2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
Banff Center, Banff, Canada, October 5-8, 2017

Automatic Hyper-parameter Tuning for Soft Sensor Modeling based on Dynamic Deep Neural Network*

Kangcheng Wang, *Student Member, IEEE*, Chao Shang, *Member, IEEE*, Fan Yang, *Member, IEEE*, Yongheng Jiang, and Dexian Huang

***Abstract*— Deep learning has been proposed for soft sensor modeling in process industries. However, conventional deep neural network (DNN) is a static network and thereby can not embrace evident dynamics in processes. Motivated by nonlinear autoregressive with exogenous input (NARX) model and neural nets based dynamic modeling, a dynamic network called NARX-DNN is put forward by further utilizing historical process samples and quality samples in a period of time. A modified hyper-parameter tuning method is proposed to choose optimal hyper-parameters of NARX-DNN with little manual intervention, which automatizes the training procedure and reduces computational cost. The quality prediction error of validation data is interpreted from different aspects, and the most appropriate delay of historical data can be determined automatically. The effectiveness of the proposed method is validated by case studies on a sulfur recovery unit and a debutanizer column. As training, validation and test data sets are selected by the original orders of data samples, the accurate prediction results of NARX-DNN demonstrate its ability in dealing with operation condition changes which are common in real processes.**

I. INTRODUCTION

In order to improve the control efficiency and product quality of process industries, some essential quality variables need to be measured on-line to make real-time monitoring and quality prediction. Soft sensor, a common mathematical modeling method to predict these quality variables, arises from inferential control and has become a hot spot since 1990s [1] [2]. Soft sensors can be classified into model-driven ones and data-driven ones. Detailed mechanisms and explicit prior knowledge of process are needed to build model-driven soft sensor models. By comparison, the complicated mechanical analysis is no longer needed when building data-driven models [1]. Therefore, data-driven modeling techniques have gained popularity since its emergence. Many conventional regression models such as partial least squares (PLS) [3], single hidden layer neural network [4], and support vector machine (SVM) [5] have been successfully used for building soft sensors.

Nevertheless, industrial processes own manifest nonlinearity, while many soft sensors are linear models. Some nonlinear models such as single hidden layer neural network have limited nonlinear representation ability and over-fitting often occurs. Besides, multi-rate sampling is common in process industries, while only quality samples and their corresponding process samples are utilized. Deep learning technique is a new technique to train DNN [6] and has been increasingly popular in many fields such as image classification [7] and speech recognition [8] in recent years. Shang et al. [9] proposed a novel DNN based data-driven soft sensor applied in process industries and better quality prediction precisions are archived. However, compared to traditional models including PLS, SVM and single hidden layer neural network, the procedures of tuning hyper-parameters of DNN are rather complicated and needs solid mathematics foundation as well as rich experiences. An automatic structure and parameters tuning method for DNN based soft sensor has been developed in [10].

Although nonlinearity and multi-rate sampling issue can be successfully resolved by [9] and [10], these two approaches are based on static networks, which do not embrace dynamic information. Industrial processes have evident dynamics, that is, system states at the current snapshot must be closely related to that of previous snapshots [11]. While soft sensors mentioned above assume that processes work in steady states and rich inherent dynamics in process industries remain unused. There are several types of dynamic soft sensor modeling approaches: the first one is historical input augmented dynamic soft sensors, such as dynamic PLS (DPLS) [12], dynamic PCA (DPCA) [13] and impulse response template [14][15]. These approaches focus on the fact that the current values of the variables in a dynamic system depend on the past values. Therefore, the input data for modeling should be formed by current and past data vectors. Although process variables at previous snapshots are utilized, quality historical data remain unused and thus only limited dynamics are considered. The second approach is neural nets based dynamic soft sensor modeling approach, Narendra [16] introduced neural networks based nonlinear dynamical systems identification and control modeling. Bhat and Mcvory [17] proposed an autoregressive-moving average (ARMA) approach back-propagation dynamic modeling. The nonlinear autoregressive with exogenous input (NARX) model is utilized in [18] and a neural network structure determination methodology is presented. In the procedure of DNN based soft sensor modeling, the DBN obtained in pre-training is actually a latent variable model, which will embrace more dynamics when historical data are added.

*This work was supported in part by the National Natural Science Foundation of China (61673236 and 61433001), the National Basic Research Program of China (2012CB720505), the seventh framework programme of the European Union (P7-PEOPLE-2013-IRSES-612230), and Tsinghua University Initiative Scientific Research Program.

K. Wang, C. Shang, F. Yang, Y. Jiang and D. Huang are with Tsinghua National Laboratory for Information Science and Technology (TNList) and Department of Automation, Tsinghua University, Beijing 100084, P.R. China. Corresponding author should be addressed to D. Huang at huangdx@tsinghua.edu.cn

Moreover, the automation level of tuning method presented in [10] still needs to be improved to well fit the DNN containing dynamics.

In this paper, DNN based soft sensor will be extended to its dynamic version. Motivated by NARX model and neural nets based dynamic modeling, a dynamic DNN is built by adding historical process samples and quality samples, and this dynamic DNN will embrace inherent information existed in processes. In addition, hyper-parameter tuning method in [10] is modified. The hidden neuron number of dynamic DNN can be well tuned based on the input dimension, and other hyper-parameters can be adjusted almost automatically.

The rest of this paper proceeds as follows. In Section II, DNN and DNN based soft sensor modeling as well as the hyper-parameter tuning method are briefly revisit. Then dynamic DNN based soft sensor modeling is introduced and the corresponding modified hyper-parameter tuning method is proposed in Section III. Later, case studies on a SRU benchmark and a debutanizer column are employed in Section IV to validate the effectiveness of the proposed method, followed by conclusions in the final section.

II. HYPER-PARAMETER TUNING FOR DNN BASED SOFT SENSOR

A. Brief Revisit of DNN

A deep belief network (DBN) is stacked by several restricted Boltzmann machines (RBMs) and RBMs are formed in a hierarchical structure, as shown in Fig. 1. Each RBM θ_l has two layers: visible layer $\mathbf{v}_l$ and hidden layer $\mathbf{h}_l$, which represent input and latent variables, respectively. $\mathbf{h}_l$ is taken as the input of next RBM θ_{l+1}. Therefore, RBM itself has a structure of latent variable model, which can be described by probability density function $P(\mathbf{v}_l, \mathbf{h}_l)$. The pre-training process of DBN is in a greedy layer-wise manner, as illustrated in Fig. 1. θ_1 is firstly trained, then its hidden layer $\mathbf{h}_1$ is taken as the input of θ_2. After that, θ_2 is trained and $\mathbf{h}_2$ is taken as the input of θ_3. At last, θ_3 is trained in the same way. By means of maximizing probability $P(\mathbf{v}_l)$ of input layers, features are extracted from inputs of RBM. It should be noted that the above training procedure of DBN is unsupervised as no target variables are used. When the pre-training procedure is accomplished, parameters of DBN are used to build a regression model, i.e., a L-layer DNN. Fig. 2 illustrates the supervised training process. The first $(L-1)$ layers of DNN are initialized by weights of the pre-trained DBN, and the weights of the L-th layer are randomly initialized. Lastly, back-propagation algorithm is used to fine-tune the DNN.

B. Revisit of DNN Based Soft Sensor Modeling

A systematic DNN based soft sensor modeling procedure has been developed in [9]. The whole modeling process can be divided into two steps. Firstly, a DBN is pre-trained to extract features in process data, and the weights obtained are taken as the initial value of the regression network in the next step. Then gradient-descent optimization is performed and a regression model is established by target quality samples and their corresponding process samples. These two steps corresponds to unsupervised learning and supervised learning, respectively.

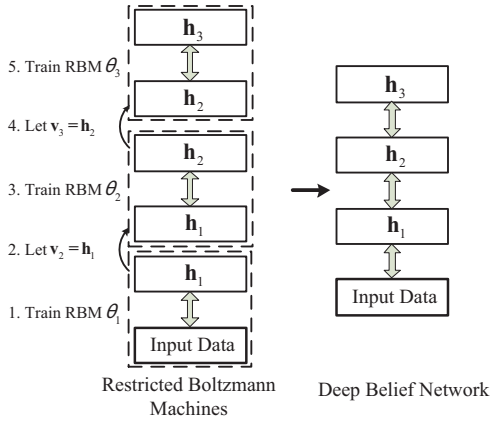

Fig. 1. The hierarchical structure of DBN

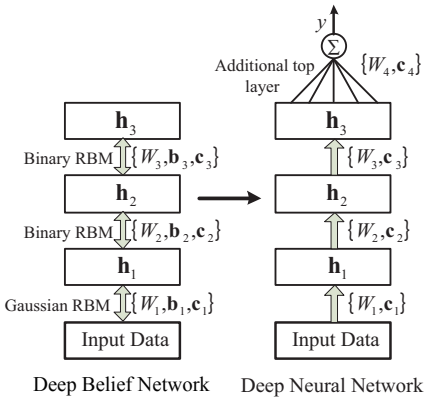

Fig. 2. DNN initialization with a pre-trained DBN

C. Overview of Automatic Hyper-parameter Tuning Method

An automatic hyper-parameter tuning method for DNN based soft sensor in chemical industries is developed in [10]. The whole tuning procedure can be summarized as follows:

- Select the number of RBM n_{RBM} and the type of each RBM T_{RBM}, which determine the structure of DNN. For common chemical processes, two-hidden-layer DBN is enough to obtain satisfying results, that is, $n_{RBM} = 2$. As data are continuous in chemical industries, the bottom RBM is chosen as Gaussian unit, and the other RBM is chosen as binary unit.
- The number of neurons of each hidden layer n_{neu} is selected by grid search. According to experimental analysis as well as the analogy to PCA, the search regions are set as $[m - n_{adj}, m + n_{adj}]$, where m is the number of input variables. $n_{adj} = \lceil m/2 \rceil$ is the adjustment of neuron numbers, where $\lceil \cdot \rceil$ represents the ceiling function.
- Contrastive divergence (CD) algorithm is utilized in the unsupervised learning procedure. The learning rate ε

and the maximum iteration *epoch* of each layer are selected as moderate values by the convergence trend of reconstruction error.

- In the next supervised learning process, back-propagation (BP) algorithm is adopted to fine-tune DNN and random search is used to determine the satisfactory learning rate η. The random search region is manually set as log(0.1) to log(0.001).
- Considering the uncertainty of DNN results caused by random initialization of RBM weights and top layer weights. The whole modeling process is repeated $iterMax = 10$ times and the result with the least validation error is chosen.
- In order to reduce the computational cost, different combinations of hidden neuron number are prepared in advance and parallel toolbox in MATLAB® is used.

III. MODIFIED HYPER-PARAMETER TUNING FOR DYNAMIC DNN BASED SOFT SENSOR MODELING

A. Dynamic DNN Based Soft Sensor Development

The autoregressive with exogenous input (ARX) model is one of the most widely used linear dynamic models because it can match the structure of many processes in real-world [19]. The conventional ARX model can be formed as:

$$A(z^{-1})y(k) = B(z^{-1})x(k) + v(k), \tag{1}$$

where $x(k)$ and $y(k)$ denote the input and output of ARX model, respectively. $v(k)$ is a sequence of independent and identically distributed random variables which has zero mean and variance σ_v^2. z^{-1} is the backward shift operator and polynomials of z^{-1} are noted as $A(z^{-1})$ and $B(z^{-1})$, which can be expressed as:

$$\begin{cases} A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a} \\ B(z^{-1}) = b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b} \end{cases}, \tag{2}$$

where n_a and n_b are model orders and $a_i (i = 1, \cdots, n_a)$ and $b_j (j = 1, \cdots, n_b)$ are polynomial coefficients. The optimal ARX predictor is described by

$$\hat{y}(k|k-1) = B(z^{-1})x(k) + (1 - A(z^{-1}))y(k), \tag{3}$$

which can also be expressed as:

$$\begin{aligned} \hat{y}(k|k-1) = {} & b_1 u(k-1) + \cdots + b_n u(k-n) \\ & - a_1 y(k-1) - \cdots - a_n y(k-n), \end{aligned} \tag{4}$$

assuming $n_a = n_b = n$ [19]. In order to perform nonlinear dynamic system identification, the ARX model can be extended to its nonlinear form, which is called nonlinear ARX (NARX) model and is described by

$$\hat{y}(k|k-1) = f(u(k-1), \cdots, u(k-n), y(k-1), \cdots, y(k-n)). \tag{5}$$

where $f(\cdot)$ denotes an unknown nonlinear function.

In the case when process variables and quality variables have the same sampling rate, historical process data and quality data can be utilized in a difference equation. Motivated by the form of NARX model, together with the successful applications of neural nets based dynamic modeling [17] [18], a dynamic DNN (hereafter called NARX-DNN) is established by taking $f(\cdot)$ in equation (5) as DNN. The

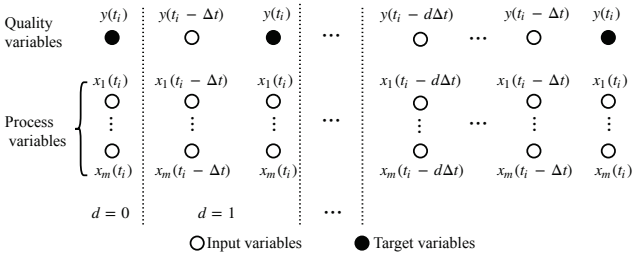

Fig. 3. NARX-DNN Modeling Data Selection

modeling data selection of NARX-DNN is illustrated in Fig. 3. In this work, only one target variable is considered in each soft sensor model. Assuming that there exist N quality variable samples denoted as $\{y(t_1), y(t_2), ..., y(t_N)\}$, whose sampling time is $\{t_1, t_2, \cdots, t_N\}$. The sampling interval of process variables is Δt and m process variables sampled at snapshot t constitutes an m-dimensional vector $[x_1(t), x_2(t), \cdots, x_m(t)]^{\mathrm{T}}$. For conventional static DNN (i.e., delay $d = 0$), only m process variables are utilized as input variables, and only data samples at current snapshot t_i, i.e., $\mathbf{x}(t_i) = [x_1(t_i), x_2(t_i), \cdots, x_m(t_i)]^{\mathrm{T}}$, are used. That is, the input of DNN can be written as follows:

$$\mathbf{X}_{DNN} = \begin{bmatrix} \mathbf{x}(t_i) & \mathbf{x}(t_{i+1}) & \cdots & \mathbf{x}(t_{i+N-1}) \end{bmatrix} \in \mathbb{R}^{m \times N}. \tag{6}$$

However, NARX-DNN with delay $d = 1$ not only utilizes the data of static DNN, but also process data and quality data at previous snapshot $t_i - \Delta t$. The number of input variables is thus augmented to $2m+1$. Similarly, the number of input variables is extended to $3m+2$ when $d = 2$, and so forth. Therefore, the number of input variables is $m(d+1)+d$. More specifically, the input of NARX-DNN can be expressed as:

$$\mathbf{X}_{NARX-DNN} = \begin{bmatrix} \mathbf{X}(d) \\ \mathbf{y}(d) \end{bmatrix} \in \mathbb{R}^{(m(d+1)+d) \times N}, \tag{7}$$

where

$$\mathbf{X}(d) = \begin{bmatrix} \mathbf{x}(t_i) & \mathbf{x}(t_{i+1}) & \cdots & \mathbf{x}(t_{i+N-1}) \\ \mathbf{x}(t_i - \Delta t) & \mathbf{x}(t_{i+1} - \Delta t) & \cdots & \mathbf{x}(t_{i+N-1} - \Delta t) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}(t_i - d\Delta t) & \mathbf{x}(t_{i+1} - d\Delta t) & \cdots & \mathbf{x}(t_{i+N-1} - d\Delta t) \end{bmatrix} \in \mathbb{R}^{m(d+1) \times N}, \tag{8}$$

and

$$\mathbf{y}(d) = \begin{bmatrix} y(t_i - \Delta t) & y(t_{i+1} - \Delta t) & \cdots & y(t_{i+N-1} - \Delta t) \\ y(t_i - 2\Delta t) & y(t_{i+1} - 2\Delta t) & \cdots & y(t_{i+N-1} - 2\Delta t) \\ \vdots & \vdots & \ddots & \vdots \\ y(t_i - d\Delta t) & y(t_{i+1} - d\Delta t) & \cdots & y(t_{i+N-1} - d\Delta t) \end{bmatrix} \in \mathbb{R}^{d \times N}. \tag{9}$$

The target variable in each model above remains the same: quality variable at current snapshot t_i, which can be described by:

$$\mathbf{y} = \begin{bmatrix} y(t_i) & y(t_{i+1}) & \cdots & y(t_{i+N-1}) \end{bmatrix} \in \mathbb{R}^{1\times N}. \quad (10)$$

B. A Tailored Hyper-parameter Tuning Method

According to the above statements, it is evident that the number of input variables will be increased when d increases, which means the hyper-parameter tuning method proposed in [10] should be modified to fit NARX-DNN. Several changes are made and are listed as follows:

- The search region of neuron number of each hidden layer n_{neu} is changed to $[m_d - n_{adj}, m_d + n_{adj}]$, where m_d is the number of input variables, and $m_d = m$ when $d = 0$. Based on massive experiments, $n_{adj} = \lceil m/2 \rceil$ still works on NARX-DNN. In other words, it is still a good choice to get satisfying results while considering computational cost.
- As the scale of process data is much lower than image, and there are only limited process variables closely relevant to the quality variable. As a result, neuron numbers in the search region are close to each other, which leads to similar quality prediction results. Therefore, the results of performing all combinations only once will be close to that of repeating each combination of neuron numbers several times, and the former has a lower computational cost.
- As the supervised learning procedure will not stop until validation error begins to increase, DNN with small learning rate will approximate to the real value of modeling data constantly, which may lead to serious over-fitting as well as a long training time. For unsupervised learning, over-fitting problem will also occur if learning rates are too small. Therefore, learning rates including ε and η are selected by a new approach in this work. They are firstly initialized with a large number such as 1, then they will be reduced by half until no divergence emerges. In this way, large enough learning rates can be obtained almost automatically, which is different from manual tuning in [10].

Based on those three improvements listed above, the automation level of hyper-parameter tuning is improved and the computational complexity can be decreased.

C. Performance Evaluation

The root mean squared error (RMSE) criterion is adopted to evaluate the performance of NARX-DNN with different d. The RMSE of training, validation and test data are defined as $\mathrm{RMSE_{TR}}$, $\mathrm{RMSE_{VA}}$ and $\mathrm{RMSE_{TE}}$, respectively.

In order to determine the optimal hyper-parameters and the appropriate delay d, $\mathrm{RMSE_{VA}}$s are applied from four aspects as follows:

- During the supervised learning procedure, once the $\mathrm{RMSE_{VA}}$ begins to increase, which indicates there occurs an over-fitting, the supervised learning should stop.
- For a certain combination of hidden neuron number n_{neu}, different values of unsupervised learning rate ε and maximum iteration *epoch* are tried and the trained model with minimum $\mathrm{RMSE_{VA}}$ is chosen. The optimal hyper-parameters of each combination of n_{neu} are thus determined.
- For a certain d, trained models with different combinations of n_{neu} can be obtained in the previous aspect, and the one with minimum $\mathrm{RMSE_{VA}}$ is selected. All of the optimal hyper-parameters are thus determined.
- Selected models with different ds are obtained in the previous aspect. When d becomes larger, $\mathrm{RMSE_{TR}}$ will decrease continuously while $\mathrm{RMSE_{VA}}$ will decrease first and then increase because over-fitting will happen when d is too large. Therefore, the one with minimum $\mathrm{RMSE_{VA}}$ is chosen as the best model. The most appropriate d is thus selected.

At last, the model with optimal hyper-parameters and the most appropriate d is used to make on-line quality predictions.

IV. CASE STUDIES

A. SRU Benchmark Process

In this section, a sulfur recovery unit (SRU) [20] desulfurization system is adopted to illustrate the effectiveness of the presented method. Acid gases streams which can cause acid rain are removed by SRU. There are two types of acid gases: the first one is MEA gas, which is rich in H_2S, and the other one is SWS gas, which is rich in ammonia as well as in H_2S. As sulfur dioxide and sulfide are harmful to sensors, which often have to be repaired, it is necessary to build soft sensor models that can make on-line estimations of H_2S and SO_2 concentrations. Thus, there are two quality variables, denoted as y_1 and y_2, to be predicted respectively. The simplified diagram of SRU is shown in Fig. 4 [21]. Five process variables relevant to quality variables are gas flow MEA_GAS, air flow AIR_MEA, secondary air flow, gas flow in SWS zone and air flow in SWS zone.

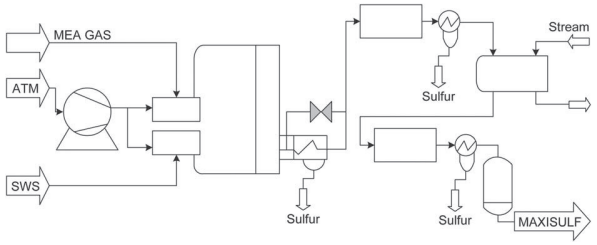

Fig. 4. The simplified diagram of the SRU

There are 10081 data samples collected in total. The training, validation and test data sets are selected by the original orders of data samples for two reasons: the first reason is to simulate the off-line model training and on-line quality prediction procedure in practice, and the other is to consider the operation condition changes which are common in real processes. About first 60% of the data samples are

used for pre-training in unsupervised learning and gradient descent in supervised learning, while 20% samples in the middle are used for validation in order to avoid over-fitting. The last 20% samples are used for testing the accuracy of quality prediction.

For each quality variable, a conventional static DNN and NARX-DNNs with different ds are built. For a certain d, optimal hyper-parameters are determined by the modified tuning method, and the trained model with minimum $\mathrm{RMSE_{VA}}$ is selected. The selected models are used to make on-line quality prediction at last. For each delay d, the time spent on model training and parameter selection is about 4h on a PC with Intel® Xeon® 3.30GHz, 16GB RAM using MATLAB® R2016b. Table I shows the optimal hidden neuron numbers and the corresponding RMSEs of y_1. First, with different d, NARX-DNN has less RMSEs than static DNN. Second, when d increases from 0 to 4, the training error is decreasing continuously. However, the validation error will decrease first and then increase, which means over-fitting occurs. As the minimum $\mathrm{RMSE_{VA}}$ occurs when $d=2$, the corresponding model is finally selected to be used on-line. The real-time quality prediction results on test data of y_1 are shown in Fig. 5, which demonstrates that static DNN fails to make accurate on-line quality predictions. On the contrary, NARX-DNN embraces the inherent dynamic information and higher prediction precisions can be achieved.

TABLE I

QUALITY PREDICTION RESULTS OF y_1 IN SRU PROCESS

d	m_d	n_{neu1}	n_{neu2}	$\mathrm{RMSE_{TR}}$	$\mathrm{RMSE_{VA}}$	$\mathrm{RMSE_{TE}}$
0	5	3	3	0.0518	0.0510	0.0576
1	11	10	11	0.0297	0.0249	0.0295
2	17	20	18	0.0281	0.0242	0.0285
3	23	21	21	0.0254	0.0277	0.0335
4	29	29	28	0.0249	0.0307	0.0340

Similar methods are used in the modeling of y_2, and Table II shows the optimal hyper-parameters and RMSEs. It can be seen that the RMSEs of NARX-DNN with $d=4$ is only about one third of static DNN, and over-fitting also occurs when d is larger than 4. Therefore, the most appropriate on-line quality prediction model of y_2 is NARX-DNN with $d=4$. The quality prediction on test data of y_2 are illustrated in Fig. 6. Owing to the lack of process dynamics, static DNN gives inaccurate predictions, while the true values of y_1 are well tracked by NARX-DNN.

B. Debutanizer Column

A debutanizer column is used to further demonstrate the practicability of the proposed method. The debutanizer column is a desulfuration and naphtha splitting device in refinery industry. The flowchart of this device is illustrated in Fig. 7 [20]. The quality variable is the concentration of butane in the bottom flow of the debutanizer column. Seven process variables relevant to the quality variable are top temperature, top pressure, reflux flow, flow to next process, 6th tray temperature, bottom temperature, and bottom pressure.

TABLE II

QUALITY PREDICTION RESULTS OF y_2 IN SRU PROCESS

d	m_d	n_{neu1}	n_{neu2}	$\mathrm{RMSE_{TR}}$	$\mathrm{RMSE_{VA}}$	$\mathrm{RMSE_{TE}}$
0	5	3	2	0.0581	0.0487	0.0589
1	11	9	14	0.0282	0.0224	0.0320
2	17	20	15	0.0261	0.0208	0.0280
3	23	22	25	0.0184	0.0163	0.0256
4	29	32	30	0.0119	0.0144	0.0181
5	35	33	34	0.0118	0.0167	0.0205
6	41	42	40	0.0116	0.0169	0.0251

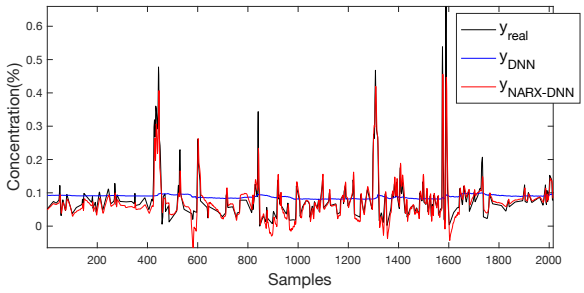

Fig. 5. Quality prediction results of y_1 in SRU process

There are 2394 data samples collected in total. Similarly, each data set is selected by the original orders of data samples, and the percentage of each data set is the same as that of SRU case study. A normal static DNN and NARX-DNNs with different ds are built. Table III shows the optimal hidden neuron numbers and the corresponding RMSEs of quality variable. It could be clearly seen that the RMSEs of NARX-DNN are lower than those of static DNN. Moreover, the training error is decreasing continuously when d increases from 0 to 6, while over-fitting occurs when the validation error comes up from $d=5$. Therefore, the trained model with $d=4$ is chosen as the optimal model to make real-time quality prediction on test data. It can be seen that the $\mathrm{RMSE_{TE}}$ with $d=4$ is only about a quarter of that of $d=0$. The comparisons of quality prediction results are shown in Fig. 8, which illustrates that the optimal model is able to achieve relatively high quality prediction accuracies even when the operation condition changes over time, while static DNN has poorer prediction precisions.

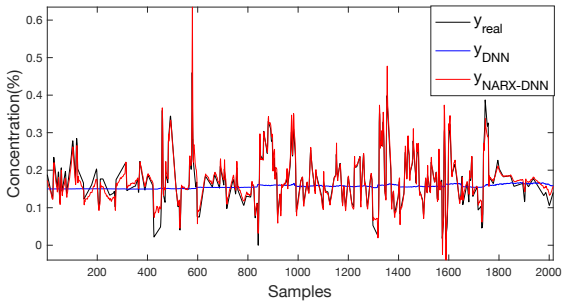

Fig. 6. Quality prediction results of y_2 in SRU process

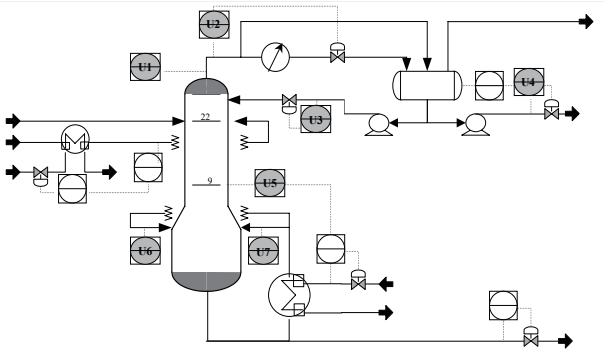

Fig. 7. The flow diagram of the debutanizer column

TABLE III
QUALITY PREDICTION RESULTS IN THE DEBUTANIZER COLUMN

d	m_d	n_{neu1}	n_{neu2}	RMSE_{TR}	RMSE_{VA}	RMSE_{TE}
0	7	8	8	0.1295	0.1524	0.2090
1	15	18	16	0.0453	0.0722	0.1011
2	23	22	22	0.0327	0.0664	0.0775
3	31	27	30	0.0203	0.0525	0.0670
4	39	36	35	0.0141	0.0423	0.0585
5	47	48	47	0.0117	0.0481	0.0864
6	55	52	56	0.0107	0.0625	0.0873

V. CONCLUSIONS

A dynamic DNN named NARX-DNN was introduced in this paper and the optimal hyper-parameters as well as the most appropriate delay of NARX-DNN could be chosen with little manual intervention. Performance evaluation was made and the RMSE of validation data is explained from several different aspects. Case studies showed that static DNN failed to make accurate quality predictions as static network lost dynamic information in processes. However, NARX-DNN embraced process dynamics and is more suitable for soft sensor modeling. Although the model training procedure may take a long time on a PC, the time spent on real-time quality prediction using the trained model is almost the same as single hidden layer neural network. Therefore, we believe that the proposed method will have promising applications in many fields like soft sensor modeling, process monitoring and fault diagnosis which have relatively higher on-line requirements.

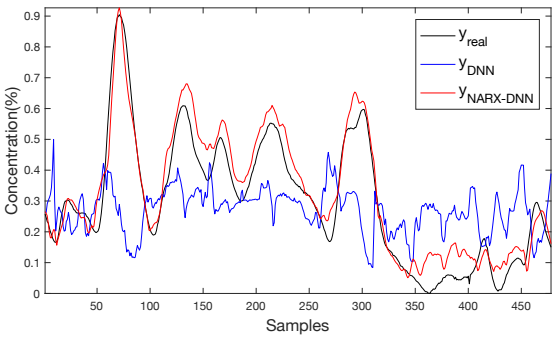

Fig. 8. Quality prediction results of butane concentration in the bottom flow of the debutanizer column

REFERENCES

[1] P. Kadlec, B. Gabrys, and S. Strandt, "Data-driven soft sensors in the process industry," *Computers & Chemical Engineering*, vol. 33, no. 4, pp. 795–814, 2009.
[2] H. J. Galicia, Q. P. He, and J. Wang, "A reduced order soft sensor approach and its application to a continuous digester," *Journal of Process Control*, vol. 21, no. 4, pp. 489–500, 2011.
[3] L. E. Frank and J. H. Friedman, "A statistical view of some chemometrics regression tools," *Technometrics*, vol. 35, no. 2, pp. 109–135, 1993.
[4] S. J. Qin, "Neural networks for intelligent sensors and control—practical issues and some solutions," *Neural Systems for Control*, pp. 213–234, 1997.
[5] G. Q. Zhu, S. R. Liu, and Y. U. Jin shou, "Data modeling based on support vector machine and its application to soft sensing," *Journal of East China University of Science & Technology*, 2002.
[6] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, p. 504, 2006.
[7] Z. Pan, Y. Liu, G. Liu, M. Guo, and Y. Li, *Topic Network: Topic Model with Deep Learning for Image Classification*. Springer International Publishing, 2015.
[8] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8599–8603.
[9] C. Shang, F. Yang, D. Huang, and W. Lyu, "Data-driven soft sensor development based on deep learning technique," *Journal of Process Control*, vol. 24, no. 3, pp. 223–233, 2014.
[10] K. Wang, C. Shang, W. Ke, Y. Jiang, and D. Huang, "Automatic structure and parameters tuning method for deep neural network based soft sensor in chemical industries," in *Submitted to the 28th Chinese Process Control Conference (CPCC), Chongqing*, 2017.
[11] C. Shang, F. Yang, X. Gao, X. Huang, J. A. Suykens, and D. Huang, "Concurrent monitoring of operating condition deviations and process dynamics anomalies with slow feature analysis," *AIChE Journal*, vol. 61, no. 11, pp. 3666–3682, 2015.
[12] M. H. Kaspar and W. H. Ray, "Chemometric methods for process monitoring and high-performance controller design," *AIChE Journal*, vol. 38, no. 10, pp. 1593–1608, 1992.
[13] W. Ku, R. H. Storer, and C. Georgakis, "Disturbance detection and isolation by dynamic principal component analysis," *Chemometrics & Intelligent Laboratory Systems*, vol. 30, no. 1, pp. 179–196, 1995.
[14] Y. Ma, D. Huang, and Y. Jin, "Discussion about dynamic soft sensing modeling," *Journal of Chemical Industry and Engineering*, vol. 56, no. 8, 2005.
[15] W. Lu, Q. Yang, D. Huang, and Y. Jin, "A dynamic soft-sensing method based on impulses response template and parameter estimation with modified de optimization," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 10 983–10 988, 2008.
[16] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.
[17] N. V. Bhat and T. J. Mcavoy, "Use of neural nets for dynamic modeling and control of chemical process system," *Computers & Chemical Engineering*, vol. 14, no. 4, pp. 573–582, 1990.
[18] ——, "Determining model structure for neural models by network stripping," *Computers & Chemical Engineering*, vol. 16, no. 4, pp. 271–281, 1992.
[19] O. Nelles, *Nonlinear System Identification*. Springer Berlin Heidelberg, 2004.
[20] L. Fortuna, S. Graziani, A. Rizzo, and M. G. Xibilia, *Soft sensors for monitoring and control of industrial processes*. Springer Science & Business Media, 2007.
[21] C. Shang, B. Huang, F. Yang, and D. Huang, "Probabilistic slow feature analysis-based representation learning from massive process data for soft sensor modeling," *AIChE Journal*, vol. 61, no. 12, pp. 4126–4139, 2015.