Soft sensor development and applications based on LSTM in deep neural networks

Wensi Ke
Tsinghua National Laboratory for Information Science and Technology (TNList) Department of Automation
Tsinghua University, Beijing, 100084, P.R. China
kws15@mails.tsinghua.edu.cn

Dexian Huang
Tsinghua National Laboratory for Information Science and Technology (TNList) Department of Automation
Tsinghua University, Beijing, 100084, P.R. China
huangdx@tsinghua.edu.cn

Fan yang
Tsinghua National Laboratory for Information Science and Technology (TNList) Department of Automation
Tsinghua University, Beijing, 100084, P.R. China
yangfan@tsinghua.edu.cn

Yongheng Jiang
Tsinghua National Laboratory for Information Science and Technology (TNList) Department of Automation
Tsinghua University, Beijing, 100084, P.R. China
jiangyh@tsinghua.edu.cn

***Abstract*—In chemical industrial processes, some quality variables are difficult to measure, and thus soft sensors have been proposed as an effective solution. Deep learning has been introduced in soft sensors to deal with the complex nonlinearity of the process, yet lacking the ability for dynamics. This paper introduces Long Short-Term Memory (LSTM) and develops a deep neural network structure based on LSTM as a soft sensor method to deal with strong nonlinearity and dynamics of the process. The effectiveness of the improved modeling method is validated by a sulfur recovery unit benchmark. Then it is applied in a real case of coal gasification, which shows that it is especially suitable for dynamic soft sensor modeling.**

Keywords—LSTM, RNN, Deep neural network, Soft sensor, Dynamics

I. Introduction

For the processes in modern chemical industry, in order to control the reaction process effectively, soft sensors have been studied and implemented for years. Some essential quality variables, which are important in chemical reactions, are often difficult to measure frequently due to hardware restriction and cost. Soft sensors can solve this problem by building models between the available information and target variables. Soft sensors can be classified into two classes: the first-principle models and data-driven models. The first-principle models are based on mechanism prior knowledge, which are difficult to build and solve for complicated industrial process. Data-driven models, on the other hand, are built on historical industrial process data, thus acquiring much popularity due to independence on prior knowledge [1].

Many statistical and machine learning techniques have been successfully applied for data-driven soft sensors, such as principal component regression (PCR), partial least squares (PLS) regression, support vector machine (SVM), and artificial neural network (ANN). PCR is a regression method that projects the variables into lower dimensions and find principal components, while cannot reflect the relation between X and Y spaces. On the basis of PCR, PLS maximizes the covariance between X and Y spaces, and is widely applied in soft sensors [2]. However, they both need large amounts of data and are weak in nonlinear processes. SVM uses quadratic programming to obtain boundary support vectors. The model training is simple and will not fall into the local optima. With small data samples, SVM enjoys high efficiency and robustness, and thus has been widely applied [3]. While for large data sets, the computation cost is unaffordable. ANN is a machine learning method widely used in pattern recognition, which is nonlinear structure and can approximate nonlinear continuous function by arbitrary precision, and thus is suitable for nonlinear industrial process [4]. However, traditional gradient descent method is slow in convergence, and the noise in industrial data often cause local optima for ANN.

In recent years, deep learning is becoming more and more popular in many fields, such as image classification [5] and natural language processing (NLP) [6], gaining a great success. Compared to traditional ANN, deep neural network (DNN) has more depth of layers and advanced convergence method, thus has a stronger ability to approximate and converge. For soft sensors in chemical industrial processes, DNN model shows great performance in complex highly-nonlinear process, containing richer information in deep layers of network and large amounts of data [7]. Deep belief network (DBN) is implemented in soft sensors, using both unsupervised training and supervised training for DNN, proving deep learning is especially suitable for soft sensor modeling. Although the complex nonlinearity is solved by DBN, another character of chemical industrial process, dynamics, remains unsolved.

The models mentioned above are all static models, on the hypothesis of steady state and static process. However, real industrial processes are always dynamic, which cannot be ignored. Therefore, dynamic soft sensor methods are developed, such as dynamic PLS [8] and impulse response template (IRT) [9]. These methods combine dynamics with traditional models and have a good effect in industrial processes, but suffer from highly complex modeling and huge amount of computation for large data sets.

This work was supported in part by the National Natural Science Foundation of China (61673236 and 61433001), the seventh framework program of the European Union and Tsinghua University Initiative Scientific Research Program.

Recurrent neural network (RNN) [10], and its improved version, long short-term memory (LSTM) [11], are put forward in machine learning, and have been widely applied in many regions. RNN contains memory and forgetting structures, and can make use of past states and information for the present state. This character makes it suitable for sequences processing, such as natural language processed word by word. For soft sensors, the RNN is well suitable for dynamics in industrial processes. In an industrial process, the current system state and output are closely related to the previous one or more states. The recurrent part of RNN put previous states in memory and participates into the calculation of the current state, keeping the ability of handling strong nonlinearity and massive amount of data. The complex correlations among the chemical process variables will be mined by deep neural network, and the dynamics in real-time processes will be considered and be well used. In this paper, the improved version of RNN, namely, LSTM, will be applied in soft sensors. Compared to traditional RNN, LSTM can solve the problem of gradient vanishing in RNN, which is a severe shortcoming when dealing with long-time model and deep network.

The rest of this article is organized as follows. In Section II, RNN and LSTM model structures are revisited. The deep neural network modeling procedure of LSTM as well as some key tricks are detailed in Section III. Then the effectiveness of this model is validated by a SRU benchmark in Section IV. In Section V, a real case study of coal gasification and discussions are provided. Some conclusions are drawn in the final section.

II. Overview of RNN and LSTM Structure

A. A brief view of RNN

A recurrent neural network is composed of an input layer, an output layer and hidden layers, as shown in Fig. 1. Unlike common neural networks, the arrow pointing back to *s* indicates that the output is determined by input X and the former state *s*. Unfolding the structure of RNN by time step makes it clear. The former state s_{t-1} is transmitted to the present state by weight W, thus carrying the information from the previous state. The present state takes present input s_t through weight W_i and the former state, then calculates the output o_t through weight W_o. Further derivation and back propagation through time (BPTT) can be referred to [10].

B. Revisit of LSTM

Long short-term memory is a RNN architecture proposed in [11]. Many following studies have been made to improve the network and function, e.g., [12] proposed Peephole LSTM with forget gates. Unlike traditional neural units in RNN, LSTM has its own unique LSTM units, which excels at remembering information for long or short durations of time. The stored past value and gradient will not vanish over iterations of BPTT, which is the defect of traditional RNN.

Fig. 2 shows the structure of a unit of Peephole LSTM [10], which is applied in this paper. The central part Cell controls the memory of the unit. Input Gate and Output Gate controls the flow of information in and out, combining with previous states

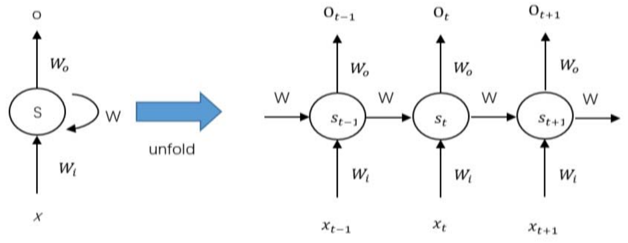

Fig. 1. The simplified structure of a recurrent neural network

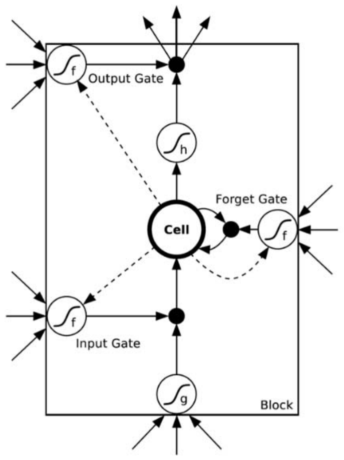

Fig. 2. The structure of LSTM unit

with weights. Forget Gate determines how the Cell will keep the previous states. The following formulas from [10] explain the units.

Input gates:

$$a_l^t = \sum_{i=1}^{I} w_{il} x_i^t + \sum_{h=1}^{H} w_{hl} b_h^{t-1} + \sum_{c=1}^{C} w_{cl} s_c^{t-1} \tag{1}$$

$$b_l^t = f(a_l^t) \tag{2}$$

Here subscript *h* stands for hidden units. *L* stands for Input Gate, and *c* stands for cell. Clearly, inputs from X, information from other hidden units and past state of cell compose the input to the Input Gate. *F* is an activation function, usually tanh or Rectified Linear Units (ReLU).

Forget Gates:

$$a_\emptyset^t = \sum_{i=1}^{I} w_{i\emptyset} x_i^t + \sum_{h=1}^{H} w_{h\emptyset} b_h^{t-1} + \sum_{c=1}^{C} w_{c\emptyset} s_c^{t-1} \tag{3}$$

$$b_\emptyset^t = f(a_\emptyset^t) \tag{4}$$

These formulas are similar to Input Gates, while $\emptyset$ stands for Forget Gates. These information helps the Forget Gate to decide whether to keep the state of Cell or not.

Cells:

$$a_c^t = \sum_{i=1}^{I} w_{ic} x_i^t + \sum_{h=1}^{H} w_{hc} b_h^{t-1} \tag{5}$$

$$s_c^t = b_\emptyset^t s_c^{t-1} + b_l^t g(a_c^t) \tag{6}$$

The input of Cell is composed of input *X* and information from other hidden units, while the value of cell is determined by Forget Gate multiplying the previous state of cell, and Input gate combining the value of cell.

Output Gates:

$$a_w^t = \sum_{i=1}^{I} w_{iw} x_i^t + \sum_{h=1}^{H} w_{hw} b_h^{t-1} + \sum_{c=1}^{C} w_{cw} s_c^t \tag{7}$$

$$b_w^t = f(a_w^t) \tag{8}$$

Here, *w* stands for Output Gates. Input *X*, information from other hidden units and states from cell (till now) multiplying the corresponding weights compose the input to the Output Gate.

Cell Outputs:

$$b_c^t = b_w^t h(s_c^t) \tag{9}$$

The final output is determined by the values of Output Gate and Cell, representing the effect of Output Gate. Forward propagation formulas above are listed to describe the structure of LSTM units. Further explanations and back propagation derivation can be referred to [10].

III. Improved Deep LSTM Network Structure and Training Methods

The network is arranged in a sequential method. The first layer is the input layer, indicating the dimension of input. The following are the recurrent layers. The last layer is the fully-connected layer, with the output of one dimension and linear activation. In this paper, LSTM is applied in thebe recurrent layer as an improved version of RNN. Parameters are set in the layer, including the activation functions and bias etc. The number of neurons of each layer is set according to the dimension of input data, usually by experience and experiments.

Instead of one layer of recurrent layer, deep neural network structure is applied. For complex chemical industrial processes, shallow neural network is insufficient in approximating highly varying functions, lacking a powerful representation efficacy. To solve this drawback, more neurons and more layers are introduced. Thus, the deep neural network structure is needed.

However, deep neural network with more than two hidden layers is usually hard to learn, and has different degrees of over-fitting. In this paper, the neural network is designed to have two recurrent layers. In this structure, the forward propagation and back propagation are clear and reliable for LSTM units in deep neural network. The sequential method also makes the complicated deep network clear and easy to build. With the recurrent unit of LSTM as hidden layers of deep network, the method is able to solve complex processes with strong dynamics.

For training a neural network, the traditional method is stochastic gradient decent (SGD). But for a deep recurrent neural network, SGD converges slowly and easily reaches local optima. Therefore, an optimizer named Adam [13] has been introduced. Compared to SGD, Adam is much more computationally efficient, converges better within the same training epochs. Also Adam has less memory requirements and is well suitable for large industrial data sets

The loss function used in training models is mean squared error (MSE), defined as

$$\mathrm{MSE} = \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}, \tag{10}$$

where *N* is the number of data samples, y_i and $\hat{y}_i$ are real value and predicted values, respectively. For training data sets, MSE shows the training effect and accuracy. For validating data, the MSE reflects if the neural network is over-fitting. During the procedure of training, if MSE of training data keeps descending while MSE of validating data increases apparently, over-fitting occurs and the training should stop. Finally, MSE of testing data are used to evaluate the performance and generalization ability of the neural network.

Even with advanced optimizer, over-fitting still exists in deep neural network. Another technique named Dropout [14] was introduced when over-fitting is severe. Dropout is to disconnect some neurons by a certain percentage during the training, and for every epoch of training, each neurons has a probability to be dropout and miss the training. This training trick performs well in large deep neural networks to alleviate over-fitting.

IV. Case Study on the SRU Benchmark

We tested the performance of RNN and LSTM on a sulfur recovery unit (SRU) desulfurization system as a benchmark, detailed in [15]. The SRU takes in two kinds of acid gases as input: MEA gas rich in H_2S, and SWS gas rich in H_2S and NH_3. Burnt in reactors, the gas is transformed into pure sulfur, and the final gas stream contains residual H_2S and SO_2. As the sensors are easily eroded in acid gas, soft sensors can be used to estimate the concentration of H_2S and SO_2 in the output gas stream. Fig. 3 shows a simplified scheme for the SRU, and Table 1 lists the quality variables used in the SRU soft sensing models.

The data set contains 10081 samples, 5 process variables and 2 target variables. According to [15], the data was collected with a sampling period of 1 minute continuously, considering the dynamic operating conditions. The data cleaning is not necessary, only normalization is applied for the data preprocessing. In this paper, the first 60% of the data are used for training the network, and the following 20% of the data are used for validation, preventing the network from over-fitting. The last 20% samples are used for testing the performance of the network.

PLS was used for comparison as a traditional static method. Some dynamic methods are also applied for comparison. Dynamic PLS and dynamic ANN are listed as traditional dynamic methods, which have been used in soft sensors and show good performance, as shown in [16] and [17]. These methods are popular and reliable dynamic soft sensor methods. In the dynamic part of these methods, the time step window for input is set as 5. The ANN has one hidden layer of full-connection containing 50 neurons, referring to the dimension of input. Simple RNN with single layer is also used as comparison with the improved version LSTM in deep neural network. Simple RNN and deep LSTM also have 50 neurons in the hidden layers, same as dynamic ANN. Adam is chosen as the optimizer, and regularization is applied. The settings and structure of deep LSTM can be referred to Sec. 3.

The experiment was conducted on a laptop, with Intel i5 @2.3GHz, 8G RAM and Nvdia GeForce GTX 1060. The software environment is Windows 10, Python 3.5, CUDA 8.0, Tensorflow 1.0.1 and Keras 2.0.4. The training time is within the range of 10s to 1min, taking advantage of GPU computing. LSTM takes more computation time than simple RNN, and more layers take more time. Time cost for prediction can be ignored, which is within 1s. The networks for comparison are trained the same way as the deep LSTM. The final architecture of our method has two hidden layers, each contains 50 neurons. The training epochs are set at 20, because more epochs will cause obvious rising of MSE. Fig. 4 shows the loss curve of training procedure of RNN and deep LSTM on SO_2 data, and on H_2S data the curve is similar. The results are listed in Tables II and III, relating Fig. 5 and Fig. 6, which show the comparison between the original test data and predictions.

During the learning procedure, the training loss of our method descends faster than Simple RNN. The final performance is also better, which shows it has a better ability of figuring nonlinearity and dynamics. The validating loss is not stable, but the overall trend of the curve is first descending and then rising. The spot of the best validating performance gets at epoch 8 or 9, and deep LSTM still performs better.

As for both SO_2 and H_2S experiment results, our method performs better than the traditional static and dynamic methods with a greater advantage. PLS performs worst, while dynamic PLS and other methods is a lot better. This shows the advantage and necessity of dynamic soft sensors for the process. In dynamic methods, Dynamic PLS performs worst because its structure is too simple compared to neural networks. Simple RNN performs better than dynamic ANN, because its recurrent structure is better than dynamic input with ANN, showing the effectiveness of recurrent network. Deep LSTM performs best, because its deep network structure is able to handle complex situations. Also, the LSTM unit is better than RNN with better memory and control of the past information. Figures also indicate that predictions proposed by our method follow the trend of the original data well, and extreme values and exceptional operating states can be handled well, while in some cases predictions show small amount of deviation from the original data.

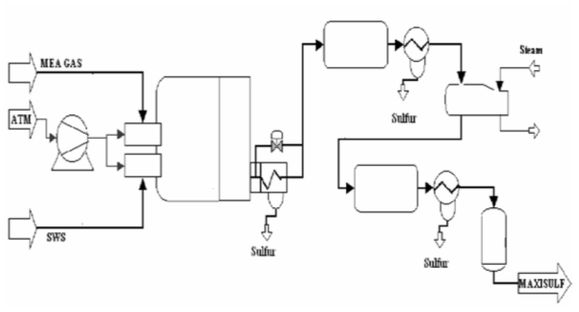

Fig. 3. A simplified scheme for the SRU

TABLE I. QUALITY VARIABLES USED IN THE SRU SOFT SENSING MODELS

Variable	Description
u_1	gas flow
u_2	air flow
u_3	secondary air flow
u_4	gas flow in SWS zone
u_5	air flow in SWS zone

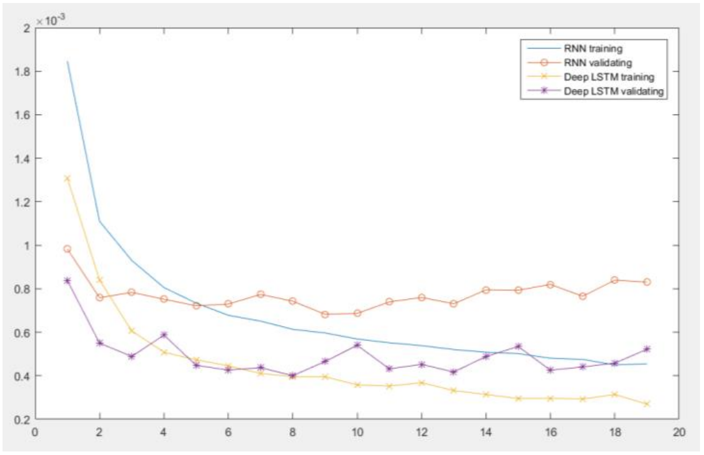

Fig. 4. Training procedure of RNN and deep LSTM on SO_2 data

TABLE II. QUALITY PREDICTION RESULTS OF SO_2 IN SRU

MSE	PLS	Dynamic PLS	Dynamic ANN	Simple RNN	Deep LSTM
SO_2 training	0.0033	0.00161	0.00073	0.00061	0.00043
SO_2 validating			0.00105	0.00068	0.00042
SO_2 testing	0.0034	0.00190	0.00116	0.00078	0.00042

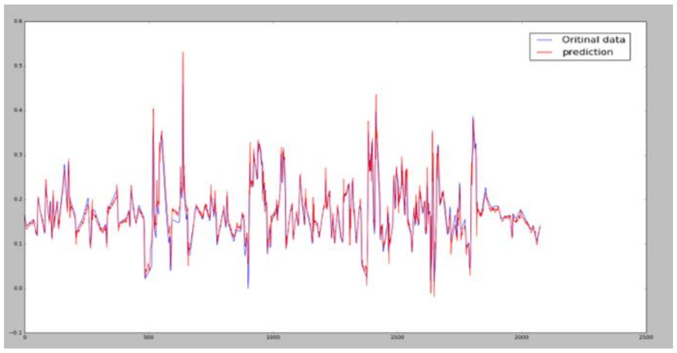

Fig. 5. Deep LSTM quality prediction results of SO_2 in SRU benchmark

TABLE III. QUALITY PREDICTION RESULTS OF H_2S IN SRU

MSE	PLS	Dynamic PLS	Dynamic ANN	Simple RNN	Deep LSTM
H_2S training	0.0025	0.00180	0.00112	0.00038	0.00029
H_2S validating			0.00119	0.00074	0.00065
H_2S testing	0.0122	0.00748	0.00126	0.00098	0.00068

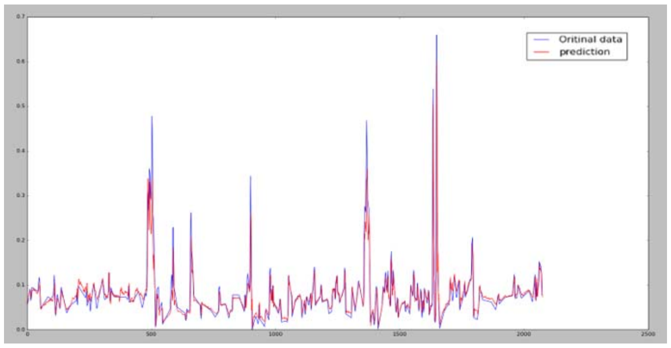

Fig. 6. Deep LSTM quality prediction results of H_2S in SRU benchmark

V. CASE STUDIES ON THE REAL COAL GASIFICATION

This section focuses on a real case study of soft sensor modeling of RNN and LSTM based on a chemical process of coal gasification, which is an important chemical industry in China. For lacking of petroleum, coal is a major source of energy and industrial raw materials, and coal gasification is a widely applied industrial technology in coal clean utilization. Take SHELL gasification furnace for instance. It is of a high daily processing capability, high efficiency and wide range of coal. However, owing to complicated chemical production environment, key process variables like temperature and gas composition are hard to measure online, so the chemical process in the reaction furnace cannot be judged and predicted in time, which cause low efficiency and malfunctions.

The traditional way is offline laboratory analysis, which has a large time delay and cannot reflect the state of process in time. Another approach is online analyzers and soft sensors, which are more and more widely used. Because the reaction process is very complex with strong dynamics, traditional methods suffer from high malfunction and difficulty in modeling. An advanced soft sensor with the ability to handle complex nonlinearity and strong dynamics is needed. Therefore, RNN and LSTM soft sensors are greatly suitable in this situation.

The Data in this experiment is taken from the SHELL gasification furnace in a chemical plant in Yunnan, China. Arranged in SQL, hundreds of process variables and massive amount of original industrial data are provided. The target variable is CO_2 content in the generated gas, indicating the states of chemical reaction process in the furnace, which are hard to observe and measure. From hundreds of process variables, 17 variables are selected for modeling according to a physical background, shown in Table IV. Some of the variables, such as pulverized coal temperature, are merged in average by several sensors.

The Data are selected from a period of continuous production in coal gasification. There are 40000 samples in total, divided into three parts in chronological order. The first 75% data, that is 30000 samples, are used for training the model. The following 5000 samples are used for validation, and the last 5000 samples are used as test data. In the same environment mentioned in section 3, the training time is around 40-200s, while the prediction time is within 1s.

In this section, the deep LSTM and other methods for comparison use the same parameters and architectures as sec. VI. Moreover, in this real case, networks suffer from various degrees of over-fitting, and dropout is introduced to solve the problem. Through this trick, the training accuracy descends and testing accuracy increases, meaning over-fitting has been greatly alleviated.

The results are listed in Table V. PLS performs worst in the real dynamic case, while dynamic PLS is a lot better. In the real case the dynamic methods have greater advantages. As the process is very complex, deep neural networks with LSTM performs much better than shallow RNN, having better modeling and generalization ability. Dynamic PLS shares the same over-fitting problem with Simple RNN. The training MSE is close for three methods, but the testing MSE is too large for Dynamic PLS and Simple RNN. Dynamic ANN shows a better performance, but still not satisfactory. The accuracy is still to be improved. LSTM in deep neural network shows the best performance, for its deep structure and advanced LSTM units, also with the helping of the dropout method to prevent over-fitting. Fig. 7 shows the comparison of LSTM and real test data, indicating prediction by this method well tracks the original test data.

TABLE IV. PROCESS VARIABLES SELECTED IN COAL GASIFICATION

Variable	Description
x_1	Pulverized coal flow rate
x_2	Pulverized coal temperature
x_3	Pulverized coal pressure
x_4	Oxygen flow rate
x_5	Oxygen temperature
x_6	Oxygen pressure
x_7	Steam flow rate
x_8	Steam temperature
x_9	Steam pressure

x_{10}	Nitrogen flow rate
x_{11}	Nitrogen temperature
x_{12}	Nitrogen pressure
x_{13}	Slag volume rate
x_{14}	Furnace surface temperature
x_{15}	Furnace pressure
x_{16}	Oxygen-coal ratio
x_{17}	Middle pressure

TABLE V. QUALITY PREDICTION RESULTS OF COAL GASIFICATION

MSE	**PLS**	**Dynamic PLS**	**Dynamic ANN**	**Simple RNN**	**Deep LSTM**
Training	0.05342	0.00058	0.00071	0.00067	0.00025
Validating			0.00072	0.00061	0.00041
Testing	0.08812	0.00261	0.00097	0.00223	0.00036

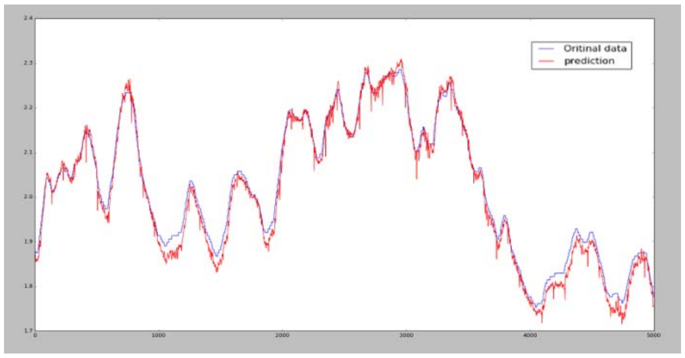

Fig. 7. Quality prediction results by deep LSTM with dropout in Coal Gasification

VI. CONCLUSIONS

This paper develops a deep neural network based on LSTM as a dynamic soft sensor modeling method. Advantages of this method is analyzed compared to traditional dynamic soft sensor methods. With a proper network structure and optimization techniques, the proposed method is applied on the SRU benchmark, demonstrating the advantages and effectiveness. Then it is applied on a real case study of Coal Gasification and shows satisfactory performance and speed. Thus, deep LSTM is well suitable for soft sensors in online variables predictions for processes with severe nonlinearity and dynamics.

REFERENCES

[1] Petr Kadlec, Bogdan Gabrys, and Sibylle Strandt. Data-driven soft sensors in the process industry. Computers & Chemical Engineering, 33(4):795–814, 2009.

[2] Kresta, J. V., Marlin, T. E., & MacGregor, J. F. Development of inferential process models using PLS[J]. Computers and Chemical Engineering, 1994, 18(7)

[3] Yan, W.W., H.H. Shao, and X.F. Wang, Soft sensing modeling based on support vector machine and Bayesian model selection. Computers & Chemical Engineering, 2004. 28(8): p. 1489-1498.

[4] Assis A J De, Filho R M. Soft sensors development for on-line bioreactor state estimation. Computers and Chemical Engineering, 2000, 24(2-7).

[5] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]// International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012:1097-1105.

[6] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space[J]. Computer Science, 2013.

[7] Shang C, Yang F, Huang D, et al. Data-driven soft sensor development based on deep learning technique[J]. Journal of Process Control, 2014, 24(3):223-233.

[8] Michael H. Kaspar and W. Harmon Ray. Chemometric methods for process monitoring and high-performance controller design. AIChE Journal, 38(10):1593–1608, 1992.

[9] Wenxiang Lu, Qing Yang, Dexian Huang, and Yihui Jin. A dynamic soft-sensing method based on impulses response template and parameter estimation with modified de optimization. IFAC Proceedings Volumes, 41(2):10983–10988, 2008.

[10] Graves A. Supervised Sequence Labelling with Recurrent Neural Networks[M]. Springer Berlin Heidelberg, 2012.

[11] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural Computation, 1997, 9(8):1735.

[12] Gers F A, Schmidhuber E. LSTM recurrent networks learn simple context-free and context-sensitive languages[J]. IEEE Transactions on Neural Networks, 2001, 12(6):1333-40.

[13] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. Computer Science, 2014.

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15, 1929-1958, 2014.

[15] Luigi Fortuna, Salvatore Graziani, Alessandro Rizzo, and Maria Gabriella Xibilia. Soft sensors for monitoring and control of industrial processes. Springer Science & Business Media, 2007.

[16] Galicia H J, He Q P, Wang J. Comparison of the performance of a reduced-order dynamic PLS soft sensor with different updating schemes for digester control[J]. Control Engineering Practice, 2012, 20(8):747-760.

[17] Dai X, Wang W, Ding Y, et al. “Assumed inherent sensor” inversion based ANN dynamic soft-sensing method and its application in erythromycin fermentation process[J]. Computers & Chemical Engineering, 2006, 30(8):1203-1225.