LSTM and Statistical Learning for Dynamic Inferential Modeling with Applications to a 660MW Boiler

Jicheng Li [*], **Peng Tan** [**], and **S. Joe Qin** [***]

[*] *School of Data Science, City University of Hong Kong, Kowloon, Hong Kong. e-mail: jichengli3-c@my.cityu.edu.hk.*
[**] *State Key Laboratory of Coal Combustion, School of Energy and Power Engineering, Huazhong University of Science and Technology, Wuhan, China. e-mail: tanpeng@hust.edu.cn*
[***] *School of Data Science and Hong Kong Institute for Data Science, City University of Hong Kong, Kowloon, Hong Kong. e-mail: joe.qin@cityu.edu.hk. Corresponding author.*

Abstract: Statistical learning methods have been widely studied and practiced in the past for inferential modeling. In recent years, deep learning methods have been implemented for inferential sensor modeling. As a popular deep learning model, the long short-term memory (LSTM) network is capable of handling data nonlinearity and dynamics and is therefore applied for dynamic inferential modeling. In this paper, we analyze and compare LSTM with other statistical learning methods for the dynamic NOx emission prediction of a 660 MW industrial boiler. Support vector regression (SVR), partial least squares (PLS), and Least absolute shrinkage and selection operator (Lasso) with embedded dynamics are compared with LSTM for dynamic inferential modeling. The experimental results indicate that SVR, PLS, and Lasso outperform LSTM. By disabling the LSTM gates to realize a simple memory structure, the LSTM performance is significantly improved. The main goal of the paper is to demonstrate that a deep neural network that is effective in other domains requires close scrutiny and detailed study to show its superiority in process applications.

Keywords: Dynamic inferential modeling; LSTM; Statistical learning; Dynamic modeling

1. INTRODUCTION

In modern process industries, the measurement of crucial quality variables has been critical for process monitoring and control (Kresta et al. (1994); Qin and McAvoy (1996); Kano et al. (1998); Sun and Ge (2021)). Although hardware sensors and laboratory tests are options to measure the quality variables, they are often subject to technical difficulties such as harsh environment, high maintenance, high cost, and long time delays in the measurements, hindering real-time requirements for process monitoring and control. Therefore, inferential sensors have been popularly used in the industry to map easy-to-measure online process measurements to hard-to-measure quality variables. Aspen Technology alone reported over 16,000 installed inferential sensors in conjunction with the model predictive controllers (Zhao (2021)). In addition, sensor validation functions can be incorporated to validate and replace faulty sensors (Qin et al. (1997); Pranatyasto and Qin (2001)). As inferential sensors significantly improve the control of product quality, they have been successfully applied in various process industries (Ghosh et al. (2020)).

In the past three decades, many inferential sensor models have been studied by researchers since process data are more and more available. Some of the classical inferential sensor modeling methods include PLS, canonical correlation analysis, and artificial neural network (Sun and Ge (2021)). In addition, sparse statistical learning methods such as the Lasso effectively select relevant variables for optimal prediction (Hastie et al. (2015)). To build dynamic inferential sensors, PLS with lagged input and output variables (Kano et al. (1998)), dynamic versions of PLS (Kaspar and Ray (1993); Dong and Qin (2018)), and recurrent neural networks (RNN) (Tan et al. (2019)) have been developed.

Recurrent neural networks (RNNs) are neural learning methods for dynamic inferential modeling. However, RNNs suffer from the problem of gradient vanishing or gradient explosion, which limits them to short sequence modeling. Thus, the LSTM model was developed by Hochreiter and Schmidhuber (1997) with successful applications in speech recognition (Graves et al. (2013)) and handwriting recognition (Carbune et al. (2020)). In industrial processes, there has been a proliferation of LSTM-based models that have been employed in dynamic inferential sensor modeling. For example, Yuan et al. (2019) proposed a supervised LSTM network that the quality and input variables are used to learn the hidden dynamics for inferential sensor modeling. Tan et al. (2019) employed LSTM method to predict the NOx emission of a 660MW pulverized coal-fired power boiler. Alternative methods such as support vector regression (SVR) were compared. However, dynamics was

left out in the SVR model, while LSTM is an inherently dynamic model with many memory cells and recurrent hidden nodes.

In this paper, statistical methods including SVR, PLS, and Lasso models with embedded dynamics are applied to the 660MW boiler data, and the results are compared with LSTM for dynamic inferential modeling. Since LSTM has complex memory and gate structures, as introduced in the next section, they are prone to multiple local minima in the gradient-based training. With extensive searches of the optimal hyperparameters with various initialization of the weights, our results show that the LSTM with the optimal outcome is inferior to the competing statistical methods with built-in dynamic relations. However, by disabling some of the gates in LSTM, comparable results are achieved with very simple LSTM models.

The rest of the paper is structured as follows. In Section 2, the LSTM, SVR, PLS, and Lasso are introduced for dynamic inferential modeling. Then, the effectiveness and performance of the introduced methods are evaluated and compared through an industrial 660MW boiler case study in Section 3. Section 4 analyzes the lack of performance of LSTM in this application. Finally, conclusions are given in Section 5.

2. METHODOLOGY

2.1 Long short-term memory

The LSTM network is a variant of RNN, which incorporates gate units into the state dynamics to handle the gradient vanishing problem in RNN. Fig.1 shows the detailed structure of the LSTM network. It comprises memory cells, forget gates, input gates, and output gates. The cells can store information over time, and the gates regulate the flow of information into and out of the cells. From

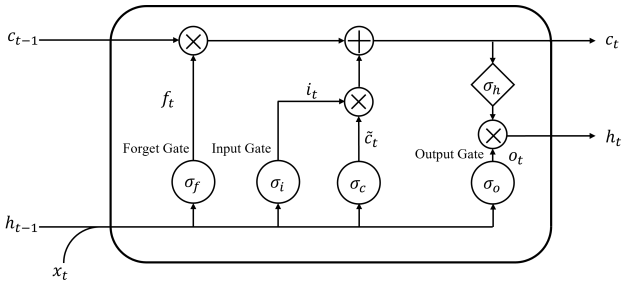

Fig. 1. Structure of the long short-term memory network

the Fig. 1, at each point in time t, the inputs of the LSTM network are the current input vector $x_t \in \Re^m$, the previous hidden state $h_{t-1} \in \Re^s$, and the previous cell state c_{t-1}. The cell state c_{t-1} is affected by the gates. The forget gates regulate how much of the previous cell state c_{t-1} should be forgotten. The input gates determine the updated information of the cell state. The output gates control the amount of information to flow from the current cell state c_t. The equations for the gates and the updates of the cell state and hidden state are shown as follows:

$$f_t = \sigma_f(W_f x_t + U_f h_{t-1} + b_f) \quad (1)$$
$$i_t = \sigma_i(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$
$$o_t = \sigma_o(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$
$$\widetilde{c}_t = \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \quad (4)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \widetilde{c}_t \quad (5)$$
$$h_t = o_t \circ \sigma_h(c_t) \quad (6)$$

where $\circ$ represents element-wise multiplication, σ_f, σ_i, σ_o, σ_c, σ_h are activation functions for the forget gate, input gate, output gate, cell state and hidden state, respectively. Generally, the sigmoid function is used as the gates activation function, whereas σ_c and σ_h are the hyperbolic tangent function. W_f, W_i, W_o and W_c represent the input weights, U_f, U_i, U_o and U_c represent the recurrent weights, and b_f, b_i, b_o and b_c denote the biases.

In this study, the LSTM network for prediction is illustrated in Fig. 2. X_t denotes the model input sequence at each time point t, consisting of the current and previous operational variables. N is the number of time steps or the size of the *look-back window*. In this study, the LSTM layer is followed by a fully connected layer with linear activation functions to produce the output prediction $\hat{y}_t$. The LSTM model is implemented on Keras with a TensorFlow backend.

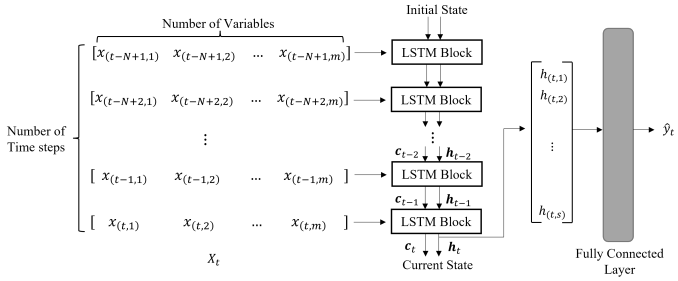

Fig. 2. LSTM network for NOx emission prediction

2.2 SVR, PLS and Lasso for dynamic inferential modeling

LSTM has internal long and short-term dynamic memory, which makes it suitable for dynamic inferential modeling. For a fair comparison, the simple SVR, PLS, and Lasso models need to have built-in dynamics. Typically lagged input and output variables should be used as augmented model inputs for dynamic modeling. The dynamic inferential model can be expressed as:

$$\hat{y}_t = \hat{f}(y_{t-1}, y_{t-2}, ...y_{t-n}, x_t, x_{t-1}, ...x_{t-n}) \quad (7)$$

where n represents the number of lags, and $\hat{f}$ is the related inferential sensor model for SVR, PLS, and Lasso. Although the Lasso is a basic algorithm from a family of sparse learning methods (Hastie et al. (2015)), it is used as a baseline sparse approach in this paper to compare with LSTM.

SVR Support Vector Regression (SVR) (Drucker et al. (1997)) is an application of SVM to regression problems. The basic idea of SVR is to find a line or a hyperplane in a high-dimensional feature space that minimizes the distance to data points. The detailed procedures are as follows.

Consider a set of training data $(x_1, y_1), \cdots, (x_n, y_n)$. The general SVR function takes the following form:

$$f(x_t) = W \cdot \Phi(x_t) + b \quad (8)$$

where W denotes the weights, $\Phi(\cdot)$ represents a nonlinear transformation from primal space to the high-dimensional feature space, and b is the bias. The goal is to minimize the following objective

$$\min_{W,b} R(f) = \frac{1}{2}\|W\|^2 + C\sum_{i=1}^{m}\Gamma(f(x_t) - y_t) \tag{9}$$

where C is a positive regularized parameter and $\Gamma(\cdot)$ is the cost function. The ϵ-insensitive quadratic loss function is defined as follows:

$$\Gamma(f(x_t) - y_t) = \begin{cases} |f(x_t) - y_t| - \epsilon, & |f(x_t) - y_t| \geq \epsilon \\ 0, \text{ otherwise} \end{cases} \tag{10}$$

After combining the equations (10), construct a Lagrangian dual function to solve the quadratic optimization problem (9). The Kernel functions used in SVR are specified in the high-dimensional feature space without explicitly knowing the transformation $\Phi(\cdot)$. The radial basis function is used as the kernel for nonlinear regression, which can be expressed by

$$k(x_t, x) = exp(-\gamma\|x - x_t\|^2), \tag{11}$$

where γ is a hyperparameter to be tuned.

PLS PLS (Geladi and Kowalski (1986)) regression is a method that fits a linear regression model by projecting the input data and the output data to a lower-dimensional latent space. PLS regression method is suitable to handle high-dimensional and collinear data. The algorithm implementation process is as follows.

Considering a pair of input and output data matrices X and Y, PLS regression builds a linear model by decomposing matrices X and Y into the bilinear form:

$$X = t_1 p_1^T + E_1; Y = u_1 q_1^T + F_1, \tag{12}$$

where t_1 and u_1 represent the latent score vectors of the first PLS factor, p_1 and q_1 are the corresponding loading vectors. E_1 and F_1 are the residuals. Then the PLS builds an inner linear relationship between u_1 and t_1:

$$u_1 = b_1 t_1 + r_1, \tag{13}$$

where b_1 is the regression coefficient obtained by minimizing the residual r_1. After calculating the first PLS factor, this procedure is repeated to decompose the residuals to obtain subsequent PLS factors. Assuming the procedure ends with k factors, then the PLS decomposes the original X, Y into

$$X = t_1 p_1^T + t_2 p_2^T + \cdots + t_k p_k^T + E, \tag{14}$$

$$Y = b_1 t_1 q_1^T + b_2 t_2 q_2^T + \cdots + b_k t_k q_k^T + F, \tag{15}$$

where $\widehat{Y} = b_1 t_1 q_1^T + b_2 t_2 q_2^T + \cdots + b_k t_k q_k^T$ represents the prediction results, E and F are the corresponding residuals.

Lasso Lasso (Tibshirani (1996)) is a regression analysis method that performs variable selection via regularization in order to improve the prediction accuracy of the resulting statistical model. Lasso regression is a type of linear regression that performs l_1 regularization. l_1 regularization can result in sparse models with few coefficients. Larger penalty results in coefficient values closer to zero, which produces a simpler model. The goal of the Lasso is to minimize:

$$\sum_{t=1}^{n}(y_t - \sum_{j=1}^{m} x_{tj}\beta_j)^2 + \lambda\sum_{j=1}^{m}|\beta_j|, \tag{16}$$

where λ is a regularization parameter that shrinks the l_1 penalty term.

3. NOX EMISSION PREDICTION ON A 660MW BOILER

3.1 Boiler process data description

The boiler process that equips with an analytical instrument to measure NOx emission is shown in Fig. 3. The measured data from a 660 MW tangential pulverized coal-fired utility boiler is provided in Tan et al. (2019). It includes 10,000 samples covering seven days of operations with a sampling interval of one minute. There are 50 operational variables, such as pressure, temperature, flow rate sensors, and boiler load. Before modeling, we preprocess the data by cleaning the outliers and removing constant variables. We scale the data to zero mean and unit standard deviation. After preprocessing, 26 operational variables and past NOx emissions are used as the model input.

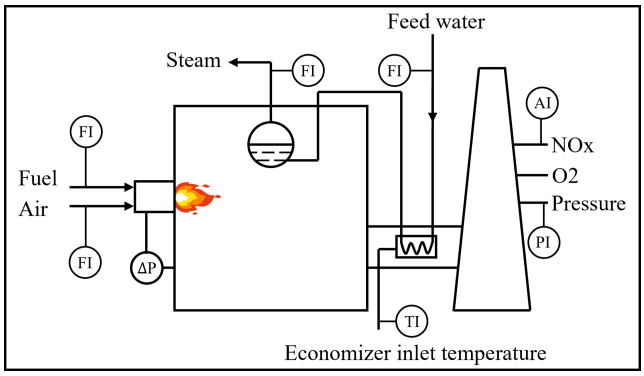

Fig. 3. Schematic of an industrial boiler with NOx emission

The correlations of 26 process variables and A&B side NOx emission are shown in Fig. 4. From Fig. 4, it is clear that variables 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 21, 22 are correlated with NOx emission, and they are also highly collinear. The collinearity is due to variable 1, the boiler load, which leads to variations of other process variables. In addition, NOx emissions at A and B sides are highly correlated because they are produced by the same process, although located at two separate sides at the furnace exit.

In this study, we build an inferential sensor model to predict NOx emissions of the A and B sides. 80% of the data are used for model training, while the remaining 20% are used for testing. The coefficient of determination (R^2) and root mean square error (RMSE) are employed as the indicators to evaluate the model performance, which are shown below.

$$\texttt{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}, \tag{17}$$

$$R^2 = 1 - \frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{n}(y_t - \bar{y})^2}, \tag{18}$$

where y_t is the actual NOx emission value, $\hat{y}_t$ represents the predicted value, and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_t$. To calculate R^2 for a test set, $\bar{y}$ of the training data should be used.

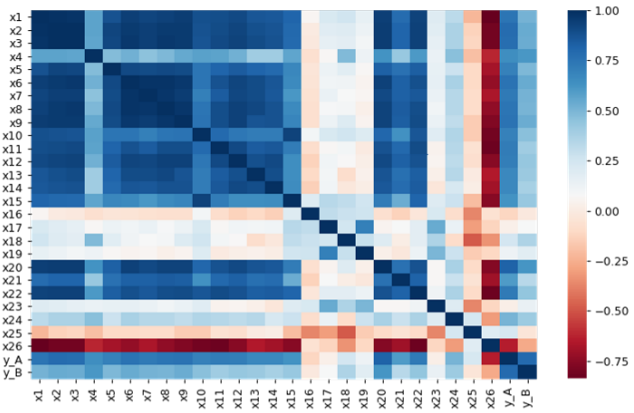

Fig. 4. Correlation matrix heatmap

3.2 Results and Comparison

To build the LSTM model, there are three critical hyperparameters we need to tune: learning rate, time steps, and the number of nodes. A grid search for the best RMSE is used to select the optimal hyperparameters. The range of candidate values and the optimal values of hyperparameters are given in Table 1. These optimal hyperparameters will be employed in the LSTM model.

Table 1. Search ranges and optimal values of the LSTM network hyperparameters

	Search range	A side	B side
Learning rate	0.1, 0.01, 0.001, 0.0001	0.001	0.001
Time steps	1, 2, ..., 10	3	6
No. of nodes	$2^3, 2^4, ..., 2^{10}$	512	1024

Grid search and five-fold cross-validation are employed for the statistical methods to select the optimal model hyperparameters before prediction. For the SVR model, the range of both γ and C are $2^{-8}, 2^{-7}, \cdots, 2^0, \cdots, 2^8$. The range of lags is from 1 to 10. On the A side, the best parameter combination is: lags=3, $\gamma = 0.00390625$, C=64. On the B side, the best parameter combination is: lags=2, $\gamma = 0.00390625$, $C = 128$. For the PLS model, on the A side, the optimal hyperparameter combination is: lags=5, the number of factors = 49. On the B side, the hyperparameters are 4 and 32. For the Lasso model, we need to determine the optimal penalty coefficient λ. The $log\lambda$ is composed of 60 points ranging from -8 to 0 with the same interval. The selected lags of the A&B side are 6&7, and the corresponding optimal λ are 0.000440 and 0.000660, respectively.

Table 2. R^2 values of the LSTM, SVR, PLS, and Lasso models

Method	LSTM	SVR	PLS	Lasso
A side Training R^2	0.9868	0.9962	0.9942	0.9937
B side Training R^2	0.9900	0.9886	0.9857	0.9853
A side Test R^2	0.8617	0.9336	0.9728	0.9688
B side Test R^2	0.8963	0.9320	0.9661	0.9622

The prediction performance of LSTM, SVR, PLS, and Lasso is compared in Table 2. From Table 2, test R^2 values of statistical models are all greater than 0.93, which indicates that combining dynamics with these traditional static models performs well in dynamic industrial processes. Additionally, PLS and Lasso have better prediction accuracy than SVR in this application because PLS and Lasso can effectively solve the data collinearity problem. Comparing the test R^2 values of LSTM and statistical models, the results show that all statistical learning models perform better than LSTM on both the A and B sides of NOx emissions. The results are surprising, which leads us to further investigate whether the LSTM structure and the related gradient-based learning algorithms would perform as expected.

The predicted NOx emission of training data and test data with the LSTM model are illustrated in Fig. 5. It can be seen that the blue line matches the grey dots in the training data on both A and B sides. Test data matching is not as good as the training data, especially in the stationary regions.

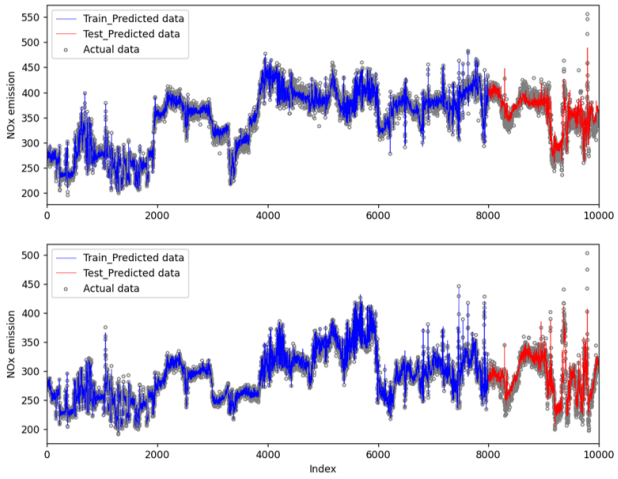

Fig. 5. LSTM: A side (Top) and B side (Bottom) actual & predicted training (in blue) and test (in red) data

In order to explore the performance of Lasso, we show the detailed Lasso prediction results in Fig. 6 for both A and B sides. The predictions on the training data (blue curve) and the testing data (red curve) match well with the actual NOx emission, which shows excellent prediction performance.

Fig. 7 shows the variable selection results with the Lasso. Around 100 variable coefficients of each model are close to zero. The coefficients with large absolute values are concentrated in the most recent instants, suggesting that the prediction results are mainly related to the recent data. From Fig. 7, Variable 1 (boiler load) strongly relates to the NOx emission. Through the coefficients of Variable 1 and other highly correlated variables will be arbitrarily selected by lasso, it is indeed the critical factor for NOx emission in the actual industry process. Moreover, the coefficient of Variable 27 (lagged NOx emission) decreases with the increase of lag time, which means the current NOx emission is highly correlated with its recent emission. Further Variable 25 is strongly associated with NOx on Side A, while Variable 26 is strongly associated with NOx on Side B.

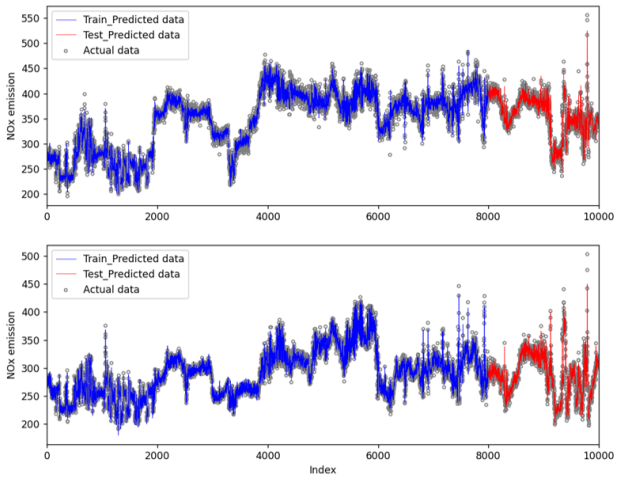

Fig. 6. Lasso: A side (Top) and B side (Bottom) actual & predicted training (in blue) and test (in red) data

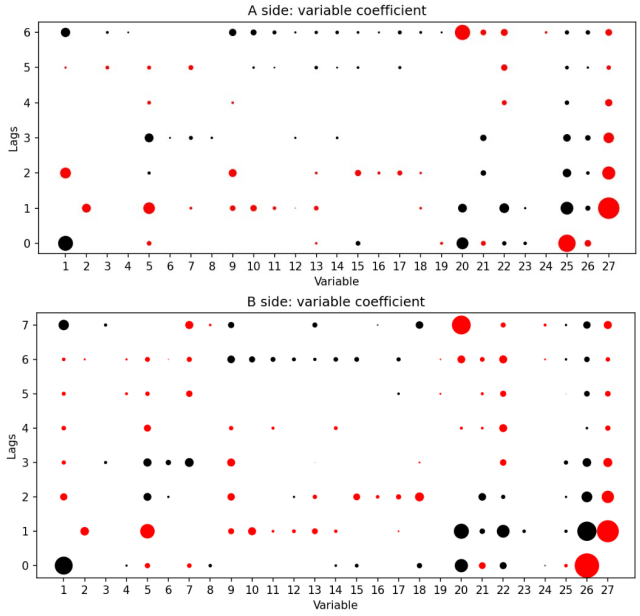

Fig. 7. Variable coefficients of the Lasso models on A side (Top) and B side (Bottom). The size of the dots represents the magnitude of the coefficients, with red color indicating positive coefficients and black color indicating negative coefficients.

4. DISSECTING LSTM FOR THE NOX PREDICTION

LSTM should outperform statistical models because of its strong ability to learn process dynamics. However, the application to a 660MW boiler for NOx emission prediction has shown the opposite results. This section dissects the internal states of LSTM for the simplest cases and inspects why it cannot outperform these statistical learning methods in this application.

Greff et al. (2016) has demonstrated that the gates and activation functions enhanced the performance of LSTM with applications in speech recognition, handwriting recognition, and polyphonic music modeling. In order to test whether the gates and activation functions in the inferential sensor application also perform as they were designed. Two simplest LSTM cases have been experimented with for A side NOx emission prediction to inspect the roles of the gates and activation functions. The LSTM network is the same as in Fig. 2, but we use only one hidden layer with one LSTM node. The two LSTM cases are:

LSTM Case I: Fixing Input Gate and Output Gate as $i_t = 1$ and $o_t = 1$.

LSTM Case II: In addition to Case I, disabling the Forget Gate with $f_t = 0$ and choosing a linear activation function $\sigma_h(x) = x$. In this case, the LSTM Equations (4) to (6) are reduced to

$$h_t = \sigma_c(W_c x_t + U_c h_{t-1} + b_c). \tag{19}$$

Note that this simple LSTM model still has recurrence in the hidden LSTM node. Fig. 8 shows the cell weights of the two LSTM cases, which represent the relationship between the corresponding variables and the cell state. From Fig. 8, the two cases have similar cell weights despite the opposite signs. The hyperbolic tangent and weights in the subsequent feedforward layer can flip the signs of the hidden state values. Variables 1 (unit load) and 27 (lagged NOx emission) are much more significant than other variables because they are weighted strongly. These results are consistent with the Lasso estimated coefficients and physical process knowledge, which indicates that the neural network successfully learned the correlations between the response and the predictor variables. The prediction performance of these two cases with one LSTM node is compared with the full LSTM with multi-nodes, as described in Section III. The training and test R^2 values of the three LSTM architectures are summarized in Table 3. Among the three models, Case II with the disabled forget gate shows the best prediction accuracy, which Case I with the enabled forget gate still performs better than the full LSTM model. It is noted that the optimized full LSTM model uses train-test validation to avoid over-training, and the number of hidden nodes is optimized as well. Theoretically, the full LSTM model should perform better than the two simple LSTM cases without the input and output gates. However, the results of the experiments reveal that these gates do not perform as they are intended to do in this 660MW boiler application. The results are consistent with Lasso and other statistical learning methods. One potential reason accounting for the result is that a simplified LSTM variant can avoid multiple local minima in the gradient-based training.

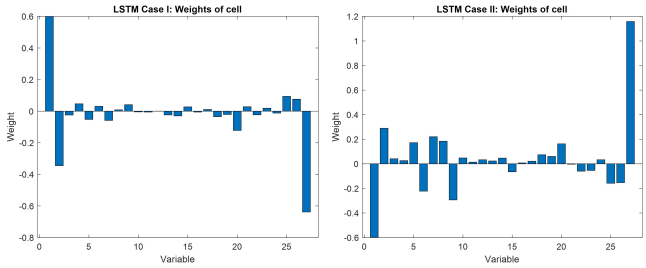

Fig. 8. The cell weights of LSTM Case I (Left panel) and LSTM Case II (Right panel)

Table 3. Comparison of R^2 values of LSTM with three different architectures

Architecture	Full LSTM	LSTM Case I	LSTM Case II
Training R^2	0.9868	0.9690	0.9680
Test R^2	0.8617	0.9020	0.9100

5. CONCLUSIONS

In this paper, fair comparisons of dynamic inferential modeling using LSTM and several statistical learning models (including SVR, PLS, and Lasso) are reported on an industrial 660MW boiler process to predict NOx emission. The results show that SVR, PLS, and Lasso with incorporated dynamics exhibit better prediction accuracy than LSTM, even though extensive searches for optimal LSTM hyperparameters were performed. On the other hand, by disabling the LSTM input and output gates and using a single hidden node, the performance is better than that of the fully optimized LSTM with the input and output gates and many hidden nodes. The performance is sustained by further disabling the forget gate. With all LSTM gates disabled, it is reduced to the external RNN structure (Qin et al. (1992)). These experiments show that the performance is improved by manually disabling the gates. Moreover, the complexity in parameter tuning and lack of transparent interpretation of LSTM could limit its adoption in practice. On the contrary, statistical learning methods such as Lasso show excellent prediction accuracy and strong model interpretability.

Although LSTM is mathematically capable of learning complex models, this study shows that it does not easily recover simple statistical models from finite and noisy data with gradient-based training algorithms. The absence of superior performance by LSTM is also reported in Ljung et al. (2020) where LSTM is compared to alternative methods for system identification. The main message of this paper is not to propose a new superior method but to demonstrate that a deep learning model developed in other domains requires scrutiny and detailed study to show its superiority in process applications.

ACKNOWLEDGMENTS

The authors acknowledge the financial support for this work from the City University of Hong Kong under Project 9380123 and an NSF China Grant (U20A20189).

REFERENCES

Carbune, V., Gonnet, P., Deselaers, T., Rowley, H.A., Daryin, A., Calvo, M., Wang, L.L., Keysers, D., Feuz, S., and Gervais, P. (2020). Fast multi-language LSTM-based online handwriting recognition. *International Journal on Document Analysis and Recognition*, 23(2), 89–102.

Dong, Y. and Qin, S.J. (2018). Regression on dynamic PLS structures for supervised learning of dynamic data. *Journal of Process Control*, 68, 64–72.

Drucker, H., Burges, C.J., Kaufman, L., Smola, A., Vapnik, V., et al. (1997). Support vector regression machines. *Advances in neural information processing systems*, 9, 155–161.

Geladi, P. and Kowalski, B.R. (1986). Partial least-squares regression: a tutorial. *Analytica chimica acta*, 185, 1–17.

Ghosh, S., Yang, S., and Bequette, B.W. (2020). Inferential modeling and soft sensors. *Smart Manufacturing*, 323–351.

Graves, A., Jaitly, N., and Mohamed, A.r. (2013). Hybrid speech recognition with deep bidirectional LSTM. In *2013 IEEE workshop on automatic speech recognition and understanding*, 273–278. IEEE.

Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., and Schmidhuber, J. (2016). Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10), 2222–2232.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). Statistical learning with sparsity. *Monographs on statistics and applied probability*, 143, 143.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.

Kano, M., Miyazaki, K., Hasebe, S., and Hashimoto, I. (1998). Inferential control system of distillation compositions using dynamic partial least squares regression. *IFAC Proceedings Volumes*, 31, 375–384.

Kaspar, M.H. and Ray, W.H. (1993). Dynamic PLS modelling for process control. *Chemical Engineering Science*, 48(20), 3447–3461.

Kresta, J., Marlin, T., and MacGregor, J. (1994). Development of inferential process models using PLS. *Computers & Chemical Engineering*, 18(7), 597–611.

Ljung, L., Andersson, C., Tiels, K., and Schön, T.B. (2020). Deep learning and system identification. *IFAC-PapersOnLine*, 53(2), 1175–1181. 21st IFAC World Congress.

Pranatyasto, T. and Qin, S. (2001). Sensor validation and process fault diagnosis for FCC units under MPC feedback. *Control Eng. Practice*, 9, 877–888.

Qin, S.J. and McAvoy, T. (1996). Nonlinear FIR modeling via a neural net PLS approach. *Computers & Chemical Engineering*, 20(2), 147–159.

Qin, S.J., Yue, H., and Dunia, R. (1997). Self-validating inferential sensors with application to air emission monitoring. *Industrial & engineering chemistry research*, 36(5), 1675–1685.

Qin, S.Z., Su, H.T., and McAvoy, T.J. (1992). Comparison of four neural net learning methods for dynamic system identification. *IEEE Trans. on Neural Networks*, 3(1), 122–130.

Sun, Q. and Ge, Z. (2021). A survey on deep learning for data-driven soft sensors. *IEEE Transactions on Industrial Informatics*, 17(9), 5853–5866.

Tan, P., He, B., Zhang, C., Rao, D., Li, S., Fang, Q., and Chen, G. (2019). Dynamic modeling of NO_x emission in a 660 mw coal-fired boiler with long short-term memory. *Energy*, 176, 429–436.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267–288.

Yuan, X., Li, L., and Wang, Y. (2019). Nonlinear dynamic soft sensor modeling with supervised long short-term memory network. *IEEE Transactions on Industrial Informatics*, 16(5), 3168–3176.

Zhao, H. (2021). An industry perspective on AI, machine learning and data science towards Industry 4.0. In *IFAC Workshop Series on Control Systems and Data Science towards Industry 4.0*.