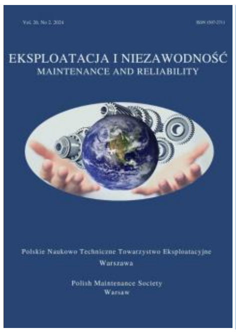


Eksploatacja i Niezawodnosc – Maintenance and Reliability
Volume 26 (2024), Issue 2
journal homepage: http://www.ein.org.pl

Article citation info:
Wang C, Liu J, Research on Online Condition monitoring for Complex System based on Modified Broad Learning Systems, Eksploatacja i Niezawodnosc – Maintenance and Reliability 2024: 26(2) http://doi.org/10.17531/ein/183600

Research on Online Condition monitoring for Complex System based on Modified Broad Learning Systems

Indexed by: Web of Science Group

Chong Wang[a], Jie Liu[a,*]

[a] Beihang University, China

Highlights

- An online condition monitoring system is designed based on broad learning system (BLS).
- Two methods for optimizing BLS are proposed based on correlation and causality.
- The effectiveness of the methods is verified by both simulation data and real data.

Abstract

Complex systems contain numerous interacting components, thus deep learning methods with powerful performance and complex structure are often used to achieve condition monitoring. However, the deep learning methods are always too time-consuming and hardware-demanding to be loaded into complex systems for online training and updates. To achieve accurate and timely monitoring of complex system state, based on broad learning system (BLS), an online condition monitoring method is proposed in this paper. General BLSs are based on a randomly generated hidden-layer, usually perform poorly in high-dimensional data classification tasks. In this work, based on correlation and causality, two modified BLSs are proposed and mixed to establish the online monitoring system. Specifically, logistic regression (LR) and structural causal model (SCM) are considered to form rough predictions of the system state, thus to replace the randomly generated ones with no practical significance. The effectiveness of the proposed online monitoring method is verified by both simulation data and real data.

Keywords
broad learning system, causality, complex system, condition monitoring, high-speed train

1. Introduction

Complex system refers to the system composed of multiple interacting mechanical and electronic components, which have high reliability, complexity, and intelligence. Nowadays, complex system have been widely integrated in industrial production. To ensure the system safety, condition monitoring technology is widely used in complex systems 1. And, with the development of data-collection equipment and powerful computer, artificial intelligence (AI) methods have attracted lots of attentions in condition monitoring research 1323. For complex system with high dimensional coupled monitoring data, deep learning methods are mostly considered to establish condition monitoring models because of their powerful capabilities of effective information mining and non-linear relationship fitting 10.

For example, with convolutional neural network (CNN), a condition monitoring model of rolling bearing was established in 31. The minimum entropy deconvolution (MED)-based CNN was proposed in 24 for fault diagnosis of axial piston pumps. In 25, based on radial basis function neural network (RBFNN), an effective power system monitoring and control method was proposed. In 19, based on EMD and deep neural network (DNN), hand-crafted (low-level) features and high-level features were

(*) Corresponding author.
E-mail addresses: C. Wang, (ORCID: 0000-0002-4286-4902) chongzi@buaa.edu.cn, J. Liu (ORCID: 0000-0003-0895-7598) liujie805@buaa.edu.cn,

extracted and fused for condition monitoring of machines. And, by using deep belief networks (DBNs), a fault detection method for axial piston pumps was proposed in 25.

The effectiveness of deep learning methods had been justified in many state-of-art research. And, the successive proposals of advanced and powerful deep learning methods provide effective ways to further improve the model accuracy 517. But, the condition monitoring models are commonly trained in an offline manner. This is because that, to obtain satisfactory prediction performances, model training and updating process require much computing resources and time. Powerful computers are difficult to load into the complex systems, and even if they can, their onboard security and availability will become new challenges. For complex systems, it is usually required that the condition monitoring model can be updated in an online manner to adapt the variable operation environment. Therefore, online models with high computational efficiency have gradually attracted the attention of researchers.

Among them, broad learning system (BLS) proposed in 2 has become one of the most popular methods. There is only one hidden-layer in BLS for feature extraction, which is composed of initial node groups, characteristic node groups and incremental node groups. The model training process is to calculate the pseudo-inverse matrix of the hidden layer nodes. And, the model update process is achieved by adding incremental node groups in the hidden-layer,which is a incremental learning process. In the incremental learning process, i.e. when adding an incremental node group, the whole BLS model does not need to be retrained. By calculating the pseudo inverse matrix of the added incremental node group, the BLS can be updated efficiently 23. Thus, BLSs usually have high training and updating efficiency.

However, the hidden-layer nodes (specially the initial node groups) in BLS are randomly generated from the input features, resulting in that BLSs perform unsatisfactory in high-dimensional data classification tasks. To fully explore the available information of the high-dimensional data, the hidden-layer in BLS is need to be set with a large number of nodes 9. And, with the increase of the number of BLS hidden-layer nodes, the improvement of model performance is no longer obvious after reaching a certain level (the experiment results can be seen in 2). As mentioned in 9, the low efficiency of feature extraction process is one of the main limitations of BLS. Thus, it is necessary to explore methods for improving feature extraction efficiency of BLS models.

The existing methods includes initial node optimization, structure compactness, ensemble learning, etc 9. Among them, initial node optimization is the most convenient method without increasing the computational burden. At present, most state-of-art studies focus on model input optimization, and can be categorized into two types. One focuses on integrating powerful feature extraction methods with BLS for processing time series, text, image data. For example, CNN is commonly used as the feature extractor to process image data 1627, and recurrent neural network (RNN) is used for processing text data 7. The other is committed to feature extraction of the original data for improving the effectiveness of the input data in BLS. Specifically, feature extraction methods are used to simplify the input data of BLSs for improving model training and updating efficiency, including variational mode decomposition (VMD) 33, Hilbert transform (HT) 33, singular value decomposition (SVD) 11, principal component analysis (PCA) 32, etc.

These above studies rely on the performance of the used feature extraction methods to mine fault information. It is difficult to ensure that all valuable information in high-dimensional data of complex system can be effectively extracted. And, the feature extraction efficiency of BLS itself has not been improved.

In this work, based on the perspectives of correlation and causality, two modified BLSs (MBLS) are proposed to improve the overly random generation of the initial node groups in BLS. The MBLS based on correlation (logistic regression, LR) uses all the variables to avoid information loss. And, the MBLS based on causality (structural causal model, SCM) is stable. For synthesizing the advantages, the two MBLSs are integrated to construct the condition monitoring model. Moreover, to obtain more practical causal information of complex systems, the existing causal discovery method is improved by adding empirical constraints. Based on the simulation data, it is proved that the proposed method has some certain universality in complex system condition monitoring. And, using actual monitoring data of an HST brake control system, the high accuracy and efficiency of the proposed condition monitoring method are confirmed.

The remainder of this paper is organized as follows. The proposed online condition monitoring method is described in Section 2. Section 3 analyzes the model applicability. And, Section 4 illustrates the application results in a real high-speed train. Conclusions and some further research directions are drawn in Section 5.

2. Online condition monitoring framework for complex systems

The structure of the proposed online condition monitoring framework for a complex system is shown in Figure 1. Firstly, sensors and monitoring tools are equipped to collect the historical monitoring data of the complex system. And, the categorical and numerical variables in the heterogeneous monitoring data are en-coded into numerical features. Then, based on LR and SCM, two modified BLSs (MBLS) are proposed to improve the feature extraction efficiency of the initial node groups in BLS. Specifically, the calculation formula of the initial node groups is replaced by the LR equation or the structural causal equation (SCE) to ob-tain efficient fault characteristics. Moreover, for improving the performance of the condition monitoring method, the two MBLSs are integrated to detect the real-time health state of the complex system. The integrated model is recorded as Mix-MBLS.

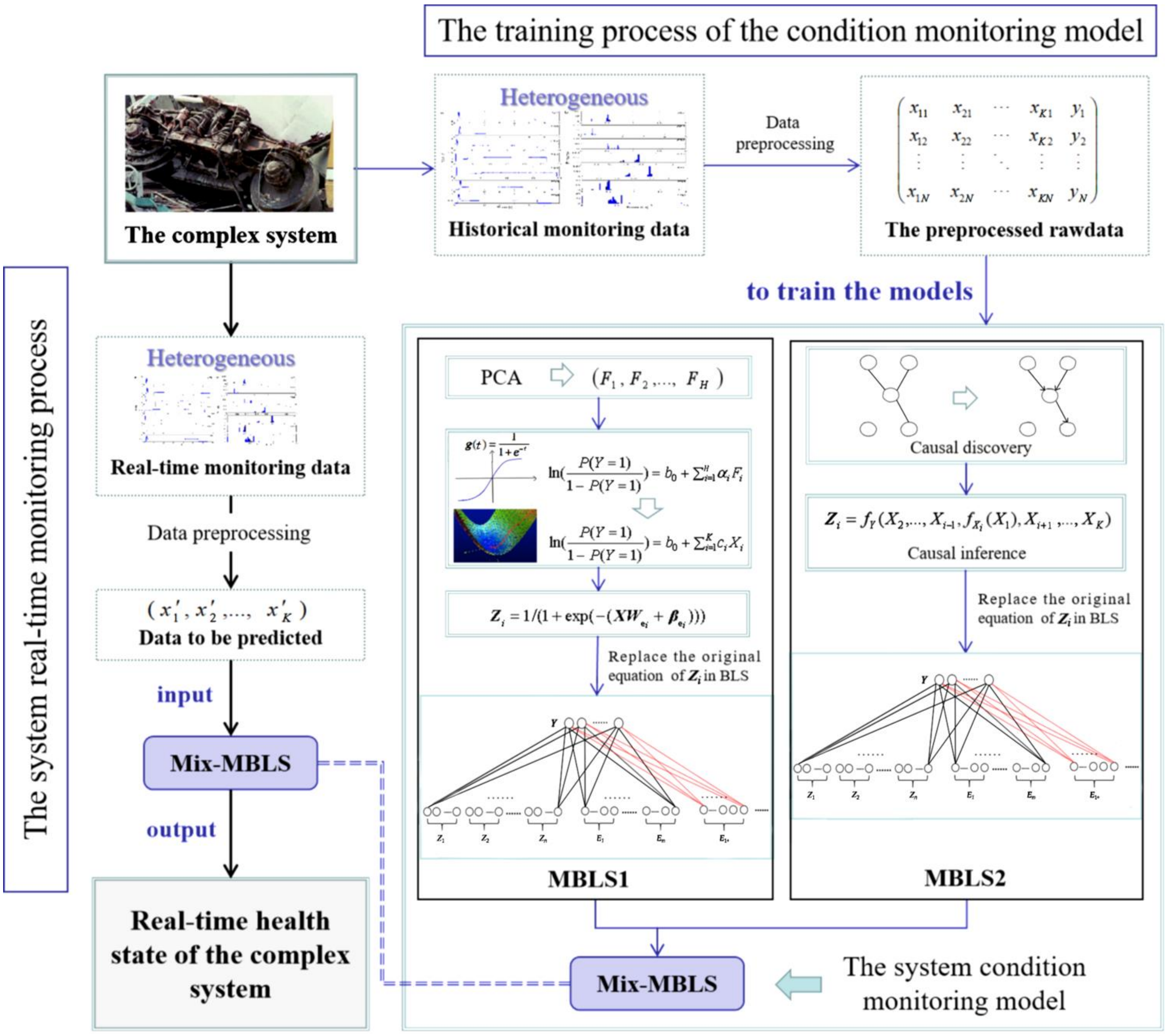
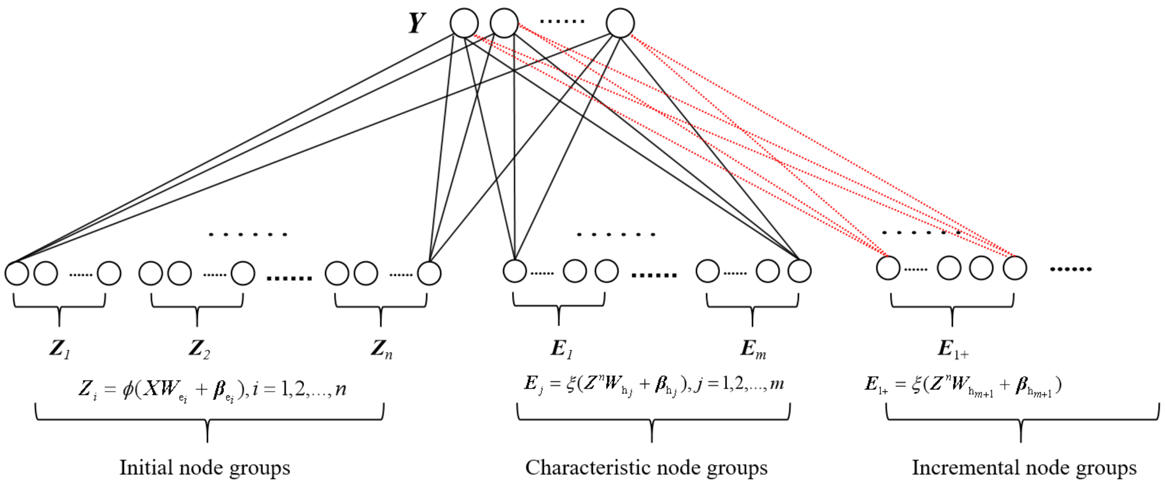

The training process of the condition monitoring model
The complex system
Heterogeneous
Historical monitoring data
Data preprocessing
The preprocessed rawdata
to train the models
The system real-time monitoring process
Heterogeneous
Real-time monitoring data
Data preprocessing
$(x'_1, x'_2, \dots, x'_K)$
Data to be predicted
input
Mix-MBLS
output
Real-time health state of the complex system
PCA
$(F_1, F_2, \dots, F_H)$
$g(t) = \frac{1}{1+e^{-t}}$
$\ln(\frac{P(Y=1)}{1-P(Y=1)}) = b_0 + \sum_{i=1}^{H} \alpha_i F_i$
$\ln(\frac{P(Y=1)}{1-P(Y=1)}) = b_0 + \sum_{i=1}^{K} c_i X_i$
$Z_i = 1/(1+\exp(-(XW_{e_i} + \beta_{e_i})))$
Replace the original equation of Z_i in BLS
MBLS1
Causal discovery
$Z_i = f_Y(X_2, \dots, X_{i-1}, f_{X_i}(X_1), X_{i+1}, \dots, X_K)$
Causal inference
Replace the original equation of Z_i in BLS
MBLS2
Mix-MBLS
The system condition monitoring model

Fig.1. The online condition monitoring framework for an example complex system.

It should be noted that the simple and interpretable models, the LR and the SCM are adopted to make the feature extraction process transparent. The LR is based on the correlation of variables, and can use all monitoring variables to avoid information loss. The SCM is based on causality and has stronger interpretability and stability. However, it only takes into the cause features of the system state, which may lead to information loss. In order to synthesize the advantages of the two models, the integrated model Mix-MBLS is obtained by fusing the two MBLSs at the decision level using parallel method. Specifically, if any MBLS output is fault state, the final prediction result of the system health state is fault.

2.1. The basic BLS

There have been many theoretical foundations and useful variants since the emergence of BLS 9. Since this work focuses on optimizing the initial node groups in the BLS hidden-layer, the basic BLS model is considered in this work. This work will be committed to improve the effectiveness and efficiency of BLS. In fact, the proposed optimizations for the initial node groups can also be adopted in other variants of BLS.



Y
Z_1
Z_2
Z_n
E_1
E_m
E_{1+}
$Z_i = \phi(XW_{e_i} + \beta_{e_i}), i = 1,2,\ldots,n$
$E_j = \xi(Z^n W_{h_j} + \beta_{h_j}), j = 1,2,\ldots,m$
$E_{1+} = \xi(Z^n W_{h_{m+1}} + \beta_{h_{m+1}})$
Initial node groups
Characteristic node groups
Incremental node groups

Fig.2. Structural overview of the basic BLS with incremental learning.

The hidden-layer of a basic BLS is composed of three node groups, which are initial node groups, characteristic node groups and incremental node groups, as shown in Figure 2.

These initial node groups $\boldsymbol{Z}_1$, $\boldsymbol{Z}_2$, ..., $\boldsymbol{Z}_n$ ($\boldsymbol{Z}^n \overset{\Delta}{=} (\boldsymbol{Z}_1, \boldsymbol{Z}_2, \cdots, \boldsymbol{Z}_n)$) are nonlinear feature maps based on the original input data $\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_K)$, which can be calculated by Eq. 1.

$$\mathbf{Z}_i = \varphi(\mathbf{X}\mathbf{W}_{e_i} + \boldsymbol{\beta}_{e_i}), i = 1,2,\ldots,n \tag{1}$$

In Eq. 1, $\varphi()$ is a nonlinear mapping function called activation function, $\boldsymbol{Z}_i$ is the i-th mapped feature group in $\boldsymbol{Z}^n$. These two matrices, $\boldsymbol{W}_{e_i}$ and $\boldsymbol{\beta}_{e_i}$, are separately the weight matrix and the corresponding deviation matrix, and are both generated randomly. For the original input data ($\boldsymbol{X}$, $\boldsymbol{Y}$) ($\boldsymbol{X} \in \boldsymbol{R}^{N\times K}$, $\boldsymbol{Y} \in \boldsymbol{R}^{N\times Q}$), let $\boldsymbol{W}_{e_i} \in \boldsymbol{R}^{K\times v}$and $\boldsymbol{\beta}_{e_i} \in \boldsymbol{R}^{1\times V}$, then the mapped feature group $\boldsymbol{Z}_i$ has v nodes.

The characteristic node groups E_1, E_2, ..., E_m ($\boldsymbol{E}^m \overset{\Delta}{=} (\boldsymbol{E}_1, \boldsymbol{E}_2, \ldots, \boldsymbol{E}_m)$) are nonlinear feature maps of the initial node groups, which can be calculated by Eq. 2.

$$\mathbf{E}_j = \xi(\mathbf{Z}^n\mathbf{W}_{h_j} + \boldsymbol{\beta}_{h_j}) \tag{2}$$

In Eq. 2, E_j is the j-th characteristic node group, $\xi()$is an activation function as in Eq. 1. The weight matrix $\boldsymbol{W}_{h_j}$ and the corresponding deviation matrix $\boldsymbol{\beta}_{h_j}$ are also both generated randomly. $\boldsymbol{W}_{h_j} \in \boldsymbol{R}^{nv\times\eta}$ and $\boldsymbol{\beta}_{h_j} \in \boldsymbol{R}^{1\times\eta}$, the characteristic node group E_j contains η nodes.

Then, the BLS model can be expressed as:

$$\mathbf{Y} = \mathbf{H}\mathbf{W} = (\mathbf{Z}^n|\mathbf{E}^m)\mathbf{W} \tag{3}$$

where $\boldsymbol{W} \in \boldsymbol{R}^{L\times Q}$ is the node weight matrix to connect $\boldsymbol{H} = (\boldsymbol{Z}^n|\boldsymbol{E}^m) \in \boldsymbol{R}^{N\times L}$ to the output $\boldsymbol{Y} \in \boldsymbol{R}^{N\times Q}$, $L = nv + m\eta$ is the total number of hidden-layer nodes in the BLS.

BLS model fitting is essentially to obtain the analytical solution with the pseudo inverse matrix of hidden-layer nodes based on (X, Y). Based on the least square method with l_2-norm regularization, the analytical solution of a classical BLS without incremental learning is shown in Eq. 4:

$$\mathbf{W} = \begin{cases} \qquad\qquad\quad \mathbf{H} \\ T^{T^{-1}} \quad \mathbf{N<L} (c\mathbf{I} + \mathbf{H}^{T^{-1}})^{T} \quad \mathbf{N\geq L} \end{cases} \tag{4}$$

where I is the L-order identity matrix, the value c represents the penalty coefficient of the l_2-norm regularization term.

When the classical BLS model cannot achieve the required accuracy, incremental node groups can be inserted to improve the model performance. Incremental node groups are the supplement to the characteristic node groups, which can be considered as the characteristic node group after the m-th. Thus, the first incremental node group E_{1+} with p nodes can be calculated analytically by Eq. 5:

$$\mathbf{E}_{1+} = \xi(\mathbf{Z}^n \mathbf{W}_{h_{m+1}} + \boldsymbol{\beta}_{h_{m+1}}) \tag{5}$$

with the weight matrix $\boldsymbol{W}_{h_{m+1}} \in \boldsymbol{R}^{nv\times p}$ and the deviation matrix $\boldsymbol{\beta}_{h_{m+1}} \in \boldsymbol{R}^{nv\times 1}$.

The new BLS model can be expressed as:

$$\mathbf{Y} = (\mathbf{H}|\mathbf{E}_{1+})\mathbf{W}^{m+1} \triangleq \mathbf{H}^{m+1}\mathbf{W}^{m+1} \tag{6}$$

where $\boldsymbol{H}^{m+1} = (\boldsymbol{H}|\boldsymbol{E}_{1+})$ is the new hidden-layer nodes and $\boldsymbol{W}^{m+1}$ is the new node weight matrix.

Then, $\boldsymbol{W}^{m+1} \in \boldsymbol{R}^{(L+p)\times Q}$ can be obtained by Eq. 7:

$$\mathbf{W}^{m+1} = \begin{bmatrix} \mathbf{W} - \mathbf{D}\mathbf{B}^T \\ \mathbf{B}^T \end{bmatrix} \tag{7}$$

where $\boldsymbol{D} = (\boldsymbol{H})^+\boldsymbol{E}_{1+}$, $\boldsymbol{C} = \boldsymbol{E}_{1+} - \boldsymbol{H}\boldsymbol{D}$, $\boldsymbol{B}^{T}\begin{cases}(\boldsymbol{C})^+, & \boldsymbol{C}\neq\boldsymbol{0} \\ (1+\boldsymbol{D}^{T^{-1}})^{T+} & \end{cases}$, $(*)^+$ is the pseudo inverse of matrix *, and $\mathbf{0} \in \boldsymbol{R}^{1\times p}$ is the zero matrix where all the elements are 0.

Similarly, $\boldsymbol{W}^{m+2}$ can be calculated based on $\boldsymbol{W}^{m+1}$ after adding the second incremental node group. It can be seen that the pseudo inverse of previous nodes can be used directly, only the pseudo inverse of the additional incremental node group needs to be calculated, thus the training process can be greatly accelerated 23.

2.2. Optimizations of the initial node groups in BLS

It can be seen from section 2.1 that the initial node groups in the basic BLS are generated with random weight matrix $\boldsymbol{W}_{e_i}$ and deviation matrix $\boldsymbol{\beta}_{e_i}$. Therefore, lots of initial node groups are usually required to ensure the prediction accuracy of the model, which is likely to make model training and updating time-consuming. Based on correlation and causality, this paper proposes two strategies for improving efficiency and effectiveness of the initial node groups.

2.2.1. Initial node groups generation based on logistic regression

To improve the feature extraction efficiency, the convenient and interpretable logistic regression (LR) equation is applied to generate the initial node groups. LR 6 is a kind of generalized linear models, which can create a mathematical model based on independent input variables to predict the occurrence probability of the event corresponding to the dependent variable. The goal of LR is to find the best-fitting linear model to describe the relationship of the probability distribution of the binary category dependent variable Y with the independent input variables $X_1, X_2, ..., X_K$.

The LR model can be defined as:

$$Y' = b_0 + \sum_{i=1}^{K} \alpha_i X_i = b_0 + \alpha_1 X_1 + \alpha_2 X_2 + \ldots + \alpha_K X_K \tag{8}$$

$$Y' = ln(\frac{P(Y=1)}{P(Y=0)}) = ln(\frac{P(Y=1)}{1-P(Y=1)}) \tag{9}$$

where b_0 is the constant term, $\alpha_1, \alpha_2, ..., \alpha_K$ are the regression coefficients. $P(Y = 1)$ and $P(Y = 0)$ are the probabilities of Y in class 1 and class 0, respectively.

The ubiquitous strong correlation among the raw features from a complex system may lead to the regression coefficients contrary to the actual situation. Therefore, in this paper, principal component analysis (PCA) is first adopted to reduce the dimension of the raw features to obtain disentangled features. A LR model can be established based on the principal components, and then by substituting the principal component score function into the LR model, the relationship between each raw features X_i ($i=1, \ldots, K$) and $P(Y = 1)$ can be obtained. The specific calculation process is as follows:

① Calculate the correlation coefficient matrix Φ of the raw feature data $X = (X_1, X_2, ..., X_K)$, as well as all the eigenvalues $\lambda_1, \lambda_2, ..., \lambda_k$ and the corresponding eigenvectors $\mu_1, \mu_2, ..., \mu_K$ of Φ. Assume that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_K \geq 0$, $\boldsymbol{\mu}_i = (u_{i1}, u_{i2}, \ldots, u_{iK})^T$, $i = 1, 2, ..., K$;

② Calculate the variance contribution rate vcr_i of each eigenvalue λ_i ($i = 1, 2, ..., K$) and the cumulative contribution rate ccr_i of the first i features;

$$vcr_i = \frac{\lambda_i}{\sum_{j=1}^{K} \lambda_j} \tag{10}$$

$$ccr_i = \frac{\sum_{s=1}^{i} \lambda_s}{\sum_{j=1}^{K} \lambda_j} \tag{11}$$

According to the eigenvalues $\lambda_1, \lambda_2, ..., \lambda_k$ and the corresponding eigenvectors $\mu_1, \mu_2, ..., \mu_q$ of the first q features when ccr_q is greater than or equal to 85%, the q principal components $F_1, F_2, ..., F_q$ can be calculated by Eq. 12.

$$\mathbf{F} = (F_1, F_2, \cdots, F_q) = \mathbf{X}(\boldsymbol{\Phi}^{-1}\mathbf{A}) = (X_1, X_2, \cdots, X_K)(\boldsymbol{\Phi}^{-1}\mathbf{A}) \tag{12}$$

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1q} \\ a_{21} & a_{22} & \cdots & a_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{K1} & a_{K2} & \cdots & a_{Kq} \end{pmatrix} = \begin{pmatrix} u_{11}\sqrt{\lambda_1} & u_{21}\sqrt{\lambda_2} & \cdots & u_{q1}\sqrt{\lambda_q} \\ u_{12}\sqrt{\lambda_1} & u_{22}\sqrt{\lambda_2} & \cdots & u_{q2}\sqrt{\lambda_q} \\ \vdots & \vdots & \ddots & \vdots \\ u_{1K}\sqrt{\lambda_1} & u_{2K}\sqrt{\lambda_2} & \cdots & u_{qK}\sqrt{\lambda_q} \end{pmatrix} \tag{13}$$

where Φ^{-1} is the inverse matrix of the correlation coefficient matrix Φ.

③ Using LR, the regression equation of principal components $F_1, F_2, \cdots, F_q$ and Y can be obtained as shown in Eq. 14.

$$ln(\frac{P(Y=1)}{1-P(Y=1)}) = b_0 + \sum_{i=1}^{q} \alpha_i F_i \tag{14}$$

And, by substituting Eq. 12 into Eq. 14, the LR equation between the raw features $X_1, X_2, ..., X_K$ and Y can be obtained as:

$$ln(\frac{P(Y=1)}{1-P(Y=1)}) = b_0 + \sum_{i=1}^{K} c_i X_i \tag{15}$$

Based on the regression coefficient vector ($c_1, c_2, ..., c_K$) and constant term b_0 from Eq. 15, the value range of column vector in the weight matrix $\boldsymbol{W}_{e\,i}$ and the deviation matrix $\boldsymbol{\beta}_{e\,i}$ can be set. To be specific, define the coefficient vector interval as $[(c_1\text{-}\varepsilon, c_2\text{-}\varepsilon, ..., c_O\text{-}\varepsilon)^T, (c_1+\varepsilon, c_2+\varepsilon, ..., c_O+\varepsilon)^T]$, the offset interval as $[b_0\text{-}\varepsilon, b_0+\varepsilon]$, in where ε is a range parameter greater than 0. This paper holds that the LR coefficients are likely to be consistent with practical experience, so the maximum positive value that does not destroy practical experience of coefficients is selected as ε, i.e. if the influence of a variable on system fault is positive (negative), its coefficient will still be positive after subtracting (adding) ε. Not all variables can be judged by experience whether the coefficient should be positive or negative, so only the variables that can provide the basis for ε value are considered.

Let the K-dimensional column vectors of $\boldsymbol{W}_{e\,i}$ be taken from the coefficient vector interval, and the 1-dimensional column vectors of $\boldsymbol{\beta}_{e\,i}$ be taken from the offset interval. In this case, the initial node groups $Z_1, Z_2, ..., Z_n$ can be calculated by Eq. 16.

$$\mathbf{Z}_i = Sigmoid(\mathbf{XW}_{e_i} + \mathbf{\beta}_{e_i}) = 1/(1 + exp(-(\mathbf{XW}_{e_i} + \mathbf{\beta}_{e_i}))), i = 1,2,...,n \tag{16}$$

here Z_i is the i-th initial node group in Z^n, and $Sigmoid(*)$ is the *Sigmoid* function with value range being (0, 1).

In this paper, the LR equation is considered as the basis feature extractor mainly because of its convenience. In fact, other regression algorithms can also be selected to obtain the regression coefficients of each variable. However, due to the low interpretability of the regression coefficients in most other regression algorithms, there will be a lack of practical guidance when setting the value ranges of $\boldsymbol{W}_{e\,i}$ and $\boldsymbol{\beta}_{e\,i}$. In addition, for a multi-classification task, each class will be identified using a LR equation, and the initial node groups generated based on different LR equations are the rough predictions of different classes. And, different LR equations are the basic feature extractors for detecting different faults.

2.2.2. Initial node groups generation based on causality

Considering causality rather than correlation, interpretable and effective features can be obtained based on structural causal model (SCM). In this case, the calculation equation Eq. 1 to get the initial node groups can be replaced by the structural causal equation (SCE).

Causality, which is objective and sequential, reveals the spatial structure of data and can extract stable and accurate information than the commonly used correlation. Based on causality, the interpretability and stability of data-driven models can be effectively improved for practical applications. Data-based causal discovery, which can excavate the fundamental causal relationship in the observed data, had been proposed to effectively replace the infeasible controlled experiments 20.

The data-based causal discovery algorithms can be roughly divided into three categories: constraint-based, score-based and hybrid algorithms 8. Constraint-based algorithms are also called conditional independence methods, which have strong statistical theory support, thus have been widely used and improved since they were proposed. Among them, the commonly used algorithms include PC (Peter and Clark) algorithm 22, fast causal inference (FCI) algorithm 28, etc. Considering that finding causality by constraint-based algorithms has strong interpretability and mathematical process, and sufficient causal features related to system health status can be collected by using advanced sensors and monitoring tools (assuming there are no unmeasured confounders), PC algorithm is applied in this paper to analyze the causality between variables $X_1, X_2, ..., X_K, Y$.

The output of PC algorithm is normally a completed partially directed acyclic graph (CPDAG), which is composed of nodes, directed edges and undirected edges (as shown in Figure 3). An undirected edge between two nodes means that there is a causal relationship between the two variables but the cause can not be identified with the available data. Under causal sufficiency assumption, PC algorithm can be divided into two parts: determining the skeleton (undirected graph) and determining a CPDAG (the details can be seen in 22).

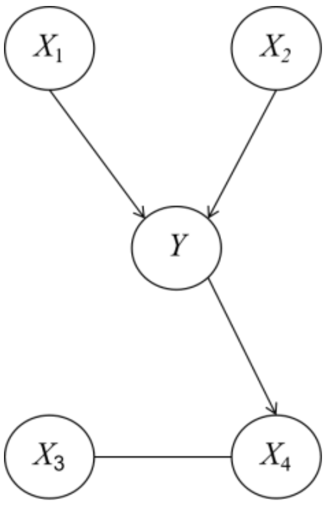


Fig. 3. An example causal CPDAG.

In this paper, empirical constraints are added to the PC algorithm to get a more stable and practical causal network. At present, there are few relevant studies on adding empirical constraints to PC algorithm, and most all of them used the method of constraining the model search space proposed in 15. The method in 15 is used and improved according to the empirical constraints designed in this work, which makes empirical constraints quickly and conveniently added to PC algorithm. Considering the limited expert experience that can be provided in complex systems, the added empirical constraints in this work are divided into the following two types:

① **Path constraint.** Constrain that there is no direct causal relationship between variable A and variable B, thus there is no edges between node A and node B in the causal network;

② **Direction constraint.** Constrain that variable A is the cause of variable B, thus there are only causal paths from A to B in the causal network. The causal path is the combination of linked edges from cause node to result node, which includes direct causal paths and indirect causal paths.

The above empirical constraints can be achieved by the following three steps:

① **Determining the skeleton under path constraints.**

By deleting the edges corresponding to the path constraints in the determined skeleton of PC algorithm, this step can be implemented quickly (as shown in Figure 4);

② **Extending the skeleton to a CPDAG under the direction constraints.**

This step is implemented by adding direction determining rules before the four ones in the second step of PC algorithm, the process is shown in Figure 5.

③ **Verifying whether all the direction constraints are satisfied.**

If a path does not meet a direction constraint, we can adjust the parameters of PC algorithm or delete the least relevant edge on the path.

```
Algorithm 1 Determining the skeleton under path constraints
 1: INPUT: Node (variable) set $V$, Conditional independence information
 2: OUTPUT: Estimated skeleton $C$, separation sets $S$ (only needed when directing the skeleton afterwards)
 3: From the complete undirected graph $\widetilde{C}$ on the node set $V$.
 4: $l = -1$; $C = \widetilde{C}$
 5: repeat
 6:     $l = l+1$
 7:     repeat
 8:         Select a new ordered pair of nodes $i, j$ that are adjacent in $C$ such that $|adj(C,i) \setminus \{j\}| \geq l$
 9:         repeat
10:             Choose new $\boldsymbol{k} \in adj(C,i) \setminus \{j\}$ with $|\boldsymbol{k}| = l$ .
11:             if $i$ and $j$ are conditionally independent given $\boldsymbol{k}$ then
12:                 Delete edge $i, j$
13:                 Denote this new graph by $C$
14:                 Save $\boldsymbol{k}$ in $S(i, j)$ and $S(j, i)$
15:             end if
16:         until edge $i, j$ is deleted or all $\boldsymbol{k} \in adj(C,i) \setminus \{j\}$ with $|\boldsymbol{k}| = l$ have been chosen
17:     until all ordered pairs of adjacent nodes $i$ and $j$ such that $|adj(C,i) \setminus \{j\}| \geq l$ and $\boldsymbol{k} \in adj(C,i) \setminus \{j\}$ with $|\boldsymbol{k}| = l$ have been tested for conditional independence
18: if edge $i, j$ is a path constraint then
19:     Delete edge $i, j$
20: end if
21: until for each ordered pairs of adjacent nodes $i, j$: $|adj(C,i) \setminus \{j\}| < l$ .
```

Fig. 4. The pseudocode to determine skeleton under path constraints.

Algorithm 2 Extending the skeleton to a CPDAG under direction constraints

```
 1: INPUT: Skeleton $G_{skel}$, separation sets $S$
 2: OUTPUT: CPDAG $G$
 3: for all pairs of non-adjacent nodes $i, j$ with common neighbour $k$ do
 4:     if $k \notin S(i, j)$ then
 5:         Replace $i$—$k$—$j$ in $G_{skel}$ by $i \rightarrow k \leftarrow j$
 6:     end if
 7: end for
 8: for all pairs of direction constraint nodes $i, j$ with common neighbour $k$ do
 9:     if $k \notin S(i, j)$ then
10:         Orient $i$—$j$ into $i \rightarrow j$ whenever $i, j$ are adjacent.
11:         Orient $k$—$j$ into $k \rightarrow j$ whenever there is a chain $i \rightarrow k$—$j$.
12:         Orient $i$—$k$ into $i \rightarrow k$ whenever there is a chain $i$—$k \rightarrow j$.
13:     end if
14: end for
15: In the resulting PDAG, try to orient as many as possible by repeated application of the
following four rules:
16: R1 Orient $j$—$k$ into $j \rightarrow k$ whenever there is an arrow $i \rightarrow j$ such that $i$ and $k$ are non-adjacent.
17: R2 Orient $i$—$j$ into $i \rightarrow j$ whenever there is a chain $i \rightarrow k \rightarrow j$.
18: R3 Orient $i$—$j$ into $i \rightarrow j$ whenever there are two chains $i$—$k \rightarrow j$ and $i$—$l \rightarrow j$ such that $k$ and $l$ are
non-adjacent.
19: R4 Orient $i$—$j$ into $i \rightarrow j$ whenever there are two chains $i$—$k \rightarrow l$ and $k \rightarrow l \rightarrow j$ such that $k$ and $l$ are
non-adjacent.
```

Fig. 5. The pseudocode to extend the skeleton to a CPDAG under direction constraints.

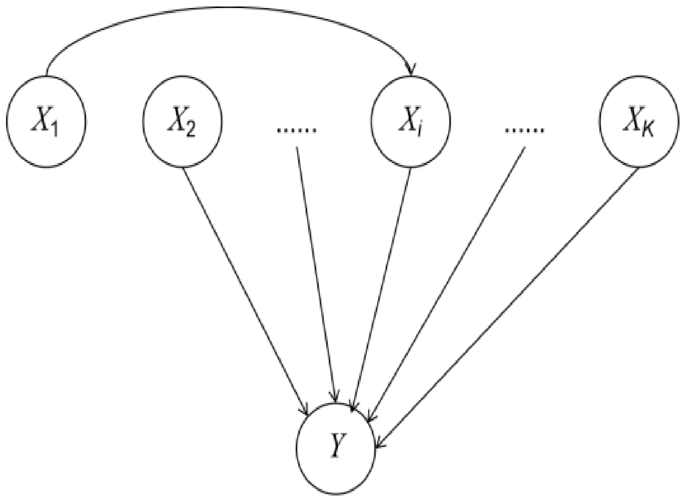


Fig. 6. A causal network example of $\{X_1, X_2, ..., X_K, Y\}$.

Assume that the causal network of $\{X_1, X_2, ..., X_K, Y\}$ obtained by the PC algorithm with empirical constraints is shown in Figure 6. In this section, to optimize the initial node groups in BLS, causal inference is carried out based on the causal network for exploring the causal effects of $X_1, X_2, ..., X_K$ on Y. In recent years, as causality has been widely concerned for exploring the interpretability of data-driven models, there have been a lot of studies on causal inference 18. In this work, the SCM 2021, which is simple and effective, is used to calculate the impact of monitoring features on the health state of the system. SCM usually assumes that the target node is directly affected by a self random variable in addition to the ancestor nodes in the causal network. For causal network in Figure 6, the causal effects of $X_1, X_2, ..., X_K$ on Y can be described by Eqs. 17 and 18:

$$P(Y=1) = f_Y(X_2, \ldots, X_{i-1}, f_{X_i}(X_1, z_{X_i}), X_{i+1}, , \ldots, X_K, z_Y) \quad (17)$$

$$X_j = f_{X_j}(z_{X_j}), j = 1,2, \ldots, i-1, i+1, \ldots, K \quad (18)$$

where $z_{X_1}, z_{X_2}, \ldots, z_{X_K}, z_Y$ are respectively random variables that directly affect $X_1, X_2, ..., X_K, Y$, which are often considered as white noises and omitted when training SCM. $f_{X_s}(*), s = 1,2, \ldots, K$ and $f_Y(*)$ are the causal models that generate $X_1, X_2, ..., X_K$ and Y.

When training the SCM (the causal models) based on actual data, different mapping functions can be selected to improve the mapping effect, including linear function, power function, exponential function, etc. Similarly, set intervals for all parameters in the SCM, the initial node groups in BLS can be optimize by randomly generating feature maps with different parameter combinations. Take linear functions as an example, after trained by the input data (***X***, ***Y***), the SCM of $X_1, X_2, \ldots, X_K$ on Y is:

$$P(Y=1) = a_2\mathbf{X}_2 + \ldots a_{(i-1)}\mathbf{X}_{i-1} + a_i(a_1\mathbf{X}_1) + a_{(i+1)}\mathbf{X}_{i+1} + \ldots + a_K\mathbf{X}_K \quad (19)$$

where a_i is the casual effect of X_i on Y.

Thus, the initial node groups $\boldsymbol{Z}_1, \boldsymbol{Z}_2, \ldots, \boldsymbol{Z}_n$ can be calculated by Eqs. 20 and 21:

$$\mathbf{Z}_i = [\mathbf{Z}_{i1}, \mathbf{Z}_{i2}, \ldots, \mathbf{Z}_{iv}], i = 1,2,\ldots,n \quad (20)$$

$$\mathbf{Z}_{ij} = a_{2j}\mathbf{X}_2 + \ldots a_{(i-1)j}\mathbf{X}_{i-1} + a_{ij}\left(a_{1j}\mathbf{X}_1\right) + a_{(i+1)j}\mathbf{X}_{i+1} + \ldots + a_{Kj}\mathbf{X}_K, j = 1,2,\ldots,v \quad (21)$$

here Z_i is the i-th initial node group in Z^n, Z_{ij} is the j-th feature node in Z_i. And, a_{ij} is randomly obtained from the casual effect interval $[a_i-\varepsilon,\ a_i+\varepsilon]$, in where ε is also a range parameter constrained by existing experience as shown in section 2.2.1.

2.2.3. Some other details

To improve the model robustness, sparse matrix technique is used for the initial node groups in the proposed methods. The initial node groups obtained based on LR or SCE are processed by sparse matrix technique to make the differences between nodes more significant.

Moreover, as verified in many existing studies, the causality is more stable than the correlation, thus the initial node group generation method based on causality may be more stable and conducive to the analysis of fault path. But, the data information used in SCM is often incomplete as shown in Eq. 19. Considering that the actual application scenarios are complex and changeable, which makes it impossible for a single method to guarantee the best results contiguously, this paper integrates the MBLSs based on regression model and causality to get the final system condition detection model, so as to improve the accuracy and robustness of fault detection.

3. Analysis on applicability

Since few heterogeneous complex system monitoring data is publicly available, and method applicability analysis needs based on diversified data, so various simulation data are generated in this section. According to Eq. 22 (called generating equation group), simulation data of the sub-application of an integrated modular avionics (IMA) software platform in Figure 7 can be generated.

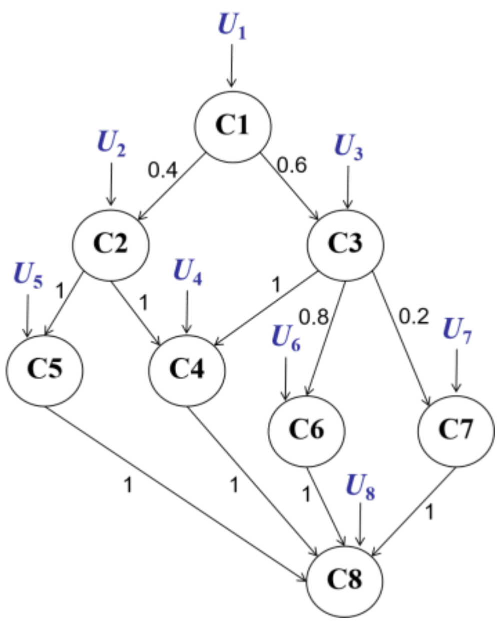


Fig. 7. The architecture of the sub-application.

$$\begin{cases} C1 = U_1 \\ C2 = U_2 + a_{12}C1 \\ C3 = U_3 + a_{13}C1 \\ C4 = U_4 + a_{24}C2 + a_{34}C3 \\ C5 = U_5 + a_{25}C2 \\ C6 = U_6 + a_{36}C3 \\ C7 = U_7 + a_{37}C3 \\ C8 = U_8 + a_{48}C4 + a_{58}C5 + a_{68}C6 + a_{78}C7 \end{cases} \quad (22)$$

where C1 is the GPS and INS data analysis module, C2 and C3 are separately the GPS and INS data correction modules, C4 is the data fusion module, C5 and C6 are separately the GPS and INS data calculation modules, C7 is a backup of C6, C8 is the subsystem status output and display module, a_{ij} is the edge weight (marked in Figure 7) of Ci to Cj, U_i is an exogenous variable which is related to the health state of Ci. Considering that there is no causal relationship between the health status of the system components, U_i is assumed to be white noise with mean value being 0 and variance being σ_i.

With $(\sigma_1, \sigma_2, \ldots, \sigma_8) = (4, 3, 3, 3, 2.5, 4, 1.5, 1)\times10$^(-2), and the fault threshold for C8 being 0.1, 10000 samples are generated as the sampling population. It is supposed that when C8 is greater than 0.1, the sub-application shows system failure. By using PC algorithm, as shown in Figure 8(a), the causal network of the simulation data is consistent with the architecture of the sub-application. And, when simulate the imbalance monitoring in practical applications, the causal network of the sampled simulation data is obtained (as shown in Figure 8(b)). Due to the simple mechanism of system modules, the gap between the two causal networks is small.

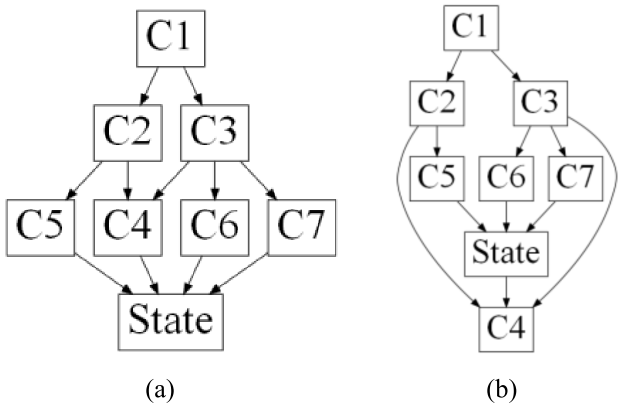


Fig.8. The causal network of the simulation data.

Based on the resampled simulation data (recorded as simulation data-1), under 5-fold cross-validation, the performances of the proposed Modified-BLSs (MBLSs) and traditional BLSs (the basic BLS and the BLSs respectively combined with PCA and sparse auto-encoder (SAE)) are shown in Table 1.

Table 1. Comparison performances of different BLSs on simulation data-1

Models	*Average Precision*	*Average Recall*	*Training time(s)*
MBLS1	0.95568	0.83333	0.136
MBLS2	0.94737	0.82609	0.123
BLS	0.94096	0.81587	0.159
PCA-BLS	0.95203	0.81470	0.146
SAE-BLS	0.94613	0.82750	0.157

It can be seen that the performances of different BLS models are similar. It is because of that the basic BLS with a fairly simple structure can well extract the fault information from simulation data-1. Here, MBLS2 is based on the causal network in Figure 8(b). Under the assumed completely linear data, the feature information is closely interconnected, so MBLS2 can have a good performance. In fact, generally, if and only if a more practical causal network is obtained, the causality based MBLS (i.e. MBLS2), which follows actual and stable causal effects, can be accurate and stable. When prior knowledge is limited, MBLS2 will contain incomplete feature information. In this case, the LR based MBLS (i.e. MBLS1) can extract information more comprehensively. Therefore, the two different methods are integrated in complex system condition monitoring.

For comparing the performances of different BLSs on high-dimensional data, a sparse weighted matrix is designed to represent the influence mechanism between K components of a virtual complex system (as shown in Eq. 23). The matrix $\boldsymbol{A}$ is set as almost an upper triangular matrix, and the last component C_K represents the system fault state.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1K} \\ a_{21} & a_{22} & \cdots & a_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ a_{K1} & a_{K2} & \cdots & a_{KK} \end{bmatrix} \tag{23}$$

where a_{ij} is the influence of component Ci on component Cj.

Similarly, the feature value of a component Ci is affected by its parent node components and an exogenous variable U_i (related to its own state). Assuming that the causal effects between the components are all linear functions, then the K-dimension monitoring data of the virtual complex system can be obtained by Eq. 24.

$$\mathbf{C} = [C_1, C_2, \ldots, C_K] = (\mathbf{I} - \mathbf{A}')^{-1}\mathbf{U} \tag{24}$$

To make the generated data not completely linear, few nonlinear mappings are added to the generating equation group. And, several component features of the generated data are piecewise discretized to simulate the heterogeneous monitoring data of the complex system. When K is 10, 20, 50 and 100 respectively, under 5-fold cross-validation, the comparison performances of different BLSs are shown in Tables 2, 3, 4 and 5. In addition, standard deviations of the generalization *Precision* and *Recall* values under 5-fold cross-validation are also compared to roughly analyze the model stability. The overall samples of generated data are used directly in this step. The simulation datasets 2 to 5 are all clean and balanced, leading to that each model can achieve an ideal effect by adding hidden layer nodes. Thus, the model structure optimization will be stopped once the model *average precision* and *average recall* reach above 0.9.

Table 2. Comparison performances of different BLSs on simulation data-2.

Models	*Average Precision*	*Average Recall*	*Training time(s)*	*standard deviation of Precision*	*standard deviation of recall*
MBLS1	0.91561	0.91013	**0.181**	0.00744	0.00379
MBLS2	0.91319	0.91143	**0.175**	0.00337	0.00603
BLS	0.91060	0.90988	**0.184**	0.01422	0.00983
PCA-BLS	0.91123	0.90731	**0.172**	0.01426	0.00906
SAE-BLS	0.90988	0.91039	**0.197**	0.01018	0.00852

Table 3. Comparison performances of different BLSs on simulation data-3.

Models	***Average Precision***	***Average Recall***	***Training time(s)***	***standard deviation of Precision***	***standard deviation of recall***
MBLS1	0.93665	0.93419	**0.235**	0.00369	0.00749
MBLS2	0.93843	0.93839	**0.230**	0.00390	0.00512
BLS	0.93542	0.93352	**0.589**	0.01281	0.01076
PCA-BLS	0.93416	0.93099	**0.462**	0.01059	0.00888
SAE-BLS	0.93927	0.93748	**0.609**	0.01080	0.01355

Table 4. Comparison performances of different BLSs on simulation data-4.

Models	***Average Precision***	***Average Recall***	***Training time(s)***	***standard deviation of Precision***	***standard deviation of recall***
MBLS1	0.94645	0.93866	**0.429**	0.00802	0.00772
MBLS2	0.95068	0.94436	**0.350**	0.00531	0.00465
BLS	0.94313	0.93594	**1.926**	0.00904	0.01420
PCA-BLS	0.93749	0.94159	**1.315**	0.00743	0.01493
SAE-BLS	0.94309	0.93500	**1.977**	0.01372	0.01083

Table 5. Comparison performances of different BLSs on simulation data-5.

Models	***Average Precision***	***Average Recall***	***Training time(s)***	***standard deviation of Precision***	***standard deviation of recall***
MBLS1	0.94075	0.93734	**0.747**	0.00980	0.00974
MBLS2	0.94194	0.94329	**0.545**	0.00947	0.00704
BLS	0.94345	0.93508	**7.873**	0.01176	0.01760
PCA-BLS	0.93837	0.93750	**5.781**	0.01205	0.01328
SAE-BLS	0.94057	0.93616	**8.193**	0.01369	0.01579

It can be seen that when the data dimension increases, traditional BLSs (BLS, PCA-BLS and SAE-BLS) need to sacrifice training efficiency to get the models that meet the requirements. And, the proposed MBLSs can maintain high efficiency on high-dimensional data. This shows that the proposed MBLSs are more applicable to high-dimensional complex data, which is consistent with complex system application scenarios. When the data complexity is relatively low, traditional BLS is sufficient. There is no exact data complexity threshold, it is necessary to judge when to use MBLSs according to specific application scenarios and model performance.

In fact, the complexity of simulation data is far less than that of real data, the improvement made by MBLSs can be more significant in real data. In addition, the simulation data is clean and evenly distributed, so the stability of the MBLSs can not be well reflected in this part. The above two issues will be further verified in the next Section.

4. Case study

Since designed according to the fault-oriented safety principle, HST brake control system is a typical complex system with multiple braking schemes. And, condition monitoring of the HST brake control system have always been one of the key technologies for safe transportation 4. In this section, using the actual monitoring data of a high-speed train (HST) brake control system within one-year operation, a comparative experiment is carried out to verify the effectiveness of the proposed two MBLSs and condition monitoring method (i.e. the Mix-MBLS). The HST brake control system and its monitoring dataset have not yet been publicly available. Under the confidentiality agreement, this paper only provides a rough description of the system structure and the monitoring data.

The high-speed train brake control system includes pneumatic and electric devices, and combines various devices by microelectronics technology to achieve different functions. There are a variety of factors that can affect the health state of the system. To comprehensively monitor these factors, sensors were installed at key locations based on expert experience, and the operation and environmental variables were recorded. Specifically, monitoring tools have been set up at key locations in the circuit system to collect information such as voltage, current, and temperature. Sensors have been set up in key components of the mechanical system to collect vibration signals, temperature, power, etc. And, state variables such as braking mode, traction level, train speed, position, and running time displayed by the software system, as well as environmental variables such as external temperature, humidity and weather are recorded. The collected raw monitoring data contains 38 variables that may be related to the system state. To comply with the confidentiality agreement, these variables cannot be listed, and are only denoted as V_1, V_2. . . V_{38}.

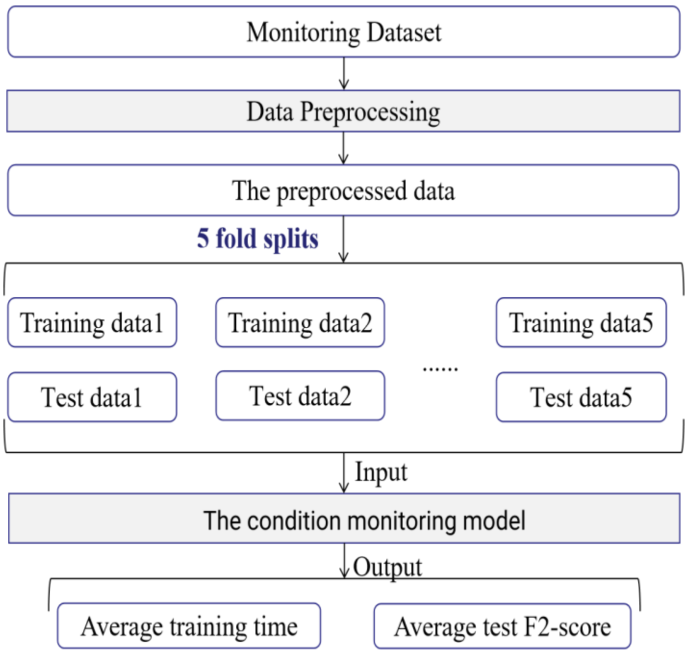


Fig. 10. The rough process to verify the effectiveness of the MBLSs.

1 The rough process to verify the effectiveness of the MBLSs is shown in Fig.10. To avoid the influence of random factors, average results under 5-fold cross-validation are concerned. The commonly adopted deep learning methods, back propagation neural network (BPNN) 34 and CNN 31, as well as the basic BLS and feature extraction-based BLSs are considered as benchmark condition monitoring models. The feature extraction-based BLSs include PCA-BLS and SAE-BLS, i.e. the BLS with PCA and sparse auto-encoder (SAE), which are commonly used and often have good effects. Here, the cumulative contribution rate (CCR) selected in PCA is 85%, the SAE uses *Sigmoid* function as the activation function and contains one hidden layer.

2 Moreover, advanced deep learning methods, which consider data structure information, are also used as comparison groups, including the correlation graph convolutional network (Corr-GCN) 29 and the causal graph convolutional network (Cal-GCN) 12. The Corr-GCN is an undirected graph convolutional network (GCN) based on the correlation network of the monitoring variables. The the correlation network is obtained by setting a threshold value (i.e. 0.5) for correlation coefficients, i.e. if and only if the correlation coefficient of V*i* and V*j* is greater than 0.5, there is an edge between V*i* and V*j*. The Cal-GCN is a directed GCN based on the causal network of the monitoring variables.

3 In addition, the structures of all the condition monitoring models are optimized by progressively adding layers, nodes and iterations until the generalization performance remains stable or decreased. And, initial hyperparameters of each model are set based on modeling experiences. The basic BLSs are initially set to contain 20 initial node groups with 10 nodes in each group, 10 characteristic node groups with 5 nodes in each group, and one incremental node group with 10 nodes. NNs (BPNN and CNN) are initially set to contain 3 hidden-layers with common used node parameters (8, 16, 32, etc.), and trained 100 times. And, GCNs (Corr-GCN and Cal-GCN) are initially set to contain 2 graph convolutional layers for structural information extraction, and 1 fully connected layer for fusing graph information.

4.1. Data description

There are 21 continuous numerical variables and 17 discrete category variables in the monitoring data. Firstly, these discrete category variables need to be converted to numerical values, by recording “TURE” as 1 and “FALSE” as 0 for the binary variables, and label coding the hierarchical variables from low to high. Then, *Z-score* standardization is considered to normalize continuous variables. Moreover, all faults are marked as abnormal states to improve the fault detection rate. The system state, i.e. the sample label is denoted as Y, with abnormal state as 1 and normal state as 0.

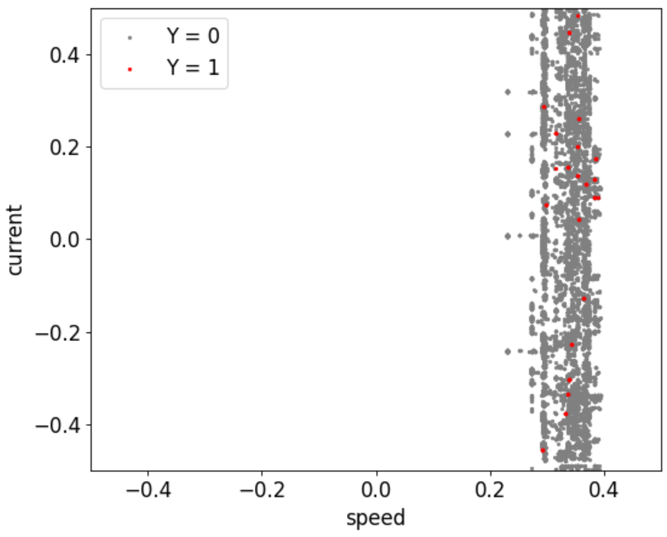


Fig. 9. The scatter diagram of (*speed*, *current*) after data preprocessing.

After the above data preprocessing, there are 15982 normal samples and 287 fault ones, which shows that the preprocessed data is seriously imbalanced. And, from Figure 9, the scatter diagrams of (*speed*, *current*)), it can be seen that there is a strong intersection between the variable values of fault samples and normal ones. In fact, the values of other variables also have strong intersection. Due to the highly imbalanced and class-overlapping of the monitoring data, *F*2-*score* is applied to compare the prediction effects of different classification models in this work.

$$F - score = (1+\beta^2)\frac{precision*recall}{(\beta^2*precision)+recall} \tag{25}$$

$$precision = TP/(TP+FP) \tag{26}$$

$$recall = TP/(TP+FN) \tag{27}$$

where β is the weight adjustment index. When $\beta = 1$, the weight of *recall* is the same as that of *precision*. And, when $\beta > 1$, the weight of *recall* is higher than that of *precision*. Actually, the *recall* of abnormal state for HST brake control system is more important, thus we set $\beta = 2$ and use *F2-score* to compare the models. *TP*, *TN*, *FP*, *FN* respectively represent the number of true positive, true negative, fault positive and fault negative.

$$F2 - score = 5 * \frac{precision*recall}{(4*precision)+recall} \tag{28}$$

4.2. Condition monitoring based on Modified-BLSs

Using the feature extraction mechanism in Section 2.2.1, with 18 PCA factors extracting from 38 original features by setting the CCR as 85%, after using the LR algorithm and transforming the PCA factors, the relationship between $\boldsymbol{V}$ and $\boldsymbol{Y}$ can be described as the regression equation shown in Eq. 29. Due to the large number of variables, here we do not show all regression coefficient values in detail.

$$ln(\frac{P(Y=1)}{1-P(Y=1)}) = \beta_0 + \sum_{i=1}^{38}\alpha_i V_i = -7.363 + 0.223V_1 - 0.003V_2 + \ldots\ldots -0.011V_{37} + 0.008V_{38} \tag{29}$$

Thus, the MBLS-1 model can be obtained by constraining the value range of weight coefficient matrix $\boldsymbol{W}_{e\,i}$ and offset matrix $\boldsymbol{\beta}_{e\,i}$.

The original 38 variables can be divided into three categories: objective features (including *environmental factors* and *operation duration*), human influence features (braking model, *braking level*, *reaction time*, etc) and system monitoring features (*current*, *voltage*, *braking status*, *power*, *frequency*, etc). According to practical experiences, objective features and human influence features are the ancestral nodes of system monitoring features, and there is no direct causal relationship between objective features and human influence features. Based on the above assumptions, the causal network of $\{V_1, V_2, \ldots, V_{38}, Y\}$ obtained by PC algorithm with empirical constraints is shown in Figure 11.

From the causal network in Figure 11, the SCE model shown in Eq. 30 can be used to describe the causal relationship.

$$P(Y=1) = f_Y(V_4, V_5, V_{12}, V_{18}, V_{23}, V_{24}, V_{33}, V_{35}, V_{36}, V_{38}, \varepsilon_Y) \tag{30}$$

To fully extract the dataset information, all the ancestor nodes of Y are traced layer by layer and used interactively to predict $P(Y=1)$. $f_Y(*)$ in Eq. 30 is also the *Sigmoid* function. Considering that the influence of time, temperature and humidity on the system failure rate generally shows a gradual upward trend of growth rate, their causal effects are set as exponential or square function. Other causal mapping functions are randomly selected in linear and nonlinear. In this case, 5 feature extraction equations which are the first 5 best fitting Y can be obtained by ergodic method, and each equation is the basis for extracting one initial node group. Similarly, the value intervals can be set for the parameters to extract initial nodes in each group. Thus, the MBLS-2 model can be obtained based on these initial node groups.

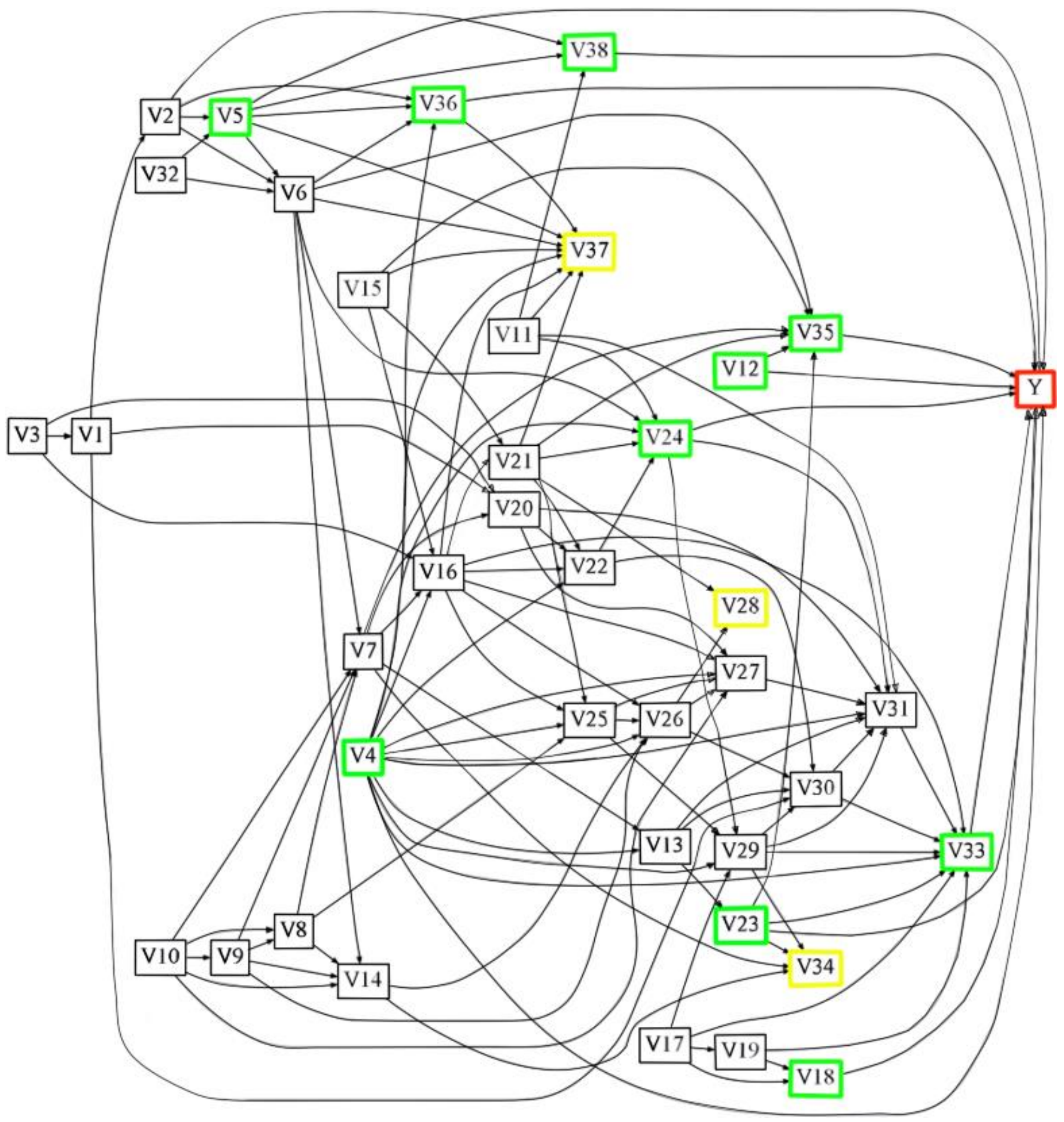


Fig. 11. The causal network of the HST brake control system(with green box representing parent node of *Y* and yellow box representing invalid node in this case).

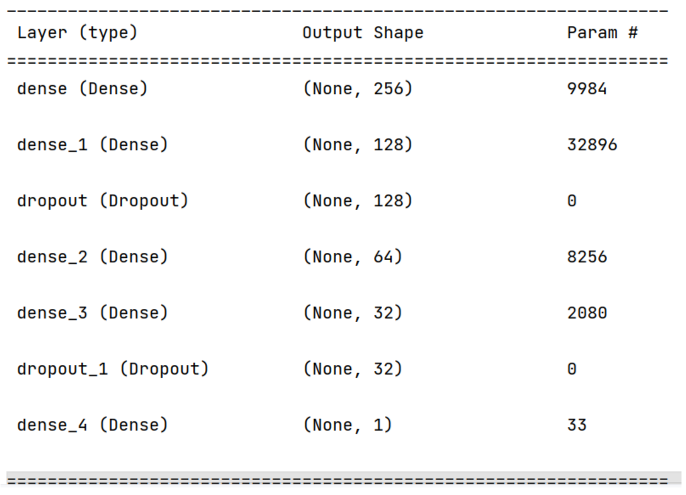

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 256)               9984

 dense_1 (Dense)             (None, 128)               32896

 dropout (Dropout)           (None, 128)               0

 dense_2 (Dense)             (None, 64)                8256

 dense_3 (Dense)             (None, 32)                2080

 dropout_1 (Dropout)         (None, 32)                0

 dense_4 (Dense)             (None, 1)                 33

=================================================================
```

Fig. 12. The final structure of the BPNN.

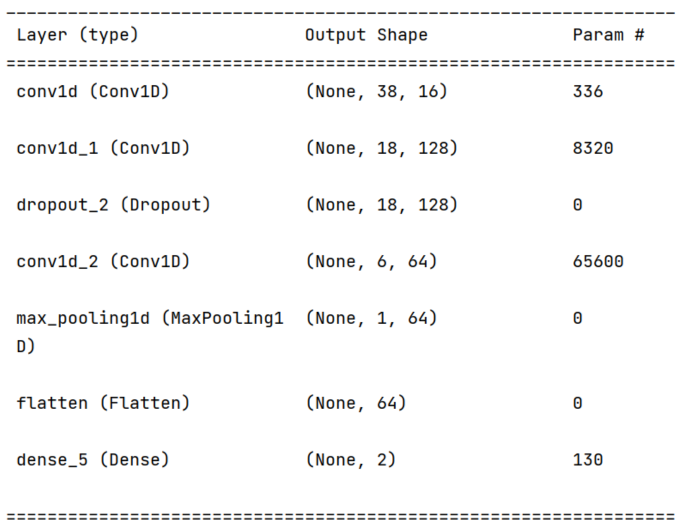

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 38, 16)            336

 conv1d_1 (Conv1D)           (None, 18, 128)           8320

 dropout_2 (Dropout)         (None, 18, 128)           0

 conv1d_2 (Conv1D)           (None, 6, 64)             65600

 max_pooling1d (MaxPooling1  (None, 1, 64)             0
 D)

 flatten (Flatten)           (None, 64)                0

 dense_5 (Dense)             (None, 2)                 130

=================================================================
```

Fig. 13. The final structure of the CNN.

The initial structures of MBLS-1 and MBLS-2 include 5 initial node groups with 3 nodes, 5 characteristic node groups with 3 nodes, and one incremental node group with 5 nodes. And, to make all possible faults be detected, the prediction result of the Mix-MBLS model is "1" if at least one of the two proposed MBLS model output is "1". By manually adjusting the model hyperparameters, the optimized BPNN and CNN are shown in Figures 12 and 13. And, GCNs have powerful feature extraction capabilities, the optimized Corr-GCN and Cal-GCN both contain 2 graph convolutional layers and 2 fully connected layers.

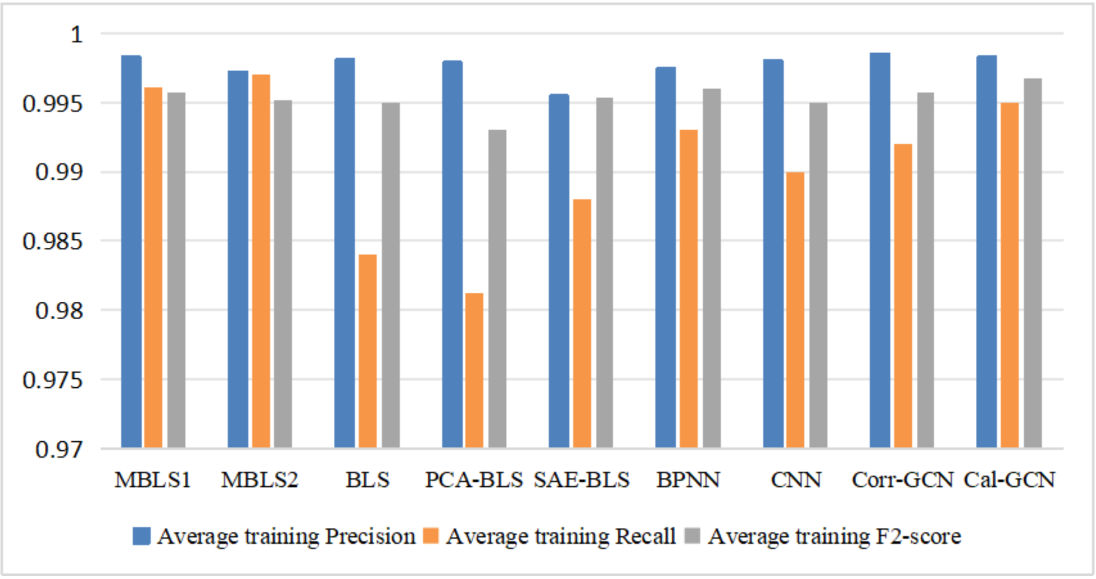


Fig. 14. Comparison training accuracy of different models.

Under the same operating environment and with 5-fold cross-validation test data, the average training effects of different models are shown in Figure 14, and the average generalization performances are shown in Table 6. It can be seen that each model has been well trained, which can be considered that there is no under-fitting problem. Since we optimized the model structures according to model generalization performances, there is no over-fitting problem in each model (This can also be seen from Figure 14).

Table 6. Comparison generalization performances of different models.

Models	*Average Precision*	*Average Recall*	*Average F2-score*	*Training time(s)*	*standard deviation of F2-score*
MBLS1	0.93333	0.93241	0.93648	**0.641**	**0.00401**
MBLS2	**0.94627**	0.94816	0.93835	**0.813**	**0.00199**
Mix-MBLS	0.94064	**0.95242**	**0.94509**	**0.931**	**0.00268**
BLS	0.87876	0.90134	0.88941	21.474	0.01491
PCA-BLS	0.89078	0.89527	0.89136	19.622	0.01428
SAE-BLS	0.89780	0.90505	0.89296	22.562	0.01869
BPNN	0.92585	0.90667	0.91219	357.584	0.01823
CNN	0.93750	0.92743	0.92604	579.739	0.01701
Corr-GCN	0.92652	0.91547	0.91733	2205.882	0.00912
Cal-GCN	0.91332	0.92487	0.92034	1433.393	0.00758

According to the results of the comparative experiment, we can see that:

1) The deep learning methods (BPNN, CNN, Corr-GCN and Cal-GCN) can deal with the data imbalance problem of this case and achieve good prediction performances. But, the model training process is time-consuming.

2) GCNs (especially Cal-GCN) that consider data structure information have high stability, but the training process is much more time-consuming.

3) By increasing the number of hidden-layer nodes, the

basic BLS may reach a high accuracy, but the training time is increased, making BLS unable to realize real-time training and updating.

4) Feature extraction and dimensionality reduction for the input data based on common feature extraction methods (PCA and SAE) can improve the model performance, but the reduction on training efficiency is very limited.

5) The proposed methods can significantly improve the mode training efficiency, which is reflected in that MBLSs can be retrained and updated online in complex system condition monitoring.

6) The performances of the proposed MBLSs, i.e. *F2-score*, *precision* and *recall*, are better than the deep learning methods. At the same time, the training time of each MBLS is much lower. It shows that the condition monitoring method based on Mix-MBLS can meet the requirements of real-time update and accurately prediction.

7) Based on the proposed MBLSs, the training and generalization *recall* measures have been significantly improved, which indicates that the improved models can identify fault information more effectively. At the same time, the standard deviation of *F2-score* has been significantly reduced, which means that MBLSs and MiX-MBLS have higher stability in this case.

8) In addition, the prediction fault set of the MiX-MBLS is just the union of the true positive and the fault positive samples in the two MBLSs. Thus, the *recall* of the MiX-MBLS is improved, while the *precision* is slightly decreased. This means that the integrated model by parallel fusion can diagnose faults more conservatively to ensure the system safety.

4.3. Transferabil

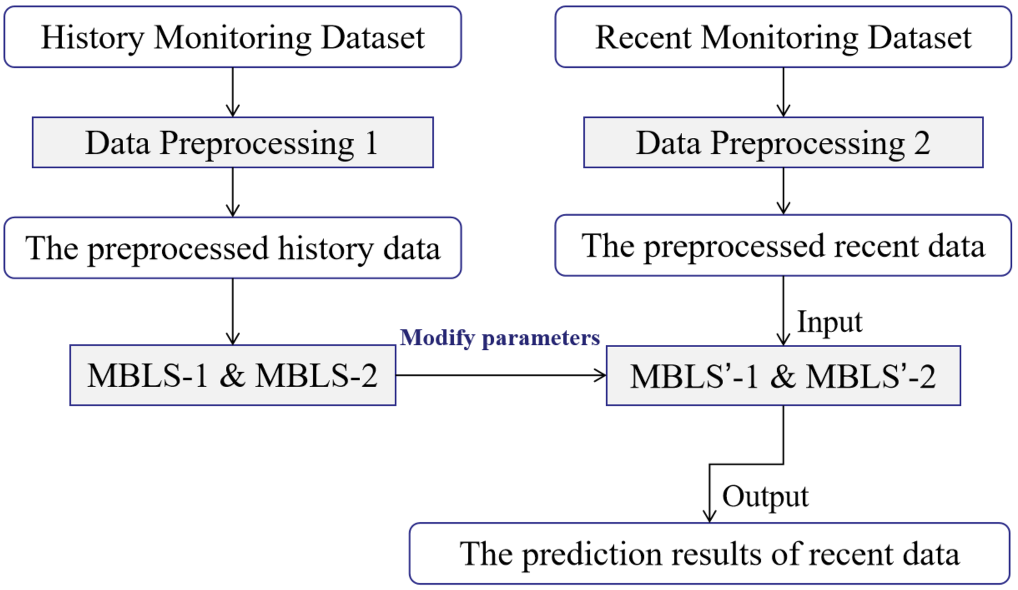

History Monitoring Dataset
Data Preprocessing 1
The preprocessed history data
MBLS-1 & MBLS-2
Modify parameters
Recent Monitoring Dataset
Data Preprocessing 2
The preprocessed recent data
Input
MBLS'-1 & MBLS'-2
Output
The prediction results of recent data

Fig. 15. The process to verify the transferability.

The main improvement of the proposed BLS lies in the feature extraction efficiency of the initial node groups. Benefit from the adopted LR and structural causal equation, the initial node groups have high interpretability. Compared with the traditional BLS, the MBLSs proposed in this paper can provide some basis for model migration and optimization. When the operating environment changes, the state monitoring model can be easily adjusted to meet new scenarios. Limited by the available data, an example on the changing of operation time is considered to verify the transferability of the propose methods.

Assuming that the data monitoring is intermittent, we want to use the historical data monitored a long time ago to build the current system condition monitoring model. This means that the degradation state and risk level of the system have significantly changed, and will be more susceptible to external environmental factors. Based on knowledge transfer, the migration models are built by fine-tuning the parameters in the original model. Concretely, fine-tuning the coefficients of environmental factors in the LR equation or the structural causal equation. How to adjust environmental factors and which environmental factors need to be adjusted are all judged by experts' experience. After the migration models are officially put into use, they can be further optimized according to the performance.

Based on the MBLSs constructed from the operation data of the previous half year, the migration models (MBLS'-1 and MBLS'-2) can be obtained by increasing the weights of environmental factors in the initial node groups. The weights of environmental factors in the initial node groups are increased by changing the absolute values of the coefficients of environmental variables (temperature, humidity and wind speed) in Eqs. 29 and 30. And, the monitoring data of the last month of the year is used as the test set to verify the effectiveness of the migration models.

	n (Y ^_MBLS1=0)	*n* (Y ^_MBLS1=1)	*n* (Y ^_MBLS2=0)	*n* (Y ^_MBLS2=1)	*n* (Y ^_Mix-MBLS)	*n* (Y ^_Mix-MBLS)
n (Y _true=0)	1365	1	1365	1	1364	0
n (Y _true=1)	2	3	2	3	2	4

(a) The confusion matrices of the original models

	n (Y ^_MBLS'1=0)	*n* (Y ^_MBLS'1=1)	*n* (Y ^_MBLS'2=0)	*n* (Y ^_MBLS'2=1)	*n* (Y ^_Mix-MBLS')	*n* (Y ^_Mix-MBLS')
n (Y _true=0)	1365	1	1366	0	1365	1
n (Y _true=1)	1	4	1	4	0	5

(b) The confusion matrices of the migration models

Fig. 14. Comparison of the number of different samples in the prediction results.

It can be seen that the performance of the migration models is better than that of the original models. By increasing the weight of environmental factors, the model is adaptive to the change of environments. Additionally, the samples collected in the emergence of a new environment can also be added to the migration MBLSs as incremental node groups to further optimize the models. In conclusion, interpretable diagnostic methods have higher application value in practice.

5. Conclusions

The timely and accurate condition monitoring for complex systems can greatly reduce the risk of potential faults. In this paper, a condition monitoring model is established based on broad learning system (BLS). The hidden-layer nodes in classical BLS need to be large enough to achieve satisfactory prediction accuracy in some scenarios, two Modified-BLS (MBLSs) are proposed in this work to improve the feature extraction efficiency of the initial node groups in BLS. In details, the LR equation and the structural causal equation are used to replace the totally random generation of the initial node groups. While a single model is often impossible to guarantee the prediction performance, the integration of different models is considered. So, the proposed MBLSs are combined (noted as Mix-MBLS).Several simulation datasets are used to verify that the proposed method can improve the feature extraction efficiency of BLS for high-dimensional data. And, based on real monitoring data of a high-speed train brake control system, by comparing with the deep learning methods (BPNN, CNN, Corr-GCN and Cal-GCN) and traditional BLSs (BLS, PCA-BLS and SAE-BLS), it can be inferred that the proposed MBLSs can achieve higher accuracy in a much shorter time. And, Mix-MBLS can achieve even better prediction results with slightly increased training time. This shows that the proposed model can be adopted to realize the online monitoring and alarm for complex systems. Moreover, this work also improves the interpretability of the initial node groups which makes the proposed model be capable of migrating effectively and conveniently to a new environment. Future research will continue to focus on improving the efficiency and interpretability of the BLS model.

References

1. Chavan N, Kale M, Deshmukh S, et al. A Review on Development and Trend of Condition Monitoring and Fault Diagnosis[J]. ECS transactions, 2022,107(1):17863-17870. DOI: 10.1149/10701.17863ecst
2. Chen C L P, Liu Z. Broad learning system: A new learning paradigm and system without going deep[C]// 2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC). IEEE, 2017. DOI: 10.1109/YAC.2017.7967609.
3. Chen C L P, Liu Z. Broad learning system: An effective and efficient incremental learning system without the need for deep architecture[J]. IEEE Transactions on Neural Networks and Learning Systems, 2018, 29(1):10-24. DOI: 10.1109/TNNLS.2017.2716952.

4. Chen H, Jiang B. A Review of Fault Detection and Diagnosis for the Traction System in High-Speed Trains, IEEE Transactions on Intelligent Transportation Systems, 2020, 21(2):450-465. DOI: 10.1109/TITS.2019.2897583.
5. Chen H, Wang T, Chen T, et al. Hyperspectral Image Classification Based on Fusing S3-PCA, 2D-SSA and Random Patch Network[J]. Remote Sensing, 2023, 15(13):3402. DOI:10.3390/rs15133402.
6. David W H, Stanley L. Applied Logistic Regression[M], second edition. Wiley Press, 2005. DOI:10.1002/0471722146.
7. Du J, Vong C M, Chen C L P. Novel Efficient RNN and LSTM-Like Architectures: Recurrent and Gated Broad Learning Systems and Their Applications for Text Classification[J]. IEEE Transactions on Cybernetics, 2021, PP(99):1-12. DOI: 10.1109/TCYB.2020.2969705
8. Glymour C, Zhang K, Spirtes P. Review of Causal Discovery Methods Based on Graphical Models[J]. Frontiers in Genetics, 2019, 10:524-538. DOI: 10.3389/fgene.2019.00524.
9. Gong X, Zhang T, Chen C L P, et al. Research Review for Broad Learning System: Algorithms, Theory, and Applications[J]. IEEE Transactions on Cybernetics, 2021, PP(99):1-29. DOI: 10.1109/TCYB.2021.3061094.
10. Ji A, Woo W L, Wong W L E, et al. Rail track condition monitoring: a review on deep learning approaches[J]. Intelligence & Robotics, 2021, 1(2):151-175. DOI: 10.20517/ir.2021.14.
11. Jiang S B, Wong P K, Liang Y. A Fault Diagnostic Method for Induction Motors Based on Feature Incremental Broad Learning and Singular Value Decomposition[J]. IEEE Access, 2019, 7:157796-157806. DOI: 10.1109/ACCESS.2019.2950240.
12. Kong W, Qiu M, Li M, et al. Causal Graph Convolutional Neural Network For Emotion Recognition[J]. IEEE Transactions on Cognitive and Developmental Systems, 2022, PP(99):1-1. DOI:10.1109/TCDS.2022.3175538.
13. Kostrzewski M, Melnik R. Condition Monitoring of Rail Transport Systems: A Bibliometric Performance Analysis and Systematic Literature Review[J]. Sensors, 2021, 21(14):4710. DOI:10.3390/s21144710.
14. Kou L, Qin Y, Zhao X, et al. A Multi-dimension End-to-End CNN Model for Rotating Devices Fault Diagnosis on High Speed Train Bogie[J]. IEEE Transactions on Vehicular Technology, 2020, 69(3):2513-2524. DOI: 10.1109/TVT.2019.2955221
15. Li J, Shi J. Knowledge discovery from observational data for process control using causal Bayesian networks[J]. Iie Transactions, 2007, 39(6):681-690. DOI: 10.1080/07408170600899532.
16. Li T, Fang B, Qian J, et al. CNN-Based Broad Learning System[C]// 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP), Wuxi, China, 2019:132-136, DOI: 10.1109/SIPROCESS.2019.8868769.
17. Li Xn, Zhao H, Deng W. BFOD: Blockchain-based Privacy Protection and Security Sharing Scheme of Flight Operation Data[J]. IEEE Internet of Things Journal, 2023, PP:1-1, DOI:10.1109/JIOT.2023.3296460.
18. Marloes H M, Preetam N. A review of some recent advances in causal inference[J]. arXiv, 2015. DOI: 10.48550/arXiv.1506.07669.
19. Maurya S, Singh V, Verma N K. Condition Monitoring of Machines using Fused Features from EMD based Local Energy with DNN[J].IEEE Sensors Journal, 2019, PP(99):1-1. DOI:10.1109/JSEN.2019.2927754.
20. Pearl J. Causality: Models, Reasoning, and Inference [M], second edition. Cambridge University Press, 2000. DOI:10.1111/1468-0297.13919.
21. Pearl J. Graphs, Causality, and Structural Equation Models[J]. Sociological Methods & Research, 1998, 27(2):226-284. DOI: 10.1177/0049124198027002004.
22. Spirtes P, Clark G. An Algorithm for Fast Recovery of Sparse Causal Graphs[J]. Social Science Computer Review, 1991:62-72. DOI: 10.1177/089443939100900106.
23. Tang R, De D L, Besinovic N, et al. A literature review of Artificial Intelligence applications in railway systems[J]. Transportation research Part C: Emerging technologies, 2022, Jul.(140):103679.1-103679.25. DOI:10.1016/j.trc.2022.103679.
24. Wang S, Xiang J. A minimum entropy deconvolution-enhanced convolutional neural networks for ault diagnosis of axial piston pumps[J]. Soft Computing, 2020, 24(20): 2983-2997. DOI: 10.1007/s00500-019-04076-2.
25. Wang S, Xiang J, Zhong Y, et al. A data indicator-based deep belief networks to detect multiple faults in axial piston pumps[J]. Mechanical Systems and Signal Processing, 2018, 112:154-170. DOI: 10.1016/j.ymssp.2018.04.038.
26. Wei S, Li J, Zhao Z, et al. Artificial Monitoring of Eccentric Synchronous Reluctance Motors Using Neural Networks[J]. Computers, Materials, & Continua, 2022, 000(007):1035-1052. DOI:10.32604/cmc.2022.024201.
27. Yang F. A CNN-Based Broad Learning System[C]// 2018 IEEE 4th International Conference on Computer and Communications (ICCC).

Chengdu, China: IEEE, 2018:2105-2109. DOI: 10.1109/CompComm.2018.8780984.

28. Zhang J. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias[J]. Artificial Intelligence, 2008, 172(16-17):1873-1896. DOI:10.1016/j.artint.2008.08.001.
29. Yu X, Tang B, Zhang K. Fault Diagnosis of Wind Turbine Gearbox Using a Novel Method of Fast Deep Graph Convolutional Networks[J]. IEEE Transactions on Instrumentation and Measurement, 2021, 70:1-14. DOI:10.1109/TIM.2020.3048799
30. Zhang Y, Qin N, Huang D, et al. Fault Diagnosis of High-speed Train Bogie Based on Deep Neural Network[J]. SCIENCE CHINA Information Sciences, 2021, 52(24):135-139. DOI:10.1016/j.ifacol.2019.12.395.
31. Zhao B, Zhang X, Zhan Z, et al. A robust construction of normalized CNN for online intelligent condition monitoring of rolling bearings considering variable working conditions and sources[J]. Measurement, 2021, 174:108973.1-108973.16. DOI: 10.1016/j.measurement.2021.108973.
32. Zhao H, Zheng J, Xu J, et al. Fault Diagnosis Method Based on Principal Component Analysis and Broad Learning System[J]. IEEE Access, 2019, 7:99263-99272. DOI: 10.1109/ACCESS. 2019.2929094.
33. Zheng W, Yuan Q, Zou L, et al. Study on a Novel Fault Diagnosis Method Based on VMD and BLM[J]. Symmetry, 2019, 11(6):747-766. DOI: 10.3390/sym11060747.
34. Zhu L, Zhou Y, Jia R, et al. Real-Time Fault Diagnosis for EVs With Multilabel Feature Selection and Sliding Window Control[J]. IEEE internet of things journal, 2022, 19(9):18346-18359. DOI: 10.1109/JIOT.2022.3160298.