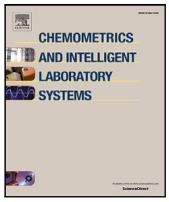
Chemometrics and Intelligent Laboratory Systems 240 (2023) 104881

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemometrics



Simultaneous fault detection and isolation based on multi-task long short-term memory neural networks

Ken Sinkou Qin [a], Yegang Du [b,*]

[a] Information Science and Technology College, Dalian Maritime University, Dalian 116026, China
[b] Future Robotics Organization, Waseda University, Tokyo 162-0041, Japan

ARTICLE INFO

Keywords:
Fault detection and isolation
Sparse autoencoder
Long short-term memory
Penicillin fermentation process

ABSTRACT

Fault detection and isolation are two significant problems in process monitoring. However, due to the complicated relationships between the process variables, it becomes a challenging problem to build a model to capture the complicated relationship between process variables and perform fault detection and isolation based on the model. This paper proposes an advanced model, called multi-task long short-term memory (MTLSTM) neural networks, that can capture the most complicated relationships between process variables and a simultaneous fault detection and isolation approach based on MTLSTM neural networks. The proposed simultaneous fault detection and isolation approach takes a sparse autoencoder (SAE) to extract features from the measurements of process variables. Then, a long short-term memory neural network is trained on the extracted features with a multi-task learning strategy. Compared with the traditional multivariate statistical models (MVSM) such as principal component analysis and independent component analysis models, the MTLSTM neural networks can capture the nonlinear autocorrelations of process variables and correlations between process variables and perform fault detection and isolation for an industrial process simultaneously. To demonstrate the advantages and superiority of the proposed simultaneous fault detection and isolation approach, a case study on fault detection and isolation for the penicillin fermentation process is carried out. Case study results show that the proposed approach significantly outperforms existing fault detection and isolation approaches.

1. Introduction

Modern large-scale and intelligent industrial processes have greatly improved production efficiency and economic benefits. For the sake of the safe operation of industrial plants, process monitoring technique has drawn great attention from engineers. Process monitoring technique is beneficial for troubleshooting and industrial plant maintenance, which have been the essential components of industrial monitoring systems. Fault detection and isolation are two important problems in process monitoring. Fault detection and isolation are beneficial for avoiding disasters caused by industrial plant faults. Fault detection and isolation methods can be grouped into model-based or model-free approaches. Model-based approaches rely on the use of a physical model of the process. While model-free approaches do not use the model of processes but only depend on the use of the measurements of process variables [1]. Fault detection and isolation are to detect whether there are faults in the production process and recognize the category of the detected faults. Common faults can be categorized into the following three types: sensor, actuator, and equipment faults. Fault detection and isolation usually rely on the use of monitoring statistics. Specifically, if a monitoring statistic exceeds its control limit, then a fault has been detected. On the contrary, there is no fault occurred [2].

For complex industrial processes, establishing accurate physical models is an almost impossible task due to the complex coupling relationship between thousands of process variables, which limits the application of model-based process monitoring methods in real-world industrial production processes. Specifically, with the wide application of discrete control systems and computer-related technologies in industrial processes, a large number of measurements of process variables can be stored in databases. Then, it is feasible to take advantage of an effective data analysis algorithm for extracting valuable information from the measurements, and use the information for process monitoring [3]. Using measurements of process variables to monitor industrial processes is called data-driven process monitoring methods. Compared with model-based process monitoring methods, data-driven process monitoring methods are easy to design and not constrained by the physical process model [4]. In other words, data-driven process

* Corresponding author.
E-mail address: yg.du@aoni.waseda.jp (Y. Du).

https://doi.org/10.1016/j.chemolab.2023.104881
Received 6 April 2023; Received in revised form 21 May 2023; Accepted 5 June 2023
Available online 12 June 2023

monitoring approaches do not rely on the use of the physical model of industrial processes, nor does it require prior knowledge about industrial processes. Therefore, data-driven process monitoring approaches have been the hottest research topic over the past decades. For example, Westerhuis et al. studied the contribution plots for multivariate statistical process control of batch processes [5]. Sun [6] studied the fault diagnosis problem with the partial least square algorithm, which can monitor the faults of quality variables. The above research results provide theoretical support for the development of process monitoring in modern industrial processes. Ku et al. [7] proposed a dynamic principal component analysis model for dynamic process monitoring. The method first stacks the observation data according to a certain time window to build an augmented observation data matrix, then use the augmented matrix to build a principal component analysis model. However, this model cannot explicitly explain the dynamics of latent variables in time series, and it is easy to cause dimensional disasters as the time window increases. Qin and Hwang reviewed the fault detection, isolation, and reconfiguration methods in the data-driven and model-based frameworks, respectively [8,9]. Li et al. [10] proposed a process monitoring method based on contribution plots combined with canonical correlation analysis and short-term memory neural networks. Jia et al. [11] proposed a dynamic kernel partial least squares algorithm for quality-related process monitoring in nonlinear dynamic processes. However, these works only capture the correlation among process variables, while ignoring the autocorrelation of process variables. The dynamic latent variable model consists of a observation equation and a latent variable autoregressive equation, which describes the dynamic properties of the process. Currently, dynamic latent variable models have been widely used to solve process monitoring problems. For example, Li et al. [12] improved the dynamic principal component analysis model and proposed a structured dynamic principal component analysis model, overcoming the shortcomings of traditional dynamic principal component analysis models.

In addition to the traditional multivariate statistical models, deep neural networks have been applied to process monitoring. For example, Song et al. applied deep neural networks to nonlinear dynamic process monitoring [13]. Yang et al. applied stacked autoencoder to extract nonlinear features and the manifold structure-related information for fault detection. The effectiveness of the proposed fault detection method was verified in the Tennessee Eastman process [14]. Luo et al. proposed a deep neural network with tensor factorization layers for sequential fault diagnosis where the proposed neural network shares efficient knowledge across the spatiotemporal features of fault data. Moreover, since the industrial data exhibit time dependency and inherent complex characteristics, the authors take advantages of tensor representation to preserve the number of the raw data and sequential dependence between observations. Then, multilinear mapping with tensor-to-tensor projection is used to transform the input and hidden tensor to the low-dimensional tensors [15]. Yu et al. proposed a manifold regularized stacked autoencoders for feature extraction and fault detection. The proposed autoencoders can extract the local and global information and intrinsic structure of process data. With the effective features, then the authors perform fault detection based on two statistics T-squared and squared prediction error. The effectiveness of the proposed fault detection method was evaluated on two benchmark processes, Tennessee Eastman process and fed-batch fermentation penicillin process [16]. Compared with multivariate statistical models, deep neural networks show advantages in feature extraction, which is beneficial for fault detection and isolation. Moreover, deep neural networks can be trained by an end-to-end manner based on the open-source machine learning platforms.

Although the existing process monitoring approaches obtain excellent achievements, they all fail to capture the nonlinear autocorrelations of process variables and correlation among process variables, simultaneously. Moreover, the order of autocorrelation is difficult to specified by experience. This paper takes advantage of a new deep neural network to learn the nonlinear autocorrelation and correlation from data. In MTLSTM, the order of autocorrelation is learned instead of specified by experience. Moreover, the proposed simultaneous fault detection and isolation approach is feasible if only if the measurements of process variables are available. Moreover, the network of the MTLST can be trained by an end-to-end manner using the open-source machine learning platform such as TensorFlow designed by Google. In conclusion, the measurements of process variables and machine learning platform allows us to perform simultaneous fault detection and isolation based on the proposed method. As a result, the simultaneous fault detection and isolation approach is feasible from the viewpoint of implementation.

The main contributions and advantages of the MTLSTM-based simultaneous fault detection and isolation approach are summarized as follows: (1) MTLSTM can extract the nonlinear autocorrelations of process variables and correlation among process variables, which is beneficial for fault detection and isolation. (2) MTLSTM can significantly improve the efficiency of process monitoring by simultaneously perform fault detection and isolation. (3) MTLSTM shows the best performance in the fault detection and isolation for the Penicillin Fermentation Process.

The remainder of this article is organized as follows. Section 2 introduces the background knowledge about the properties of process variables. In Section 3, we introduce the new multi-task long short-term memory networks. In Section 4, the multi-task long short-term memory-based process monitoring approach is proposed. In Section 5, a case study on fault detection and isolation for the penicillin fermentation process is carried out to evaluate the performance of the proposed process monitoring approach. At last, conclusions are drawn.

2. Background knowledge

2.1. Properties of process variables

In this section, we will take the penicillin fermentation process as an example to demonstrate the inherent properties of process variables. The penicillin fermentation process is a widely used benchmark for performance evaluation of process monitoring approaches [17,18]. We selected two process variables, aeration rate and agitator power, from the total 16 process variables to illustrate the properties of process variables. Three properties including nonlinearity, autocorrelation, and correlation are explored. First, the autocorrelation property of a process variable is measured by the autocorrelation and partial autocorrelation coefficients. The coefficients of aeration rate are depicted in Fig. 1 with confidence interval 95% and lag value 40. Fig. 1 indicates that the aeration rate shows significant autocorrelation i.e., the current measurement of it is highly related to its past measurements.

Correlation between aeration rate and agitator power are depicted in Fig. 2. In this figure, the histograms show the distributions of the values of aeration rate and agitator power. The other two subfigures show the correlation between the aeration rate and agitator power. There is an obvious nonlinear relation between aeration rate and agitator power where the straight line in the figures represents a linear model that attempts to fit the correlation between aeration and agitator power.

In this example, it is demonstrated that each process variable has the autocorrelation property. Moreover, there is a significant nonlinear correlation between process variables. Therefore, if we adopt a linear model to fit the nonlinear autocorrelation of process variables and the correlation between process variables, then it will cause a large modeling error. Further, the process monitoring performance based on the model will deteriorate as well.

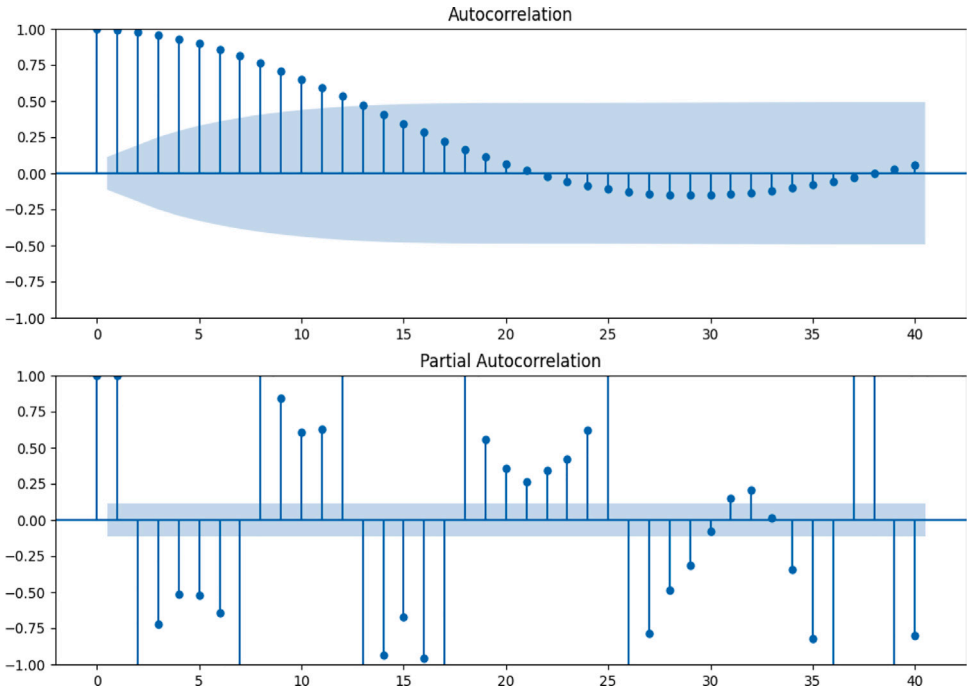


Fig. 1. Autocorrelation and partial autocorrelation of aeration rate.

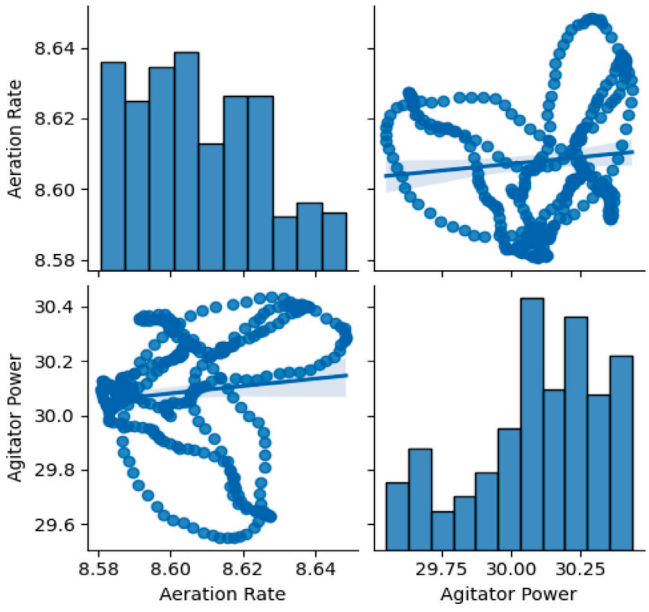


Fig. 2. Correlation among the aeration rate and agitator power.

2.2. Multivariate statistical models

The MVSM is used to describe the properties of process variable, which can generally be described by

$$\begin{aligned}\mathbf{v}_k &= g\left(\mathbf{x}_k\right)\\ \hat{\mathbf{x}}_k &= f\left(\mathbf{v}_k\right)+\mathbf{w}_k\end{aligned} \tag{1}$$

where $f(\cdot)$ and $g(\cdot)$ are two nonlinear functions, $\mathbf{x}_k \in \Re^m$ is the vector of process variable measurements at kth sampling moment, $\mathbf{v}_k \in \Re^l$ is the feature vector, $\hat{\mathbf{x}}_k$ is the approximation of $\mathbf{x}_k$, and $\mathbf{w}_k \in \Re^m$ is the error between $\mathbf{x}_k$ and $\hat{\mathbf{x}}_k$ [19].

For example, PCA can be represented by

$$\begin{aligned}\mathbf{v}_k &= \mathbf{P}\mathbf{x}_k\\ \hat{\mathbf{x}}_k &= \mathbf{P}^T\mathbf{v}_k+\mathbf{w}_k\end{aligned} \tag{2}$$

where matrix $\mathbf{P} \in \Re^{m\times l}$ consists of the eigenvectors of the covariance matrix associated to the top largest eigenvalues [20,21].

ICA can be represented by

$$\begin{aligned}\mathbf{x}_k &= \mathbf{A}\mathbf{s}_k\\ \mathbf{s}_k &= \mathbf{W}\mathbf{x}_k\end{aligned} \tag{3}$$

where $\mathbf{A}$ represents the mixing matrix, and $\mathbf{W}$ represents the de-mixing matrix [22,23].

Obviously, PCA and ICA are both linear approximations of the model (1). Moreover, they do not capture the autocorrelation of each process variable but only capture the correlation among process variables. Specifically, the current measurement is only related to the current feature, i.e. time independent. To overcome the drawbacks of PCA and ICA that focus on the correlations of process variables, vector autoregression (VAR) was developed for capturing the autocorrelations of process variables and correlations between process variables [24,25]. The model of VAR can be represented by

$$\mathbf{x}_k = \mathbf{A}_1\mathbf{x}_{k-1}+\cdots+\mathbf{A}_s\mathbf{x}_{k-s}+\mathbf{w}_k \tag{4}$$

By introducing an intermediate variable $\mathbf{v}=\mathbf{x}$, then the VAR model can be rewritten as

$$\begin{aligned}\mathbf{v}_{k-1} &= \mathbf{x}_{k-1}\\ \mathbf{x}_k &= \mathbf{A}_1\mathbf{v}_{k-1}+\cdots+\mathbf{A}_s\mathbf{v}_{k-s}+\mathbf{w}_k\end{aligned} \tag{5}$$

where $\mathbf{w}$ is a residual error that follows a zero-mean normal distribution, $\mathbf{A}_j \in \Re^{m\times m}$ represents a coefficient matrix for $j=1,\ldots,s$. Compared with PCA and ICA, VAR can capture the autocorrelation and correlation properties of process variables. However, VAR is also a linear model that cannot fit the complicated nonlinear properties of process variables.

2.3. Challenging problems in nonlinear process monitoring

As discussed above, the existing models cannot capture the complicated nonlinear autocorrelations of process variables and correlation

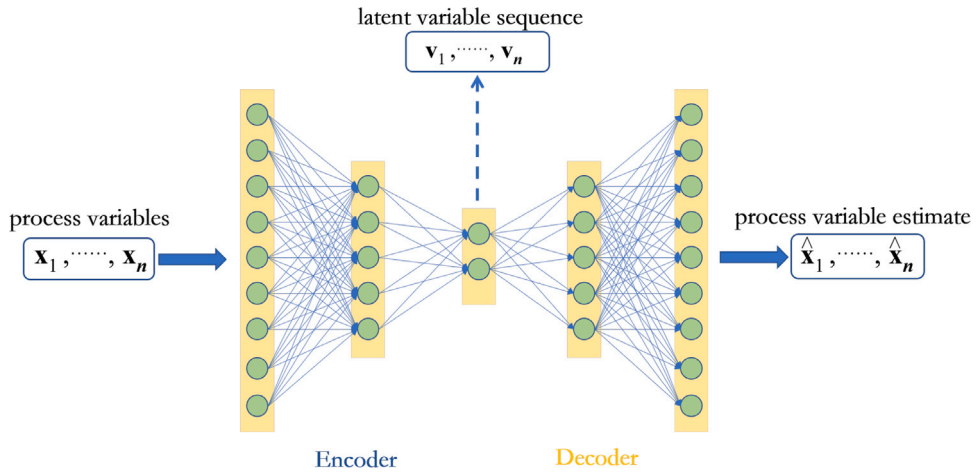


Fig. 3. Structure of the SAE.

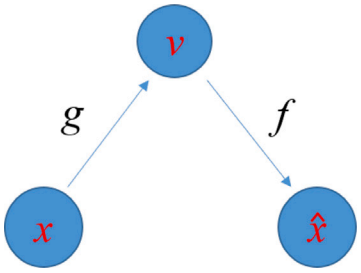


Fig. 4. An illustration of the SAE where **x** represents the input of the SAE, **v** represents a feature vector, $\hat{\mathbf{x}}$ represents the approximation of the input, f and g represents two nonlinear functions.

among process variables. As a result, the process monitoring approach based on these models will fail to monitor the industrial processes accurately. We summarize the challenging problems in process monitoring as follows: (1) For weakly nonlinear processes, it is feasible to use a linear model to approximate the nonlinear model. However, for strongly nonlinear processes, using linear models to approximate nonlinear models can introduce significant model errors; (2) It is difficult to simultaneously capture the autocorrelations of process variables and correlation among process variables by a data-driven model; (3) The existing MVSM focus on fault detection but ignore fault isolation.

2.4. Comparison between MVSM and MTLSTM

MVSM has been widely used for process monitoring over the past decades. However, they only perform well in linear static process monitoring but fail to monitor the nonlinear dynamic processes. The reasons are that MVSM only captures the correlation between process variables but ignores the nonlinear autocorrelation of process variables and correlation simultaneously. Even the nonlinear versions of MVSM improve the nonlinear modeling ability, though they still ignore the autocorrelation of process variables. Moreover, MVSM applies to fault detection or fault isolation but fails to perform fault detection and isolation simultaneously by just a model.

To improve the performance of MVSM in process monitoring, this paper proposed a new model to capture the complicated properties of process variables. Then, a high-performance fault detection and isolation approach is proposed based on the new model. The new model is called multi-task long short-term memory (MTLSTM) neural networks. MTLSTM show a lot of advantages in fault detection and isolation. We summarize the features of the traditional MVSM and MTLSTM in Table 1 for comparison. Moreover, the functionalities of them are summarized in Table 2.

Table 1
Feature comparison between MVSM and MTLSTM.

Models	Nonlinearity	Autocorrelation	Correlation
PCA	✗	✗	✓
ICA	✗	✗	✓
VAR	✗	✓	✓
RF	✓	✗	✓
FCN	✓	✗	✓
MTLSTM	✓	✓	✓

Table 2
Functionality comparison between MVSM and MTLSTM.

Models	Fault detection	Fault isolation
PCA	✓	✗
ICA	✓	✗
VAR	✓	✗
RF	✗	✓
FCN	✗	✓
MTLSTM	✓	✓

In Table 1, the token tick in each cell represents the feature that a model possesses. On the contrary, the token cross represents the feature that a model does not possess. The table indicates that the traditional MVSM are capable of fault detection and capturing the correlation between process variables but ignore the nonlinearity and autocorrelation. However, the MTLSTM can capture all of the properties of process variables and perform fault detection and isolation simultaneously. Next, we will introduce the MTLSTM and the simultaneous fault detection and isolation approach based on it.

3. The MTLSTM neural networks

In this section, we will introduce the new multi-task long short-term memory neural networks. MTLSTM consist of two components including a SAE for feature extraction and a LSTM neural network for learning the patterns of different types of sample. Structure of the SAE is depicted in Fig. 3. To allow the SAE to extract compact features from the measurements of process variables, the features are required to be sparse. Fig. 4 is an illustration of a one-layer LSTM expanded at different moments. LSTM is a type of recurrent neural networks (RNN) for time series modeling, which can capture the autocorrelation and correlation properties of process variables. Compared with the strand RNN, LSTM takes advantage of memory cells to store historical information so that it can capture the autocorrelation of process variables.

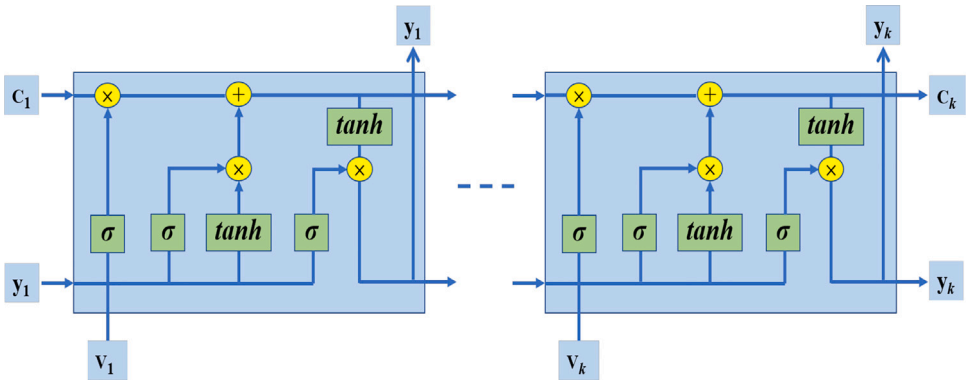


Fig. 5. An illustration of one-layer LSTM expanded at different moments.

3.1. The sparse autoencoder

The SAE in MTLSTM is depicted in Fig. 3 where the input of a SAE is a vector $\mathbf{x}$, and the output of it is the approximation of $\mathbf{x}$ [26–30]. The SAE consists of two modules including an encoder and a decoder where the encoder is responsible for feature extraction by downsampling, and the decoder is responsible for reconstructing the input. Moreover, SAE requires the features to be sparse so that it can extract compact features from the inputs.

In fact, constructing a SAE is to learn two maps from the input $\mathbf{x}$ to feature $\mathbf{v}$ and from feature $\mathbf{v}$ to $\hat{\mathbf{x}}$, respectively where $\hat{\mathbf{x}}$ is the approximation of $\mathbf{x}$. In Fig. 4, $g(\cdot)$ and $f(\cdot)$ represent two nonlinear functions. According to Fig. 4, a SAE can be formulated by

$$\begin{aligned}\mathbf{v}_k &= g\left(\mathbf{x}_k\right)\\ \hat{\mathbf{x}} &= f\left(\mathbf{v}_k\right)\end{aligned} \tag{6}$$

where $g(\cdot)$ and $f(\cdot)$ represent two nonlinear functions associated to the encoder and decoder, respectively, and $\mathbf{v}$ is the feature vector extracted from the input $\mathbf{x}$.

3.2. The long short-term memory neural network

LSTM is a type of recurrent neural networks (RNN), which is powerful for time sequence modeling [31–33]. Compared with standard RNN, LSTM is specially designed to improve the memory ability of the historical information. To this end, LSTM adopts a memory cell to store the historical information. In LSTM, at each moment, the input will first pass through the input gate, and the input gate will determine whether any information will be input to the memory cell at this moment. Whether there is information output from the memory cell at each moment depends on the output gate. The value in the memory cell is controlled by the forget gate. Specifically, the signals controlling each gate are given by the following formulas. The structure of a LSTM is depicted in Fig. 5.

The output of LSTM at kth moment can be formulated by

$$\mathbf{o}_k = f_{lstm}\left(\mathbf{v}_k, \ldots, \mathbf{v}_1\right) \tag{7}$$

where f_{lstm} represents the nonlinear map from the input of the LSTM to the output, and $\mathbf{v}_k = g\left(\mathbf{x}_k\right)$.

3.3. The multi-task long short-term memory neural networks

The MTLSTM neural networks consists of the encoder of the SAE, a LSTM, and two fully connected networks, as shown in Fig. 6. The input of MTLSTM is a vector $\mathbf{x}$, and the outputs of it are $\hat{\mathbf{x}}$ and $\mathbf{y}$ where $\hat{\mathbf{x}}$ and $\mathbf{y}$ represent the approximation and one-hot label vector of $\mathbf{x}$, respectively. The encoder is responsible for extracting features from inputs so that the LSTM can capture the properties of process variable more efficiently and accurately. The two fully connected networks are responsible for outputting the results of different tasks.

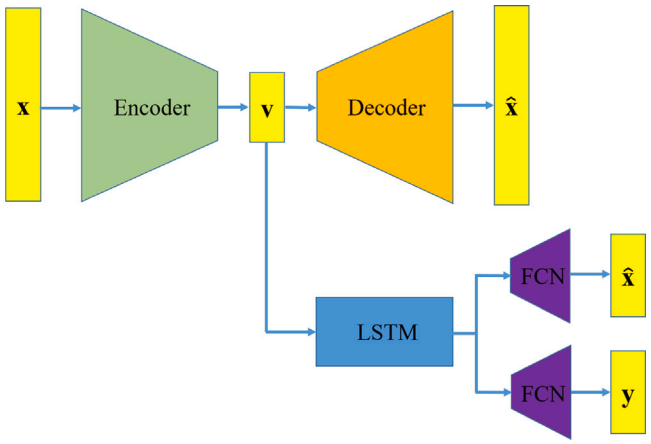


Fig. 6. The illustration of MTLSTM models.

There are a lot of advantages for MTLSTM in process modeling because they have powerful nonlinear expression ability to capture the complicated properties of process variables. Moreover, the LSTM module allows it to capture the time dependency of the measurements of process variables. In conclusion, MTLSTM are capable of constructing the data-driven model of complicated processes.

4. Simultaneous fault detection and isolation approach

To perform fault detection ad isolation based on MTLSTM simultaneously, it is required to train a MTLSTM on a set of training samples. For training the MTLSTM, we first construct a set of training samples. For example, $\mathbf{X} = \left[\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3\right] \in \Re^{n\times m}$ represents the set of training samples where $\mathbf{X}_0 \in \Re^{n_0\times m}$ consists of fault-free samples, $\mathbf{X}_j \in \Re^{n_j\times m}$ for $j = 1, 2, 3$ consists of jth type of fault samples, and n_j for $j = 0, 1, 2, 3$ represents the number of fault-free samples, fault-1 samples, fault-2 samples, and fault-3 samples, respectively. Moreover, $\mathbf{Y} = \left[\mathbf{Y}_0, \mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3\right] \in \Re^{n\times 4}$ represents the set of one-hot label vectors of the training samples where $\mathbf{Y}_j \in \Re^{n_j\times 4}$ consists of labels associated to $\mathbf{X}_j$ for $j = 0, 1, 2, 3$. Specifically, the label vectors of fault-free, fault-1, fault-2, and fault-3 samples are defined by $\mathbf{y}^0 = [0, 0, 0, 0]$, $\mathbf{y}^1 = [0, 1, 0, 0]$, $\mathbf{y}^2 = [0, 0, 1, 0]$, $\mathbf{y}^3 = [0, 0, 0, 1]$, respectively.

We first discuss the sparsity of features extracted by the SAE. From the point view of probability, we have the following relation

$$log\,P(\mathbf{x}, \mathbf{v}) = log\,P(\mathbf{v}) + log\,P(\mathbf{x}|\mathbf{v}) \tag{8}$$

To guarantee the sparsity of feature vector $\mathbf{v}$, we suppose that $\mathbf{v}$ follows the Laplace prior as follows

$$P\left(\mathbf{v}\right) = \frac{\lambda}{2} e^{-\lambda|\mathbf{v}|} \tag{9}$$

As a result, we have

$$-logP(\mathbf{v}) = \sum_j \left(\lambda|\mathbf{v}[j]| - log\frac{\lambda}{2}\right) + constant \tag{10}$$

where $[j]$ represents jth element of a vector.

In summary, the loss function for training a SAE can be constructed as

$$loss_{SAE} = \frac{1}{n}\sum_j^n \left\|\mathbf{x}_j - \hat{\mathbf{x}}_j\right\|_F^2 + \lambda\left|\mathbf{v}_j\right|_1 \tag{11}$$

where $\|\cdot\|_F$ represents the Frobenius norm of a vector, and $|\cdot|_1$ is the l_1-norm of a vector. To minimize the loss function (11), we are allowed to train a SAE on a set of training samples. Then, given a vector $\mathbf{x}$, we can get a feature vector $\mathbf{v}$ extracted by the encoder of the well-trained SAE.

Taking the loss function for SAE training into account, the loss function for training a MTLSTM can be defined by

$$\begin{aligned} loss = &\frac{1}{3n}\sum_{j=1}^n \left\|\mathbf{y}_j - \mathbf{y}^1\right\|_1 \left\|\mathbf{y}_j - \mathbf{y}^2\right\|_1 \left\|\mathbf{y}_j - \mathbf{y}^3\right\|_1 \left\|\mathbf{x}_j - \hat{\mathbf{x}}_j\right\|_F^2 \\ &+ \sum_{j=1}^n\sum_{i=1}^4 \mathbf{y}_j[i]log\left(\frac{\mathbf{y}_j[i]}{\hat{\mathbf{y}}_j[i]}\right) - \mathbf{y}_j[i]log\left(\mathbf{y}_j[i]\right) + \lambda\left|\mathbf{v}_j\right|_1 \end{aligned} \tag{12}$$

where $\hat{\mathbf{x}}$ represents the approximations of $\mathbf{x}$ from SAE and MTLSTM, $\mathbf{y}_j$ is the one-hot label of $\mathbf{x}_j$ for $j = 1, \ldots, n$. The loss function consists of two parts, i.e., a mean squared error (MSE) and a cross entropy (CE) where the MSE is responsible for reconstructing the input, and the CE is responsible for predicting the label of the input. Then, with the loss function, a MTLSTM is allowed to train on the sample set $(\mathbf{X}, \mathbf{Y})$.

In fact, during the training, if $\mathbf{x}_j$ is a fault-free sample, then the loss function becomes

$$loss = \frac{1}{n}\sum_{j=1}^n \|\mathbf{x}_j - \hat{\mathbf{x}}_j\|_F^2 + \lambda\left|\mathbf{v}_j\right|_1 \tag{13}$$

In this case, the MTLSTM focuses on the reconstruction of inputs. On the contrary, if $\mathbf{x}_j$ is a fault sample, then the loss function becomes

$$loss = \sum_{j=1}^n\sum_{i=1}^4 \mathbf{y}_j[i]log\left(\frac{\mathbf{y}_j[i]}{\hat{\mathbf{y}}_j[i]}\right) - \mathbf{y}_j[i]log\left(\mathbf{y}_j[i]\right) \tag{14}$$

In this case, the MTLSTM focuses on predicting the labels of inputs.

As a result, the loss function allows MTLSTM to reconstruct the input and predict the label of the input, simultaneously. With the loss function and training samples, then we are allowed to train a MTLSTM using the back propagation algorithm. In this paper, the MTLSTM was trained on the TensorFlow machine learning platform. After the MTLSTM gets well trained on the training data set $(\mathbf{X}, \mathbf{Y})$, then we are allowed to perform fault detection and isolation. Specifically, given a vector $\mathbf{x}_k$ of the measurements of process variables at moment k, then the output of MTLSTM with respect to $\mathbf{x}_k$ is

$$[\hat{\mathbf{x}}_k, \hat{\mathbf{y}}_k] = f_{MTLSTM}\left(\mathbf{x}_k, \ldots, \mathbf{x}_1\right) \tag{15}$$

where $\hat{\mathbf{x}}_k$ is the estimation of $\mathbf{x}_k$, $\hat{\mathbf{y}}_k$ is the predicted label of $\mathbf{y}_k$, and $f_{MTLSTM}(\cdot)$ represents the nonlinear mapping from the input to the output of MTLSTM. With the well-trained MTLSTM, we are allowed to perform fault detection and isolation, simultaneously.

Specifically, we define the reconstruction error between $\mathbf{x}_k$ and $\hat{\mathbf{x}}_k$ as follows

$$error_k = \frac{1}{m}\sum_j^m |\mathbf{x}_k[j] - \hat{\mathbf{x}}_k[j]| \tag{16}$$

where $[j]$ represents jth element of a vector, and $|\cdot|$ represents the absolute value operator. The principle for process monitoring is based on the rule that if $error_k$ exceeds its threshold then $\mathbf{x}_k$ is a faulty sample. Otherwise, it is a fault-free sample. The threshold for process monitoring can be selected as the maximum reconstruction error among the training samples.

Table 3
Procedures for simultaneous fault detection and isolation.

Off-line training	Procedures
1	Collect training data $\mathbf{x}_1, \ldots, \mathbf{x}_n$
2	Construct the model of MTLSTM
3	for i in range(Epoch):
4	for j in range(n):
5	$[\hat{\mathbf{x}}_j, \hat{\mathbf{y}}_j] = f_{MTLSTM}\left(\mathbf{x}_j\right)$
6	Calculate the loss function (12)
7	Update the parameters of MTLSTM
8	Save the well-trained model of MTLSTM
On-line monitoring	**Procedures**
1	Load the well-trained model of MTLSTM
2	Collect a sample $\mathbf{x}_{new} \in \Re^m$
3	$[\hat{\mathbf{x}}_{new}, \hat{\mathbf{y}}_{new}] = f_{MTLSTM}\left(\mathbf{x}_{new}\right)$
4	$error_{new} = \frac{1}{m}\sum_j^m |\mathbf{x}_{new}[j] - \hat{\mathbf{x}}_{new}[j]|$
5	$ind = argmax\left(\hat{\mathbf{y}}_{new}\right)$
6	if $error_{new} \leq$ threshold:
7	no faults happen
8	else:
9	$\mathbf{x}_{new}$ is a faulty sample
10	ind indicates the category of $\mathbf{x}_{new}$

If $\mathbf{x}_k$ is a fault sample, then we perform fault isolation based on $\hat{\mathbf{y}}_k$ where fault isolation aims to recognize the type of the fault sample $\mathbf{x}_k$. This can be achieved by the following formula

$$ind = argmax\left(\hat{\mathbf{y}}_k\right) \tag{17}$$

where $argmax(\cdot)$ is the function of a vector that returns the index of the maximum value of the vector, and $ind \in \{1, 2, 3\}$. The principle for fault isolation is to determine the types of $\mathbf{x}_k$ based on ind. For example, if $ind = 1$, then the current fault belongs to the first type of faults.

The proposed simultaneous fault detection and isolation approach is feasible only if the measurements of process variables are available. Moreover, the network of the MTLSTM can be trained in an end-to-end manner using the open-source machine learning platform such as TensorFlow designed by Google. As a result, the measurements of process variables and the machine learning platform allow us to perform simultaneous fault detection and isolation based on the proposed method. In other words, the simultaneous fault detection and isolation approach is feasible from the viewpoint of implementation. The implementation details of the proposed simultaneous fault detection and isolation approach are summarized in Table 3.

5. A case study on penicillin fermentation process monitoring

The existing systems for fault detection and isolation research include the penicillin fermentation process, the Tennessee Eastman process, wastewater treatment process, and continuous stirred tank reactor [1,2,17,18]. All of them have been widely used for evaluating the performance of process monitoring methods. In this case study, we select the penicillin fermentation process for fault detection and isolation research. The penicillin fermentation process was developed by Cenk Undey, Gulnur Birol, and Ali Cinar of the Process Modeling, Monitoring, and Control Research Group of the Department of Chemical and Environmental Engineering at the Illinois Institute of Technology [17]. The schematic of the penicillin fermentation process is depicted in Fig. 7. It is a widely used benchmark to evaluate the performance of process monitoring methods. A total of 16 process variables and four operating modes are involved in the penicillin fermentation process. The process variables are listed in Table 4 [18]. There are four operating modes including a fault-free operating mode and three fault-operating modes. In this case study, each of the four operating modes runs for 300 h. In each fault operating mode, the fault is added at the 101st hour, as shown in Table 5. In this case study, we first evaluate

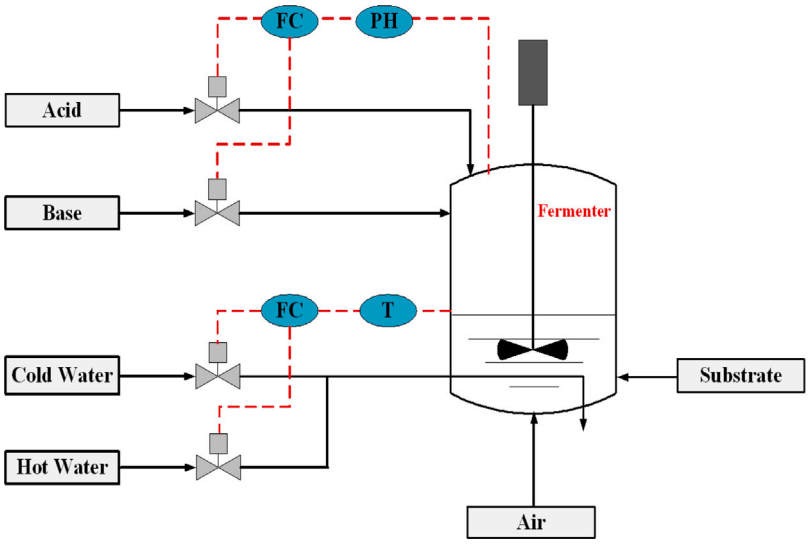


Fig. 7. Schematic of the penicillin fermentation process [17].

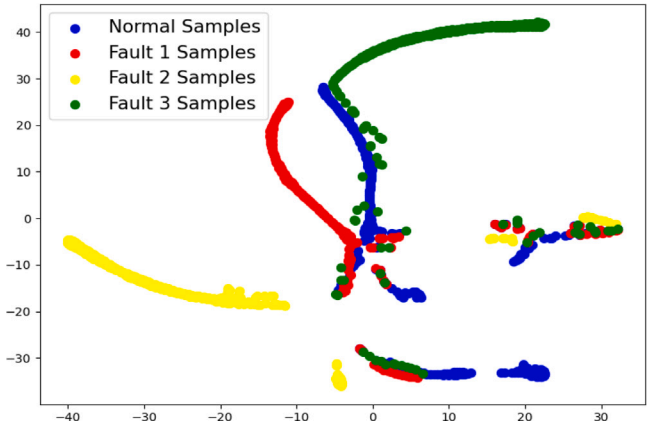


Fig. 8. Visualization of samples by t-SNE.

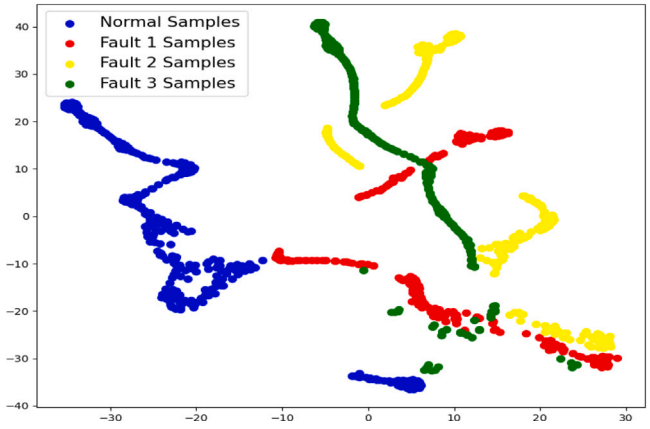


Fig. 9. Visualization of features by t-SNE.

the fault detection problem of the penicillin fermentation process. In this case study, we first compare MTLSTM with principal component analysis (PCA), and vector autoregression with lag order 2 (VAR(2)) for fault detection. Then, we compare MTLSTM with random forest (RF) and fully connected neural network (FCN) for fault isolation.

Table 4
Process variables.

No.	Variables
1.	Aeration rate (L/h)
2.	Agitator power (W)
3.	Substrate feed rate (L/h)
4.	Substrate feed temperature (K)
5.	Dissolved oxygen saturation (%)
6.	Biomass concentration (g/L)
7.	Penicillin concentration (g/L)
8.	Culture volume (L)
9.	CO_2 concentration (mmole/L)
10.	PH
11.	Temperature (K)
12.	Generated heat (kcal/h)
13.	Acid flow rate (mL/h)
14.	Base flow rate (mL/h)
15.	Cold water flow rate (L/h)
16.	Hot water flow rate (L/h)

Table 5
Fault descriptions.

Period	Fault variable	Type
101–300 h	Aeration rate	Step
101–300 h	Agitator power	Step
101–300 h	Substrate feed rate	Step

Data collection is the first step in this case study. We collected 300 fault-free samples from the fault-free operating mode. Moreover, we also collected 100 fault-free samples and 200 fault samples from each fault operating mode. We selected 300 fault-free samples collected from the fault-free operating mode, 100 fault-1 samples, 100 fault-2 samples, and 100 fault-3 samples to construct a training set. With the training sample set, then we are allowed to train the MTLSTM. While, the models of PCA, ICA, VAR(2) were trained on the 300 fault-free samples. Before fault detection, it is important to visualize the distribution of the samples collected from different operating modes. We apply the t-SNE to map the collected 16-dimensional samples to a two-dimensional plane for visualization [34], as shown in Fig. 8. Fig. 8 shows the samples from different operating modes overlapped and show complicated nonlinear manifolds. Thus, it is hard to recognize the fault operating mode base on the model trained on a set of fault-free samples because fault samples and fault-free samples almost are overlapped. Moreover, we also apply t-SNE to the features extracted from the original samples, as shown in Fig. 9. The features extracted from the different types of

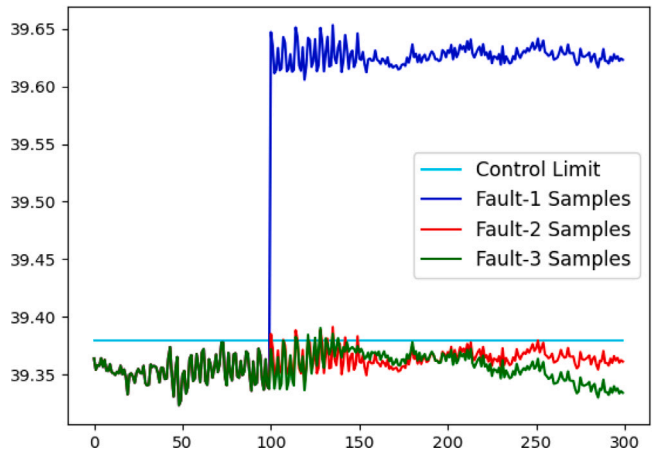


Fig. 10. Fault detection results by PCA.

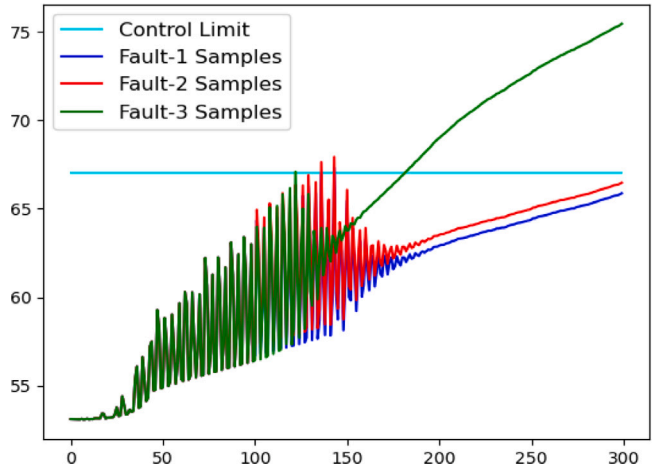


Fig. 11. Fault detection results by ICA.

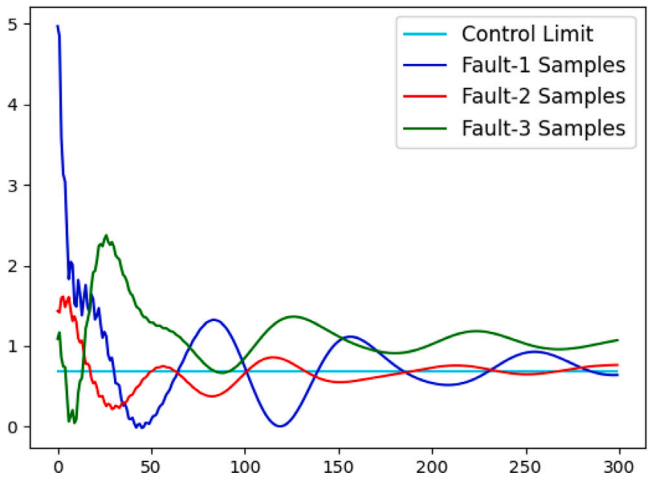


Fig. 12. Fault detection results by VAR(2).

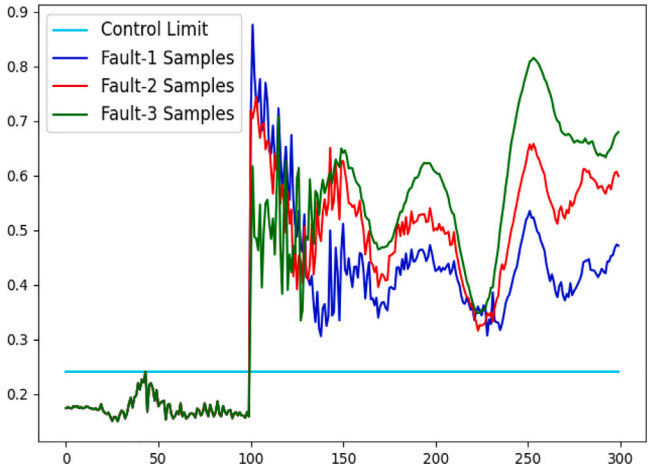


Fig. 13. Fault detection results by MTLSTM.

samples distribute in different regions and overlap less than the original samples. This indicates that fault detection and isolation based on the features instead of original samples is feasible.

After the models get well trained on the training sample set, then we applied to the models to fault detection for each fault operating mode. Based on the well-trained models, we perform process monitoring for the fault operating modes. In the fault detection, we compute the reconstruction error for the samples collected from each fault operating mode using the well trained models. Then, we compare the reconstruction errors with a threshold. If the error of a sample exceeds the threshold, then a fault is detected. Otherwise, the sample is fault free. Fault detection results based on PCA, ICA, VAR(2), and MTLSTM are depicted in Figs. 10, 11, 12, and 13, respectively.

To quantitatively evaluate the fault detection performance, we define two metrics based on the confusion matrix. The definition of a confusion matrix is illustrated in Fig. 14. For each fault detection approach, we compute the precision and recall based on the confusion matrix of fault detection results. Precision and recall are defined by the formulas (18) and (19), respectively. Higher precision and recall indicates a fault detection approach that has a better performance in this case study. To compute the confusion matrix of each approach, we assign the label 0 for the fault-free samples and 1 for the fault samples. Then, the fault detection problem is translated into a binary classification problem. The precision and recall of each fault detection approach are summarized in Table 6. Obviously, MTLSTM gets the highest precision and recalls for all three faults. However, PCA fails

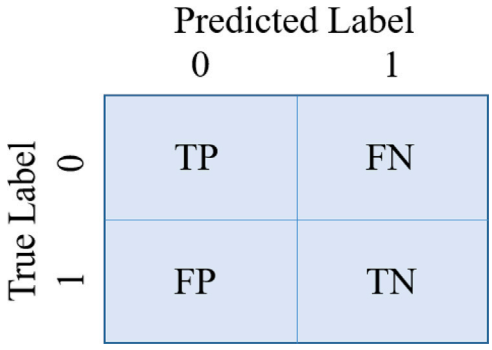


Fig. 14. Definition of the confusion matrix.

to detection the second and third types of fault only detect the first type of fault correctly. For ICA and VAR(2), they partially correct fault detection results for all three types of fault. In conclusion, MTLSTM shows the highest performance in this fault detection.

$$precision = \frac{TP}{TP + FP} \tag{18}$$

$$recall = \frac{TP}{TP + FN} \tag{19}$$

Table 6
Fault detection comparison.

Faults	PCA	ICA	AR(2)	MTLSTM
Fault 1	Precision = 1 Recall = 1	Precision = 1 Recall = 0.33	Precision = 0.34 Recall = 0.25	Precision = 1 Recall = 1
Fault 2	Precision = 1 Recall = 0.34	Precision = 1 Recall = 0.33	Precision = 0.72 Recall = 0.41	Precision = 1 Recall = 1
Fault 3	Precision = 1 Recall = 0.34	Precision = 1 Recall = 0.55	Precision = 0.20 Recall = 1	Precision = 1 Recall = 1

Table 7
Fault isolation accuracy comparison.

RF	FCN	MTLSTM
98.00%	97.33%	100.00%

Table 8
Model complexity and computation time.

Models	Model learnable parameters	Computation time
PCA	9600	0.080 s
ICA	9600	0.059 s
VAR	512	0.091 s
RF	None	0.083 s
FCN	1219	0.154 s
MTLSTM	1184	0.190 s

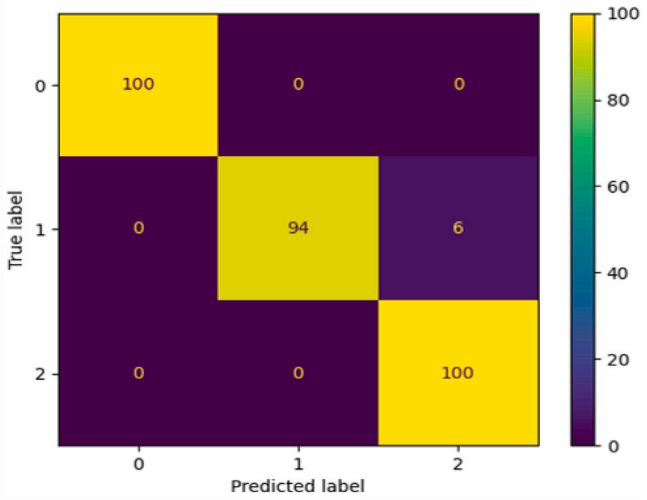


Fig. 15. Confusion matrix of the fault isolation results of RF.

After a fault is detected, then it is important to perform fault isolation. However, we have to emphasize that PCA, ICA, and VAR(2) have no fault isolation feature. Therefore, we next perform fault isolation using the random forest (RF), fully connected neural (FCN) network, and MTLSTM where the FCN consists of three layers where the neurons of the layers are 32, 16, 3, respectively. The RF and FCN were trained on a set of samples consisting of 100 fault-1 samples, 100 fault-2 samples, and 100 fault-3 samples. Then, the well-trained models were used for fault isolation where the samples for fault isolation evaluation consists of 100 fault-1 samples, 100 fault-2 samples, and 100 fault-3 samples. Then, with the predicted labels and ground truth labels of the test samples, we are allowed to calculate the confusion matrices of the fault isolation results for each method, as shown in Figs. 15, 16, and 17. Fig. 15 indicates that there are 8 samples isolated by mistake for RF-based fault isolation. For FCN-based fault isolation, there are 8 samples isolated by mistake. The MTLSTM-based fault isolation approach achieved the one hundred percent accuracy.

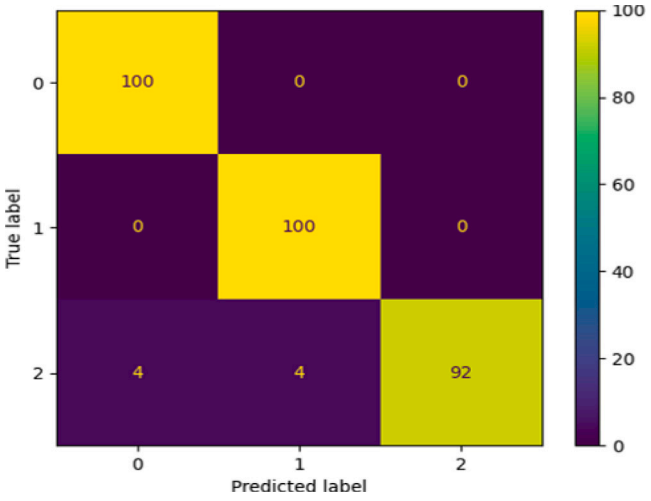


Fig. 16. Confusion matrix of the fault isolation results of FCN.

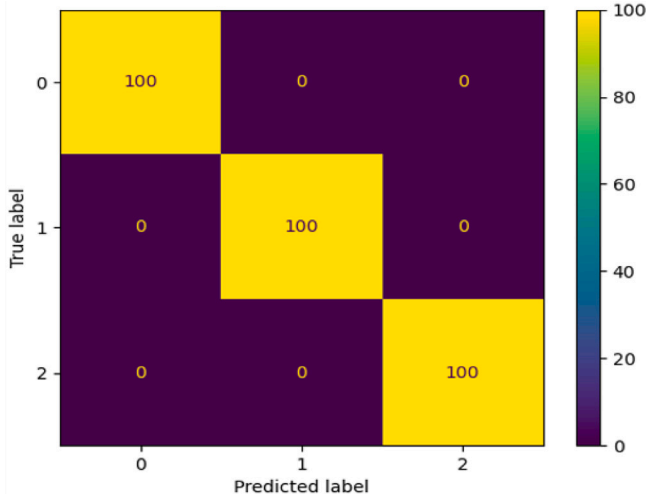


Fig. 17. Confusion matrix of the fault isolation results of MTLSTM.

To quantitatively evaluate the fault isolation accuracy (FIA), we define

$$FIA = \frac{trace\,(\mathbf{A})}{|\mathbf{A}|} \tag{20}$$

where $\mathbf{A}$ represents the confusion matrix in Fig. 15, $trace(\mathbf{A})$ represents the trace of matrix $\mathbf{A}$, and $|\mathbf{A}|$ represents the sum of the elements of matrix $\mathbf{A}$. Then, we compute the FIA of each approach based on the confusion matrix $\mathbf{A}$, we immediately get the FIA of RF, FCN, and MTLSTM that are 98.00%, 97.33% and 100.00%, respectively. The FIA of each fault isolation approach is summarized in Table 7.

The model complexity and model inference efficiency are two important metrics we are concerned with. Thus, we list the number of parameters of each model and the model inference time in Table 8 for comparison. Since RF has no learnable parameters, thus the number of learnable parameters of RF is marked as None. The computation time of each model was calculated on a PC with I7 CPU and RTX2060 GPU. Table 8 shows that PCA and ICA have more learnable model parameters and take less computation time. Even deep models have fewer model parameters, but they take much computation time because they have more complicated model structures than multivariate statistical models. In conclusion, MTLSTM achieves a higher fault detection and isolation performance at the cost of computation efficiency.

In this section, we have studied the fault detection and isolation problems of the penicillin fermentation process. Both multivariate statistical models and MTLSTM showed good fault detection and isolation performance in the case study. The experimental results demonstrate that different multivariate statistical models can tackle different process monitoring tasks such as fault detection and fault isolation. For example, PCA can detect the first fault operating mode. However, it fails to detect the second and third fault operating modes. RF shows high performance in fault isolation. Compared with the multivariate statistical models, MTLSTM can achieve fault detection and isolation simultaneously and shows higher performance in both fault detection and fault isolation of the penicillin fermentation process.

6. Conclusions

In this paper, we have verified the fault detection and isolation performance of multivariate statistical models and deep learning models. The previous researches on multivariate statistical models inspire us to develop the MTLSTM. Moreover, the investigations on fault detection and isolation based on multivariate statistical models promote fault detection and isolation research based on MTLSTM. As a result, based on the existing studies on multivariate statistical models, we proposed the MTLSTM for fault detection and isolation. MTLSTM inherits the merits of multivariate statistical models so that it significantly improvements fault detection and isolation performance. In the future, we hope to achieve more process monitoring tasks and improve process monitoring performance with an unified model.

CRediT authorship contribution statement

Ken Sinkou Qin: Conceptualization, Methodology, Original draft preparation. **Yegang Du:** Investigation, Software, Validation, Writing, Reviewing and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgement

The work was supported by the Fundamental Research Funds for the Central Universities (No. 017231167).

References

[1] S.J. Qin, Statistical process monitoring: basics and beyond, J. Chemometr. 17 (2003) 480–502.
[2] J.H. Cho, J.M. Lee, S.W. Choi, D. Lee, I.B. Lee, Fault identification for process monitoring using kernel principal component analysis, Chem. Eng. Sci. 60 (2005) 279–288.
[3] G. Jia, Q. Wang, B. Huang, Dynamic higher-order cumulants analysis for process monitoring based on a novel lag selection, Inform. Sci. 331 (2016) 45–66.
[4] R. Muradoreand, P. Fiorini, A PLS-based statistical approach for fault detection and isolation of robotic manipulators, IEEE Trans. Ind. Electron. 59 (2012) 3167–3175.
[5] J.A. Westerhuis, S.P. Gurden, A.K. Smilde, Generalized contribution plots in multivariate statistical process monitoring, Chemometr. Chemom. Int. Lab. Syst. 51 (2000) 95–114.
[6] R.R. Sun, Multiblock global orthogonal projections to latent structures for fault diagnosis, Chemometr. Chemom. Int. Lab. Syst. 204 (2020) 104092.
[7] W. Ku, R.H. Storer, C. Georgakis, Disturbance detection and isolation by dynamic principal component analysis, Chemometr. Chemom. Int. Lab. Syst. 30 (1995) 179–196.
[8] S.J. Qin, Survey on data-driven industrial process monitoring and diagnosis, Annu. Rev. Control 36 (2012) 220–234.
[9] I. Hwang, S. Kim, Y. Kim, C.E. Seah, A survey of fault detection, isolation, and reconfiguration methods, IEEE Trans. Control Syst. Technol. 18 (2010) 636–653.
[10] X. Li, F. Duan, P. Loukopoulos, I. Bennett, D. Mba, Canonical variable analysis and long short-term memory for fault diagnosis and performance estimation of a centrifugal compressor, Control Eng. Pract. 72 (2018) 177–191.
[11] Q.L. Jia, Y.W. Zhang, L.R. Zhai, L. Feng, Uncorrelated component analysis on manifold for statistical process monitoring, J. Chemometr. 31 (2017) e2918.
[12] G. Li, S.J. Qin, D.H. Zhou, A new method of dynamic latent-variable modeling for process monitoring, IEEE Trans. Ind. Electron. 61 (2014) 6438–6445.
[13] P. Song, C. Zhao, B. Huang, Sfnet: A slow feature extraction network for parallel linear and nonlinear dynamic process monitoring, Neurocomputing 488 (2022) 359–380.
[14] J. Yang, L. Wang, Nonlocal, local and global preserving stacked autoencoder based fault detection method for nonlinear process monitoring, Chemometr. Chemom. Int. Lab. Syst. 235 (2023) 104758.
[15] L. Luo, L. Xie, H. Su, Deep learning with tensor factorization layers for sequential fault diagnosis and industrial process monitoring, IEEE Access 8 (2020) 105494–105506.
[16] J. Yu, C. Zhang, Manifold regularized stacked autoencoders-based feature learning for fault detection in industrial processes, J. Process Control 92 (2020) 119–136.
[17] R. Wang, T.F. Edgar, M. Baldea, M. Nixon, W. Wojsznis, R. Dunia, A geometric method for batch data visualization, process monitoring and fault detection, J. Process Control 16 (2018) 197–205.
[18] J.M. Lee, C.K. Yoo, I.B. Lee, Fault detection of batch processes using multiway kernel pricipal component analysis, Comput. Chem. Eng. 28 (2004) 1837–1847.
[19] Y. Dong, S.J. Qin, New dynamic predictive monitoring schemes based on dynamic latent variable models, Ind. Eng. Chem. Res. 59 (2020) 2353–2365.
[20] S. Gajjar, M. Kulahci, A. Palazoglu, Real-time fault detection and diagnosis using sparse principal component analysis, J. Process Control 67 (2018) 112–128.
[21] S. Wold, K. Esbensen, P. Geladi, Principal component analysis, Chemometr. Chemom. Int. Lab. Syst. 2 (1987) 37–52.
[22] J.M. Lee, S.J. Qin, I.B. Lee, Statistical process monitoring with independent component analysis, J. Process Control 14 (2004) 467–485.
[23] P.P. Odiowei, Y. Cao, State-space independent component analysis for nonlinear dynamic process monitoring, Chemometr. Chemom. Int. Lab. Syst. 103 (2010) 59–65.
[24] L. Chen, V. Makis, Application of vector time series modeling and t-squared control chart to detect early gearbox deterioration, Int. J. Perform. Eng. 10 (2014) 105–114.
[25] Y.J. Chen, J.Z. Ming, A sparse multivariate time series model-based fault detection method for gearboxes under variable speed condition, Mech. Syst. Signal Proc. 167 (2022) 108539.
[26] Z.W. Hu, H.T. Zhao, J.C. Peng, Low-rank reconstruction-based autoencoder for robust fault detection, Control Eng. Practice 123 (2022) 105156.
[27] J.B. Yu, C.Y. Zhang, Manifold regularized stacked autoencoders-based feature learning for fault detection in industrial processes, J. Process Control 92 (2020) 119–136.
[28] D. Cacciarelli, M. Kulahci, A novel fault detection and diagnosis approach based on orthogonal autoencoders, Comput. Chem. Eng. 163 (2022) 107853.
[29] Z. Yang, P. Baraldi, E. Zio, A method for fault detection in multi-component systems based on sparse autoencoder-based deep neural networks, Reliab. Eng. Syst. Saf. 220 (2022) 108278.
[30] F. Cheng, Q.P. He, J. Zhao, A novel process monitoring approach based on variational recurrent autoencoder, Comput. Chem. Eng. 129 (2019) 106515.
[31] Y. Liu, R. Young, B. Jafarpour, Long-short-term memory encoder–decoder with regularized hidden dynamics for fault detection in industrial processes, J. Process Control 124 (2023) 166–178.
[32] J. Yu, X. Liu, L. Ye, Convolutional long short-term memory autoencoder based feature learning for fault detection in industrial processes, IEEE Trans. Instrum. Meas. 70 (2021) 1–15.
[33] S. Zhang, K. Bi, T. Qiu, Bidirectional recurrent neural network-based chemical process fault diagnosis, Ind. Eng. Chem. Res. 59 (2019) 824–834.
[34] L. Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (2008) 2579–2605.