Received January 23, 2022, accepted February 8, 2022, date of publication February 11, 2022, date of current version February 22, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3151155

A Deep Learning-Based Fault Diagnosis of Leader-Following Systems

XIAOXU LIU[1], XIN LU[1], (Student Member, IEEE), AND ZHIWEI GAO[2], (Senior Member, IEEE)
[1]Sino-German College of Intelligent Manufacturing, Shenzhen Technology University, Shenzhen 518118, China
[2]Faculty of Engineering and Environment, Northumbria University, Newcastle Upon Tyne NE1 8ST, U.K.

Corresponding author: Xiaoxu Liu (liuxiaoxu@sztu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62003218, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2019A1515110234, in part by the Shenzhen Science and Technology Program under Grant RCBS20200714114921371, and in part by the Natural Science Foundation of Top Talent of Shenzhen Technology University (SZTU) under Grant 2020106.

ABSTRACT This paper develops a multisensor data fusion-based deep learning algorithm to locate and classify faults in a leader-following multiagent system. First, sequences of one-dimensional data collected from multiple sensors of followers are fused into a two-dimensional image. Then, the image is employed to train a convolution neural network with a batch normalisation layer. The trained network can locate and classify three typical fault types: the actuator limitation fault, the sensor failure and the communication failure. Moreover, faults can exist in both leaders and followers, and the faults in leaders can be identified through data from followers, indicating that the developed deep learning fault diagnosis is distributed. The effectiveness of the deep learning-based fault diagnosis algorithm is demonstrated via Quanser Servo 2 rotating inverted pendulums with a leader-follower protocol. From the experimental results, the fault classification accuracy can reach 98.9%.

INDEX TERMS Deep learning, multisensor data fusion, fault diagnosis, leader-following system, convolution neural network, data-driven, distributed, batch normalization, image fusion, sliding window data sampling.

I. INTRODUCTION

In recent years, multiagent systems have been widely used in the fields of multiunmanned aerial vehicles [1], smart grids [2], multirobot cooperative formation [3] and sensor networks [4]. The reliability of multiagent systems depends on the performance of each agent and their communications. A fault in one agent can degrade the performance of its neighbours via communication (e.g., [5]), which threatens the whole multiagent system. Furthermore, abnormal communication can cause agent loss control protocols from other agents. Therefore, there is a stringent demand to locate and identify the faults in multiagent systems at an early stage. Accurate fault diagnosis approaches produce time to mitigate the faults, save maintenance costs, and reduce the risk of a breakdown of the whole system, which is crucial for improving the reliability of the system, e.g., [6]–[10]. In recent years, fault diagnosis of multiagent systems has received extensive attention and undergone rapid development [11]–[15].

The associate editor coordinating the review of this manuscript and approving it for publication was Shen Yin.

Most existing results about fault diagnosis of multiagent systems are based on precise mathematical models and communication information, such as an adjacency matrix. Nevertheless, the model and communication are not always available from real multiagent systems. Therefore, it is motivated to develop alternative fault diagnosis techniques, such as data-driven methods [16]–[19] for multiagent systems that are independent of model and communication information.

Deep learning algorithms can obtain the system features by training data collected from sensors and have been widely used for fault diagnosis of a single system, e.g., [20]–[24]. Among various deep-learning algorithms that have been comprehensively reviewed in recent survey papers [25] and [26], convolutional neural networks (CNNs) can automatically learn features from datasets, especially large-scale datasets, and generalise the results to the same type of unknown data. Therefore, CNN-based fault diagnosis has achieved great success in many research fields, such as rotating machinery [27], wind turbine systems [28], power transmission systems [29], and motor bearing systems [30].

In multiagent systems, the number of sensors increases considerably, which brings challenges to the transmission and training of a great amount of data. Therefore, it is important to reduce data transmission and design an appropriate data preprocessing method in deep learning-based fault diagnosis of multiagent systems. To reduce data transmission, distributed fault diagnosis has been a recent research interest, e.g., [31]–[35]. Identifying faults in one agent through training data using its neighbour is a crucial objective of distributed fault diagnosis. However, multi-sensor data fusion can reduce training complexity and training time. Image fusion can transform a number of one-dimensional sensor data into a two-dimensional image. The image keeps rich information on the data; hence, the accuracy of fault diagnosis can be guaranteed. Training the image rather than the one-dimensional data repetitively can enhance the real-time performance of the system. [27], [28] and [30] employed image fusion for CNN data preprocessing. However, the systems under consideration in the abovementioned work are single systems. According to the authors' knowledge, there is no existing work about distributed fault diagnosis of multiagent systems through image fusion-based CNNs.

Multiagent systems with leader-following protocols are widely used in engineering fields, such as unmanned helicopters [36], [37], multi-inverted pendulums [38], battery packs [39] and liquid-level systems [40]. In the abovementioned work, faults were not considered in [36], [38], [39]. Only communication fault was discussed in [37]. Distributed fault detection was designed in [40]; however, fault classification was not a concern. The objective of this paper is to use an image fusion-based deep CNN to design distributed fault classification for a leader-following multiagent system considering actuator faults, sensor faults and communication faults. Specifically, the one-dimensional historical data collected from followers are converted into two-dimensional image information by a multisensor data fusion technique. Then, a deep CNN is established to train the image data. Through the training, the type and location of faults can be identified. The faults under investigation include communication interruption faults, sensor failure faults and actuator limitation faults. Furthermore, three types of faults can exist in both the leader and the followers. Finally, a real experiment on a leader-following inverted pendulum demonstrates the effectiveness of the developed algorithms. The contribution of this article can be summarised as follows: 1. Image fusion-based deep learning for fault diagnosis of leader-following systems is a novel topic. Compared with a one-dimensional CNN, the relevance of different sensors can be preserved through a two-dimensional image fusion-based CNN. Therefore, the accuracy of fault diagnosis can be enhanced via the developed technique. Actuator faults, sensor faults, and communication faults in both leaders and followers are investigated in this article, which is a challenge for fault diagnosis. Many existing results only consider actuator and sensor faults or only communication faults. Classifying the three types of faults together is another contribution of this paper. 3. The developed fault diagnosis algorithms depend on data from followers rather than from both leaders and followers. Prior to existing work on distributed fault diagnosis, such as in [24]–[27], communication among leaders and followers was not required. Therefore, the designed fault diagnosis is fully distributed.

The organization of the paper is as follows: The system and faults under consideration are introduced in Section II. Multisensor fusion-based deep learning for the distributed fault diagnosis algorithm is developed in Section III. Section IV demonstrates real experimental work to apply the developed technique to Quanser Servo 2 rotating inverted pendulums. Section V presents the conclusion and future work.

II. THE SYSTEM DESCRIPTION AND THE PROBLEM STATEMENT

A. MULTI-AGENT SYSTEMS WITH UNKNOWN COMMUNICATION

The system under consideration is a networked homogeneous multiagent system under the leader-follower control protocol, and its topology is shown in FIGURE 1, where 0 represents the leader and $1, 2, \ldots N$ represents the followers. The communication among N followers can be denoted by an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{A}\}$, where $\mathcal{V} = \{v_i, i = 1, 2, \ldots N\}$ represents the set of followers, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of edges, and $\mathcal{A} = [a_{ij}]$ is the weighted adjacency matrix of G. $a_{ij} > 0$ if and only if $(v_i, v_j) \in \mathcal{E}$; otherwise, $a_{ij} = 0$. If $(v_i, v_j) \in \mathcal{E}$, v_i is named the neighbour of v_j, and $\mathcal{N}_j = \{v_i : (v_i, v_j) \in \mathcal{E}\}$ stands for the set of all neighbours of v_j. $\deg(v_i) = d_i = \sum_{j \in N_i} a_{ij}$ denotes the degree of the vertex of v_j.$\mathcal{D} = diag\{d_1, d_2 \ldots d_N\}$ is the degree matrix, and $\mathcal{L} = \mathcal{D} - \mathcal{A}$ represents the Laplacian matrix of $\mathcal{G}$. The above definitions are generally used in leader-following multiagent systems, such as in [41], [42]. We assume that there is a pre-existing control protocol such that the overall system is stable with desired performance (e.g., consensus, robustness, etc.) in the absence of faults, and the design of the control protocol has been widely investigated in other works, such as [41]–[44]. The aim of this paper is to design a fault diagnosis technique to detect and classify faults accurately. The design of the control protocol is not a concern of this article.

In the design of the fault diagnosis method, the physical model and communication are unknown. In other words, the physical model and communication are internal to agents but are not available in fault classifiers.

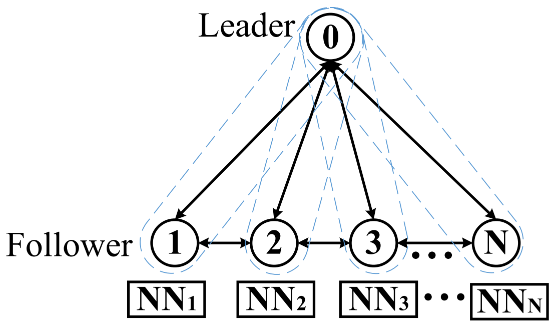


FIGURE 1. Leader-follower control protocol.

Remark 1: This method adopts distributed fault diagnosis, and each follower has a fault diagnosis device ($NN_i, i = 1, 2, \ldots N$), as shown in the FIGURE 1. In the authors previous work [49], a residual-triggered mechanism is proposed, where each agent has a predictor and a fault classifier. The designed predictor and the actual output form a residual. When the residual value generated in a follower exceeds the threshold, the fault diagnosis device on the corresponding follower can be triggered to classify the fault. When the residual of the leader exceeds the threshold, the residual network in follower1 is enabled by default. Therefore, the fault diagnosis task with N agents can be divided into a leader-follower distributed fault diagnosis mode with $N - 1$ pairs. Taking the formation mode of artificial potential field as an example, the strong repulsion field will be generated between followers only at a very short distance. Under normal circumstances, the influence is small, and only the leader will produce a large gravitational field on the follower. Therefore, the objective of this paper focus on enhancing fault classifier in the residual-triggered fault diagnosis developed in [49], namely, when a leader-follower pair is triggered.

B. THE TYPE OF FAULTS

Between agents can also be abnormal in networked systems, e.g., the interruption of interactions or false data injection between agents. Therefore, the identification and classification of communication faults is crucial for leader-following systems.

In this paper, the faults under consideration include sensor failure, actuator limitation faults and communication interruption. Specifically, the sensor fault is moulded as follows:

$$f_s(t) = \begin{cases} 0, & t \le t_0 \\ -y(t), & t \ge t_0 \end{cases} \tag{1}$$

where $f_s(t)$ represents the sensor fault, t_0 denotes the time that a sensor fault occurs, and $y(t)$ is the system output without a sensor fault. This type of fault indicates that the sensor output remains zero, which can occur when the signal is open circuited [45].

An actuator limitation fault $f_a(t)$ is modelled as follows:

$$f_a(t) = \begin{cases} 0, & u(t) \le A \\ A - u(t), & u(t) > A \end{cases} \tag{2}$$

where $u(t)$ is the real input of the actuator and A is a constant that the actuator output cannot exceed due to abnormal hardware or software conditions. This type of fault occurs due to actuator stuck failures [46] and is recognised as one of the most important factors that reduce system performance.

A communication fault $f_c(t)$ under investigation is an interrupt fault, and can be calculated as follows:

$$f_c(t) = \begin{cases} 0, & t < t_1 \\ -q(t), & t \ge t_1 \end{cases} \tag{3}$$

where t_1 is the time that the interruption occurs and $q(t)$ is the normal communication signal. This type of fault can occur due to a cyber-attack, which is also known as denial-of-service [47].

Remark 2: This paper adopts a distributed fault diagnosis, which is used to identify the fault between leader and follower. The main contribution of this paper is to investigate the fault diagnosis of leader-follower tracking. Moreover, the communication between followers is weak or no communication. As a result, the fault of communication between followers is not considered in this paper. "Follower communication failure" means that the communication in which the follower sends its status to the leader is interrupted.

The actual signal is formulated as follows:

$$Y(t) = y(t) + f_\sigma(t) \quad \sigma \in \{s, a, c\} \tag{4}$$

$Y(t)$ represent the real signal of system, $y(t)$ represent the theoretical signal of system, $f_\sigma(t)$ represent three types of fault. Take $f_s(t)$ as an example. When there is no fault, $f_s(t) = 0$. When a fault occurs, $f_s(t) = -y(t)$. As a result, the really output of sensor equal 0.

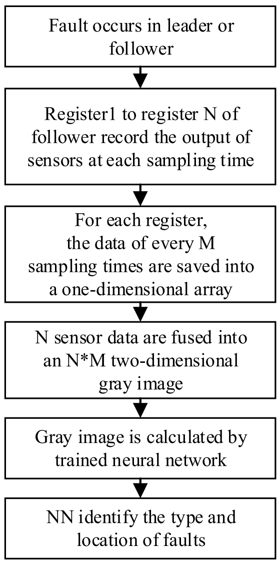

Fault occurs in leader or follower
Register1 to register N of follower record the output of sensors at each sampling time
For each register, the data of every M sampling times are saved into a one-dimensional array
N sensor data are fused into an N*M two-dimensional gray image
Gray image is calculated by trained neural network
NN identify the type and location of faults

FIGURE 2. Diagnosis flowchart when fault occurs.

Another challenge in fault diagnosis of multiagent systems is that the amount of data to be trained is great. Moreover, the relation of data from different sensors should be included in the data collection. In the rest of the paper, a multisensor fusion-based deep learning fault classification technique is proposed to identify and locate sensor faults, actuator faults and communication faults accurately. To further reduce data transmission, the designed fault classifier can recognise the fault of the leader through output of the follower without requiring their communication information and precise mathematical model. A 2D image is generated through multisensor fusion. Then, a deep-learning algorithm is designed to classify and locate faults. The diagnosis process can be described in FIGURE 2. The diagram of the overall design can be found in FIGURE 3.

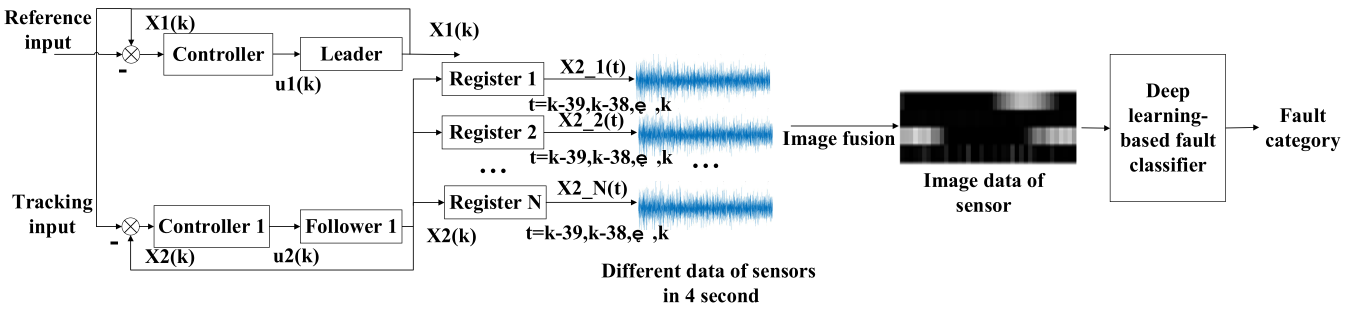


FIGURE 3. Multisensor fusion-based deep learning fault classifier.

III. DEEP LEARNING-BASED FAULT CLASSIFICATION

This section introduces the image-based sensor fusion method and the main types of faults. The image-based sensor fusion method takes advantage of the BP neural network in feature extraction and fault recognition accuracy.

A. IMAGE FUSION BASED ON MULTI-SENSOR-SIGNALS

Data preprocessing is the premise for deep learning-based fault diagnosis. Traditional data preprocessing methods for deep learning are known as normalisation and regularisation. Many modern systems, especially multiagent systems, have a number of sensors. Normalising and regularising data from multiple sensors considerably increases computational complexity and the relation among sensors can be missed. Therefore, this paper is motivated to develop an image fusion technique to transform data from a number of one-dimensional sensors into two-dimensional grey images. The fusion of multisensor data into a grey image for deep learning can improve the accuracy, richness, and efficiency of fault diagnosis. FIGURE 4 illustrates the process of converting one-dimensional data into two-dimensional image grey values.

Each sensor obtains M sampling points through sliding window technology(in the section IV). Then M sampling points are calculated and converted into pixel values through formula (5). Finally, N sensor signals are fused into an image of size $N \times M$. When the noise signal is incorporated into the original signal, the influence to fault diagnosis depends on the size of the noise. If the value of noise is small, it will not affect the converted data into pixel values, because there is a round operation in the process of converting data into pixel values. If the noise is big, then it will result in a wrong pixel, which reduces the performance of fault diagnosis. Normally, the noises incorporated into data is in the former case. The grey value of each pixel is obtained via Equation (5) as follows:

$$\boldsymbol{Value}(\boldsymbol{i},\boldsymbol{j}) = 255 \times \boldsymbol{Round} \times \left(\frac{\boldsymbol{signal}(\boldsymbol{i},\boldsymbol{j}) - \boldsymbol{min}(\boldsymbol{signal}(\boldsymbol{i},\boldsymbol{j}))}{\boldsymbol{max}(\boldsymbol{signal}(\boldsymbol{i},\boldsymbol{j}) - \boldsymbol{min}(\boldsymbol{signal}(\boldsymbol{i},\boldsymbol{j})))}\right) \tag{5}$$

In Equation (4), $\boldsymbol{i} = \mathbf{1}, \mathbf{2}, \ldots$ stands for time $\boldsymbol{iT}$, where $\boldsymbol{T}$ is the sampling time and $\boldsymbol{j}$ represents the $\boldsymbol{j}$-th sensor. $\boldsymbol{signal}(\boldsymbol{i},\boldsymbol{j})$ is the data collected from the $\boldsymbol{j}$-th sensor at time $\boldsymbol{iT}$. Meanwhile, $\boldsymbol{i}$ and $\boldsymbol{j}$ correspond to the number of rows and the number of columns in the 2-D image, respectively; $\boldsymbol{Value}(\boldsymbol{i},\boldsymbol{j})$ stands for the grey value of the image at $\boldsymbol{i},\boldsymbol{j}$, and $\boldsymbol{Round}(\cdot)$ is a rounding function, which ensures that every data point can be converted into the corresponding grey value, and there is no case in which the decimal cannot generate the corresponding grey value.

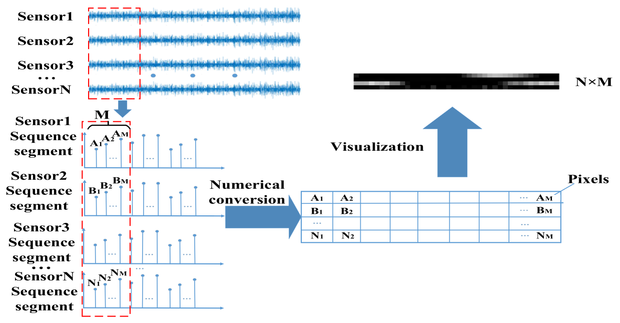


FIGURE 4. Conversion to 2D image process diagram.

B. CONVOLUTIONAL NEURAL NETWORK-BASED DEEP LEARNING

Through the aforementioned image fusion, a 2-D image containing rich information of the system can be generated. Then, deep learning-based fault diagnosis can be designed by the fused image. The existing deep learning method for image recognition is to reshape the image to be nearly square before recognition. However, the adjacent features can be lost by reshaping. The sensor returns continuous sequence data, and local visual field features are important for feature extraction of convolution. The reshaping process reduces the accuracy of fault diagnosis. Therefore, this paper proposes a deep learning fault diagnosis algorithm without reshaping the image to increase the accuracy of fault diagnosis.

To train a large-scale dataset, a CNN is used for deep learning. The CNN used in this paper is composed of an input layer, a convolution layer, a pooling layer, a batch normalisation layer, a full connection layer and an output layer. The main function of the input layer is to preprocess the input data and convert the data of different sizes into the data of the same format and size. The processed image is called the feature map. Two methods of data preprocess are used, i.e., padding and normalisation.

1) PADDING

Padding is used to fill in one or more circles of 0 value around the image, which can ensure that the height or depth of data will not decrease when convolution calculation is carried out. More importantly, the image edge information can be preserved. If there is no padding, the edge features of the next layer disappear, which does not affect the subsequent convolution. Generally, in convolutional neural networks, $\boldsymbol{P}$ is used to represent the number of paddings. The schematic diagram of padding is shown in FIGURE 5.

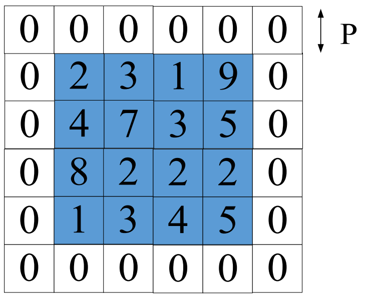


FIGURE 5. **The schematic diagram of padding.**

2) NORMALISATION

Normalisation reduces the value of each pixel to the range from 0 to 1 after preprocessing the data, which is conducive to data processing. The common normalisation methods are function transformation, Z score standardisation and max min standardisation.

The convolution layer is composed of a number of convolution kernels, which are also known as filters. The pixels of the image after padding and normalisation are input to the convolution layer.

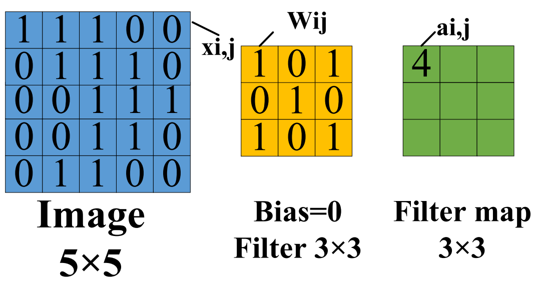


FIGURE 6. **Convolution calculation diagram.**

A schematic diagram of the convolution calculation process is presented in FIGURE 6 and Equation (6).

In FIGURE 6, the element in the upper left corner of filter map $\boldsymbol{a_{0,0}}$ is calculated as follows:

$$\begin{aligned}\boldsymbol{a_{0,0}} &= \sum_{m=0}^{2}\sum_{n=0}^{2} \boldsymbol{W_{m,n}x_{i+m,j+n}} + \boldsymbol{W_b} \\ &= \boldsymbol{W_{0,0}x_{0,0}} + \boldsymbol{W_{0,1}x_{0,1}} + \ldots + \boldsymbol{W_{2,1}x_{2,1}} + \boldsymbol{W_2} \\ &= \boldsymbol{1+0+1+0+1+0+0+0+1+0} \\ &= \boldsymbol{4} \end{aligned} \tag{6}$$

where $\boldsymbol{x_{i,j}}$ represents the grey value of the pixel at row $\boldsymbol{i}$ and column $\boldsymbol{j}$; $\boldsymbol{W_{m,n}}$ represents the weight at row $\boldsymbol{m}$ and column $\boldsymbol{n}$ and $\boldsymbol{W_b}$ represents the bias. $\boldsymbol{f(\cdot)}$ is a leaky rectified linear unit (leaky ReLU) activation function, which is shown in Equation (7) and FIGURE 7. In Equation (7), $\boldsymbol{c}$ is a fixed constant, and $\boldsymbol{c\epsilon(1,+\infty)}$. The main function of leaky ReLU is to add nonlinearities to the network model so that the network has a better feature recognition ability. It has a broad acceptance domain for solving the gradient disappearance problem in the training process. Equation (7) is calculated as follows:

$$f(x) = \begin{cases} \frac{x}{c}, & x \leq 0 \\ x, & x > 0 \end{cases} \tag{7}$$

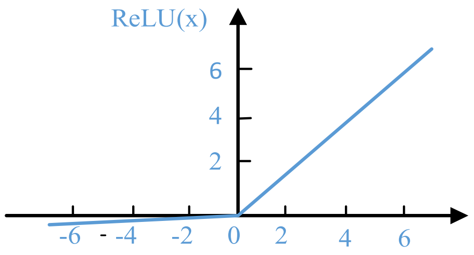


FIGURE 7. **Leaky ReLU activation function.**

Remark 3: There are two common activation functions, which are known as sigmoid and ReLU activation functions. When the value of the convolution layer is large, the gradient of the network disappears by using the sigmoid function, which will cause the slow convergence speed of the model. ReLU function remains linear at $(\mathbf{0}, +\infty)$. Therefore, the ReLU activation function solves the problem with a slow learning convergence of the neural network caused by gradient disappearance. However, the output of neurons can be negative, and all negative activation becomes zero by the ReLU function. Compared with traditional ReLU function, which induces the disappearance of negative neurons, leaky ReLU, a variant of ReLU, can preserve negative neurons, hence enhancing the training performance.

To obtain a more obvious feature map, a batch normalisation layer [48] is added after the convolution layer. The function of a batch normalisation layer is similar to the normalisation of data in the input layer. It normalises the features learned by the previous convolution layer, which is also conducive to feature learning by the next convolution layer. However, the output of the convolution layer is different from the input of the network. The output of the convolution layer is the extracted feature, while the input of the network is the non-extracted data. Therefore, the output of the convolution layer cannot be normalised directly via the method of input. For the aforementioned reason, batch normalisation, which enables the network to learn and recover the characteristic distribution, is employed in this paper.

Pooling is implemented for downsampling. Through pooling, useless information from the calculated feature map can be removed and the amount of data can be reduced to enhance the operation speed. Among widely used pooling methods, maximum pooling can effectively preserve and enhance the feature information in downsampling. Therefore, maximum pooling is used in the applied convolutional networks. An example of the method is shown in FIGURE 8.

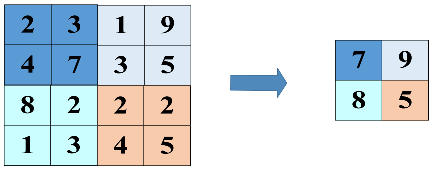


FIGURE 8. **Max pooling.**

The function of the fully connected layer is to send the feature map obtained after a series of operations, such as convolution, into the fully connected layer through flattening. Different feature planes can be mapped to the same feature plane after full connection linear transformation, which is conducive to combining features trained by different filters for analysis.

The output layer applies the softmax activation function to solve multiclassification problems. This layer maps the outputs of multiple neurons into the (0,1) interval, which can be regarded as the probability that the current output belongs to a category. Then, multiclassification can be carried out. The calculation method is given in (8), where a_i is the value of the ***ith*** input node of the output layer and A_i is the output of the CNN. This is calculated as follows:

$$A_i = \frac{e^{a_i}}{\sum e^{a_i}} \tag{8}$$

All of the nodes in the previous layer are connected to the next neuron node by a weight. The overall structure of the convolutional neural network is shown in FIGURE 9.

Remark 4: For the convolution neural network, we use the classical neural network LeNet-5 to modify the structural parameters. i.e. the first layer is composed of six 3×3 filters where the stride is 1; the second layer consists sixteen 3×3 filters, where the stride is 1. In order to increase performance of fault diagnosis, an additional layer with forty 3×3 filters is added to LeNet-5 in the developed technique.

IV. EXPERIMENTAL RESULTS

A. SYSTEM AND FAULT DESCRIPTION

In this section, the developed deep learning fault diagnosis is applied to Quanser Servo 2 rotating inverted pendulums to demonstrate the effectiveness. The agents are connected through a leader-follower control protocol. The hardware-in-loop diagram is shown in FIGURE 10. We conduct fault diagnosis by collecting the data of follower. The parameter of sensor shown in TABLE 1. The categories of faults under investigation include: fault-free, leader sensor failure, leader actuator stuck, leader communication failure, follower sensor failure, follower actuator stuck, and follower communication failure, which is shown in TABLE 2.

TABLE 1. **Parameter and meaning.**

parameter	Representative meaning
θ	Horizontal displacement of inverted pendulum
α	Vertical displacement of inverted pendulum
$\dot{\theta}$	Horizontal velocity of inverted pendulum
$\dot{\alpha}$	Vertical velocity of inverted pendulum

TABLE 2. **The type of faults and category label.**

Sample type	Sample length	Number of sample	Category label
leader and follower work normally	40	4960	1
leader sensor failure	40	4960	2
leader actuator stuck	40	4960	3
leader communication failure	40	4960	4
follower sensor failure	40	4960	5
follower actuator stuck	40	4960	6
follower communication failure	40	4960	7

B. DATA SAMPLE

The Quanser Servo 2 inverted pendulum is connected with MATLAB Simulink through USB, and a hardware-in-loop experiment can be implemented. The real-time data of the inverted pendulum is collected from Simulink with a sampling time of 0.005 s. The storage of software is limited, which indicates that the maximum length of data is certain. However, sufficient data should be obtained to train the neural network such that the generalisability is satisfied. To generate sufficient data from limited software storage, a sliding window data sampling approach is employed to amplify the data. Specifically, a sampling window of length f moves on the collected data of length L, and the moving step is S (see

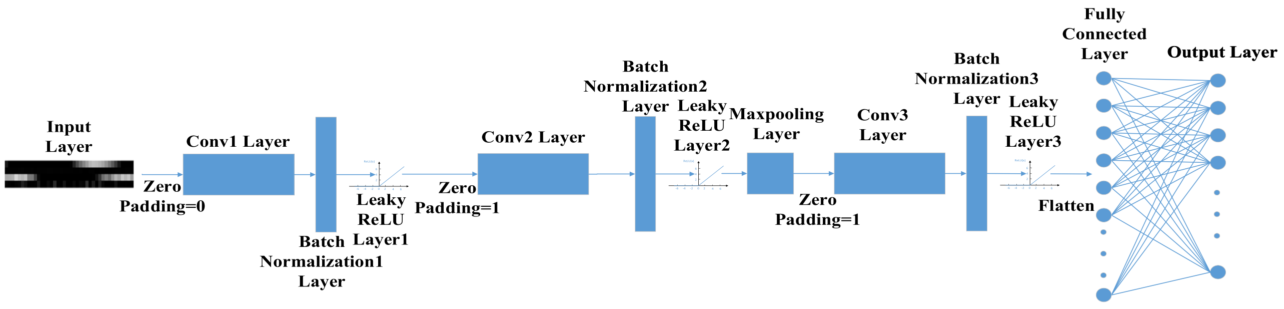


FIGURE 9. Structure of convolution neural network.

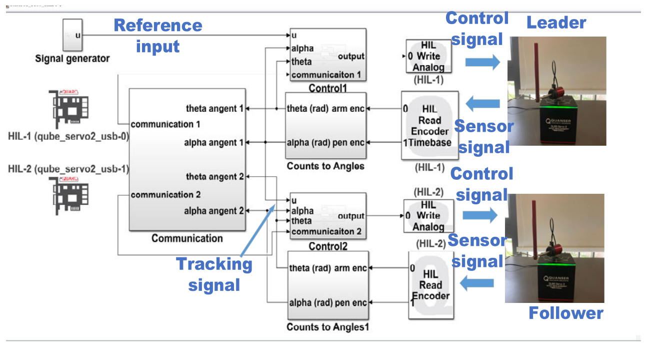


FIGURE 10. Leader-following system control diagram. (a) The communication block simulates the communication between agents (b) HIL write analog and HIL read encoder represent the input of the actuator and the output of sensors, respectively.

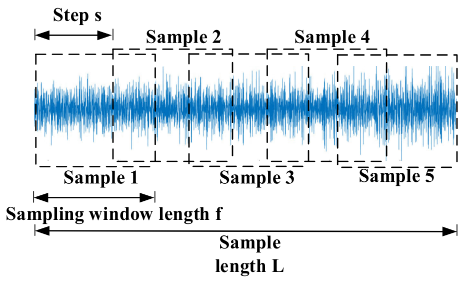


FIGURE 11. Data expansion diagram.

FIGURE 11). As a result, $\boldsymbol{n}$ groups of data can be obtained in the equation as follows:

$$\boldsymbol{n} = \left[\frac{L-f}{S}\right] \tag{9}$$

In each fault scenario, real inverted pendulum data is collected over 29 s, and the sampling time is 0.005 s. The number of sampling points is 5800, i.e., $\boldsymbol{L} = \mathbf{5800}$. The number of sampling points in the sampling window is 800, i.e.,$\boldsymbol{f} = \mathbf{800}$. The sampling step $\boldsymbol{S} = \mathbf{1}$. Therefore, 5000 groups of data ($\boldsymbol{n} = \mathbf{5000}$) are generated, which increases the amount of data by 114.28 times. However, the calculation ability of the software is limited. Therefore, the sampling time of the sliding window after data expansion is extended to 0.1 s, which indicates that the sample length is 40 [49].

C. EXPERIMENTAL RESULTS AND ANALYSIS

Different types of faults are injected to the system, and data of 4 sensors in the follower is recorded. In each fault scenario, data after expansion is used to train the neural network, and the features of the fault can be exacted. The data is divided into three parts: 80% (training set) of the data is used to train the network and update the model weight parameters and 8% (validation set) of the data is used to evaluate the model performance. The remaining 12% (test set) is used to test the accuracy of final neural network. In this experiment, the fault classification performances of the BP neural network, the traditional CNN network with image fusion, and the batch normalisation embedded CNN network with image fusion are compared.

Remark 5: In this paper, 2-D is considered in the data conversion. High dimensions are not considered because there are only four sensors for data acquisition in the experimental equipment, and it is difficult to increase the dimension of data. Moreover, if the dimension of data is increased, the sensors that obtain the data should have the same characteristics. For example, if three channel RGB image ($\boldsymbol{n} \times \boldsymbol{n} \times \mathbf{3}$) is used, fault characteristics in the three channels should be similar with each other. Therefore, the increased dimension is not applicable in this experiment.

1) BP NEURAL NETWORK FAULT DIAGNOSIS

BP neural network is used to train the original image. The image of size 4*40*1 is flatted into 160 inputs. The physical meaning of data form sensor 1 ($\boldsymbol{\theta}$), sensor 2 ($\boldsymbol{\alpha}$), sensor 3 ($\dot{\boldsymbol{\theta}}$) and sensor 4 ($\dot{\boldsymbol{\alpha}}$) can be found in TABLE 1. Sensor 1,3 and 2,4 are independent; Sensor 1 is the integral of sensor 3; and sensor 2 is the integral of sensor 4. The structure of BP shows in TABLE 3.

After 571 seconds' training, we can draw the training figures. The accuracy curve and loss function curve of BP network-based fault classification are shown in FIGURE 12 and FIGURE 13, respectively. To illustrate the performance of BP neural network-based fault diagnosis, the fault misclassification matrix is drawn in FIGURE 14.

The coordinate values from 1 to 7 in FIGURE 14 are the label numbers in TABLE 2, representing different fault types

TABLE 3. The structure of the BP.

Layer	Output Shape	Param#
input_1	4, 40, 1	0
Flatten	160	0
Fullyconnect_1	70	11270
activation_1	70	0
Fullyconnect _2	25	1775
activation_2	25	0
Fullyconnect _3	15	390
activation_3	15	0
Output	7	112
Total params: 13,547		
Trainable params: 13,547		
Non-trainable params: 0		

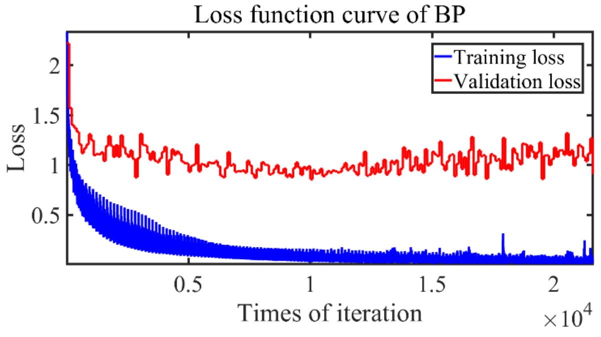


FIGURE 12. Loss function curve of BP neural network. (a) Training loss represents the loss value curve of the training set obtained by the gradient descent method of momentum. (b) Validation Loss represents the loss value curve of the validation set obtained by the gradient descent method of momentum. (c) Iteration represents update times of neural network weight.

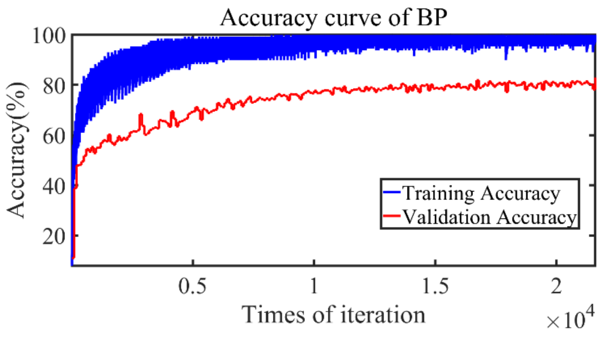
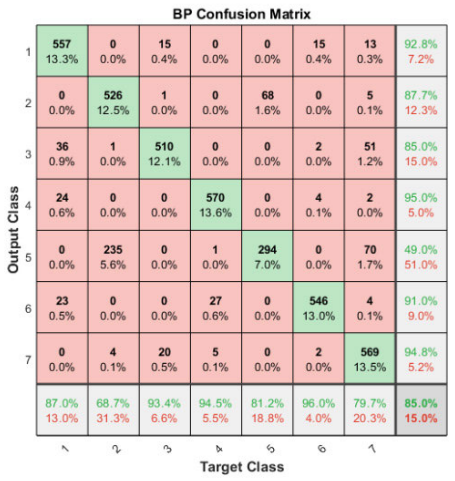


FIGURE 13. Fault accuracy curve of BP neural network. (a) Training accuracy represents the recognition accuracy of the network to the fault of the training set. (b) Validation accuracy represents the recognition accuracy of the network to the fault of the validation set. (c) Iteration represents update times of neural network weight.

of the leader-following system. The number in the shadow is the number of actual sample tags that match the predicted sample tags. The BP neural network can roughly identify the seven types of faults, and the accuracy of types 3,4 and 6 is good. Nevertheless, the accuracy of other faults is low, especially for type 2 faults. The overall identification accuracy is only 85%.



Output Class \ Target Class	1	2	3	4	5	6	7	
1	557 13.3%	0 0.0%	15 0.4%	0 0.0%	0 0.0%	15 0.4%	13 0.3%	92.8% 7.2%
2	0 0.0%	526 12.5%	1 0.0%	0 0.0%	68 1.6%	0 0.0%	5 0.1%	87.7% 12.3%
3	36 0.9%	1 0.0%	510 12.1%	0 0.0%	0 0.0%	2 0.0%	51 1.2%	85.0% 15.0%
4	24 0.6%	0 0.0%	0 0.0%	570 13.6%	0 0.0%	4 0.1%	2 0.0%	95.0% 5.0%
5	0 0.0%	235 5.6%	0 0.0%	1 0.0%	294 7.0%	0 0.0%	70 1.7%	49.0% 51.0%
6	23 0.5%	0 0.0%	0 0.0%	27 0.6%	0 0.0%	546 13.0%	4 0.1%	91.0% 9.0%
7	0 0.0%	4 0.1%	20 0.5%	5 0.1%	0 0.0%	2 0.0%	569 13.5%	94.8% 5.2%
	87.0% 13.0%	68.7% 31.3%	93.4% 6.6%	94.5% 5.5%	81.2% 18.8%	96.0% 4.0%	79.7% 20.3%	85.0% 15.0%

FIGURE 14. Fault misclassification matrix trained by BP. (a) The number of 1–7 in the picture represent the label of fault. (b) The value in the lower right corner represents the final network accuracy.

2) CNN-BASED FAULT DIAGNOSIS

It can be seen that the fully connected network cannot have a satisfactory diagnosis performance, especially when the characteristics of faults are closed. Now, it is time to implement the multisensor data fusion-based CNN algorithms to achieve fault classification.

First, the one-dimensional time domain signal data of different fault types are converted into two-dimensional images, and the different fault images are shown in Figure 15. The converted image is divided into three parts: 80%, 8% and 12%. 80% (training set) of the data is used to train the network and update the model weight parameters and 8%(validation set) of the data is used to evaluate the model performance. The remaining 12%(test set) is used to test the accuracy of the final neural network.

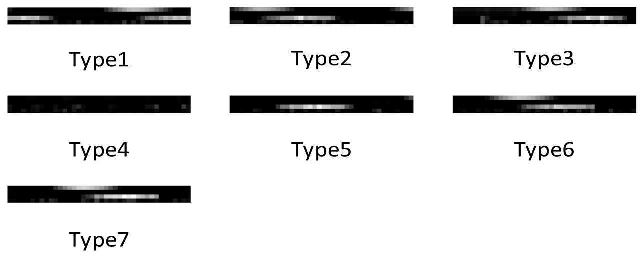


FIGURE 15. The picture of fault types.

Then, we use the convolutional neural network, which has outstanding performance in image recognition, as the training network, and the specific network structure parameters are shown in Table 4.

After 280 seconds' training, the final network accuracy reaches 91.5%. The fault accuracy curve is shown in FIGURE 16, and the loss function curve is shown in FIGURE 17. By connecting the trained neural network to the inverted pendulum model for the experiment, the confusion matrix can be generated in FIGURE 18.

TABLE 4. The structure of the CNN.

Layer	Output Shape	Param#
input_1	4, 40, 1	0
conv2d_1	2, 38, 6	60
activation_1	2, 38, 6	0
zero_padding2d_2	4, 40, 6	0
conv2d_2	2, 38, 16	880
activation_2	2, 38, 16	0
max_pooling2d_2	1, 19, 16	0
zero_padding2d_3	3, 21, 16	0
conv2d_3	1, 19, 40	5800
activation_3	1, 19, 40	0
Flatten	760	0
Output	7	761

Total params: 7,501
Trainable params: 7,501
Non-trainable params: 0

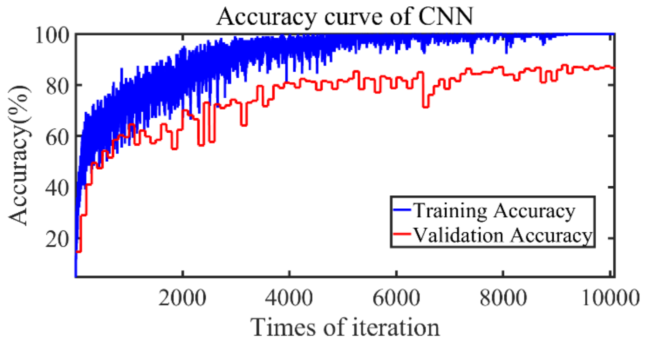


FIGURE 16. Fault accuracy curve of CNN.

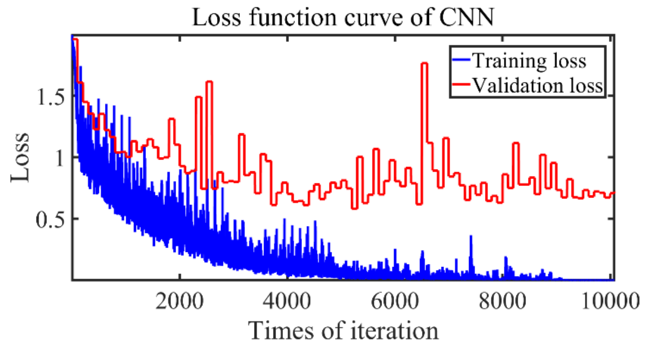
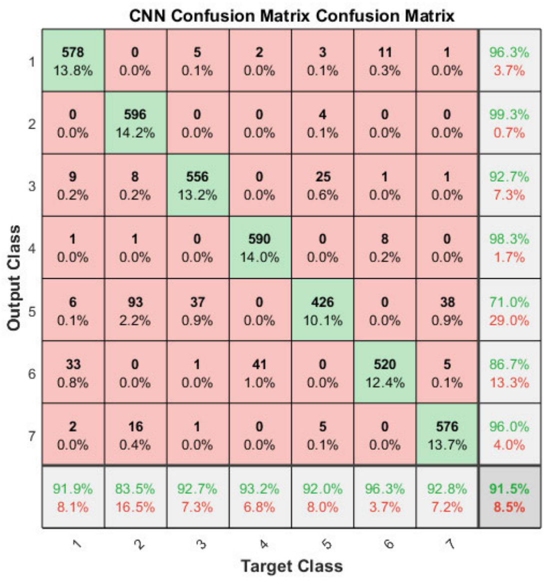


FIGURE 17. Loss function curve of CNN.

Comparing FIGURE 18 with FIGURE 14, we find that the convolution neural network has better fault diagnosis performance than the fully connected neural network. There is a certain misjudgment for individual faults, but the error rate is in an acceptable range. Compared with the fully connected


CNN Confusion Matrix Confusion Matrix
Output Class
Target Class

Output \ Target	1	2	3	4	5	6	7	
1	578 13.8%	0 0.0%	5 0.1%	2 0.0%	3 0.1%	11 0.3%	1 0.0%	96.3% 3.7%
2	0 0.0%	596 14.2%	0 0.0%	0 0.0%	4 0.1%	0 0.0%	0 0.0%	99.3% 0.7%
3	9 0.2%	8 0.2%	556 13.2%	0 0.0%	25 0.6%	1 0.0%	1 0.0%	92.7% 7.3%
4	1 0.0%	1 0.0%	0 0.0%	590 14.0%	0 0.0%	8 0.2%	0 0.0%	98.3% 1.7%
5	6 0.1%	93 2.2%	37 0.9%	0 0.0%	426 10.1%	0 0.0%	38 0.9%	71.0% 29.0%
6	33 0.8%	0 0.0%	1 0.0%	41 1.0%	0 0.0%	520 12.4%	5 0.1%	86.7% 13.3%
7	2 0.0%	16 0.4%	1 0.0%	0 0.0%	5 0.1%	0 0.0%	576 13.7%	96.0% 4.0%
	91.9% 8.1%	83.5% 16.5%	92.7% 7.3%	93.2% 6.8%	92.0% 8.0%	96.3% 3.7%	92.8% 7.2%	91.5% 8.5%

FIGURE 18. Fault confusion matrix trained by the CNN.

neural network, the fault classification accuracy of the image fusion-based CNN has improved substantially. The accuracy is acceptable even for some faults with similar features.

3) CNN FAULT DIAGNOSIS WITH BATCH NORMALISATION

However, there are still fault data that cannot be identified accurately by utilising traditional convolutional neural networks. To improve the accuracy of recognition, we are motivated to further improve the feature extractability of the convolution network. Therefore, we add a layer of batch normalisation after each convolution layer, which is introduced in part III. The hierarchy of the network is documented in TABLE 5. The fault accuracy curve, loss function curve, and fault misclassification matrix are demonstrated in FIGUREs 19-21.

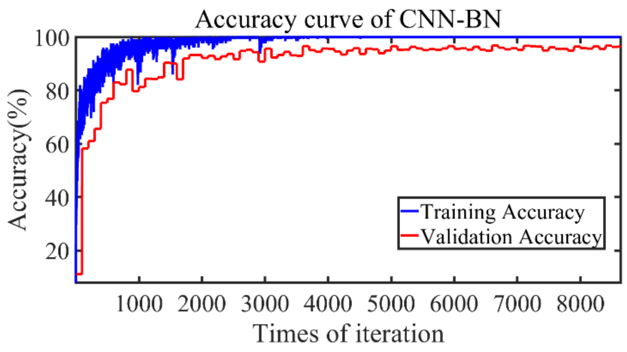


FIGURE 19. Fault accuracy curve of the CNN with batch normalisation.

It can be seen from FIGURE 21 that, after 318 seconds' training, the recognition accuracy of the network has reached 99%, which is a considerable improvement compared with the BP and the traditional CNN. Based on the developed image fusion-based CNN with batch normalisation, the

TABLE 5. The structure of the CNN with batch normalisation.

Layer	Output Shape	Param#
input_1	4, 40, 1	0
conv2d_1	2, 38, 6	60
batch_normalization_1	2, 38, 6	24
activation_1	2, 38, 6	0
zero_padding2d_1	4, 40, 6	0
conv2d_2	2, 38, 16	880
batch_normalization_2	2, 38, 16	64
activation_2	2, 38, 16	0
max_pooling2d	1, 19, 16	0
zero_padding2d_2	3, 21 16	0
conv2d_3	1, 19, 40	5800
batch_normalization_3	1, 19, 40	160
activation_3	1, 19, 40	0
Flatten	760	0
Output	7	760

Total params: 7,749
Trainable params: 7,625
Non-trainable params: 124

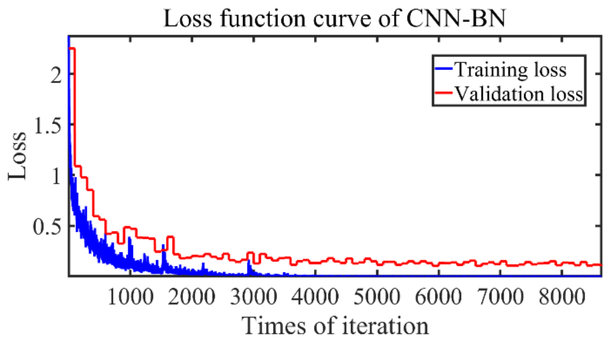


FIGURE 20. Loss function curve of the CNN with batch normalisation.

fault classification accuracy of the seven types of faults is satisfactory.

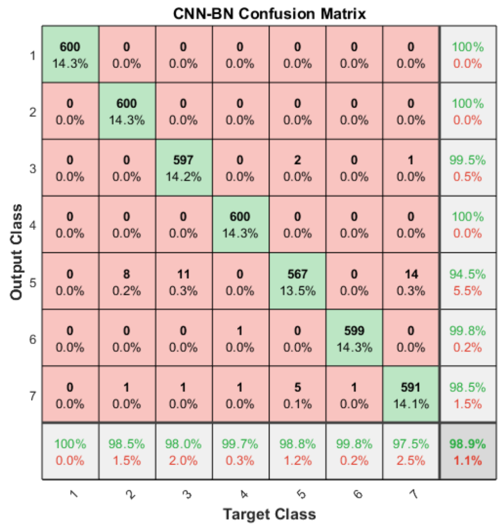
Remark 6: The time complexity can be reflected in Figures 12-20, and the space complexity can be reflected in TABLEs 3-5 as follows:

$$\text{time} \sim O\left(\sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l\right) \tag{10}$$

$$\text{space} \sim \mathrm{O}\left(\sum_{l=1}^{D} K_l^2 \cdot C_{l-1} \cdot C_l + \sum_{l=1}^{D} M^2 \cdot C_l\right) \tag{11}$$

where M represents the edge length of the feature map and K represents the side length of each filter. C represents the number of outputs of the current layer, *and* l represents the number of outputs of the current layer. D represents the depth of network. We can conclude the following from the experimental parameters:

$$time_{bp} > time_{bn-cnn} > time_{cnn};$$
$$space_{bp} > space_{bn-cnn} > space_{cnn}.$$



Output Class \ Target Class	1	2	3	4	5	6	7	
1	600 14.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	600 14.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	0 0.0%	597 14.2%	0 0.0%	2 0.0%	0 0.0%	1 0.0%	99.5% 0.5%
4	0 0.0%	0 0.0%	0 0.0%	600 14.3%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
5	0 0.0%	8 0.2%	11 0.3%	0 0.0%	567 13.5%	0 0.0%	14 0.3%	94.5% 5.5%
6	0 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	599 14.3%	0 0.0%	99.8% 0.2%
7	0 0.0%	1 0.0%	1 0.0%	1 0.0%	5 0.1%	1 0.0%	591 14.1%	98.5% 1.5%
	100% 0.0%	98.5% 1.5%	98.0% 2.0%	99.7% 0.3%	98.8% 1.2%	99.8% 0.2%	97.5% 2.5%	98.9% 1.1%

FIGURE 21. Fault misclassification matrix trained by the CNN with batch normalisation.

The time complexity determines the training or prediction time of the model, and the space complexity determines the number of model parameters. The time complexity can be reflected in Figures 12-17, and the space complexity can be reflected in TABLE 2 and TABLE 3. The computation complexity of the BN CNN is medium among the three methods, and the accuracy is the best.

It should be noted that at present, only two rotating inverted pendulums exist in our laboratory. Therefore, only one leader and one follower are used in the experiment. However, we are recently employing heterogeneous manipulators (robotic arms with 4-6 freedom) and multiunmanned aerial vehicles, which will be used in future research. Moreover, improvement of the network structure to enhance fault diagnosis accuracy is also under further investigation.

V. CONCLUSION

This research presents a distributed deep learning fault diagnosis technique for leader-following systems, where the CNNs with batch normalisation and image fusion methods are integrated to enhance training efficiency and accuracy. Three typical faults, including communication faults, sensor faults and actuator faults, are investigated in this article. For the leader-following system with communication coupling, the fault diagnosis of the leader can be achieved by observing the follower. Real experimental work has illustrated that the recognition rate of the developed fault diagnosis method is important prior to the BP neuro network and the traditional CNN. In future research, multiagent systems with other topologies and the improvements made to the CNN to enhance fault diagnosis accuracy will be considered.

REFERENCES

[1] F. Real, A. R. Castano, A. Torres-Gonzalez, J. Capitan, P. J. Sanchez-Cuevas, M. J. Fernandez, M. Villar, and A. Ollero, "Experimental evaluation of a team of multiple unmanned aerial vehicles for cooperative construction," *IEEE Access*, vol. 9, pp. 6817–6835, 2021.

[2] Y. Xu, W. Zhang, M.-Y. Chow, H. Sun, H. B. Gooi, and J. Peng, "A distributed model-free controller for enhancing power system transient frequency stability," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1361–1371, Mar. 2019.

[3] F. Zitouni, S. Harous, and R. Maamri, "A distributed approach to the multi-robot task allocation problem using the consensus-based bundle algorithm and ant colony system," *IEEE Access*, vol. 8, pp. 27479–27494, 2020.

[4] B. Shen, Z. Wang, D. Wang, and H. Liu, "Distributed state-saturated recursive filtering over sensor networks under round-robin protocol," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3605–3615, Aug. 2020.

[5] H.-J. Li, W. Xu, S. Song, W.-X. Wang, and M. Perc, "The dynamics of epidemic spreading on signed networks," *Chaos, Solitons Fractals*, vol. 151, Oct. 2021, Art. no. 111294.

[6] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3757–3767, Jun. 2015.

[7] X. Liu, Z. Gao, and M. Z. Q. Chen, "Takagi–Sugeno fuzzy model based fault estimation and signal compensation with application to wind turbines," *IEEE Trans. Ind. Electron.*, vol. 64, no. 7, pp. 5678–5689, Jul. 2017.

[8] X. Liu, Z. Gao, and A. Zhang, "Observer-based fault estimation and tolerant control for stochastic Takagi–Sugeno fuzzy systems with Brownian parameter perturbations," *Automatica*, vol. 102, pp. 137–149, Apr. 2019.

[9] K. Sun, L. Liu, J. Qiu, and G. Feng, "Fuzzy adaptive finite-time fault-tolerant control for strict-feedback nonlinear systems," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 4, pp. 786–796, Apr. 2021.

[10] M. Liu, L. Zhang, P. Shi, and Y. Zhao, "Fault estimation sliding-mode observer with digital communication constraints," *IEEE Trans. Autom. Control*, vol. 63, no. 10, pp. 3434–3441, Oct. 2018.

[11] P. Zou, P. Wang, and C. Yu, "Distributed fault detection for linear time-varying multi-agent systems with relative output information," *IEEE Access*, vol. 9, pp. 42933–42946, 2021.

[12] G. Dong, H. Li, H. Ma, and R. Lu, "Finite-time consensus tracking neural network FTC of multi-agent systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 653–662, Apr. 2021.

[13] Y. Xu and Z.-G. Wu, "Distributed adaptive event-triggered fault-tolerant synchronization for multiagent systems," *IEEE Trans. Ind. Electron.*, vol. 68, no. 2, pp. 1537–1547, Feb. 2021.

[14] H.-J. Li, L. Wang, Y. Zhang, and M. Perc, "Optimization of identifiability for efficient community detection," *New J. Phys.*, vol. 22, no. 6, Jun. 2020, Art. no. 063035.

[15] H.-J. Li, L. Wang, Z. Bu, J. Cao, and Y. Shi, "Measuring the network vulnerability based on Markov criticality," *ACM Trans. Knowl. Discovery Data*, vol. 16, no. 2, pp. 1–24, Jul. 2021.

[16] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part II: Fault diagnosis with knowledge-based and hybrid/active approaches," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3768–3774, Jun. 2015.

[17] Z. Chen, Y. Cao, S. X. Ding, K. Zhang, and T. Koenings, "A distributed canonical correlation analysis-based fault detection method for plant-wide process monitoring," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 2710–2720, May 2019.

[18] H. J. Li, Z. Bu, Z. Wang, and J. Cao, "Dynamical clustering in electronic commerce systems via optimization and leadership expansion," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5327–5334, Aug. 2020.

[19] H.-J. Li, Z. Wang, J. Pei, J. Cao, and Y. Shi, "Optimal estimation of low-rank factors via feature level data fusion of multiplex signal systems," *IEEE Trans. Knowl. Data Eng.*, early access, Aug. 13, 2020, doi: 10.1109/TKDE.2020.3015914.

[20] H. Wang, C. Liu, D. Jiang, and Z. Jiang, "Collaborative deep learning framework for fault diagnosis in distributed complex systems," *Mech. Syst. Signal Process.*, vol. 156, Jul. 2021, Art. no. 107650.

[21] S. Tang, S. Yuan, and Y. Zhu, "Deep learning-based intelligent fault diagnosis methods toward rotating machinery," *IEEE Access*, vol. 8, pp. 9335–9346, 2020.

[22] R. Razavi-Far, E. Hallaji, M. Farajzadeh-Zanjani, M. Saif, S. H. Kia, and H. Henao, and G.-A. Capolino, "Information fusion and semi-supervised deep learning scheme for diagnosing gear faults in induction machine systems," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6331–6342, Aug. 2019.

[23] X. Cao, P. Shi, Z. Li, and M. Liu, "Neural-network-based adaptive backstepping control with application to spacecraft attitude regulation," *IEEE Trans. Neur. Net. Lear.*, vol. 29, no. 9, pp. 4303–4313, Nov. 2017.

[24] B. Xiao and S. Yin, "A deep learning based data-driven thruster fault diagnosis approach for satellite attitude control system," *IEEE Trans. Ind. Electron.*, vol. 68, no. 10, pp. 10162–10170, Oct. 2021.

[25] M. Mansouri, M. Trabelsi, H. Nounou, and M. Nounou, "Deep learning-based fault diagnosis of photovoltaic systems: A comprehensive review and enhancement prospects," *IEEE Access*, vol. 9, pp. 126286–126306, 2021.

[26] K. Choi, J. Yi, C. Park, and S. Yoon, "Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines," *IEEE Access*, vol. 9, pp. 120043–120065, 2021.

[27] M. Xia, T. Li, L. Xu, L. Liu, and C. W. de Silva, "Fault diagnosis for rotating machinery using multiple sensors and convolutional neural networks," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 1, pp. 101–110, Feb. 2018.

[28] H. Wang, S. Li, L. Song, L. Cui, and P. Wang, "An enhanced intelligent diagnosis method based on multi-sensor image fusion via improved deep learning network," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 6, pp. 2548–2657, Jun. 2020.

[29] X. Tao, D. Zhang, Z. Wang, X. Liu, H. Zhang, and D. Xu, "Detection of power line insulator defects using aerial images analyzed with convolutional neural networks," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 4, pp. 1486–1498, Apr. 2020.

[30] L. Wen, X. Li, L. Gao, and Y. Zhang, "A new convolutional neural network-based data-driven fault diagnosis method," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5990–5998, Jul. 2018.

[31] M. Gao, S. Yang, and L. Sheng, "Distributed fault estimation for time-varying multi-agent systems with sensor faults and partially decoupled disturbances," *IEEE Access*, vol. 7, pp. 147905–147913, 2019.

[32] M. Khalili, X. Zhang, M. M. Polycarpou, T. Parisini, and Y. Cao, "Distributed adaptive fault-tolerant control of uncertain multi-agent systems," *Automatica*, vol. 87, pp. 142–151, Jan. 2018.

[33] H.-J. Ma and L. Xu, "Cooperative fault diagnosis for uncertain nonlinear multiagent systems based on adaptive distributed fuzzy estimators," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1739–1751, Apr. 2020.

[34] D. Ye, M.-M. Chen, and H.-J. Yang, "Distributed adaptive event-triggered fault-tolerant consensus of multiagent systems with general linear dynamics," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 757–767, Mar. 2019.

[35] H. Hassani, R. Razavi-Far, M. Saif, and G.-A. Capolino, "Regression models with graph-regularization learning algorithms for accurate fault location in smart grids," *IEEE Syst. J.*, vol. 15, no. 2, pp. 2012–2023, Jun. 2021.

[36] C.-W. Kuo, C.-C. Tsai, and C.-T. Lee, "Intelligent leader-following consensus formation control using recurrent neural networks for small-size unmanned helicopters," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 2, pp. 1288–1301, Feb. 2021.

[37] J. A. V. Trejo, M. Adam-Medina, C. D. Garcia-Beltran, G. V. G. Ramirez, B. Y. L. Zapata, E.-M. Sanchez-Coronado, and D. Theilliol, "Robust formation control based on leader-following consensus in multi-agent systems with faults in the information exchange: Application in a fleet of unmanned aerial vehicles," *IEEE Access*, vol. 9, pp. 104940–104949, 2021.

[38] Q. Shen, Y. Shi, R. Jia, and P. Shi, "Design on type-2 fuzzy-based distributed supervisory control with backlash-like hysteresis," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 2, pp. 252–261, Feb. 2021.

[39] Q. Ouyang, Z. Wang, K. Liu, G. Xu, and Y. Li, "Optimal charging control for lithium-ion battery packs: A distributed average tracking approach," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3430–3438, May 2020.

[40] I. A. Raptis and E. Noursadeghi, "Distributed fault diagnosis of nonlinear stochastic systems with monitoring networks," *IEEE Access*, vol. 8, pp. 135933–135947, 2020.

[41] W. Zou, Z. Xiang, and C. K. Ahn, "Mean square leader–following consensus of second-order nonlinear multiagent systems with noises and unmodeled dynamics," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 12, pp. 2478–2486, Dec. 2019.

[42] Y. Yang, H. Modares, D. C. Wunsch, and Y. Yin, "Optimal containment control of unknown heterogeneous systems with active leaders," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 3, pp. 1228–1236, May 2019.

[43] W. Zou, P. Shi, Z. Xiang, and Y. Shi, "Consensus tracking control of switched stochastic nonlinear multiagent systems via event-triggered strategy," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 3, pp. 1036–1045, Mar. 2020.

[44] Y. Yang, H. Modares, K. G. Vamvoudakis, Y. Yin, and D. C. Wunsch, "Dynamic intermittent feedback design for H_∞ containment control on a directed graph," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3752–3765, Aug. 2019.

[45] J. Díez-González, R. Álvarez, N. Prieto-Fernández, and H. Perez, "Local wireless sensor networks positioning reliability under sensor failure," *Sensors*, vol. 20, no. 5, p. 1426, Mar. 2020.

[46] X. Yu, Z. Liu, and Y. Zhang, "Fault-tolerant flight control design with finite-time adaptation under actuator stuck failures," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1431–1440, Jul. 2017.

[47] A.-Y. Lu and G.-H. Yang, "Observer-based control for cyber-physical systems under Denial-of-Service with a decentralized event-triggered scheme," *IEEE Trans. Cybern.*, vol. 50, no. 12, pp. 4886–4895, Dec. 2020.

[48] R. T. Schirrmeister and J. T. Springenberg, "Deep learning with convolutional neural networks for EEG decoding and visualization," *Hum. Brain Mapping*, vol. 38, no. 11, pp. 5391–5420, Nov. 2017.

[49] X. Lu, X. Liu, B. Li, and J. Zhong, "Data-driven state prediction and sensor fault diagnosis for multi-agent systems with application to a twin rotational inverted pendulum," *Processes*, vol. 9, no. 9, p. 1505, Aug. 2021.

XIAOXU LIU received the B.S. and M.S. degrees from the Department of Mathematics, Northeastern University, China, in 2012 and 2014, respectively, and the Ph.D. degree from the Faculty of Engineering and Environment, Northumbria University, Newcastle Upon Tyne, U.K., in 2018. She is an Associate Professor at the Sino-German College of Intelligent Manufacturing, Shenzhen Technology University, China. Her research interests include robust fault diagnosis, fault-tolerant control, nonlinear systems, stochastic systems, fuzzy modeling, distributed systems, and data-driven methods.

XIN LU (Student Member, IEEE) received the B.S. degree from the Department of Automation, School of Electrical Engineering, Yanshan University, China, in 2019. He is currently pursuing the M.S. degree with the Department of Machinery, School of Applied Technology (SZTU), Shenzhen University, China. He is also a Joint Training of Graduate Student at the Sino-German College of Intelligent Manufacturing, Shenzhen Technology University, China. His research interests include data-driven fault diagnosis, deep learning, and reinforcement learning.

ZHIWEI GAO (Senior Member, IEEE) works with the Faculty of Engineering and Environment, Northumbria University, Newcastle Upon Tyne, U.K., as a Reader and the Head of the Electrical Power and Control Systems Research Group. His research interests include condition monitoring and fault diagnosis, fault tolerant control, model-based methods, machine learning and data-driven approaches, and their applications in power converters and wind turbine energy systems. He is an Academic Editor or an Editorial Member of over ten refereed journals, such as IEEE Transactions on Automatic Control, IEEE Transactions on Industrial Electronics, IEEE Transactions on Industrial Informatics, and *Renewable Energy*.