Real-time Quality Inspection of Motor Rotor Using Cost-effective Intelligent Edge System

Qingyun Zhu, Jingfeng Lu, Xiaoxian Wang, *Member, IEEE*, Hui Wang, Siliang Lu, *Senior Member, IEEE*, Clarence W. de Silva, *Fellow, IEEE*, Min Xia, *Senior Member, IEEE*

***Abstract*—Induction motors (IMs) are used extensively as driving actuators in electric vehicles. Motor rotors are prone to defects in the die casting procedure, which can significantly reduce the production quality. Benefitting from the development of Internet of things (IoT) techniques and edge computing, this study designed an instrumentation system for the fast inspection of rotor defects to meet the objectives of efficient and high-quality rotor production. First, an electromagnetic sensing device is designed to acquire the induced voltage signal of the rotor under investigation. Second, a residual multiscale feature fusion convolutional neural network model is designed to extract the hierarchical features of the signal, to facilitate defect recognition. The developed algorithm is deployed into a cost-effective edge computing node that includes a signal acquisition circuit and a Raspberry Pi microcontroller. The conducted experimental studies show that this implementation can achieve an inference time of less than 200 ms and accuracy of more than 99 %. It is shown that the designed system exhibits superior performance when compared with conventional methods. The developed, compact and flexible handheld solution with enhanced deep learning techniques shows outstanding potential for use in real-time rotor defect detection.**

***Index Terms*—IM, rotor defect detection, IoT, multiscale feature fusion, convolutional neural network, edge computing**

I. Introduction

ALL seem highly concerned about and value resource conservation and carbon reduction; thus, the electric vehicle (EV) industry is rapidly flooding the market [1, 2]. The pivotal components of an EV are the drive motors, among which induction motors (IMs) are the most widely used [3]. The motor rotor, as the key component in an IM, considerably affects the performance and efficiency of the IM [4]. Die casting is a mainstream production process of IM rotors. As the die cast rotor goes through a complex manufacturing process, multiple defects could occur/emerge in the rotor bars during production, such as porosity and destruction [5], which will result in low yield and must be detected in a timely manner. Thus, methods for accurate evaluation of rotor quality are necessary, and considerable research has been devoted to the detection of rotor faults [6].

Deep learning (DL) techniques are widely employed in fault diagnosis [7]. For instance, Shao et al. presented a framework based on an improved convolutional neural network (CNN) with transfer learning for fault diagnosis of rotor bearing systems under different operating conditions [8]. Jiao et al. put forward a new CNN for intelligent diagnosis using complementary data to integrate information fusion, feature extraction and fault classification [9]. Liu et al. proposed a new fault diagnosis framework based on the characteristics of industrial vibration signals, which they used a novel dislocated time series CNN. This model is designed to extract the relationship between signals at various intervals in periodic mechanical signals, which overcame the disadvantages of conventional CNNs. This approach is appropriate for modern electrical machines, especially under nonstationary conditions [10]. Xiao et al. firstly took cross-domain case from simulation domain to experimental domain into consideration, and developed promising joint adaptation network, which contributes to unsupervised transfer fault diagnosis [11]. Wang et al. exploited a novel convolutional deep belief network, which is applicable to fault diagnosis [12].

The mentioned methods represent a certain contribution in fault diagnosis. However, most of the available methods are employed to handle offline data and are conducted on desktops or servers. Many practical industry applications, such as quality inspection of die cast rotors require real-time performance and convenience. Most of the existing deep learning-based methods are incapable of acceptable performance and should be further improved.

With the rise of Internet of things (IoT), many of IoT techniques are used in different fields (eg., 5G [13], unmanned aerial vehicle [14], and fault diagnosis [15-19]). Meanwhile, edge computing is a new computing paradigm that enables fast detection through the deployment of algorithms that are embedded in distributed nodes [20]. Hence, the data transmitted and preprocessed by IoT techniques can be conducted by edge computing, and combined with DL models to realize high efficiency detection [21-24].

This work was supported in part by the National Natural Science Foundation of China under Grants 52075002 and 62203010, and the Project of the Outstanding Young Talents in Colleges and Universities of Anhui Province under Grant gxyqZD2022006. (*Corresponding authors:Siliang Lu; Min Xia*)

Qingyun Zhu, Jingfeng Lu, Hui Wang, and Siliang Lu are with the College of Electrical Engineering and Automation, Anhui University, Hefei 230601, China. (e-mail: z19301104@stu.ahu.edu.cn; z21301139@stu.ahu.edu.cn; wanghui@stu.ahu.edu.cn; lusliang@mail.ustc.edu.cn)

Xiaoxian Wang is with the College of Electronics and Information Engineering, Anhui University, Hefei, 230601, China, and also with the Department of Precision Machinery and Precision Instrumentation, University of Science and Technology of China, Hefei 230027, China. (e-mail: xiaoxian@ahu.edu.cn)

C.W. de Silva is with the Department of Mechanical Engineering, The University of British Columbia, Vancouver V6T 1Z4, Canada (email: desilva@mech.ubc.ca)

M. Xia is with the Department of Engineering, Lancaster University, Lancaster LA1 4YW, U.K. (e-mail: m.xia3@lancaster.ac.uk)

In light of the presented discussion, the present study introduced an approach that incorporates DL models and edge computing to achieve real-time rotor defect detection (RDD), focusing on factory application. The main process of the proposed method is as follows: 1) a sensor is designed for an IM stator to output the induced voltage signals, 2) an embedded system is designed for the data acquisition and transmission, 3) data pretreatment is conducted, and 4) a designed CNN model that is trained on a desktop with offline data is used to process the online signals on a Raspberry Pi microcontroller, producing the quality inspection result of the motor.

The novelty and primary contributions of the present study are the following. 1) A new sensor is designed based on an IM stator to output the induced voltage signals by directly changing the wiring method of the three-phase winding, which substantially simplifies the detection process. A sensing approach that is of low cost and handy for industry implementation. 2) A residual multiscale feature fusion CNN (RMFFCNN) model is designed to extract distinct and hierarchical features from the sensor signals. Besides high recognition accuracy, this DL model has benefits including fast convergence in training, suitability for limited training samples, and good anti-noise capacity. 3) The design of a compact system, including a microcontroller unit (MCU) and an analog-to-digital converter (ADC), which are used for the data acquisition and transmission, and a Raspberry Pi used as the platform for the CNN model. The developed solution is demonstrated to exhibit high accuracy, flexibility, cost-effectiveness, and efficiency and is superior to state-of-the-art methods that typically include complex signal sensing and large machine learning models.

The remaining sections of the present paper are organized as shown below. Section II introduces the designed sensor, the designed embedded system, and the experimental rotors. Section III introduces the novel RMFFCNN model and the edge computing framework. Section IV presents the results of the RMFFCNN training and real-time RDD. Section V compares the developed method with existing methods. Section VI discusses the research trends and future works. Finally, Section VII provides the conclusion.

II. Hardware Design and Experimental Rotors

In the present section, the sensor and the embedded system, which are designed in the present work, and the nine IM rotors that are used in the experiments are described, to demonstrate the system design and the experimental setup.

A. Designed Sensor

The sensor that is designed in the present work is based on an IM stator, as shown in Fig. 1(a), in which the IM stator is converted directly into an electromagnetic sensor [25]. Different from the traditional techniques of wiring and electrification of the stator windings, where the rotor windings are not externally excited [3], the innovative wiring method is depicted in Fig. 1(b). Phases A and B are powered by a constant DC voltage to provide a constant magnetic field to the sensor, as in a synchronous motor. When the rotor spins to intercept the magnetic field in the sensor, an induced voltage $U_{s,c}$ is caused in phase C according to the principle of electromagnetic induction. The induced voltage signal $U_{s,c}$ is used as the sensed signal. The designed sensor is simple and convenient, and presents considerable potential in real-time RDD for practical industry application. *Note*: A Hall effect sensor may be used as well as for the present purpose [3].

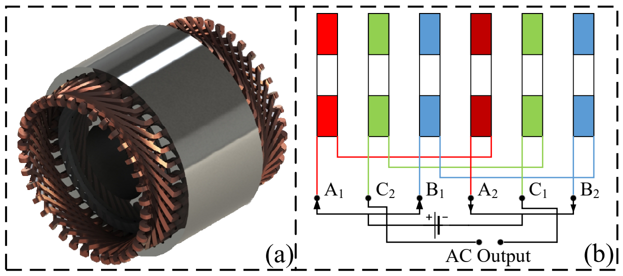


Fig. 1. The designed sensor based on (a) IM stator; (b) wiring of the sensor.

B. Experimental Setup and Edge Computing Node

The designed hardware devices are displayed in Fig. 2. Besides the designed sensor for generating the rotor detection signal, an embedded system is designed to implement real-time CNN inference, including an ADC (AD7606, Analog Devices, Inc.), an MCU (STM32H7, STMicroelectronics, Inc.), and a single-board microcomputer (SBC) (Raspberry Pi 4 Model B, Element14, Inc.) A 22.5-inch LCD monitor connected to the SBC via a micro-HDMI port is used to show the real-time results. A laptop serves as the host, and is used to store the experimental data and results that are unnecessary in real-time RDD.

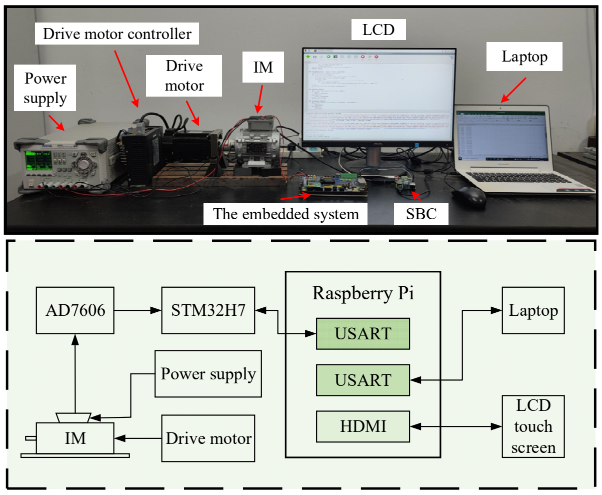


Fig. 2. Experimental setup.

First, the drive motor rotates the rotor that is attached by a mechanical coupling, and a servo motor controller is used to control the speed of the rotor rotation. The sensor is powered at 30 V by a regulated DC power supply. The induced voltage signal $U_{s,c}$ that is generated in the designed sensor is then real-time sampled by the ADC. Second, the MCU is used for acquiring and transmitting the signals to the SBC. Specifically, the ADC is controlled by the MCU, for accurate setting of the real-time period. In the present manner, the SBC is involved in real-time operation. In the present study, the sampling frequency is initially configured to 200 kHz. Finally, the designed CNN model is utilized to handle the induced voltage signal on the SBC. When the real-time input data are processed, the results are immediately displayed on the LCD monitor. Universal synchronous asynchronous receiver transmitters are used for the communication between the SBC, the MCU, and the laptop. The STM32H7, the AD7606, and the SBC are powered at +3.3 +5 V, and +5 V, respectively. In addition, the Raspberry Pi may be controlled by the desktop visual interface through Windows Remote Desktop Connection. The entire system possesses the simplicity and flexibility of the design, which is particularly appropriate in real-time RDD in practical applications.

C. *Experimental Rotors*

The IM parameters in the experiment are presented in Table I, and the nine IM rotors used are depicted in Fig. 3. The different types of defects of rotors include those that are healthy, those with various degrees of porosity faults, and those with various degrees of broken bar faults. The rotor faults are created as follows. 1) The rotor porosity defect is manufactured by drilling a hole in the rotor bar using a drilling machine. The hold does not disconnect the entire rotor bar, allowing the current to run through the rotor bar. The degree of rotor porosity defect is indicated by the number of holes. 2) The rotor with broken bar is manufactured by drilling hole whose dimension is larger than that of the bar, which leads to the rotor bar being completely broken. Thus, the entire rotor bar becomes open-circuit connection. The degree of rotor broken bar defect is also indicated by the number of drilled holes. This configuration can simulate the actual rotor defects. In the present study, the corresponding rotor faults are labeled 0–8 as shown in Fig. 3.

TABLE I
IM PARAMETERS

No. of phases	Rated power (W)	Rated voltage (VAC)	Rated current (A)
3	90	380	0.39

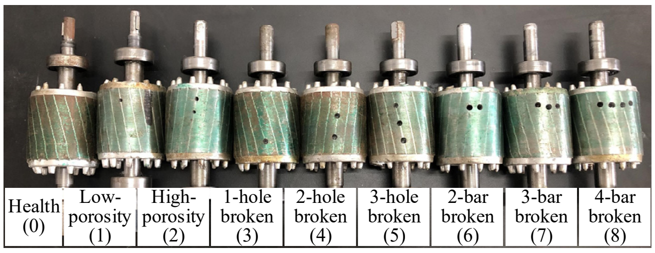


Fig. 3. Experimental IM rotors.

III. PROPOSED ALGORITHMS FOR REAL-TIME RDD

In the present study, a novel data pretreatment method is adopted before subjecting to CNN training, to reduce the redundancy, prevent overfitting, enhance the robustness, and improve the classification accuracy. In addition, the novel RMFFCNN method and edge computing framework are presented.

A. *Induced Voltage Signal Pretreatment*

The signal pretreatment process is shown in Fig. 4. The initial induced voltage $U_{s,c}$ waveforms collected from the data acquisition system are shown in Fig. 5. The rotors include a healthy rotor and defect ones with different fault degrees. Obviously, the time domain features of $U_{s,c}$, such as the amplitude and the shape, differ for the various rotor defect. The oscillation modes of the signals are also different, indicating that the signal features are located at different scales on frequency domain.

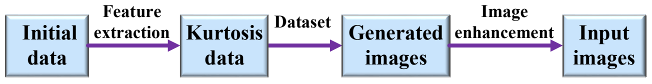


Fig. 4. Flowchart of signal pretreatment.

The induced voltage signal from the IM is discretized as follows:

$$U_K[n] \tag{1}$$

where n = 1,2, ..., N_k, and N_k is the number of samples. Then, kurtosis is extracted from the initial data. This method can obtain a reasonable data processing effect based on multiple experimental results. The kurtosis U_K may be expressed as

$$K = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i - \bar{x}}{\sigma_t}\right)^4 \tag{2}$$

where x_i is the signal value, x is the average value of the signal values. N is the number of signals used. σ_t is the standard deviation. In this present study, a kurtosis is taken for every five data points ($N = 5$).

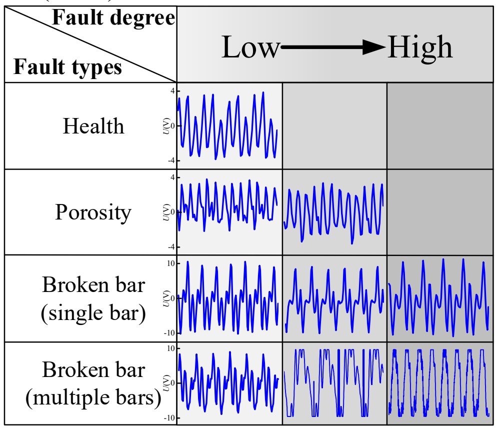


Fig. 5. Initial induced voltage waveforms.

Second, the three-dimensional (3D) kurtosis data is obtained by duplicating the one-dimensional (1D) kurtosis data, and the 3D kurtosis data are transformed into the two-dimensional matrix of a three-channel image, which can be expressed as

$$\begin{cases} U_K(i,j,1) = u_K[p+(j-1)\times q+i] \\ U_K(i,j,2) = u_K[p+(y-1)\times q+i] \\ U_K(i,j,3) = u_K[p+(y-1)\times q+i] \end{cases} \tag{3}$$

in which i =1, 2, ..., q (q are the numbers of rows in the two-dimensional matrix); j = 1, 2, ..., n (n are the numbers of columns in the two-dimensional matrix); and p is the location of the stochastically chosen preliminary point. With the two-dimensional matrix acquired, its values are mapped to numbers in the range between 0 and 255 to make the image, which are expressed as

$$U_K(u,v,w) = \frac{U_K(u,v,w) - \min(U_K(:,:,w))}{\max(U_K(:,:,w)) - \min(U_K(:,:,w))} \times 255 \tag{4}$$

where $w = 1,2,3$. The generated images of the nine experimental IM rotors are shown in Fig. 6, in which either slight or significant differences can be observed between the different fault type images. The white dots dispersed in the black background have different distributed modes for different types of defects. The dense or sparse levels of the white dots indicate that the images present multi-scale features. Hence, a multi-scale feature fusion approach is investigated to effectively extract the image features and pave the way for high-accuracy RDD. In the present study, the size of each image is set to 28 × 28 pixels, consisting of 3,920 induced voltage signal lengths.

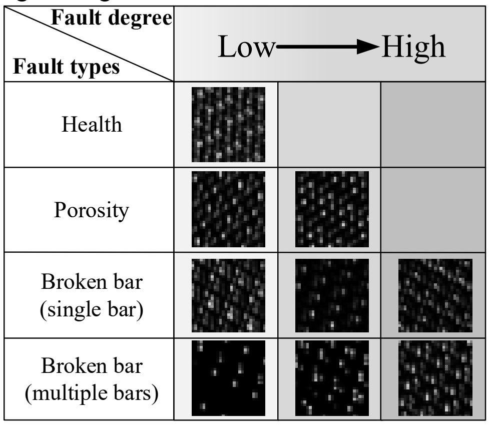

Fault degree
Fault types
Low → High
Health
Porosity
Broken bar (single bar)
Broken bar (multiple bars)

Fig. 6. Generated images of the nine experimental rotors.

Third, as the generated images have the characteristics of a small amount of information and a simple information distribution, scaling and translation transformation methods are used to process the generated images, address the problem of network overfitting and resist the interference caused by external factors of the front sensor. Hence, four times scaling and translation transformation is applied to each generated image to obtain the input images, which are shown in Figs. 7(a)–(d). It can be noticed that the generated image is zoomed and shifted after processing. This method can further expand the dataset and strengthen the robustness of the network.

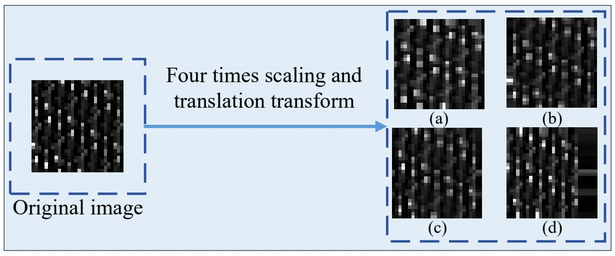


Fig. 7. Four times scaling and translation transformation of the generated image.

B. RMFFCNN

In view of the features of the input images and the limited computing resources of the designed embedded system, the RMFFCNN based on a residual structure and multiscale feature fusion is designed. The main structure of the module consists of 1) multiscale feature fusion and 2) a residual structure consisting of mixed features and the shortcut, which differs from existing methods that use only multiscale feature fusion in the input layer. The residual structure lacks a deep integration with the other structures; thus, its performance cannot be fully exploited. The RMFFCNN performs multiscale feature fusion in the input layer, innovatively uses multiscale feature fusion as the basic unit that is constituted in the network, and combines multiscale feature fusion with the residual structure to form the hidden layer of the network. This can considerably improve the diversity and parameter utilization of the convolutional kernel, reduce the training difficulty of the network parameters, and further enhance the RMFFCNN performance. The construction of the RMFFCNN is described in Fig. 8. This model is primarily comprised of three parts, introduced as follows.

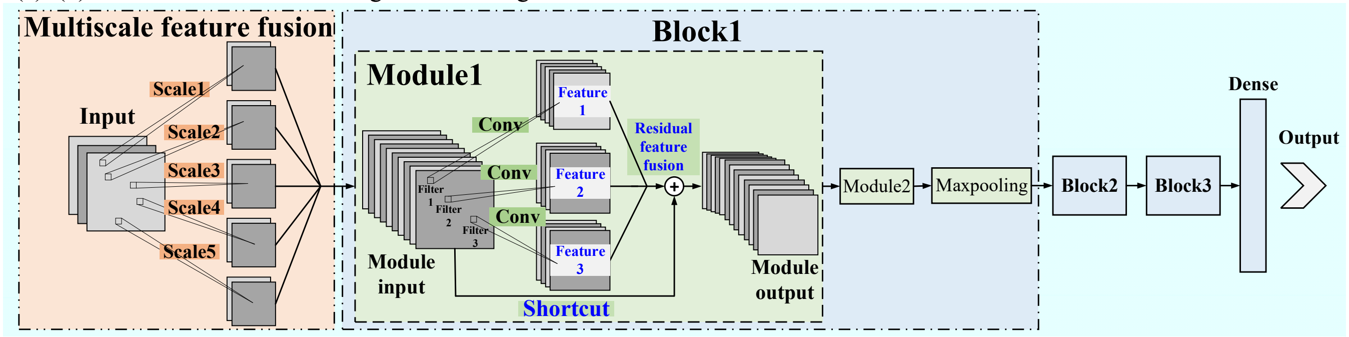


Fig. 8. RMFFCNN architecture.

1) The input layer is the first part, where multiscale feature fusion is conducted. Multiscale feature fusion provides a solution to the low hardware efficiency problem for no uniform sparse data computation by using convolutional kernels of different sizes to extract the target features from different scales of the receptive domain [26]. This layer has a relatively wide network width and uses a large convolutional kernel size, which is expressed as

$$\sigma_j^i\left(x_{ms}\right) = f\left(k_j^i * x_{ms} + b_j^i\right) \tag{5}$$

$$y_{ms} = \text{concat}\left\{\sigma_j^i(x_{ms}) \middle| i \in (1,2,\ldots 5), j \in (1,2)\right\} \tag{6}$$

where $\sigma_j^i(\cdot)$ is the ith channel output on the jth scale, b_j^i is the corresponding bias, "*" is the convolution operation, x_{ms} is the input of the multiscale feature mixing layer, y_{ms} is the output of the multiscale feature mixing layer, k_j^i represents the jth convolution kernel on the ith scale. In addition, $f(\cdot)$ represents the activation function, and it is a ReLU function when not specified otherwise throughout the RMFFCNN, and concat$\{\cdot\}$ is the matrix splicing operation.

<

2) The hidden layer is based on the residual structure and the feature fusion structure. The essence of a residual connection is mapping shallow features directly on the deep features in the network structure, which is a well-established solution to the problem of the recognition accuracy not increasing with the deepening of the number of layers in the network and computational resources being consumed, vanishing gradient, and exacerbated overfitting [27]. The module is the basic unit of the hidden layer, multiple modules constitute a block after the maximum pool operation, and the hidden layer is composed of multiple blocks according to the modular design idea, which may be expressed as [28]

$$y_{h,l}^{m} = x_{h,l}^{m} + \mathrm{concat}\left\{\sigma_{j}^{i}(x_{h,l}^{m}) \middle| 1 < i \le I_{h,l}^{m}, 0 < j \le J_{h,l}^{m}\right\} \tag{7}$$

$$y_{h}^{b} = \mathrm{maxpooling}\left\{y_{h,L}^{m}\right\} \tag{8}$$

where $y_{h,l}^{m}$ is the output of the lth module in the hth block, l *is* between the interval (1, L_h), L_h is the maximum number of modules in the hth block, $y_{h,L}^{m}$ is the output of the last module in the block, y_{h}^{b} is the output of the hth block, h is between the interval (1, H), H is the maximum value of the number of blocks, $x_{h,l}^{m}$ is the input of the lth module in the hth block, $I_{h,l}^{m}$ is the number of features of the different scales in the lth module of the hth block, $J_{h,l}^{m}$ is the total number of channels in the features mixing in the lth module in the hth block, and $\mathrm{maxpooling}\{\cdot\}$ is the maximum pooling operation.

3) The output layer is fully connected to the hidden layer in order to output a probability vector as the prediction result, by expanding the convolution result into a 1D vector and activating the output, expressed as

$$x_0 = \mathrm{flatten}\left\{y_{H}^{b}\right\} \tag{9}$$

$$y = \mathrm{softmax}(f(wx_0 + b)) \tag{10}$$

where y_{H}^{b} is the output of the last layer of the network block, flatten $\{\cdot$- indicates the expansion of the output tensor of the hidden layer into a 1D vector, b is the corresponding bias, x_0 is the input of the output layer, w is the weight of the output layer, and y is the final output of the network obtained after activation by the softmax function. Additionally, Module 2 has the same structure as Module 1, but with different model parameters. The detailed configuration parameters of the whole network are illustrated in Table II, in which '1×1', '2×2',…, represent the size of the convolution kernels. '3×3 reduce' and '5×5 reduce' stand for the number of 1×1 filters in the reduction layer used before '3×3' and '5×5' convolutions, respectively. '10976FC' and '9FC' represent that the number of fully connected neurons are 10976 and 9, respectively.

The pseudo code for implementation of the proposed RMFFCNN model is shown in Algorithm 1. Steps 1-18 show the creation process of the Module. Firstly, the input tensor is operated differently for 4 paths, and then it is connected to the input for residuals. The output is generated after maximum pooling and normalization. Steps 19-31 show the forward propagation process of the whole network. Firstly, multi-scale feature extraction is conducted from the input data, and then the data stream is processed by several Modules. Finally, the 3D tensor is expanded into a 1D tensor, and then mapped to a classification probability vector and output. This paper adopts a modular engineering idea to combine the residual and multi-scale feature extraction structures into code blocks. Such an architecture provides feasibility to adjust the model size for implementing onto the edge computing nodes with different computing power and storage resources, thereby improving the model robustness and engineering applicability.

Algorithm 1

Pseudo code for the RMFFCNN model

```
1:  // Module of the model
2:  Procedure Module(input):
3:      pathway1 ← Conv2D (filter = 1×1, input)
4:      pathway1 ← Relu(pathway1)
5:      pathway2 ← Conv2D (filter = 1×1, input)
6:      pathway2 ← Conv2D (filter = 3×3, pathway2)
7:      pathway2 ← Relu (pathway2)
8:      pathway3 ← Conv2D (filter = 1×1, input)
9:      pathway3 ← Conv2D (filter = 5×5, pathway3)
10:     pathway3 ← Relu (pathway3)
11:     Pathway4 ← MaxPooling2D (size = 3×3, input)
12:     Pathway4 ← Conv2D (filter = 1×1, pathway 4)
13:     pathway4 ← Relu (pathway4)
14:     // Residual connections to the merge matrix of all paths
15:     Mout ← input + Concat{pathway_1, pathway_2, pathway_3, pathway_4}
16:     Mout ← MaxPooling (size = 3×3, Mout)
17:     Mout ← BatchNormalization (Mout)
18: Return Mout
19: // Input: Initialize a placeholder with the shape (28, 28, 3)
20: // Multiscale feature fusion
21: for i=1: N do
22:     ∀i ∈ N , // Multi-scale feature extraction using different N kernels
23:     out_i ← Conv2D (filter = i×i, input)
24:     out_i ← Relu (out_i)
25: end for
26:     out ← Concat{out_1, out_2, ..., out_N},
27: // The data flow executes step 2~18 repeatedly
28: // Expanding a 3D tensor into a 1D vector
29: out_1D←Flatten(out_3D)
30: // Mapping to classification vector through full connection layer
31: out_model←Dense (out_1D)
```

TABLE II
PARAMETERS OF RMFFCNN CONFIGURATION

Type		1×1	2×2	3×3 reduce	3×3	4×4	5×5 reduce	5×5
Input layer		2	2	Null	2	2	Null	2
Block-1	B1-Module1	16	Null	24	32	Null	4	8
	B1-Module 2	16	Null	24	32	Null	4	8
Block-2	B2-Module 1	32	Null	48	64	Null	8	16
	B2-Module 2	32	Null	48	64	Null	8	16
	B2-Module 3	32	Null	48	64	Null	8	16
Block-3	B3-Module 1	64	Null	96	128	Null	16	32
	B3-Module 2	64	Null	96	128	Null	13	32
Flatten								
Linear		10976FC						
Softmax		9FC						

C. *Edge Computing Framework*

The edge computing framework for real-time RDD includes RMFFCNN training and real-time inference, which is depicted in Fig. 9. First, the training of RMFFCNN is carried out on a configured desktop computer to minimize the computational time, as shown in the middle part of Fig. 9. Then, a well-trained RMFFCNN is obtained from the training, and the offline data are used for validation before the real-time RDD. Finally, the validated RMFFCNN model is deployed into the SBC, to conduct the real-time RDD. Second, numerous sampling points generated in the designed sensor are acquired and transmitted by the data acquisition and transmission equipment (ADC and MCU) and finally, to the Raspberry Pi.

Next, the induced voltage signals are preprocessed to obtain the input images, as mentioned in the previous section. The images are processed by the RMFFCNN model on the Raspberry Pi for real-time RDD, as shown in the bottom part of Fig. 9. The training of RMFFCNN and real-time inference are carried out on the Keras platform using Python, which is displayed in the upper right hand part of Fig. 9. This approach combines different software (Windows and Linux) and hardware (x64 CPUs and ARM CPUs) to achieve versatility and interoperability in edge computing [29], which is suitable for real-time RDD in an industrial production line.

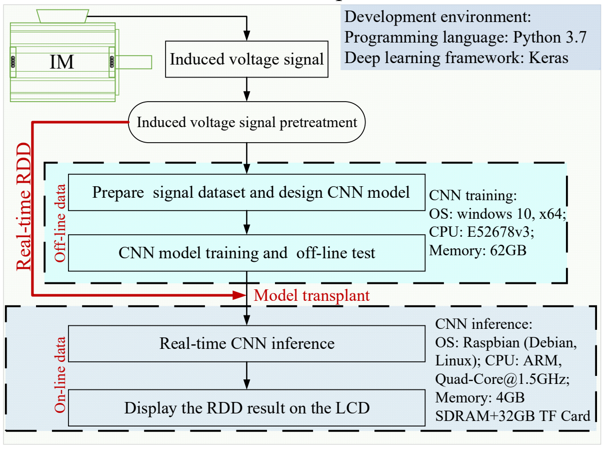

IM
Induced voltage signal
Development environment:
Programming language: Python 3.7
Deep learning framework: Keras
Induced voltage signal pretreatment
Real-time RDD
Off-line data
Prepare signal dataset and design CNN model
CNN model training and off-line test
CNN training:
OS: windows 10, x64;
CPU: E52678v3;
Memory: 62GB
Model transplant
On-line data
Real-time CNN inference
Display the RDD result on the LCD
CNN inference:
OS: Raspbian (Debian, Linux); CPU: ARM, Quad-Core@1.5GHz;
Memory: 4GB
SDRAM+32GB TF Card

Fig. 9. Edge computing framework.

IV. Effectiveness Validation

In this section, RMFFCNN training, validation, and testing are conducted using offline signals. In addition, the RMFFCNN is executed on the Raspberry Pi to handle the online signals that are acquired and transmitted by the embedded system, to achieve real-time RDD.

A. *RMFFCNN Training, Validation, and Testing*

A total of 20,000 images without overlaps are generated from the acquired induced voltage signals for each rotor defect fault. That is, a total of 180,000 offline images are used for the RMFFCNN training, validation, and testing, with a ratio of 6:2:2, on the desktop computer. In this study, the batch size is set as 32 after many repeated experiments. The optimal batch size may change depending on the hardware, such as different CPUs and GPUs. The training trends of the RMFFCNN model are illustrated in Fig. 10, and Table III displays the training, validation, and testing accuracy. It can be concluded that the RMFFCNN has a fast convergence speed and high classification accuracy.

TABLE III
Training, Validation, and Testing Accuracies

Method	Training accuracy (%)	Validation accuracy (%)	Testing Accuracy (%)
RMFFCNN	100.00	100.00	99.69

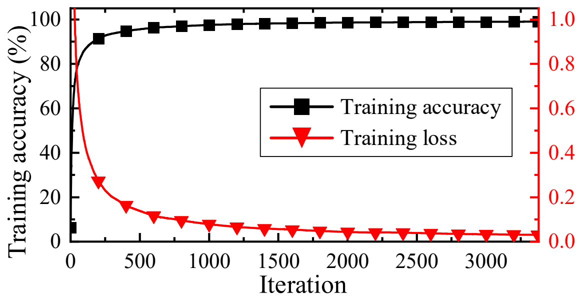
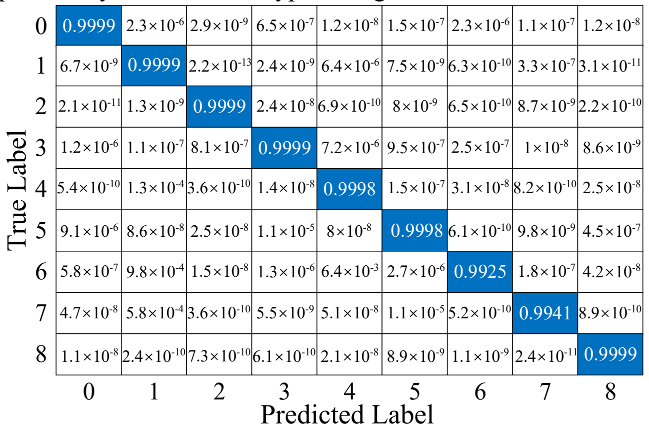


Fig. 10. RMFFCNN training accuracy and loss curves.

B. *Real-time Inference in Edge Computing Node*

The files generated by the RMFFCNN model are replicated and stored on the SD memory card of the embedded system. Next, real-time inference is realized in the designed embedded system under the edge computing framework as described in Section III. C. When the real-time inference is completed, the results are immediately displayed on the LCD monitor. Subsequently, 4,000 images for each fault type are collected and processed in real-time for real-time inference, and the classification accuracy of each category is calculated and analyzed, as shown in Fig. 11. All the predicted rotor defect types are corresponded to the actual ones, and the overall probability of nine defect types is higher than 99 %.


True Label
Predicted Label

	0	1	2	3	4	5	6	7	8
0	0.9999	2.3×10^{-6}	2.9×10^{-9}	6.5×10^{-7}	1.2×10^{-8}	1.5×10^{-7}	2.3×10^{-6}	1.1×10^{-7}	1.2×10^{-8}
1	6.7×10^{-9}	0.9999	2.2×10^{-13}	2.4×10^{-9}	6.4×10^{-6}	7.5×10^{-9}	6.3×10^{-10}	3.3×10^{-7}	3.1×10^{-11}
2	2.1×10^{-11}	1.3×10^{-9}	0.9999	2.4×10^{-8}	6.9×10^{-10}	8×10^{-9}	6.5×10^{-10}	8.7×10^{-9}	2.2×10^{-10}
3	1.2×10^{-6}	1.1×10^{-7}	8.1×10^{-7}	0.9999	7.2×10^{-6}	9.5×10^{-7}	2.5×10^{-7}	1×10^{-8}	8.6×10^{-9}
4	5.4×10^{-10}	1.3×10^{-4}	3.6×10^{-10}	1.4×10^{-8}	0.9998	1.5×10^{-7}	3.1×10^{-8}	8.2×10^{-10}	2.5×10^{-8}
5	9.1×10^{-6}	8.6×10^{-8}	2.5×10^{-8}	1.1×10^{-5}	8×10^{-8}	0.9998	6.1×10^{-10}	9.8×10^{-9}	4.5×10^{-7}
6	5.8×10^{-7}	9.8×10^{-4}	1.5×10^{-8}	1.3×10^{-6}	6.4×10^{-3}	2.7×10^{-6}	0.9925	1.8×10^{-7}	4.2×10^{-8}
7	4.7×10^{-8}	5.8×10^{-4}	3.6×10^{-10}	5.5×10^{-9}	5.1×10^{-8}	1.1×10^{-5}	5.2×10^{-10}	0.9941	8.9×10^{-10}
8	1.1×10^{-8}	2.4×10^{-10}	7.3×10^{-10}	6.1×10^{-10}	2.1×10^{-8}	8.9×10^{-9}	1.1×10^{-9}	2.4×10^{-11}	0.9999

Fig. 11. The confusion matrix of the RMFFCNN model for real-time inference.

V. Performance Evaluation

The performance of the proposed method in real-time RDD applications is affected by the instrument system, DL model, and edge computing platform. To further demonstrate the superiority of the proposed method in these three aspects, a comparison among the proposed method and state-of-the-art methods is carried out.

<

A. Introduction of Comparative Methods

The comparative methods are briefly introduced as shown below.

The first one is the convolutional attention neural network (CANN) method [30]. This method proposed by Tran et al. achieved higher fault diagnosis accuracy for IM diagnosis by combining the continuous wavelet transform with a CANN model.

The second one is the Enhanced CNN (ECNN) method that is implemented on an embedded system for real time motor fault diagnosis [29]. Specially, the performance on embedded systems is compared between the proposed method and ECNN method.

GoogLeNet model in Ref. [31] developed an inception structure and the concept of feature fusion, and it is used in a broad range of fields. GoogLeNet model is compared with the RMFFCNN in the present work, to explore the influence of the residual structure on the performance of the feature fusion structure.

AlexNet model in Ref. [32] proposed a regularization method called dropout. The method, which is a typical image recognition technique and widely applied in fault diagnosis, demonstrates excellent network performance.

ResNet model in Ref. [27] first introduced the residual structure. To be as close as possible to the parameters of the RMFFCNN model, ResNet18 is used as a comparison network to explore the effect of the residual structure on the performance of the feature fusion structure.

B. Effects of the DL Models on Classification Accuracy

1) Training and Testing Performance

The training and testing procedures of the six methods are the same as those presented in previous section. The training results of the six methods for the first epoch are displayed in Fig. 12. It can be shown that among the six methods, the proposed method demonstrates the highest training accuracy. To further examine the performances of different DL models, 10 independent tests are conducted and the average testing accuracies are calculated as shown in Fig. 13. It can be seen that the proposed method has the highest average testing accuracy (96.64%) and the lowest standard deviation when compared with the other methods, thereby demonstrating excellent recognition accuracy and stability of the RMFFCNN model.

2) Effects of Training Samples Number and Noise Interference

When edge computing devices process large amounts of data, edge devices can increase device latency, energy consumption, and reduce system reliability [33]. Therefore, it is necessary for models deployed in edge devices to use fewer data to achieve high classification accuracy. To explore the influence of the dataset size on the methods and further demonstrate whether the proposed method can achieve relatively high classification accuracy by using less data, different numbers of training samples including 108000, 19200, 16000, 12800, and 9600, are used in the independent experiment. The experimental results are shown in the 2nd to 6th columns in Table IV. It is obvious that, compared with the other methods, the proposed method has reached the highest fault identification accuracy in each dataset. Moreover, GoogLeNet, AlexNet, and ResNet18 are hardly able to complete the convergence, owing to the large number of parameters and layers. Hence, it can be summarized that the proposed method exhibits outstanding performance and can achieve relatively high classification accuracy by using less data, thereby meeting the requirements of edge computing solutions.

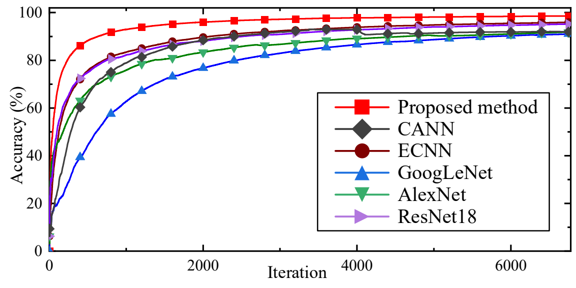


Fig. 12. Training accuracy of six methods for the first epoch.

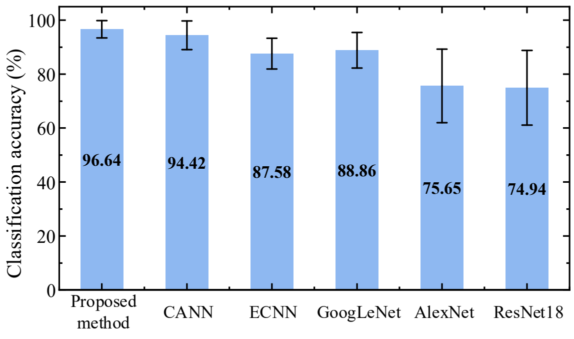


Fig. 13. Average accuracy and standard deviation of the six methods.

In practice, the signals are easily corrupted by the background noise especially in rotor production line. Given this, the anti-noise performance of different methods is also tested by adding external interference noise. In practical factory settings, the interference noise is typically a time-varying signal. Therefore, to effectively simulate real-noise interference for real-time RDD, salt-and-pepper noise is added to the dataset, which can be expressed as

$$\begin{cases} y_{ij} = \begin{cases} 255, seed \le Den \\ x_{ij}, seed > Den \end{cases} \\ seed \sim U(0,1) \end{cases} \tag{11}$$

where x_{ij} represents the elements of ith row and jth column in the input matrix before noise is added; y_{ij} represents the elements of ith row and jth column in the output matrix after noise is added; *Den* represents the noise density, that is, the proportion of white noise points in the image; and *seed* represents random numbers that conform to the uniform distribution within the interval (0, 1).

In the experiments, the noises with different intensities respect to the normalized image magnitude are injected into the training samples. The testing accuracies under different noise intensities are summarized in the 7th to 11th columns in Table IV. It can be seen that the overall testing accuracy decreases with the increase of noise intensity. Nevertheless, the proposed method still maintains the highest accuracy as compared with the other methods. This result demonstrates that the proposed method has high anti-noise capacity and robustness, which will be beneficial to practical applications.

The proposed RMFFCNN model combines the residual structure and multi-scale feature fusion mechanism. The hierarchical features are extracted simultaneously by adding convolutional kernels with different scales. In addition, the residual structure further increases the generalization and robustness of the model, and effectively alleviates the gradient explosion and training overfitting problems caused by the increased number of layers. The combination of multi-scale feature extraction and residual structure finally improves the recognition accuracy, model stability, and robustness.

TABLE IV
TESTING ACCURACY FOR DIFFERENT TRAINING SAMPLES NUMBER AND NOISE INTENSITY

Method	C = 108000 δ = 0	C = 19200 δ = 0	C = 16000 δ = 0	C = 12800 δ = 0	C = 9600 δ = 0	C = 108000 δ = 0.02	C = 108000 δ = 0.04	C = 108000 δ = 0.06	C = 108000 δ = 0.08	C = 108000 δ = 0.10
Proposed	96.64	94.14	92.44	90.82	89.43	87.31	85.32	84.87	84.25	83.97
CANN	94.42	89.03	76.24	60.33	52.32	84.89	83.26	81.37	78.65	73.53
ECNN	87.58	86.93	73.89	74.06	73.60	79.29	78.84	76.52	66.27	63.68
GoogLeNet	88.86	66.80	70.04	61.55	51.59	84.14	83.81	79.18	78.58	75.19
AlexNet	75.60	87.20	81.30	80.14	78.24	33.53	25.07	23.12	22.71	21.58
ResNet18	74.94	85.59	85.54	80.62	78.66	82.75	71.12	66.46	58.42	57.38

C is the number of training samples, δ is the noise intensity

C. *Effects of Model Size and Computing Time on Edge Computing Platform*

With the development of semiconductor technology, the computing capacity of edge computing nodes improves rapidly in recent years. RDD can be realized by implementing the DL models into the edge computing platforms. In this subsection, the effects of DL model's performance on edge computing are evaluated. The relevant experimental parameters and results for the six methods are listed in Table V. Note that the AlexNet and ResNet18 models cannot be implemented on the Raspberry Pi because their model sizes are too large. The model size in Table V is just the memory space for storing the model's parameters. The execution of the DL model requires much more available memory on the edge computing system. The results of multiple tries indicate that the maximal size of the model that can be successfully implemented on the Raspberry Pi (4 GB memory) is 100 MB. The training time is recorded on the desktop computers with x64-architecture CPUs. The inference time is recorded on the Raspberry Pi platform with ARM-architecture CPUs. It can be seen that the proposed method and ECNN methods have a small model size, lower training time and inference time. The model size and training time of the AlexNet and ResNet18 are obviously larger than those of other methods, and hence these two models are not suitable to be implemented on edge computing platforms with limited computation capacity and storage space.

TABLE V
COMPARISON OF THE MODEL SIZE, TRAINING TIME, AND INFERENCE TIME

Method	Proposed method	CANN	ECNN	GoogLeNet	AlexNet	ResNet18
Model size (MB)	7.95	27.38	2.41	68.91	9356.27	134.43
Training time (s)	94.05	137.87	64.57	160.89	484.48	965.25
Inference (ms)	170.18	213.25	145.79	876.16	Null	Null

To further examine the influence of the model size on the performance, the model size of the proposed method and ECNN is changed to approximately 2.00, 7.00, and 11.00 MB, respectively, by increasing or decreasing the network parameters. For the proposed RMFFCNN model, according to the network structure shown in Fig. 8, the model size can be adjusted while maintaining structural consistency by simply adding or deleting the Block structures. For the ECNN model, the parameters of the network are added or deleted according to the composition rules of each layer.

The comparative results are presented in Table VI. As shown in the 3rd column in Table VI, the proposed method maintains its classification accuracy (92.48 %) for the model size of ~2.00 MB, which is higher than that of the ECNN model (87.15 %). However, as the network parameters increase and the model size reaches ~7.00 MB, the accuracy of the proposed method increases to 99.46 %, whereas the accuracy of ECNN remains similar to that without adding parameters. When the network continues to deepen and the model size reaches ~11.00 MB, the accuracy of the two methods remains unchanged compared with that of the ~7.00 MB model. This phenomenon demonstrates that: 1) in terms of performance, the proposed method is superior to ECNN, and 2) the ~2.00 MB and ~7.00 MB model sizes are sufficient to maximize the performance of the ECNN and the proposed method, respectively. In addition, their performance cannot be improved by increasing the model size.

With regard to the computing time, as the model size increases from ~2.00 MB to ~11.00 MB, the training time and the real-time inference time of both methods increase. Although the computing time of ECNN is shorter than that of the proposed method for the same model size, the computing time difference is not obvious. For example, the real-time inference time of ECNN is 168.58 ms, which is approximately 2 ms less than that of the proposed method (170.86 ms) for a ~7.00 MB model size. Thus, the proposed method demonstrates better performance than ECNN in satisfying the same conditions for RDD; that is, small model size and fast computing time.

TABLE VI
COMPARISON OF THE PROPOSED AND ECNN METHODS

Model size (MB)	Index	Proposed method	ECNN
~2.00	Classification accuracy (%)	92.48	87.15
~7.00		99.27	86.81
~11.00		99.46	87.25
~2.00	Training time (s)	59.52	64.32
~7.00		93.25	72.53
~11.00		113.18	83.34
~2.00	Inference time (ms)	159.68	145.15
~7.00		170.86	168.58
~11.00		178.52	171.38

D. *Effects of Sensor and Instrument System*

The rotor inspection is realized through successive steps including signal acquisition, feature extraction, and pattern recognition. The sensor and instrument system has a great effect on recognition accuracy. To illustrate the advantage of our instrument system as shown in Figs. 1 and 2, another instrument system designed for IM rotor inspection [34] is used for a comparison in this subsection. The experimental setup and sensor constructed according to Ref. [34] are shown in Figs. 14(a) and 14(b), respectively.

The main difference between our system and the comparative system is introduced as follows. As shown in Figs. 1 and 2, our instrument system uses the IM stator as the sensor, and the rotor to be detected is inserted into the stator cavity. In the comparative system, an external electromagnetic sensor is placed close to the tested rotor, and the distance and angle of the sensor should be carefully adjusted according to the rotor shape. In contrast, our system doesn't need such a complicated operation. Besides, the signal generated from the comparative system is weaker than that of our system, which further affects the performance of defect detection.

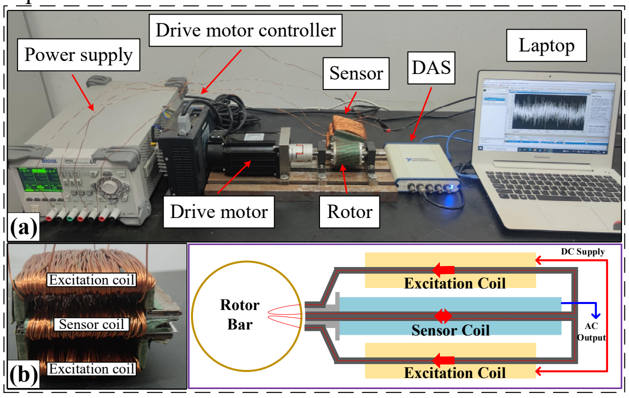


Fig. 14. Comparative RDD system referred from Ref. [34]: (a) experimental setup and (b) sensor.

The rotors in Fig. 3 are tested using the comparative instrument system in Fig. 14, and the signals are processed using the proposed DL model and comparative models. The results are summarized in Fig. 15. It can be seen that the overall recognition accuracy of the comparative system is lower than that of our system. The highest accuracy (76.34 %) is generated by the proposed RMFFCNN model, but this value is about 20 % lower than that of our instrument system.

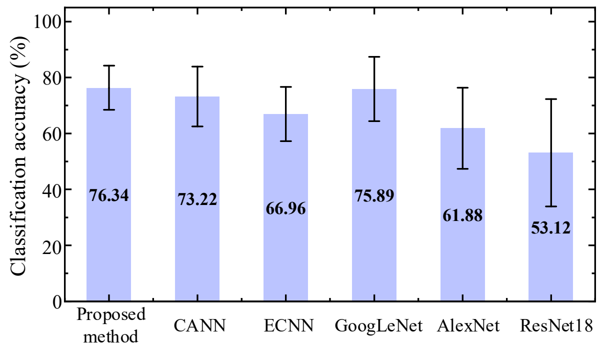


Fig. 15. Classification accuracy of the dataset generated from the comparative instrument system.

This result indicates that the instrument system is a crucial factor to guarantee signal quality along with recognition accuracy. Indeed, the experimental results in Ref. [34] demonstrated that the system in Fig. 14 can detect different types of rotor faults, but it cannot quantitatively evaluate the fault level or severity. The tested rotors in Fig. 3 contain fault rotors with different severities, and hence the comparative system cannot effectively distinguish the rotors' conditions, and finally leads to low recognition accuracy. Relatively speaking, the combination of the designed instrument system and the proposed DL model achieves a high recognition accuracy for different types of rotor defects.

VI. Discussions

To further improve the practicality, robustness, and flexibility of the proposed method, the factors that affect the system performance are discussed from three aspects: 1) DL model, 2) edge computing system, and 3) sensor and instrument system.

A. *DL Model*

For a certain dataset, the recognition accuracy is influenced by the architectures of the DL models. The merits and limits of the six comparative DL models are compared and discussed. According to the results in Tables IV and V, the DL models' performances are classified into three categories, i.e., High/Good/Large, Average, and Low/Fair/Small. The results are summarized in Table VII. The proposed RMFFCNN and CANN models have the highest recognition accuracy as compared with other models. In addition, the fault data with labels may be difficult to obtain, and hence the model performance under limited training samples is important. The proposed model can achieve fast training convergence and high classification accuracy with limited samples. In the future, the block and module architectures of the proposed model can be further investigated and optimized to improve its performance.

TABLE VII
CAPACITY QUALITATIVE ANALYSIS OF DIFFERENT DL MODELS

Method	Accuracy	Small samples	Model size	Computing time	Anti-noise
Proposed	High	Good	Small	Low	Good
CANN	High	Fair	Average	Average	Average
ECNN	Average	Average	Small	Low	Average
GoogLeNet	Average	Fair	Average	Average	Good
AlexNet	Low	Average	Large	High	Fair
ResNet18	Low	Good	Large	High	Fair

B. *Edge Computing Platform*

Edge computing provides a new paradigm for real time signal processing and RDD. Besides the high accuracy, the size of the proposed model is only 7.95 MB, thereby it is suitable for implementation on the edge computing platform. In contrast, the AlexNet and ResNet18 have a large model size and cannot even be deployed into the Raspberry Pi. Additionally, the proposed model can be further optimized due to its flexible modular architecture, and it shows potential to be implemented into simpler and cheaper edge computing platforms such as MCUs. From another aspect, the training of DL model cannot be carried out on the Raspberry Pi platform due to the limitation of computing resources. With the continuous development of hardware performance, model training is expected to be implemented on edge computing systems, which will provide more convenience for practical applications.

C. Sensor and Instrument System

The comparative results using different sensors and instrument systems indicate that the classification accuracy is determined by the quality of the sampled signal. The designed sensor and instrument system in Figs. 1 and 2 guarantee that the weak induced voltage signal of the defect rotor can be effectively detected. Nevertheless, the tested rotors in this study only contain several types of defects. A rotor may simultaneously have multiple or compound faults in practice. Therefore, further improvement of the sensor and instrument system is needed to achieve compound fault detection. From another aspect, the automation of the experimental setup can also be improved. For instance, the installation and adjustment of the tested rotor can be realized using an industrial robot. These topics remain a further study so as to improve the efficiency of rotor quality inspection and to meet Industry 4.0 requirements.

VII. Conclusion

The present study designs an intelligent real-time RDD system including a designed sensor, a novel RMFFCNN model, and a corresponding edge computing system, which is suitable for an industrial production line. First, the sensor is configured into an IM stator, with three-phase winding by an innovative wiring method to output the induced voltage from phase C. The stator is transformed into an electromagnetic sensor, which is low cost and convenient for industry implementation. Subsequently, a novel RMFFCNN model based on a residual structure and multi-scale feature fusion is designed, which has the advantages of small size, fast convergence speed, high recognition accuracy, and strong robustness. Finally, to realize real-time RDD, an edge computing framework including model training and inference is designed on a desktop computer and in the embedded system, respectively. The reliability of the entire system is tested and validated through quality classification on nine different rotor faults, along with a comparison with other state-of-the-art methods. The analysis of the results demonstrates the proposed method's potential for application in real-time RDD in an industrial production line.

Reference

[1] L. Zhu, Y. Zhou, R. Jia, W. Gu, T. H. Luan, and M. Li, "Real-Time Fault Diagnosis for EVs With Multilabel Feature Selection and Sliding Window Control", *IEEE Internet of Things Journal,* vol. 9, pp. 18346-18359, 2022.

[2] X. X. Wang, S. L. Lu, K. Chen, Q. J. Wang, and S. W. Zhang, "Bearing Fault Diagnosis of Switched Reluctance Motor in Electric Vehicle Powertrain via Multisensor Data Fusion", *IEEE Transactions on Industrial Informatics,* vol. 18, pp. 2452-2464, Apr. 2022.

[3] C. W. De Silva, *Sensors and actuators: Engineering system instrumentation*, 2nd ed.: Taylor & Francis/CRC Press, Boca Raton, FL, 2016.

[4] M. A. Rahman, A. M. Osheiba, K. Kurihara, M. A. Jabbar, H. W. Ping, K. Wang, and H. M. Zubayer, "Advances on Single-Phase Line-Start High Efficiency Interior Permanent Magnet Motors", *IEEE Transactions on Industrial Electronics,* vol. 59, pp. 1333-1345, Mar. 2012.

[5] P. K. Wong and J. S. Biao, "Fault Diagnosis of Induction Motors Under Untrained Loads With a Feature Adaptation and Improved Broad Learning Framework", *IEEE/ASME Transactions on Mechatronics,* vol. 27, pp. 3041-3052, 2022.

[6] P. Luong and W. Wang, "Smart Sensor-Based Synergistic Analysis for Rotor Bar Fault Detection of Induction Motors", *IEEE/ASME Transactions on Mechatronics,* vol. 25, pp. 1067-1075, 2020.

[7] Y. Djenouri, A. Belhadi, G. Srivastava, U. Ghosh, P. Chatterjee, and J. C. W. Lin, "Fast and Accurate Deep Learning Framework for Secure Fault Diagnosis in the Industrial Internet of Things", *IEEE Internet of Things Journal,* p. DOI: 10.1109/JIOT.2021.3092275, 2021.

[8] H. Shao, M. Xia, G. Han, Y. Zhang, and J. Wan, "Intelligent Fault Diagnosis of Rotor-Bearing System Under Varying Working Conditions With Modified Transfer Convolutional Neural Network and Thermal Images", *IEEE Transactions on Industrial Informatics,* vol. 17, pp. 3488-3496, May 2021.

[9] J. Jiao, M. Zhao, J. Lin, and C. Ding, "Deep Coupled Dense Convolutional Network With Complementary Data for Intelligent Fault Diagnosis", *IEEE Transactions on Industrial Electronics,* vol. 66, pp. 9858-9867, Dec. 2019.

[10] R. Liu, G. Meng, B. Yang, C. Sun, and X. Chen, "Dislocated Time Series Convolutional Neural Architecture: An Intelligent Fault Diagnosis Approach for Electric Machine", *IEEE Transactions on Industrial Informatics,* vol. 13, pp. 1310-1320, Jun. 2017.

[11] Y. M. Xiao, H. D. Shao, S. Y. Han, Z. Q. Huo, and J. F. Wan, "Novel Joint Transfer Network for Unsupervised Bearing Fault Diagnosis From Simulation Domain to Experimental Domain", *IEEE/ASME Transactions on Mechatronics,* p. DOI: 10.1109/TMECH.2022.3177174, 2022.

[12] F. Wang, R. Liu, Q. Hu, and X. Chen, "Cascade Convolutional Neural Network With Progressive Optimization for Motor Fault Diagnosis Under Nonstationary Conditions", *IEEE Transactions on Industrial Informatics,* vol. 17, pp. 2511-2521, Apr. 2021.

[13] S. Sakib, T. Tazrin, M. M. Fouda, Z. M. Fadlullah, and N. Nasser, "An Efficient and Lightweight Predictive Channel Assignment Scheme for Multiband B5G-Enabled Massive IoT: A Deep Learning Approach", *IEEE Internet of Things Journal,* vol. 8, pp. 5285-5297, 2021.

[14] N. Dilshad, A. Ullah, J. Kim, and J. Seo, "LocateUAV: Unmanned Aerial Vehicle Location Estimation via Contextual Analysis in an IoT Environment", *IEEE Internet of Things Journal,* p. DOI: 10.1109/JIOT.2022.3162300, 2022.

[15] P. Liu, Y. Zhang, H. Wu, and T. Fu, "Optimization of Edge-PLC-Based Fault Diagnosis With Random Forest in Industrial Internet of Things", *IEEE Internet of Things Journal,* vol. 7, pp. 9664-9674, 2020.

[16] A. A. Shah, N. A. Bhatti, K. Dev, and B. S. Chowdhry, "MUHAFIZ: IoT-Based Track Recording Vehicle for the Damage Analysis of the Railway Track", *IEEE Internet of Things Journal,* vol. 8, pp. 9397-9406, 2021.

[17] Q. Yang, C. Hu, and N. Zheng, "Data-Driven Diagnosis of Nonlinearly Mixed Mechanical Faults in Wind Turbine Gearbox", *IEEE Internet of Things Journal,* vol. 5, pp. 466-467, 2018.

[18] W. Zhang, J. Wang, G. Han, S. Huang, Y. Feng, and L. Shu, "A Data Set Accuracy Weighted Random Forest Algorithm for IoT Fault Detection Based on Edge Computing and Blockchain", *IEEE Internet of Things Journal,* vol. 8, pp. 2354-2363, 2021.

[19] L. Yang, Y. Li, and Z. Wei, "Fa-Mb-ResNet for Grounding Fault Identification and Line Selection in the Distribution Networks", *IEEE Internet of Things Journal,* vol. 9, pp. 11115-11125, 2022.

[20] X. Y. Shi, G. Qiu, C. Yin, X. G. Huang, K. Chen, Y. H. Cheng, and S. M. Zhong, "An Improved Bearing Fault Diagnosis Scheme Based on Hierarchical Fuzzy Entropy and Alexnet Network", *IEEE Access,* vol. 9, pp. 61710-61720, 2021.

[21] V. Bianchi, M. Bassoli, G. Lombardo, P. Fornacciari, M. Mordonini, and I. D. Munari, "IoT Wearable Sensor and Deep Learning: An Integrated Approach for Personalized Human Activity Recognition in a Smart Home Environment", *IEEE Internet of Things Journal,* vol. 6, pp. 8553-8562, 2019.

[22] S. Chang, S. Huang, R. Zhang, Z. Feng, and L. Liu, "Multitask-Learning-Based Deep Neural Network for Automatic Modulation Classification", *IEEE Internet of Things Journal,* vol. 9, pp. 2192-2206, 2022.

[23] I. Mehmood, A. Ullah, K. Muhammad, D. Deng, W. Meng, F. Al-Turjman, M. Sajjad, and V. H. C. d. Albuquerque, "Efficient Image Recognition and Retrieval on IoT-Assisted Energy-Constrained Platforms From Big Data Repositories", *IEEE Internet of Things Journal,* vol. 6, pp. 9246-9255, 2019.

[24] C. Hou, G. Liu, Q. Tian, Z. Zhou, L. Hua, and Y. Lin, "Multisignal Modulation Classification Using Sliding Window Detection and Complex

Convolutional Network in Frequency Domain", *IEEE Internet of Things Journal,* vol. 9, pp. 19438-19449, 2022.
[25] Q. Zhu, X. Wang, H. Wang, M. Xia, S. Lu, B. Liu, G. Li, and W. Cao, "Real-Time Defect Detection of Die Cast Rotor in Induction Motor Based on Circular Flux Sensing Coils", *IEEE Transactions on Industrial Informatics,* vol. 18, pp. 9271-9282, 2022.
[26] D. Peng, H. Wang, Z. Liu, W. Zhang, M. J. Zuo, and J. Chen, "Multibranch and Multiscale CNN for Fault Diagnosis of Wheelset Bearings Under Strong Noise and Variable Load Condition", *IEEE Transactions on Industrial Informatics,* vol. 16, pp. 4949-4960, Jul. 2020.
[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 770-778, 2016.
[28] C.-Y. Lee and T.-A. Le, "Identifying Faults of Rolling Element Based on Persistence Spectrum and Convolutional Neural Network With ResNet Structure", *IEEE Access,* vol. 9, pp. 78241-78252, 2021.
[29] S. Lu, G. Qian, Q. He, F. Liu, Y. Liu, and Q. Wang, "In Situ Motor Fault Diagnosis Using Enhanced Convolutional Neural Network in an Embedded System", *IEEE Sensors Journal,* vol. 20, pp. 8287-8296, Aug. 2020.
[30] M. Q. Tran, M. K. Liu, Q. V. Tran, and T. K. Nguyen, "Effective Fault Diagnosis Based on Wavelet and Convolutional Attention Neural Network for Induction Motors", *IEEE Transactions on Instrumentation and Measurement,* vol. 71, p. 3501613, 2022.
[31] C. Szegedy, L. Wei, Y. Jia, P. Sermanet, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1-9, 2015.
[32] T. Technicolor, S. Related, T. Technicolor, and S. Related, "ImageNet Classification with Deep Convolutional Neural Networks", *Commun.ACM,* vol. 60, pp. 84–90, 2017.
[33] S. Naveen, M. R. Kounte, and M. R. Ahmed, "Low Latency Deep Learning Inference Model for Distributed Intelligent IoT Edge Clusters", *IEEE Access,* vol. 9, pp. 160607-160621, 2021.
[34] S. T. Varghese, K. R. Rajagopal, and B. Singh, "Design and Development of Rotor Quality Test System for Die-Cast Copper Rotors", *IEEE Transactions on Industry Applications,* vol. 54, pp. 2105-2114, May-Jun. 2018.

Qingyun Zhu received the B.S. degree in electrical engineering from Chaohu University, Hefei, China, in 2019, and the M.S. degree in control engineering from the College of Electrical Engineering and Automation, Anhui University, Hefei, China.

He is currently working toward the Ph.D. degree in mechanical engineering with the Institute of Technological Science, Wuhan University, Wuhan, China. His research interests include signal processing-based machine fault diagnosis.

Jingfeng Lu received the B.S. degree in mechatronic engineering from Chongqing University of Posts and Telecommunications, Chongqing, China, in 2021.

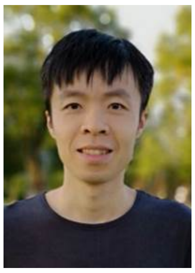
He is currently working toward the M.S. degree in mechanical engineering at the College of Electrical Engineering and Automation, Anhui University, Hefei, China. His research interest includes machine fault diagnosis based on deep learning.

Xiaoxian Wang (Member, IEEE) received the B.S. degree from the Shandong University of Science and Technology, Qingdao, China, in 2010, and the M.S. degree in 2013 from the University of Science and Technology of China, Hefei, China, where she is currently working toward the Ph.D. degree with the Department of Precision Machinery and Precision Instrumentation, all in mechanical engineering.

She is currently an Engineer with the College of Electronics and Information Engineering, Anhui University, Hefei. Her research interests include innovative design and intelligent maintenance of electromechanical system.

Hui Wang received the M.S. degree in control engineering in 2020 from Anhui University, Hefei, China, where she is currently working toward the Ph.D. degree in electrical engineering with the School of Electrical Engineering and Automation.

Her research interests include signal processing-based motor fault diagnosis.



Siliang Lu (Senior Member, IEEE) received the B.S. and Ph.D. degrees in mechanical engineering from the University of Science and Technology of China, Hefei, China, in 2010 and 2015, respectively.

He is currently an Associate Professor with the College of Electrical Engineering and Automation, Anhui University, Hefei. He served as an Associate Editor for *IEEE Transactions on Instrumentation and Measurement*, and an editorial board member for *Journal of Dynamics, Monitoring and Diagnostics*. His research interests include machinery-condition-based monitoring and fault diagnosis, signal processing, IoT and edge computing, and robotics.

Clarence W. de Silva (Fellow, IEEE) received the Ph.D. degrees in mechanical engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 1978, also the Ph.D. degree in information engineering from the University of Cambridge, Cambridge, U.K., in 1998, and the honorary D.Eng. degree from the University of Waterloo, Waterloo, ON, Canada, in 2008.

Since 1988, he has been a Professor of mechanical engineering, the Senior Canada Research Chair, and the Natural Sciences and Engineering Research Council (NSERC)-BC Packers Chair in Industrial Automation with the University of British Columbia, Vancouver, BC, Canada. Dr. de Silva is a fellow of the IEEE, the American Society of Mechanical Engineers, the Canadian Academy of Engineering, and the Royal Society of Canada.

Min Xia (Senior Member, IEEE) is currently a Lecturer (Assistant Professor) in the School of Engineering at Lancaster University, UK. He received B.S. degree in Industrial Engineering from Southeast University, China (2009); M.S. degree in Precision Machinery and Precision Instrumentation from the University of Science and Technology of China, China (2012); and Ph.D. degree in Mechanical Engineering from the University of British Columbia, Canada (2017). He has led 11 research projects in the UK, Canada, and Japan with total funding of £9 million.

He has served various editorial roles including Associate Editor of *IEEE Transactions on Instrumentation and Measurements*. His research interests include smart manufacturing, machine diagnostics and prognostics, deep neural networks, process monitoring and optimization.