IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 11, 2017, accepted January 7, 2018, date of publication January 30, 2018, date of current version March 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2794765

Sequential Fault Diagnosis Based on LSTM Neural Network

HAITAO ZHAO[1], SHAOYUAN SUN[2], AND BO JIN[3]

[1]Automation Department, School of Information Science and Engineering, East China University of Science and Technology, Shanghai 200037, China
[2]College of Information Sciences and Technology, Donghua University, Shanghai 201620, China
[3]School of Computer Science and Software Engineering, East China Normal University, Shanghai 200062, China

Corresponding author: Haitao Zhao (haitaozhao@ecust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under 61375007 and in part by the Basic Research Programs of Science and Technology Commission Foundation of Shanghai under Grant 15JC1400600.

ABSTRACT Fault diagnosis of chemical process data becomes one of the most important directions in research and practice. Conventional fault diagnosis and classification methods first extract features from the raw process data. Then certain classifiers are adopted to make diagnosis. However, these conventional methods suffer from the expertise of feature extraction and classifier design. They also lack the adaptive processing of the dynamic information in raw data. This paper proposes a fault diagnosis method based on long short-term memory (LSTM) neural network. The novel method can directly classify the raw process data without specific feature extraction and classifier design. It is also able to adaptively learn the dynamic information in raw data. First, raw process data are used to train the LSTM neural network until the cost function of LSTM converges below certain predefined small positive value. In this step, the dynamic information of raw process data is adaptively learned by LSTM. Then testing data are used to obtain the diagnosis results of the trained LSTM neural network. The application of LSTM to fault identification and analysis is evaluated in the Tennessee Eastman benchmark process. Extensive experimental results show LSTM can better separate different faults and provide more promising fault diagnosis performance.

INDEX TERMS Process monitoring, fault diagnosis, recurrent neural network, long short-term memory (LSTM) neural network.

I. INTRODUCTION

Monitoring process conditions is crucial to its normal operation. Over last few years, data-driven statistical process monitoring (SPM) has been widely applied to fault diagnosis for industrial process operations and production results [1]–[4]. Due to the data-based nature of SPM, it is relatively convenient to apply to real processes of large scale comparing to other methods based on theoretical modelling or rigorous derivation of process systems [5]–[9].

MacGregor and Cinar [1] provided latent variable models which reduced the high dimensional processes into low dimensional models. Yin *et al.* [2] provided a comparison study on different FE algorithms for process monitoring and fault diagnosis. All of their methods were tested on the Tennessee Eastman (TE) benchmark process. Feital *et al.* [5] considered the multimodal modeling for FE in process monitoring. The process conditions were divided into three components describing between-cluster variation, within-cluster variation, and model residuals. Qin [10] were mainly focused on the reconstruction- and contribution-based FE methods. A hierarchical monitoring framework was proposed in [10] as a way for fault analysis including detectability, reconstructability and identifiability conditions.

The task of SPM is challenging mainly because of the ''curse of dimensionality'' problem and the ''data rich but information poor'' problem. Many methods have been proposed to embed original process data into a lower dimensional feature space and then performing fault detection or fault diagnosis in that feature space. Principal component analysis (PCA) [11], [12], partial least squares (PLS) [13], independent component analysis (ICA) [14] and linear discriminant analysis (LDA) [15] are the most widely used feature extraction methods in the fields of fault detection and fault diagnosis.

Considering the dynamic properties of raw process data, dynamic principal component analysis (DPCA) [16], [17]

and dynamic linear discriminant analysis (DLDA) [3], [16] were adopted. Both of them improved the fault diagnosis performance by constructing extended vectors through concatenating current process data and certain number of previous process data in order to incorporate the dynamic information of raw data in feature extraction. Other methods adopted augmented matrices instead of vectors to utilize dynamic information by extending each sample into a matrix comprising current and past process data within certain intervals [18]. Both the vector-based augmentation and matrix-based augmentation may aggravate the ''curse of dimensionality'' problem and make the feature extraction methods unstable [19]. Moreover, the structures of the augmentations are fixed in advance, i.e. the adoption of the dynamic information are not adaptively learned from raw process data.

After feature extraction, different diagnosis or classification methods have been developed to determine the root cause of faults. Fault diagnosis can be viewed as a supervised learning task whose target is to classify a new testing observations to one of the existing classes. Various classifiers such as support vector machines (SVM) [20], Bayesian discriminant functions [18], neural networks (NNs) [21], and ANFIS [22] have been applied for fault diagnosis of chemical processes. Among these classifiers, the neural networks of multilayer perceptron (MLP) type have received considerable attention. Numerous studies of fault diagnosis have shown the simplicity and efficiency of MLP and its variants to model the extracted features. Nevertheless, the majority of these works make use of conventional ''feature + classifier'' strategy, such as ''feature of PCA + ANFIS'' [22] and ''feature of ICA + MLP'' [21]. The reason why the extracted features are suitable for the following classifiers are not stated clearly. Moreover, the conventional classifiers used in fault diagnosis have no relationship with the dynamic information of chemical process data. For example, Conventional static neural networks (such as MLP) take each data independently for training and ignore the correlation information between different data.

Recently, however, a trend in the deep learning community has emerged towards an *end-to-end* manner, which combines feature extraction and classifier design into one neural network. The motivation behind this idea is that the neural network automatically learns both the features of raw data and the classifier which better suit the fault diagnosis task and hence lead to improved performance. In order to deal with the dynamic information of time series, Recurrent neural network (RNN) [23] architectures such as long short-term memory (LSTM) neural network and its variants [24], [25] have exhibited state-of-the-art performance on a wide range of complicated sequential problems including signal processing, speech classification and video captioning. LSTM can adaptively learn the dynamic information of time sequences by non-linear gating units regulate the information into and out of the memory cells of LSTM.

In this paper, We propose a fault diagnosis method based on long short-term memory (LSTM) neural network. The main contributions of this paper are as follows:

1) Contrary to the conventional practises of fault diagnosis, where features of raw process data are extracted individually and then simply fed to a classifier, our method is trained in an end-to-end manner which provides a framework to learn the representation of raw input data and classifier simultaneously.
2) the dynamic information of process data are adaptively utilized and learned by LSTM. The parameters of activation functions of different cells in LSTM can be trained to represent highly correlated features that are essential for fault diagnosis.
3) In order to reduce the internal covariate shift of LSTM, batch normalization procedure is adopted in our method to improve the convergence of LSTM. Covariate shift is a common problem of deep neural networks, in which the following layers are continually affected by the shifting distributions of the previous layers. Covariate shift may degrades the efficiency of training and makes LSTM unable to learn effectively. Batch normalization solves the covariate shift problem by standardizing the activations going into each layer, enforcing their means and variances to be invariant to changes in parameters of other layers. Experimental results show that our method converges significantly fast and has good performance.

The application of our LSTM-based fault diagnosis method to fault identification and analysis is evaluated on the Tennessee Eastman (TE) benchmark process. Extensive experimental results show our method can better separate different faults with our design and provide more promising fault diagnosis performance.

II. RECURRENT NEURAL NETWORK

Recurrent neural network (RNN) [23], [25], [26] is able to process sequential data one sample at a time. In this way, RNN adaptively models dynamic information of sequential data on multiple scales. The architecture of standard RNN is presented in Figure 1. The node $\mathbf{z}_t$ receives input from the current sample $\mathbf{x}_t$ as well as the hidden state value of the hidden layer in the previous state $\mathbf{h}_{t-1}$. Thus RNN is neural network with loops, adaptively allowing information to persist long period of time.

Given an input sequence $X = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T]$, an RNN defines a sequence of hidden states $\mathbf{h}_t$ by

$$\mathbf{h}_t = \psi(\mathbf{z}_t) = \psi(W_h \mathbf{h}_{t-1} + W_x \mathbf{x}_t + \mathbf{b}) \tag{1}$$

where $W_h \in \mathbb{R}^{d_h \times d_h}$, $W_x \in \mathbb{R}^{d_h \times d_x}$, $\mathbf{b} \in \mathbb{R}^{d_h}$ and the initial state $\mathbf{h}_0 \in \mathbb{R}^{d_h}$ are parameters of RNN. A popular choice of the activation function $\psi(\cdot)$ is tanh.

From Figure 2, it is easy to find that an RNN can be thought of as multiple copies of the same network, each passing a message to a successor.

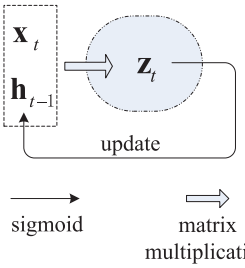


FIGURE 1. The architecture of standard recurrent neural network.

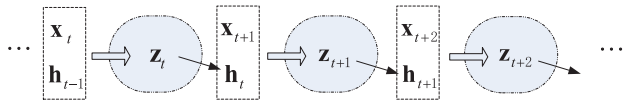


FIGURE 2. The feedforward structure of standard recurrent neural network.

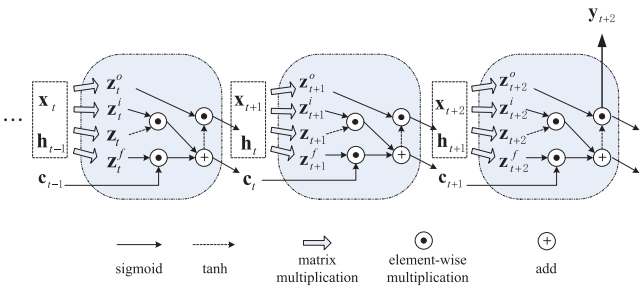


FIGURE 3. The architecture of LSTM.

RNNs are designed for sequential data modelling. However, training RNNs using stochastic gradient descent (SGD) is notoriously hard because of the well-known problem of vanishing/exploding gradients [27]. The exploding gradient problem is relatively easy to solve by constraint over the norm of the gradients. On the other hand, the vanishing gradient problem can be mitigated through architectural variations such as LSTM, GRU and *i*RNN/*u*RNN [25], [27].

III. VANILLA LSTM

A. CONCEPT OF VANILLA LSTM

After refinement and popularization, the variant of LSTM, vanilla LSTM [24] is widely used in sequential data processing. The architecture of the vanilla LSTM is illustrated in Figure 3.

Vanilla LSTM has this chain like structure like RNN, but the repeating module is totally different. Instead of having a single neural network layer, there are four, interacting in a very special structure. The internal structure of vanilla LSTM is based on a set of connected cells. Different from simple RNN which overwrites the cell information directly, each cell of the vanilla LSTM contains three gates serving as the controllers for information propagation within the network. In what follows, we focus on the architecture of the vanilla

TABLE 1. TEP fault modes in 2 cases (RV means Random Variation).

Case	Fault	Description	Type
1	1	A/C Feed ratio, B composition constant (Stream 4)	Step
	2	B composition, A/C ratio constant (Stream 4)	Step
	6	A feed loss (Stream 1)	Step
	7	C header pressure loss (Stream 4)	Step
	8	A, B, C feed composition (Stream 4)	RV
2	3	D feed temperature (Stream 2)	Step
	4	Reactor cooling water inlet temperature	Step
	5	Condenser cooling water inlet temperature	Step
	9	D feed temperature (Stream 2)	RV
	10	C feed temperature (Stream 4)	RV
	11	Reactor cooling water inlet temperature	RV
	12	Condenser cooling water inlet temperature	RV

LSTM with recurrent transition given by

$$\begin{pmatrix} \mathbf{z}_t^o \\ \mathbf{z}_t^i \\ \mathbf{z}_t \\ \mathbf{z}_t^f \end{pmatrix} = \mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t + \mathbf{b} \tag{2}$$

$$\mathbf{c}_t = \sigma\left(\mathbf{z}_t^f\right) \odot \mathbf{c}_{t-1} + \sigma\left(\mathbf{z}_t^i\right) \odot \tanh\left(\mathbf{z}_t\right) \tag{3}$$

$$\mathbf{h}_t = \sigma\left(\mathbf{z}_t^o\right) \odot \tanh\left(\mathbf{c}_t\right) \tag{4}$$

where $\mathbf{W}_h \in \mathbb{R}^{4d_h \times d_h}$, $\mathbf{W}_x \in \mathbb{R}^{4d_h \times d_x}$, $\mathbf{b} \in \mathbb{R}^{4d_h}$ and the initial states $\mathbf{h}_0 \in \mathbb{R}^{d_h}$, $\mathbf{c}_0 \in \mathbb{R}^{d_h}$ are the parameters of the network. The $\odot$ operator denotes the Hadamard product (element-wise multiplication). The $\sigma(\cdot)$ is the sigmoid function [25].

Unlike simple RNN, the vanilla LSTM has an additional memory element $\mathbf{c}_t$ whose update is approximately linear that allows the gradient to flow back through time easily. What is more, different from RNN which overwrites the cell at every time step, the update of the cell of the vanilla LSTM is regulated by three gates:

1) The forget gate $\sigma\left(\mathbf{z}_t^f\right)$ controls the extent to which information is transferred from the previous time step;
2) The input gate $\sigma\left(\mathbf{z}_t^i\right)$ determines the flow of information of the current input data $\mathbf{x}_t$;
3) The output gate $\sigma\left(\mathbf{z}_t^o\right)$ controls the information to be obtained from the cell.

This carefully design allows the vanilla LSTM to robustly remove or add information during long period of time.

B. BATCH NORMALIZATION

Covariate shift [28] is a common problem in deep neural networks where the features presented to a network change in distribution. In order to deal with the covariate shift, the parameters of the neural networks must be adjusted to minimize the loss function at hand but also to adapt to the changing distribution of the features. In deep neural networks, the changing of the parameters of one layer largely affects the distribution of all layers following it.

Batch normalization [28] is a recently proposed network reparametrization procedure to reduce the covariate shift problem. Batch normalization standardizing the activation functions using empirical estimates of the mean $\hat{E}(\mathbf{h})$ and

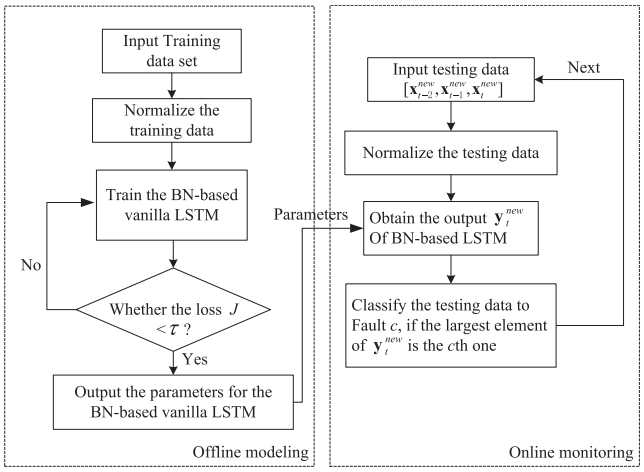


FIGURE 4. The steps of BN-based vanilla LSTM for fault diagnosis.

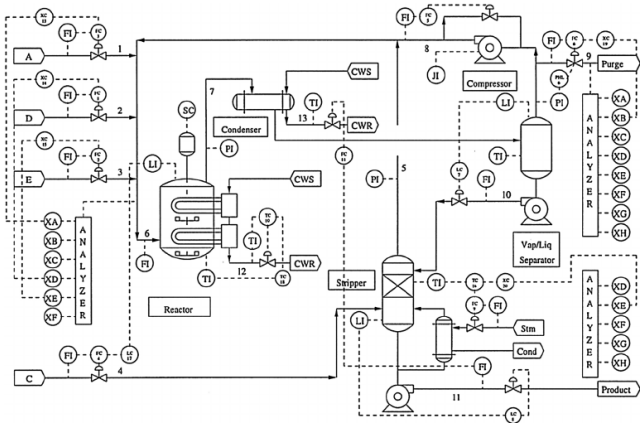


FIGURE 5. A diagram of the TEP simulator.

standard deviation $\hat{Var}(\mathbf{h})$ for each layer. The batch normalizing transform is as follows

$$\mathrm{BN}(\mathbf{h}; \eta, \rho) = \rho + \eta \odot \frac{\mathbf{h} - \hat{E}(\mathbf{h})}{\sqrt{\hat{Var}(\mathbf{h}) + \epsilon}} \tag{5}$$

where $\mathbf{h} \in \mathbb{R}^d$ is the vector of features to be normalized, $\eta \in \mathbb{R}^d$, $\rho \in \mathbb{R}^d$ are parameters that determine the mean and standard deviation of the normalized features, and $\epsilon \in \mathbb{R}$ is a regularization parameter. The division should be understood to proceed element-wise.

Since the sample mean $\hat{E}(\mathbf{h})$ and sample variance $\hat{Var}(\mathbf{h})$ are estimated of the training features of each layer, the backpropagation algorithm can be derived through these statistics and preserving the convergence properties of gradient decent.

C. BATCH-NORMALIZATION-BASED VANILLA LSTM

In order to avoid unnecessary redundancy and over fitting, we set $\rho = 0$ in our design of batch normalization, and denote the simplified batch normalization as $\mathrm{BN}(\cdot; \eta)$. We adopt the

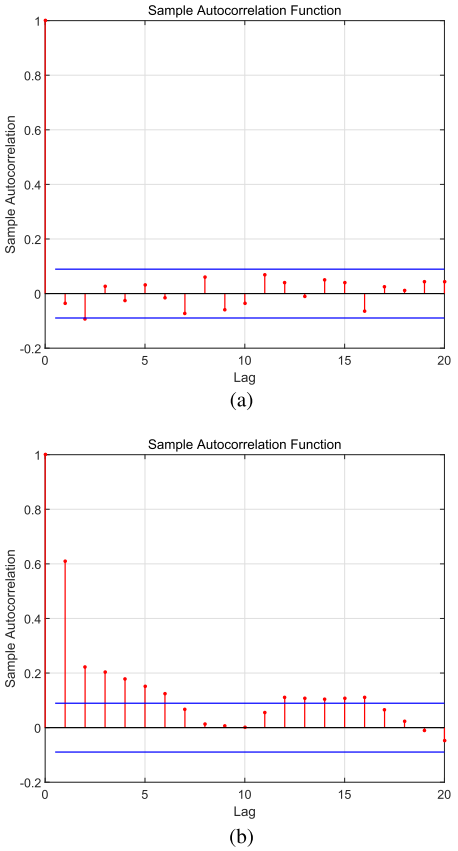


FIGURE 6. Autocorrelation charts for different variables. (a) Autocorrelation chart for the variable of product separator level. (b) Autocorrelation chart for the variable of A feed (Stream 1).

batch normalization into the vanilla LSTM as follows:

$$\begin{pmatrix} \mathbf{z}_t^o \\ \mathbf{z}_t^i \\ \mathbf{z}_t^f \\ \mathbf{z}_t^j \end{pmatrix} = \mathrm{BN}(\mathbf{W}_h \mathbf{h}_{t-1}; \eta_h) + \mathbf{W}_x \mathbf{x}_t + \mathbf{b} \tag{6}$$

$$\mathbf{c}_t = \sigma\left(\mathbf{z}_t^f\right) \odot \mathbf{c}_{t-1} + \sigma\left(\mathbf{z}_t^i\right) \odot \tanh\left(\mathbf{z}_t\right) \tag{7}$$

$$\mathbf{h}_t = \sigma\left(\mathbf{z}_t^o\right) \odot \tanh\left(\mathrm{BN}\left(\mathbf{c}_t; \eta_c\right)\right) \tag{8}$$

Since the training process data are normalized before training, we do not perform normalization on the term $\mathbf{W}_x \mathbf{x}_t$. In our design, we normalize the recurrent term $\mathbf{W}_h \mathbf{h}_{t-1}$. Normalizing this term gives the vanilla LSTM better control over the relative contribution of the terms using the η_h parameter. In order to preserve the dynamics of LSTM and maintain the gradient flow through $\mathbf{c}_t$, batch normalization is not performed in the update of $\mathbf{c}_t$. In this paper, we set η_h and η_c all equal 0.9.

For LSTM, outputs can be obtain in each time step, or in any time steps. There are four types of LSTM, one-to-one, one-to-many, many-to-many, many-to-one. In this paper, we use the many-to-one type of LSTM. We compute outputs after certain time steps (In Figure 3, we compute outputs every 3 time steps.). The output $\mathbf{y}_{t+2}$ can be obtained by

$$\mathbf{y}_{t+2} = \sigma\left(W_{yh} \mathbf{h}_{t+2} + \mathbf{b}_y\right), \tag{9}$$

where $W_{yh} \in \mathbb{R}^{d_y \times d_h}$, $\mathbf{b}_y \in \mathbb{R}^{d_y}$.

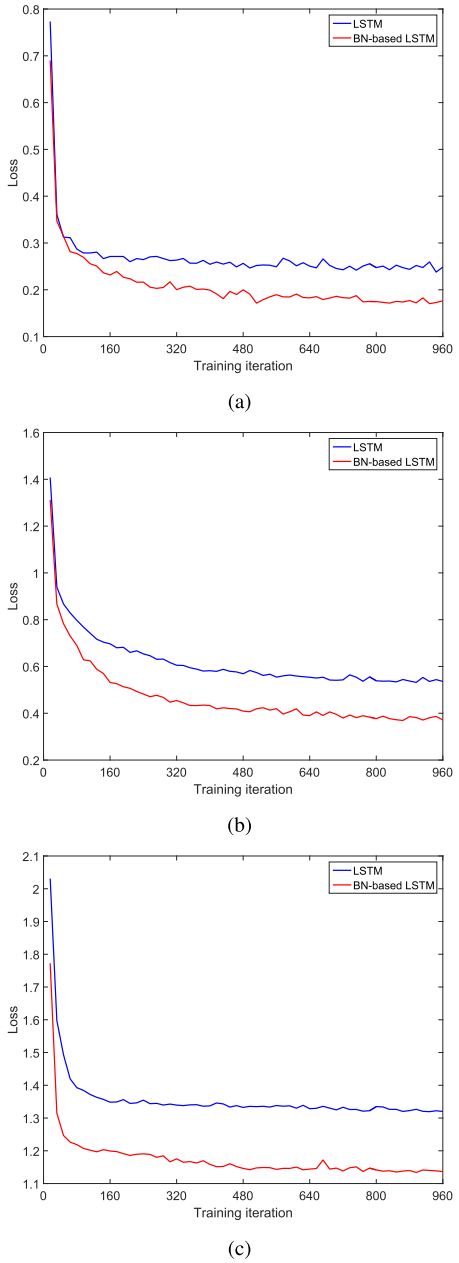


FIGURE 7. Comparison of convergence between a baseline LSTM and BN-based vanilla LSTM on different training sets during training (in the first 960 training iteration). (a) Comparison of convergence in Case 1. (b) Comparison of convergence in Case 2. (c) Comparison of convergence of 21 faults.

During training we estimate the sample mean and sample variance for each time step. At test time we use these statistics by averaging the estimates over the training set.

D. LOSS FUNCTION

Let $\omega_1, \omega_2, \cdots, \omega_C$ be C classes of fault. In the equation below, $1\{\cdot\}$ is the indicator function, so that $1\{\text{a } \textit{true statement}\} = 1$ and $1\{\text{a } \textit{false statement}\} = 0$. In this case, $d_y = C$, that is $\mathbf{y}_{t+2} = \left[y_{t+2}^{(1)}, y_{t+2}^{(2)}, \cdots, y_{t+2}^{(C)}\right]^T \in$

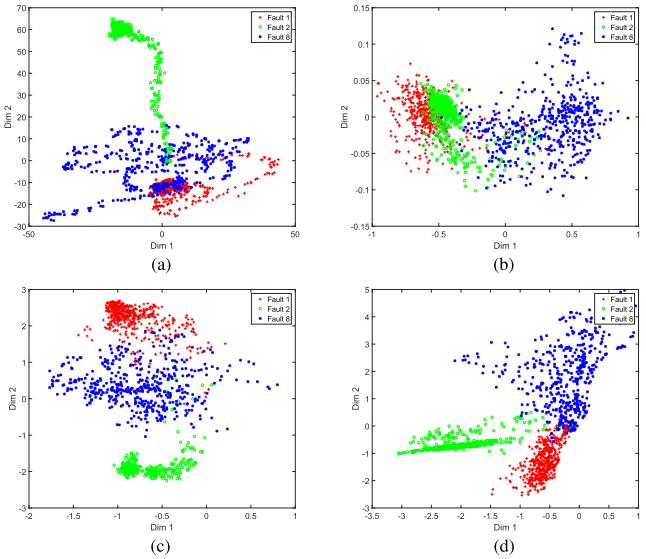


FIGURE 8. Visualization of the training samples of Fault 1, 2 and 8 in Case 1. Sub-Figure (a) and (b) plot the training data on the first two dimensions of PCA and the first two dimensions of LDA respectively. Sub-Figure (c) and (d) plot the outputs of the hidden layer of MLP and those of BN-based LSTM of the training samples, respectively. In this experiment, we design MLP and BN-based LSTM only with 2 nodes in the hidden layer, i.e. $d_h = 2$. (a) Visualization plot of DPCA. (b) Visualization plot of DLDA. (c) Visualization plot of the hidden layer of MLP. (d) Visualization plot of the hidden layer of BN-based LSTM.

$\mathbb{R}^C$. Our loss function will be:

$$J\left(\mathbf{W}_h, \mathbf{W}_x, W_{yh}, \mathbf{b}, \mathbf{b}_y\right) = -\frac{1}{T-2}\sum_{i=1}^{T-2}\sum_{j=1}^{C} 1\left\{\mathbf{x}_{i+2} \in \omega_j\right\} \ln \frac{y_{i+2}^{(j)}}{\sum_{k=1}^{C} y_{i+2}^{(k)}}. \quad (10)$$

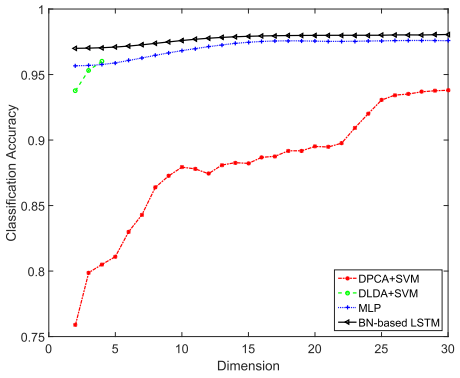
In this paper, we use Bengio [29], i.e. adaptive moment estimation, for mini-batch stochastic-gradient-based optimization of the loss function, based on adaptive estimates of lower-order moments. For more detail of Adam, please refer to [25].

E. FAULT DIAGNOSIS BASED ON LSTM

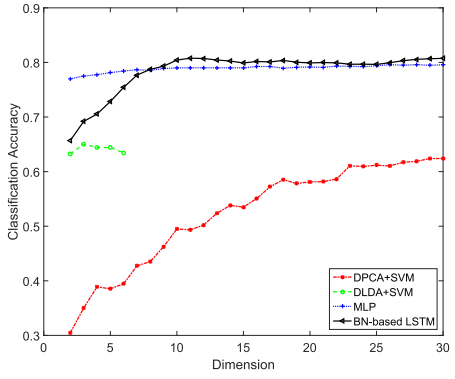
With the proposed batch-normalization-based vanilla LSTM, the diagnosis of fault data is straightforward. The offline modeling and online monitoring flow charts are shown in Figure 4. The procedures of offline modeling and online monitoring are as follows:

- Offline modeling:

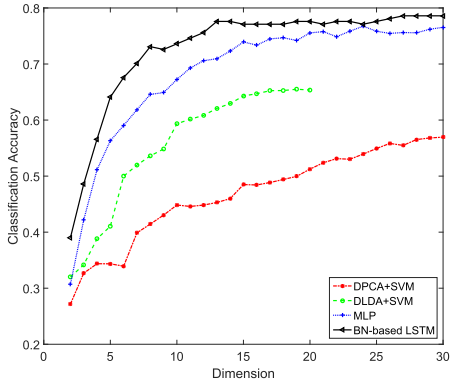
1) Collect process data as training data.
2) Normalize each feature of the training data.
3) Train the batch-normalization-based vanilla LSTM with Adam.
4) Compute the loss J in Equation (10). If $J > \tau$ or the number of iterations $l <$ MaxIter, goto 3. (τ is a small positive number and MaxIter is the predefined maximum number of iterations.)
5) Output the parameters for the BN-based vanilla LSTM.



(a)



(b)



(c)

FIGURE 9. Diagnosis results of DPCA + SVM, DLDA + SVM, MLP, and BN-based LSTM in different cases. (a) Diagnosis results of different methods in Case 1. (b) Diagnosis results of different methods in Case 2. (c) Diagnosis results of 21 faults by different methods.

- Online monitoring:

1) Sample a new augmented testing data $\left[\mathbf{x}_{t-2}^{new}, \mathbf{x}_{t-1}^{new}, \mathbf{x}_{t}^{new}\right]$ $(t \geq 3)$.
2) Normalize the testing data according to the means and variances of the training features.
3) Obtain the output $\mathbf{y}_t^{new}$, where $\mathbf{y}_t^{new} = \left[\left(y_t^{new}\right)^{(1)}, \left(y_t^{new}\right)^{(2)}, \cdots, \left(y_t^{new}\right)^{(C)}\right]^T$.
4) Classify the testing data to Fault $\hat{c}$, where $\hat{c} = \arg\max_j \left\{\left(y_t^{new}\right)^{(j)}\right\}$.

IV. SIMULATION AND DISCUSSION

This section compares the fault classification performance of DPCA + SVM, DLDA + SVM, MLP and BN-based vanilla LSTM on the benchmark Tennessee Eastman process (TEP). DPCA and DLDA are classical dynamic feature extraction methods. In DPCA and DLDA, we concatenate three samples to form the extended vectors for feature extraction. After the feature extraction of DPCA and DLDA, we use SVM as the classifier for fault diagnosis respectively. For SVM classifier [30], we utilize the SVM program in scikit-learn [31] with the rbf kernel and set the parameter $\gamma = 1/d_f$, where d_f is the number of the extract features of DPCA or DLDA. For the MLP method, we adopt the feed-forward neural network with one hidden layer and directly train this neural network with the raw process data using sigmoid function as the activation function. The code of our propsed BN-based vanilla LSTM in PyTorch can be found at https://github.com/haitaozhao/LSTM_fault_detection.

TEP has been widely used by process monitoring community as a source of publicly available data for comparing different algorithms. The simulated TEP is mainly based on a practical industrial process in which the kinetics, operation and units have been altered for specific reasons. The data generated by TEP are nonlinear, strong coupling and dynamic [16], [32]. A flow sheet of TEP with its implemented control structure is shown in Figure 5. The MATLAB codes can be downloaded from http://depts.washington.edu/control/LARRY/TE/download.html. There are five major units in TEP: a chemical reactor, condenser, recycle compressor, vapor/liquid separator, and stripper. Besides normal data, the simulator of TEP can also generate 21 different types of faults in order to test process monitoring algorithms.

A total of 52 variables including 22 continuous process measurements, 19 compositions and 11 manipulated variables[1] were selected as the monitoring variables in our simulation. Our first simulation includes 12 programmed fault modes. We group these 12 fault modes into two cases. The fault modes in each case are listed in Table 1. Fault modes in Case 1 are related to composition of feed and flow rate, while fault modes in Case 2 have relationship with temperature, 480 training samples and 800 testing samples are utilized in our experiments. Our second simulation takes all 21 faults for fault diagnosis. More details about TEP can be referred to [16].

A. DYNAMIC PROBLEM OF TEP

Dynamic problem or serial statistical correlation problem is common in TEP. Autocorrelation chart is a simple method to check whether correlations are present in each variable of the raw data. If significant autocorrelation is shown in the autocorrelation chart, the dynamic problem should be considered in fault diagnosis. Autocorrelation is a mathematical tool for finding cross-correlation which can be considered as the cross-correlation of a variable with itself at different points in

[1] the agitation speed was not included because it was not manipulated

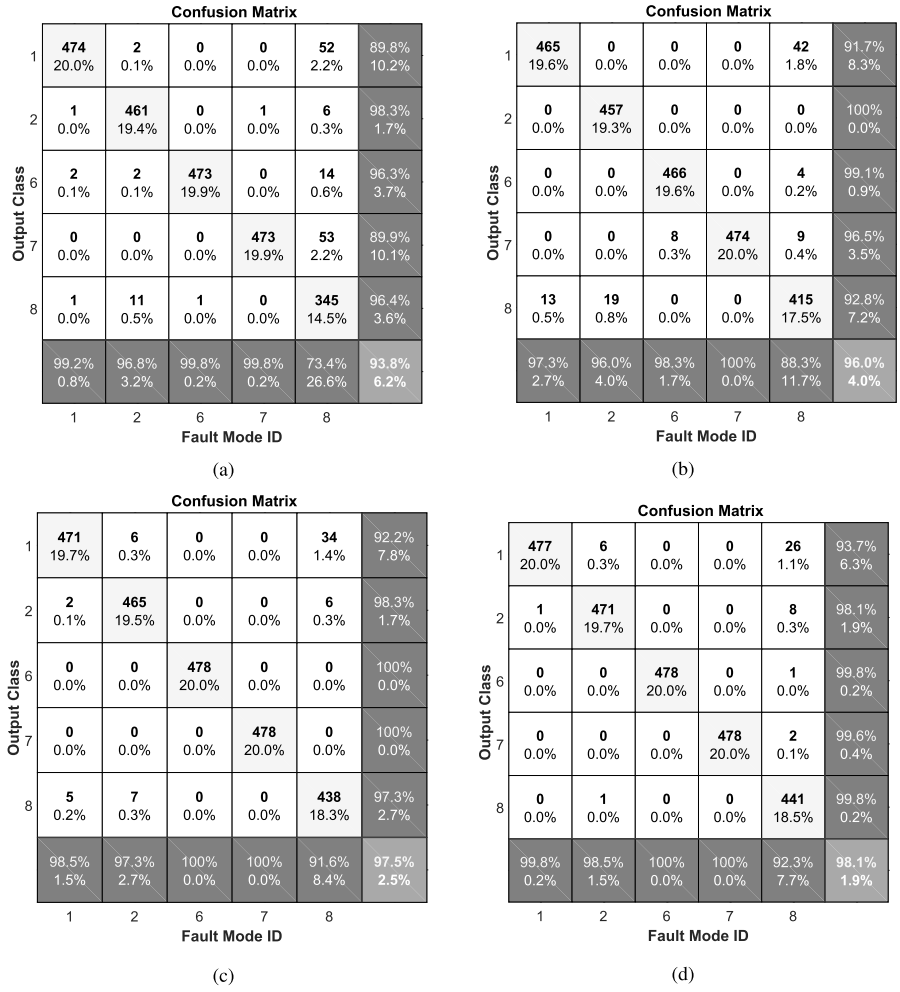


FIGURE 10. Confusion matrices of different methods in Case 1. (a) Confusion matrix of DPCA + SVM in Case 1. (b) Confusion matrix of DLDA + SVM in Case 1. (c) Confusion matrix of MLP in Case 1. (d) Confusion matrix of BN-based LSTM in Case 1.

time sequence. Let τ_t and τ_{t+k}, where $k = 0, \cdots, K$ and τ_t is a stochastic process. The formula [33] for the auocorrelation for lag k is

$$\hat{r}_k = \frac{c_k}{c_0}$$

where $c_k = \frac{1}{T-1}\sum_{t=1}^{T-k}(\tau_t - \bar{\tau})(\tau_t - \bar{\tau})$, $\bar{\tau}$ is the mean of τ_t $(t = 1, 2, \cdots, T)$, and c_0 is the sample variance of the time series. The standard error for testing the significance of a single lag-h autocorrelation, $\hat{r}_h$, is approximately

$$SE_r = \sqrt{\left(1 + 2\sum_{i=1}^{h-1}\hat{r}_i^2\right)/N}.$$

For example, Figure 6 shows the number of time lags for past and future data points determined from autocorrelation function (ACF) of two measurements of product separator level and A feed (Stream 1), respectively. In the figure, approximate 95% confidence intervals are drawn at $\pm 2SE_r$ (blue lines). Figure 6(a) shows that the ACF cuts off with no lag. This behavior indicates that there is no serial correlation in the measurement of product separator level. This measurement can be considered as independent sampled variable. As for the measurements of A feed (Stream1), the ACFs in Figure6(b) show time lags in this measurement.

Conventional methods, such as PCA and LDA, consider the data as independent sampled variables. In this way, no correlation information are taken into account. DPCA or DLDA algorithm try to utilize the dynamic information through extended vectors. However, due to the prefixed structure of the extension, DPCA or DLDA cannot deal with the different characteristics of the serial correlations of different variable. Thanks for the recurrent structure of LSTM and the adaptive training strategy, our proposed algorithm can take fully consideration of the dynamic information of different variables for further fault diagnosis.

B. EFFECTS OF BATCH NORMALIZATION

In deep learning, such as LSTM, where the features presented to a network change in distribution, covariate shift is a com-

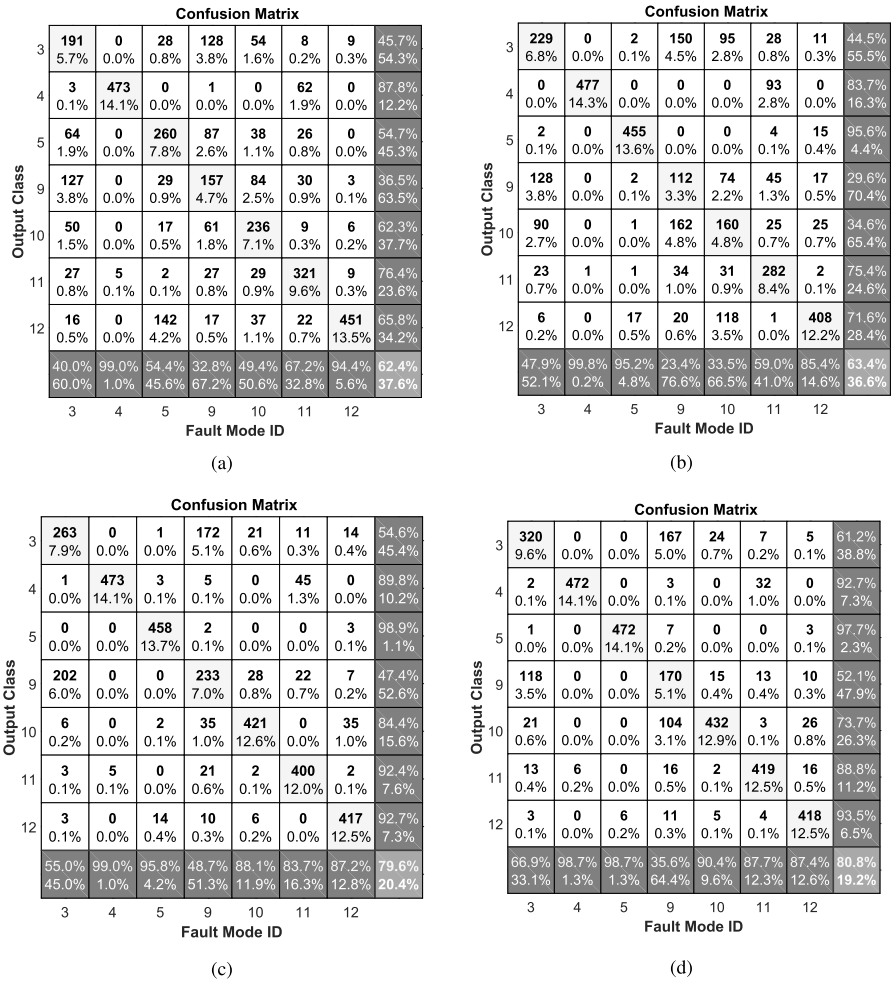


FIGURE 11. Confusion matrices of different methods in Case 2. (a) Confusion matrix of DPCA + SVM in Case 2. (b) Confusion matrix of DLDA + SVM in Case 2. (c) Confusion matrix of MLP in Case 2. (d) Confusion matrix of BN-based LSTM in Case 2.

mon problem. Covariate shift may degrades the efficiency of training and makes LSTM unable to learn effectively. Batch normalization solves the covariate shift problem by standardizing the activations going into each layer, enforcing their means and variances to be invariant to changes in parameters of other layers. Batch normalization is a recent proposed technique to tackle the covariate shift problem and make LSTM more stable and easier to converge.

In the following experiments, we train conventional LSTM and BN-based LSTM respectively. In the training, the min-batch size is set to 30 and 60 epochs of training (total $480/30 \times 60$ training iterations) are show in Figure 7. Experimental results in Figure 7 show that BN-based LSTM converges significantly faster to a baseline LSTM in all three experiments. It is easy to find that compared with conventional LSTM, the more fault modes we used in training, the faster the convergence rate of BN-based LSTM is.

Experiments are run on a computer with Inter Core i7-6700 CPU, 16GB memory and NVIDIA GeForce GTX 1070 GPU. Programming language is Python 3.5 with deep learning package ''PyTorch''. Thanks for GPU-accelerated computing, the total training time of NCA is 12.18 seconds.

C. FAULT CLASSIFICATION AND ANALYSIS

Figure 8(a) and Figure 8(b) shows the visualization results of the training samples of Fault 1, 2 and 8 in Case 1 on the first two dimensions of DPCA and DLDA, respectively (Samples of Fault 6 and Fault 7 in Case 1 are not plotted since they have no overlapping with any other faults.). Figure 8(c) and Figure 8(d) shows the outputs of the hidden layers of MLP and BN-based LSTM, respectively. In this case, the number of nodes in the hidden layer is set to 2, i.e. $d_h = 2$ for both MLP and BN-based LSTM. It is easy to find that the overlapping of samples of different classes are much less in BN-based LSTM than those by DPCA, DLDA, and MLP. Figure 8 indicates that the discriminant power in BN-based LSTM is larger than that in DPCA, DLDA, and MLP. The visualization intuitively shows that BN-based LSTM should be more suitable and effective for fault diagnosis.

The diagnosis results of DPCA + SVM, DLDA + SVM, MLP and BN-based LSTM on the testing data of different cases are shown in Figure 9. For DPCA, we provide the performance under different reduced dimensions from 2 to 30. We also provide the performance of MLP and BN-based LSTM under different numbers of nodes in the hidden layer.

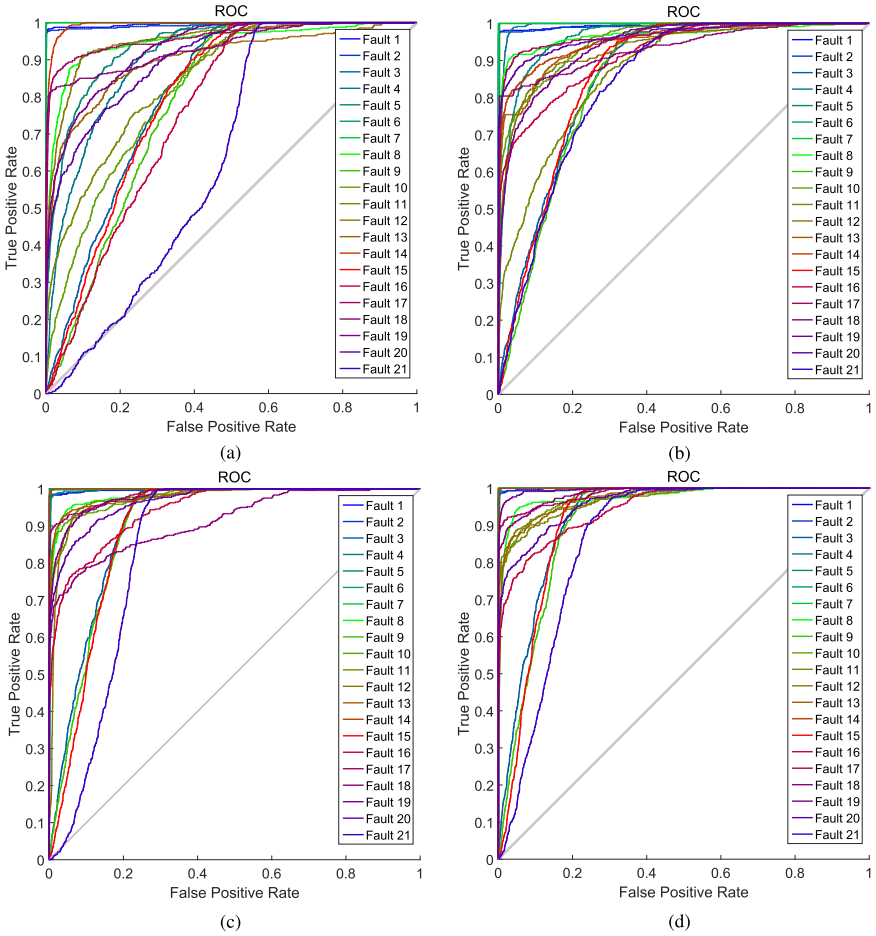


FIGURE 12. ROC plots of different methods on all 21 faults. (a) ROC plot of DPCA + SVM on 21 faults. (b) ROC plot of DLDA + SVM on 21 faults. (c) ROC plot of MLP on 21 faults. (d) ROC plot of BN-based LSTM on 21 faults.

In Figure 9, the dimension of MLP and BN-based LSTM is the number of nodes in the hidden layer. Note that since the reduced dimension of DLDA can not exceed $C-1$ (C is the number of fault modes.), the diagnosis results of DLDA + SVM are largely affected by the dimension of the reduced features. The experimental results show that the proposed BN-based LSTM method can provide the best performance of all the methods in all three conditions.

In order to give more detailed analysis of the four tested methods, we also provides the confusion matrices of DPCA + SVM, DLDA + SVM, MLP, and BN-based LSTM in Figure 10 and Figure 11 for Case 1 and Case 2. For DPCA, the reduced dimension is 30, while for DLDA, the reduced dimension is $C-1$ (C is the number of fault modes. For example, $C=5$ in Case 1.). For MLP and BN-based LSTM, the confusion matrices is obtained under the neural networks with 30 hidden nodes, respectively. Confusion matrix [34] takes target and output data into consideration. The target data are ground truth labels. The output data are the outputs from the tested method that performs classification. In the confusion matrix, the rows show the predicted class, and the columns show the ground truth. The diagonal cells show where the true class and predicted class match. The off diagonal cells show instances where the tested algorithm has made mistakes. The column on the right hand side of the confusion matrix shows the accuracy for each predicted class, while the row at the bottom of the confusion matrix shows the accuracy for each true class. The cell in the bottom right of the confusion matrix shows the overall accuracy.

From the bottom rows of the confusion matrices in Figure 10 and Figure 11 we can obtain the performances of different algorithms on different fault modes and also obtain the overall accuracies from the bottom right cell of the confusion matrices. It is easy to find BN-based LSTM achieves the best overall accuracies both in Case 1 and Case 2. Moreover, we can find that the classification accuracies are largely varied in different fault modes. In Case 2, DPCA + SVM cannot effectively classify Fault 5 and the classification accuracy is 54.4%. However, DLDA + SVM, MLP, and BN-based LSTM can identify this fault with much higher recognition rates of 95.2%, 95.8%, and 98.7%, respectively.

Figure 11 also shows both Fault 3 and Fault 9 in Case 2 are hard to diagnosis for all four algorithms. DPCA + SVM and DLDA + SVM can not classify these two faults, the accuracies are all below 50%. MLP has 202 samples of Fault 3

misclassified to Fault 9 and 172 samples of Fault 9 misclassified to Fault 3. While BN-based LSTM has 118 samples of Fault 3 misclassified to Fault 9 and 167 samples of Fault 9 misclassified to Fault 3. According to Table 1, both Fault 3 and Fault 9 are related to D feed temperature (Stream 2). The only difference is that the type of Fault 3 is step noise while Fault 9 is random variation. It means these two fault modes are easy to mixed up and make the diagnosis hard to perform [16]. Our experimental results also confirm this point of view.

For the experiments on all 21 faults, due to the limitation of space, the receiver operating characteristic (ROC) [34] plots are drawn in Figure 12 instead of providing confusion matrices. ROC is a metric often used to check the quality of a method on each fault mode. For each fault mode, ROC applies different threshold values across the interval [0,1] to outputs. For each threshold, two values are calculated, the True Positive Ratio (TPR) (the number of outputs greater or equal to the threshold, divided by the number of samples in this fault mode), and the False Positive Ratio (FPR) (the number of outputs less than the threshold, divided by the number of samples in other fault modes). ROC curves tend to go from the bottom left corner to the top right corner of the box. The top left corner of the ROC box is the point where TPR $=$ 100% and FPR $=$ 0%. This point can be considered as a perfect fault diagnosis. The closer the ROC curve comes to the top left corner, the better the diagnosis is overall. The closer the curve gets to the centre grey diagonal line, the worse the diagnosis. Figure 12 shows that, although DPCA + SVM, DLDA + SVM, MLP, and BN-based LSTM may have different diagnosis performances on different fault modes, the overall performance of BN-based LSTM are the best one.

We summarize the studies of different cases below:

1) Due to the design and utilization of batch normalization, our proposed BN-based LSTM converges much faster than conventional LSTM for TEP data.
2) Different from DPCA and DLDA, which concatenate prefixed number of samples to incorporate dynamic information, BN-based LSTM can adaptively learn the dynamic information of different variables through different gates in the cells of LSTM. Because of the adaptive learning ability, BN-based LSTM becomes a more powerful tool for fault diagnosis.
3) Although no single algorithm gives optimal performance for all fault modes consisting of diverse numbers of fault conditions. BN-based LSTM outperforms DPCA + SVM, DLDA + SVM and MLP with regard to the best performance and emerges as the clear winner.

V. CONCLUSION

In this paper, we propose BN-based LSTM neural network for fault diagnosis. Unlike the traditional methods which take conventional "feature + classifier" strategy for fault classification, our method is trained in an end-to-end manner which provides a framework to learn the representation of raw input data and classifier simultaneously. Moreover, due to the usage of LSTM, the dynamic information of process data can be adaptively utilized. In order to reduce the internal covariate shift of LSTM and accelerate the convergence of LSTM, batch normalization is designed and used in LSTM neural network to achieve fault diagnosis tasks.

We compare BN-based LSTM with several other algorithms, such as DPCA + SVM, LDA + SVM, and MLP, on TEP. Based on the experiments, it is clear that BN-based LSTM outperforms DPCA + SVM, DLDA + SVM, and MLP for fault diagnosis performance. BN-based LSTM can be considered as an alternative to the prevalent data driven fault diagnosis techniques. In the future work, BN-based LSTM can be extended to deal with batch process monitoring problems.

REFERENCES

[1] J. MacGregor and A. Cinar, "Monitoring, fault diagnosis, fault-tolerant control and optimization: Data driven methods," *Comput. Chem. Eng.*, vol. 47, pp. 111–120, Dec. 2012.
[2] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process," *J. Process Control*, vol. 22, no. 9, pp. 1567–1581, 2012.
[3] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annu. Rev. Control*, vol. 36, no. 2, pp. 220–234, 2012.
[4] Z. Ge, Z. Song, and F. Gao, "Review of recent research on data-based process monitoring," *Ind. Eng. Chem. Res.*, vol. 52, no. 10, pp. 3543–3562, 2013.
[5] T. Feital, U. Kruger, J. Dutra, J. C. Pinto, and E. L. Lima, "Modeling and performance monitoring of multivariate multimodal processes," *AIChE J.*, vol. 59, no. 5, pp. 1557–1569, 2013.
[6] M. Askarian, G. Escudero, M. Graells, R. Zarghami, F. Jalali-Farahani, and N. Mostoufi, "Fault diagnosis of chemical processes with incomplete observations: A comparative study," *Comput. Chem. Eng.*, vol. 84, pp. 104–116, Jan. 2016.
[7] J. Yan *et al.*, "Towards effective prioritizing water pipe replacement and rehabilitation," in *Proc. IJCAI*, 2013, pp. 2931–2937.
[8] Z. Ge, "Review on data-driven modeling and monitoring for plant-wide industrial processes," *Chemometrics Intell. Lab. Syst.*, vol. 171, pp. 16–25, Dec. 2017.
[9] Z. Ge, Z. Song, S. X. Ding, and B. Huang, "Data mining and analytics in the process industry: The role of machine learning," *IEEE Access* , vol. 5, pp. 20590–20616, 2017.
[10] S. J. Qin, "Statistical process monitoring: Basics and beyond," *J. Chemometrics*, vol. 17, no. 8, pp. 480–502, 2003.
[11] X. Deng, X. Tian, and S. Chen, "Modified kernel principal component analysis based on local structure analysis and its application to nonlinear process fault diagnosis," *Chemometrics Intell. Lab. Syst.*, vol. 127, pp. 195–209, Aug. 2013.
[12] X. Gao and J. Hou, "An improved SVM integrated GS-PCA fault diagnosis approach of Tennessee Eastman process," *Neurocomputing*, vol. 174, pp. 906–911, Jan. 2016.
[13] S. Yin, X. Zhu, and O. Kaynak, "Improved PLS focused on key-performance-indicator-related fault diagnosis," *IEEE Trans. Ind. Electron.*, vol. 62, no. 3, pp. 1651–1658, Mar. 2015.
[14] M. A. A. Rad and M. J. Yazdanpanah, "Designing supervised local neural network classifiers based on EM clustering for fault diagnosis of Tennessee Eastman process," *Chemometrics Intell. Lab. Syst.*, vol. 146, pp. 149–157, Aug. 2015.
[15] C. Zhao and F. Gao, "A nested-loop fisher discriminant analysis algorithm," *Chemometrics Intell. Lab. Syst.*, vol. 146, pp. 396–406, Aug. 2015.
[16] L. H. Chiang, R. D. Braatz, and E. L. Russell, *Fault Detection and Diagnosis in Industrial Systems*. London, U.K.: Springer, 2001.
[17] Y. Dong and S. J. Qin, "A novel dynamic PCA algorithm for dynamic data modeling and process monitoring," *J. Process Control*, to be published.

[18] J. Guo, L. Qi, and Y. Li, "Fault detection of batch process using dynamic multi-way orthogonal locality preserving projections," *J. Comput. Inf. Syst.*, vol. 11, no. 2, pp. 577–586, 2015.
[19] G. Rong, S.-Y. Liu, and J.-D. Shao, "Dynamic fault diagnosis using extended matrix and tensor locality preserving discriminant analysis," *Chemometrics Intell. Lab. Syst.*, vol. 116, pp. 41–46, Jul. 2012.
[20] L. H. Chiang, M. E. Kotanchek, and A. K. Kordon, "Fault diagnosis based on fisher discriminant analysis and support vector machines," *Comput. Chem. Eng.*, vol. 28, no. 8, pp. 1389–1401, 2004.
[21] R. Eslamloueyan, "Designing a hierarchical neural network based on fuzzy clustering for fault diagnosis of the Tennessee–Eastman process," *Appl. Soft Comput.*, vol. 11, no. 1, pp. 1407–1415, 2011.
[22] C. K. Lau, K. Ghosh, M. A. Hussain, and C. R. C. Hassan, "Fault diagnosis of Tennessee Eastman process with multi-scale PCA and ANFIS," *Chemometrics Intell. Lab. Syst.*, vol. 120, pp. 1–14, Jan. 2013.
[23] K.-I. Funahashi and Y. Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural Netw.*, vol. 6, no. 6, pp. 801–806, 1993.
[24] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[26] S. Xiao, J. Yan, X. Yang, H. Zha, and S. M. Chu, "Modeling the intensity function of point process via recurrent neural networks," in *Proc. AAAI*, 2017, pp. 1597–1603.
[27] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[29] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 437–478.
[30] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[31] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[32] P. R. Lyman and C. Georgakis, "Plant-wide control of the Tennessee Eastman problem," *Comput. Chem. Eng.*, vol. 19, no. 3, pp. 321–331, 1995.
[33] W. Hardle and L. Simar, *Applied Multivariate Statistical Analysis*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2007.
[34] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2012.

HAITAO ZHAO received the M.S. degree in applied mathematics and the Ph.D. degree in pattern recognition and artificial intelligence from the Nanjing University of Science and Technology, Nanjing, China, in 2000 and 2003, respectively. He is currently a Full Professor with the School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China. His major research interests include machine learning and pattern recognition.

SHAOYUAN SUN received the M.S. and Ph.D. degrees in optical engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1999 and 2002, respectively. She is currently a Professor with the College of Information Sciences and Technology, Donghua University, Shanghai, China. Her major research interests include multi-source information acquisition, processing, and fusion.

BO JIN received the B.S. degree in electronic engineering and the Ph.D. degree in control theory and control engineering from Shanghai Jiao Tong University, China, in 2004 and 2014, respectively. He is currently a Lecturer with the School of Computer Science and Software Engineering, East China Normal University. His major research interests include machine learning, online learning, face recognition, visual tracking, and medical diagnosis modeling.