$$q_f\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}^s\right) \sim p\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}^s, \mathbf{a}_{1:t}\right) = p\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}^s, \mathbf{a}_t\right) \tag{20}$$

The sampling would also be drawn from the filtered distribution. Therefore, the reconstruction of each process data point and each quality data point would be used to compute the loss function given in Eq.(14), along with the KL divergence between the transition prior and the smoothed posterior transition (off-line learning) or between the transition pior and the filtered posterior transition (on-line learning).

3. CASE STUDIES

In this section, two cases are presented. Both cases are compared with the supervised-RNN (S-RNN) model and the dynamic KPLS (DKPLS) model to show the merits of the proposed method in predicting the quality data.

3.1. Numerical case

To simulate a case analogous to a dynamic process, assume the dynamic changes occur in the latent variables. The dynamic change is shown by

$$z_1^k = 0.5\sin(z_1^{k-1}) + 0.05\cos(0.02 z_1^{k-1} + \theta) + 0.2 + n_1 \tag{21}$$

where n_t is the noise or uncertainty affecting the dynamic changes, $n_1 \sim N\left(0, 0.01\right)$, and θ represents the noise of the phase changes in the cosine function for the dynamic characteristics, $\theta \sim N(0.4, 0.18)$. The value of the initial latent variable is generated by $z_1^0 = 0.25$. A single process variable and a quality variable are generated as the process and quality data based on the latent variables with the following equations:

$$\begin{aligned} x_1^k &= 0.5\sin(z_1^k)\cos(z_1^k) + 0.5 z_1^k \cosh(z_1^k) + w_1 \\ y_1^k &= \cos^2(z_1^k) + 0.6\log(z_1^k + 50) + \frac{(x_1^k)^3}{(x_1^k + z_1^k + 0.3)} + w_2 \end{aligned} \tag{22}$$

where $w_1 \sim N(0, 0.002)$ and $w_2 \sim N(0, 0.005)$.

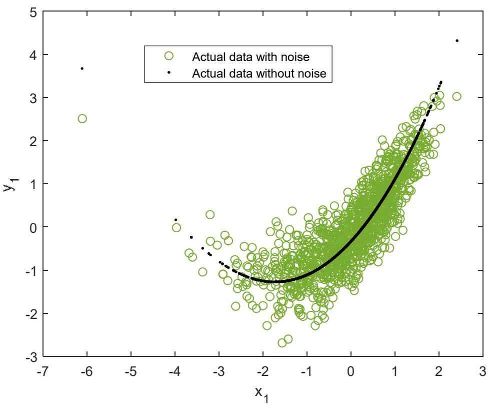


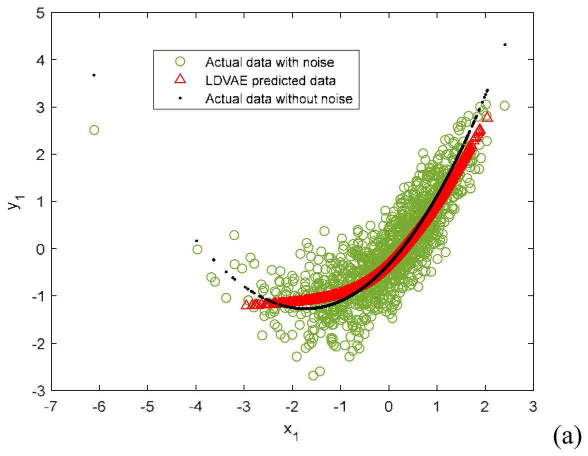
Fig. 4. Process and quality variables in the numerical case.

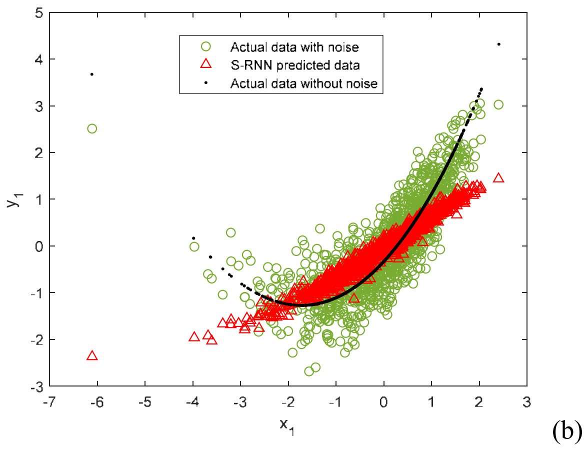
With the process equations, 1,200 data samples are generated. The graphical representation of the process and quality data is given in Fig. 4. The first 1,000 sequential datasets are used to train the model. The remaining 200 sequential data are used to test the model. The length of the moving window in the RNN network τ is set to be 20. The sequential dataset is divided into several batch sets with each batch size equal to τ. In LDVAE, each encoder, transition prior, transition posterior, and decoder consist of 3 hidden layers with 30 neural nodes. Hyperbolic tangent activation functions are used in the nodes on each layer. Both the forward and the backward RNN networks have 20 neural nodes. The number of training iterations is set to be 800. The number of the encoded variable is set to be 1. These model parameters are chosen with the lowest lower bound value. The optimizer selected for training the model is AdamOptimizer with a learning rate of 0.002. The performance of the model prediction is evaluated by the root-mean-squared error (RMSE) value using the test data from noise-contaminated and noise-free datasets. The RMSE formulation is given as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2} \tag{23}$$

where N is the number of data, y_i is noise-free quality data, and $\hat{y}_i$ is the predicted quality of the model. The model with a lower RMSE value is preferred. To show the accuracy of prediction, the proposed LDVAE is tested against the S-RNN. The S-RNN structure is trained through the mean squared error (MSE) between the predicted output and the noise-contaminated quality data. The training iteration and the learning rate are kept the same to ensure a fair comparison. The prediction results of the three models are compared in Fig. 5. The proposed LDVAE significantly outperforms S-RNN and DKPLS as the predicted data of LDVAE are closer to the actual data without noise. Also, the RMSE of LDVAE is

0.2148, of conventional S-RNN is 0.5261 and of DKPLS is 0.5912.





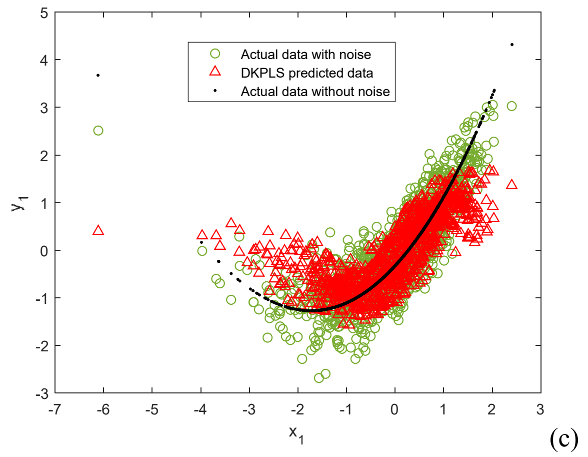


Fig. 5. Predictions of (a) LDVAE, (b) S-RNN and (c) DKPLS compared with noise-contaminated testing data in the numerical case.

3.2 *Industrial Case*

Data from an ammonia synthesis plant are used to study the effectiveness of the proposed method. Ammonia is an essential ingredient that has a lot of uses; for example, it can be used to produce fertilizer. One of the main important processes of ammonia synthesis is pre-decarburization. The carbon dioxide is absorbed for the further production process. The flowchart of this process in Fig. 6 is shown with 4 main units (the feed gas separator, the PG separator, the heat exchanger, and the absorption column). The absorption of CO_2 mainly occurs in the absorption column.

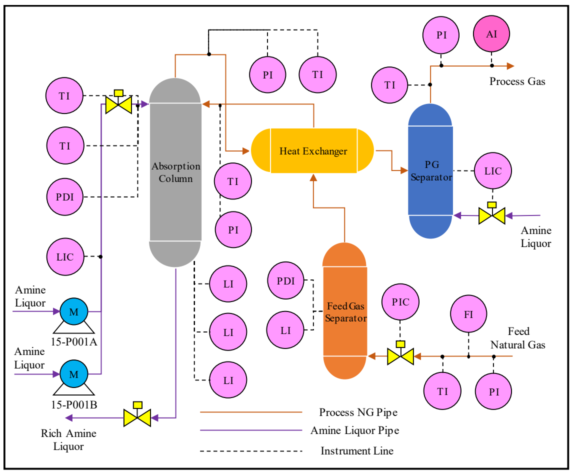


Fig. 6. The decarburization unit of the ammonia synthesis process

To maximize the effectiveness of the ammonia synthesis process, it is important to maximize the amount of absorbed CO_2. The goal is to reduce the amount of CO_2 gas in the process gas outlet, and a total of 19 process variables are selected based on prior knowledge of the variables affecting the CO_2 absorption efficiency. To protect the confidential information about the detail regarding the process, the tag number and its sensor descriptions are excluded from the flowchart. There are 2,000 data samples, with complete information on the quality data. The training data consist of 1,900 data samples, with the remaining 100 samples used as the testing data. The number of the window length τ is set to be 20. Each encoder, transition prior, transition posterior, or decoder consists of 3 hidden layers with 30 neural nodes with the hyperbolic tangent activation function. Both the forward and the backward RNN networks have 20 neural nodes. The number of training iterations is set to be 300. The number of encoded variable dimensions is set to be 4. These model parameters are chosen with the lowest lower bound value. The rest of the supervised method uses all the data for training with the same number of window sizes.

The comparative prediction results of the quality data points are shown in Fig. 7. The maroon star indicates the prediction result of the training data and the red star indicates the prediction result of the testing data. Just like the result of the numerical case, the prediction result of S-RNN is poor because the model overfits the noise in the data. At the same time, the prediction result of DKPLS is so poor although the best kernel function and the number of principal components are selected by trial and error. To measure the performance of the model in predicting the quality data, the RMSE value of each comparative model is calculated. The RMSE of the LDVAE is 0.5119, of S-RNN is 0.7603 and of DKPLS is 0.9120. The RMSE value of the proposed LDVAE model is the lowest among the comparative methods. Due to the limited spaces, the merits of LDVAE in reducing computational load and preventing model over-fitting are omitted here.

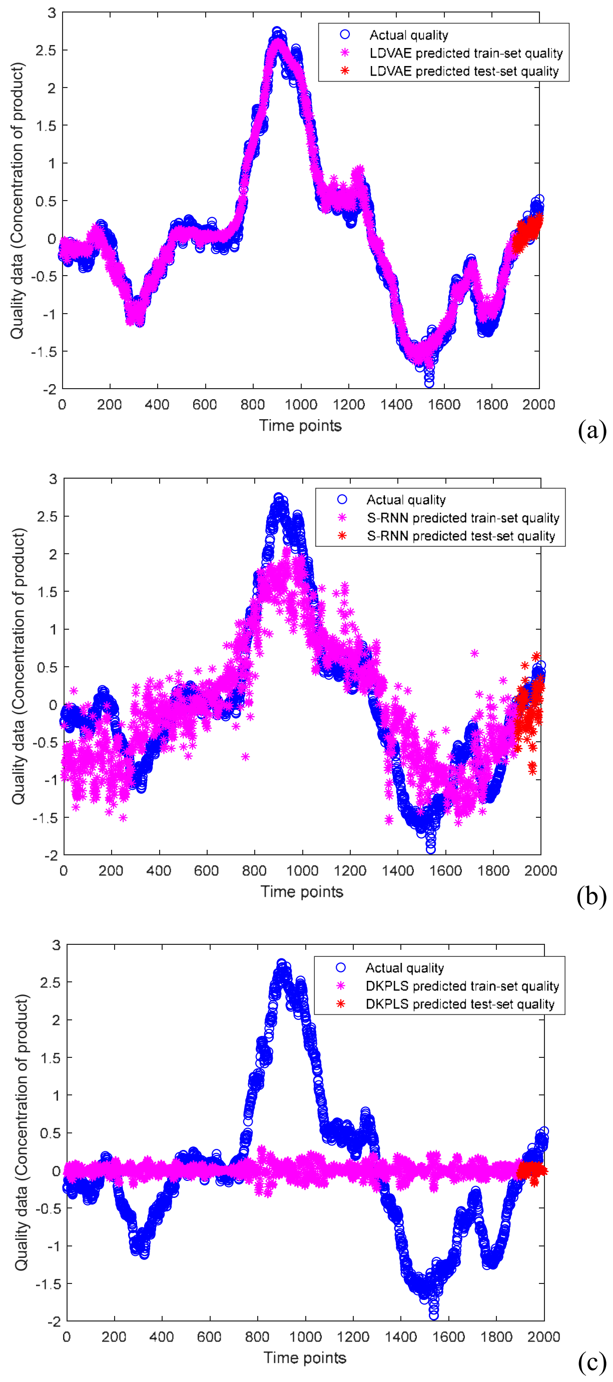


Fig. 7. Comparative prediction results of the quality data at (a) LDVAE (b) S-RNN and (c) DKPLS models

4. CONCLUSIONS

In this research, an innovative soft sensor modeling algorithm called LDVAE is proposed. The prediction of LDVAE is shown to model the actual data distribution despite the effect of disturbances and noises in data. The dynamic modeling is conducted in the latent space, so it can lower computational load and remove the effect of noise in modeling. Lastly, the numerical and industrial ammonia cases show that the proposed LDVAE method can make a more accurate prediction than the conventional supervised RNN network and the DKPLS model. In the future, underfitting indices of the LDVAE-based soft sensor model can be developed to investigate the time point for soft sensor maintainance.

REFERENCES

Agand, P., Shoorehdeli, M. A., &Khaki-Sedigh, A. (2017). Adaptive recurrent neural network with Lyapunov stability learning rules for robot dynamic terms identification. *Engineering Applications of Artificial Intelligence*, *65*(October 2016), 1–11. https://doi.org/10.1016/j.engappai.2017.07.009

Chen, R.-Q., Shi, G.-H., Zhao, W.-L., &Liang, C.-H. (2019). *A Joint Model for Anomaly Detection and Trend Prediction on IT Operation Series*. http://arxiv.org/abs/1910.03818

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A., &Bengio, Y. (2015). A recurrent latent variable model for sequential data. *Advances in Neural Information Processing Systems*, *2015-Janua*, 2980–2988.

Fraccaro, M., Kamronn, S., Paquet, U., &Winther, O. (2017). A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning. *ArXiv*, *section 5*.

Habibie, I., Holden, D., Schwarz, J., Yearsley, J., &Komura, T. (2017). A recurrent variational autoencoder for human motion synthesis. *British Machine Vision Conference 2017, BMVC 2017*. https://doi.org/10.5244/c.31.119

Jia, M., Chu, F., Wang, F., &Wang, W. (2010). On-line batch process monitoring using batch dynamic kernel principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *101*(2), 110–122. https://doi.org/10.1016/j.chemolab.2010.02.004

Jia, Q., &Zhang, Y. (2016). Quality-related fault detection approach based on dynamic kernel partial least squares. *Chemical Engineering Research and Design*, *106*, 242–252. https://doi.org/10.1016/j.cherd.2015.12.015

Kano, M., Miyazaki, K., Hasebe, S., &Hashimoto, I. (1998). Inferential control of distillation composition using partial least squares regression. *Kagaku Kogaku Ronbunshu*, *24*(3), 425–430. https://doi.org/10.1252/kakoronbunshu.24.425

Karl, M., Soelch, M., Bayer, J., &Van DerSmagt, P. (2017). Deep variational Bayes filters: Unsupervised learning of state space models from raw data. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, *ii*, 1–13.

Krishnan, R. G., Shalit, U., &Sontag, D. (2015). *Deep Kalman Filters*. *2000*, 1–17. http://arxiv.org/abs/1511.05121

Ku, W., Storer, R. H., &Georgakis, C. (1995). Disturbance detection and isolation by dynamic principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *30*(1), 179–196. https://doi.org/10.1016/0169-7439(95)00076-3.