Zhao *et al. Journal of Cloud Computing: Advances, Systems and Applications* (2020) 9:58
https://doi.org/10.1186/s13677-020-00205-7

Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH Open Access

# A multi-fault diagnosis method of gear-box running on edge equipment

Xiaoping Zhao[1,2*], Kaiyang Lv[1], Zhongyang Zhang[3], Yonghong Zhang[3] and Yifei Wang[3]

**Abstract**

Edge computing equipment is a new tool that has been widely used to monitor the operation state of industrial equipment and to diagnose and analyze faults. Therefore, the fault diagnosis algorithm used in the edge computing device plays an especially significant role in fault diagnosis. The application of deep learning method in mechanical fault diagnosis has been gradually popularized, because it has many advantages, such as strong classification ability and accurate feature extraction ability. However, many of the completed papers and models are based on single label system and are used to diagnose single target fault. The validation set is not rigorous enough, and it is difficult to accurately simulate the faults that may occur in the actual production process. Nowadays, in the era of big data, the single label system ignores the joint relationship of different fault types, and it is difficult to make a correct judgment for the location, type and degree of mechanical failure. Hence, in the process of experiment, we used the bearing data of Case Western Reserve University(CWRU) to ensure the wide range and large quantity of data sets. A fault diagnosis method of gear and bearing in the gear-box based on multi-task deep learning model is put forward. In this method, gear and bearing faults can be diagnosed simultaneously. Through a separate task layer, this method can adaptively extract the characteristics of distinct targets from the same signal, and add a Batch Normalization layer(BN) to accelerate the convergence speed of the network. Through experiments, we conclude that it is an effective method which can judge the fault situation of gear and bearing accurately in a variety of working conditions.

**Keywords:** Mechanical fault, Multi-task deep learning, Gear-box, Edge computing

## Introduction

Gear, bearing, shaft, box and other significant parts make up the gear-box. The gear-box is now an indispensable part of industrial production and operation. It is widely used in transportation and military industry, mining, metallurgy, automobile, industrial production equipment, etc. due to its compact structure, high-efficiency transmission, long service life and reliable operation. However, machine failure caused by gear-box failure often happens, mainly because the gear-box has a complex internal structure, which needs to maintain long-term and works at a very fast rate in a bad environment. The gear and bearing play a significant role in the gear-box. If part of the function fails or the gear-box runs abnormal or even the machine dies due to its fatigue and wear, it will lead to economic losses or even casualties. The introduction of edge computing equipment [1] in the diagnosis process can make the network service response more quickly and play the role of real-time response. In addition, it can also meet the fundamental requirements of application intelligence, security and privacy protection. Therefore, in order to validly avoid the above failure condition, edge computing equipment [2–4] is used to monitor and diagnose the mechanical equipment. To sum up, to ensure safe production, prevent and avoid major accidents. It is a necessary condition for product development to study efficient gear-box condition monitoring and fault identification technology.

*Correspondence: zxp@nuist.edu.cn
[1]School of Computer and Software, Nanjing University of Information Science and Technology, No.219 Ningliu Road, 210044 Nanjing, China
[2]Network Monitoring Center of Jiangsu Province, Nanjing University of Information Science and Technology No.219 Ningliu Road, 210044 Nanjing, China
Full list of author information is available at the end of the article

With the progress of machine learning, deep learning has been widely used in many different areas such as image recognition, target recognition and other areas have made spectacular achievements. Deep learning has also been proved to be applicable to speech recognition in some research reports. Numerous domestic and foreign researchers specializing in mechanical fault combine deep learning and fault diagnosis to solve practical problems under the inspiration of those studies which are mentioned above. And subsequent experiments have shown that this combination works excellently.

Shao [5], Wang [6] and Chen [7] used deep belief network to study the fault diagnosis of gear-box and rolling bearing, the authors verify the accuracy and generalization of the proposed method by comparing the network with the existing mainstream fault diagnosis. In addition, in the actual production environment, it often contains a lot of noise, which make feature extraction and information fusion become difficult, Li et al [8] . introduced deep belief network and obtained experimental results far beyond the traditional recognition methods.

When it comes to solving the fault diagnosis of asynchronous motors, the literature [9] introduced a model, which is called Sparse Autoencoder (SAE). This model can take an unsupervised way in the process of feature extraction and remove feature interferences to extract fault features more accurately, so as to improve the generalization ability of the model and diagnose the fault category more exactly. In the fault diagnosis of complicated equipments such as aero engines [10], nuclear power plants [11], wind turbines [12], gear-boxes [13], rolling bearings [14], transformers [15] and rotating machinery [16], the researches related to the above methods have been widely used and very good results have been achieved. Zhang [17] made use of the signals without preprocessing from CWRU in his research on fault diagnosis, which used multi-layer Convolutional Neural Network(CNN) for feature extraction, and the research got a great result. Wang [18] used Short Time Fourier Transform(STFT) to convert the collected time-domain signals into time-frequency maps and got higher recognition accuracy by a 2D-CNN. David et al. [19]. construct a CNN for training. Put the rolling bearing time-domain signals [20] which are transformed by STFT, wavelet packet transform and Hilbert transform in it and then train them separately. We use the control variable method to input different size spectrum and different intensity noise into the model [21] to better verify its performance. It can be concluded that using the time-frequency transform and CNN at the same time can identify fault more accurately.

The articles which are mentioned above prove that deep learning possesses mighty adaptive feature extraction and classification capabilities. Nevertheless, these researches are only focusing on the single-label system, and the recognition effect of multi-target fault is not satisfactory. In the context of the widespread use of big data, the single-label system ignores the joint relationship of different fault types, and it is difficult to make a correct judgment for the location, type and degree of mechanical failure. Therefore, this paper put forward a multi-label system [22]. This system is a deep learning network for multi-task fault diagnosis, which is applied to classifying the fault signals of different categories of bearing and gear by establishing a one-dimensional convolution. Through repeated experiments, it is proved that the network can accurately classify the different fault categories set in the workbench when a non-single fault exists in the meantime. In addition, because the model presented in this paper has the characteristics of the small number of layers and small computation, it can be quickly deployed on edge devices.

The next section in this article will introduce 1D-CNN, triplet loss function and multi-task deep learning model. In "Experiment and analysis" section, the experimental process and analysis are described. Finally, in "Conclusions" section, we summarize the experimental results and look forward to further work.

## Multi-task deep learning with one-dimensional convolution

### Convolutional neural network

Generally, CNN contains convolution layer, pooling layer and fully connected layer. Convolution layer performs convolution operation by convolution check input signals, and the convolution kernel is used to carry out convolution operation in the process of characteristics extraction. It has the characteristics of the local receptive field, parameter sharing and sparse connection [23]. Because the vibration signal of gear-box is one-dimensional, so we choose to use the one-dimensional convolution structure in the process of building the convolution neural network.

One-dimensional convolution: The features of all positions of the collected signal can be detected by convolution kernel of convolution layer, which realizes the parameter sharing of the same signal. In general, convolution kernels [24, 25] of different scales need to be set in the same convolution layer to extract different features. The formula for the convolution process is represented as Eq. (1):

$$V_{ij}^{y} = f\left(\sum_{m}\sum_{l=0}^{L_i} w_{ijm}^{l} V_{(i-1)m}^{(y+l)} + b_{ij}\right) \tag{1}$$

In the equation, $V_{ij}^{y}$ can be calculated from the output of previous layer and convolution operation of this layer, it expresses the $j-th$ feature map of layer $i$ in convolution operation at $y$ position during feature extraction. $f(\cdot)$ represents activation function, relu activation function is used in this paper, $b_{ij}$ is the bias value added during

operation, and $m-th$ refers to the index between the corresponding feature set and feature graph in the layer $i$. $w_{ijm}^{l}$ is used to describe the value of the $m-th$ characteristic graph through convolution operation in layer $i$, and $L_i$ is used to represent the size of convolution kernel set in layer $i$.

Pooling layer: pooling is a process of downsampling. It can reduce the number of parameters while retaining the main features, so as to achieve the purpose of dimensionality reduction and effectively prevent overfitting. In the model used in this paper, the maximum pool method is used to improve the stability of the whole model.

Fully connected layer: Each node of the full connection layer is connected to all nodes of the previous layer, which is used to synthesize the features extracted from the previous layer. Due to its all-connected nature, the parameters of the full-connected layer are generally the most, and their export is represented by Eq. (2).

$$h(x) = f(w \cdot x + b) \tag{2}$$

In the above formula, $h(x)$ is the output of the full connection layer and $x$ represents the input, $b$ represents the bias value during operation, $w$ is the weight multiplied by $x, f(\cdot)$ is activation function.

In order to reduce the overfitting problem of the model, dropout [26] method is introduced into the model after the fully connected layer. In training, some nodes between layers are disconnected under certain conditions, so as to enhance robustness and reduce overfitting problem.

Classifier: Softmax function [27] is often used in classification problem which has multiple category. For the input sample $x$ and its label $y$, we can get the probability value that $y$ belongs to class $j$ is $p(y = j|x)$. Therefore, for a K-classification problem, we finally get a K-dimensional vector and the sum of its elements is 1, the expression of softmax is in Eq. (3).

$$\begin{aligned} h_{\xi}\left(x^{(j)}\right) &= \begin{bmatrix} q\left(y^{(j)}=1 \middle| x^{(j)};\xi\right) \\ q\left(y^{(j)}=2 \middle| x^{(j)};\xi\right) \\ \vdots \\ q\left(y^{(j)}=m \middle| x^{(j)};\xi\right) \end{bmatrix} \\ &= \frac{1}{\sum_{t=1}^{m} e^{\xi_t^T x^{(j)}}} \begin{bmatrix} e^{\xi_1^T x^{(j)}} \\ e^{\xi_2^T x^{(j)}} \\ \vdots \\ e^{\xi_m^T x^{(j)}} \end{bmatrix} \end{aligned} \tag{3}$$

Where $\xi_1; \xi_2; \cdots \xi_m \in \Re^{n+1}$ are the parameters of our model; Notice that the term $\frac{1}{\sum_{t=1}^{m} e^{\xi_j^T x^{(j)}}}$ normalizes the distribution, so that it sums to one. In the Eq. (4):

$$q\left(y^{(j)} = t \middle| x^{(j)};\xi\right) = \frac{exp\left(\xi_j^T x\right)}{\sum_{m=1}^{K} exp\left(\xi_n^T x\right)} \tag{4}$$

**Triplet loss function**

Each triple is constructed by randomly selecting a sample from the training dataset as an anchor ($x^a$), and then randomly selecting a sample of the same type as the anchor called positive ($x^p$) and different classes of samples called negative ($x^n$). The anchor, positive, and negative constitute a complete triple. A neural network is trained for each sample in the triple, and the feature expressions of the three samples are denoted as $g\left(x_i^a\right), g\left(x_i^p\right), g\left(x_i^n\right)$.

The purpose of triplet loss is to make the distance between the characteristic expressions of $x^a$ and $x^p$ as small as possible, while making the distance between the characteristic expressions of $x^a$ and $x^n$ as large as possible. There is a minimum interval between the distances ($\alpha$ is a hyperparameter, which can be set manually). As shown in Fig. 1, the triplet learns to calculate the triplet loss multiple times to reduce the distance between similar samples and increase the distance between heterogeneous samples. In

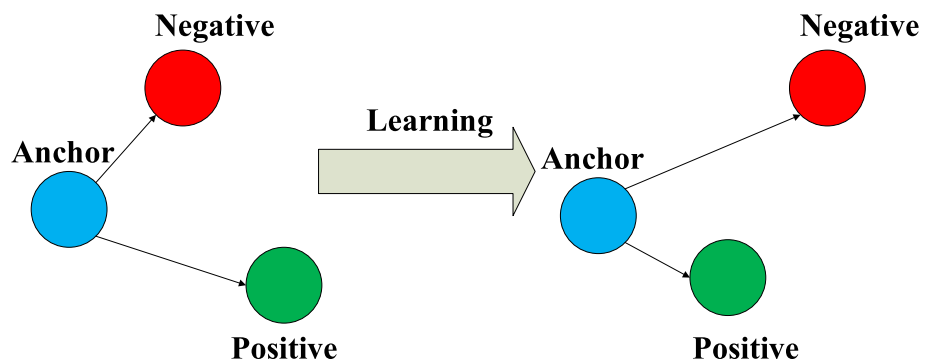



**Fig. 1** Triplet loss optimization goal

Euclidean space, a closer distance between two fault data indicates greater similarity. The formula is

$$\left\| g\left(x_j^a\right) - g\left(x_j^p\right) \right\|_2^2 + \alpha < \left\| g\left(x_j^a\right) - g\left(x_j^n\right) \right\|_2^2 \quad (5)$$

where the subscript 2 represents the L2 paradigm and normalizes the data. The corresponding objective function is

$$\sum_j^n \left[ \begin{array}{l} \left\| g\left(x_j^a\right) - g\left(x_j^p\right) \right\|_2^2 - \\ \left\| g\left(x_j^a\right) - g\left(x_j^n\right) \right\|_2^2 + \alpha \end{array} \right]_+ \quad (6)$$

In Eq. (5), the subscript (+) indicates that if zero is less than the value that in brackets, the loss is the value, and when it is less than zero, the loss is zero. It can be seen from the objective function that when the distance between the characteristic expressions of $x^a$ and $x^p$ is greater than that between the expressions of $x^a$ and $x^n$ minus $\alpha$. If the value in brackets is greater than zero, the loss will occur. And conversely the loss will be zero. When the loss is not zero, all network parameters are adjusted by a backpropagation algorithm to optimize the features [28, 29].

### Multi-task deep learning model

This paper mainly studies the possible faults of two important components of the gear-box, namely, gear fault and bearing fault. Therefore, our model is based on multi-task learning. In the process of multi-task learning, it can extract shared characteristics from different tasks. The shared characteristics extracted above have mighty generalization capacity and can act on target faults with different but related attributes. This usually leads to a wider range of use of the model. Besides, in the process of running the program, we use the shared layer [30, 31], so as to decrease the number of parameters in the calculation process of the whole model, improve the efficiency of diagnosis and predict multiple tasks in parallel.

Figure 2 shows the proposed multi-task deep learning model. Block A in the figure is the shared layer, which contains two convolution layers to extract shared features. Block B is task 1 layer, which is used for bearing faults identification. Box B is task 2 layer, which is used for gear faults identification.

In this section, the proposed model is trained by means of joint learning. The cross-entropy loss function is used to define the loss function of task 1 and task 2. In Eqs. (7); (8); (9), $p(x)$ represents the actual distribution of the target, and $q(x)$ represents the distribution predicted by the model. In the process of joint training, the loss function of individual task layer should be combined, Loss 1; Loss 2 and Loss_all is shown as follows:

$$Loss_1 = -\sum_x p_1(x) log q_1(x) \quad (7)$$

$$Loss_2 = -\sum_x p_2(x) log q_2(x) \quad (8)$$

$$Loss\_all = \frac{Loss_1 + Loss_2}{2} \quad (9)$$

Adam optimization algorithm: To minimize the objective loss function, it is necessary to select an optimization algorithm to update the network weight when training the

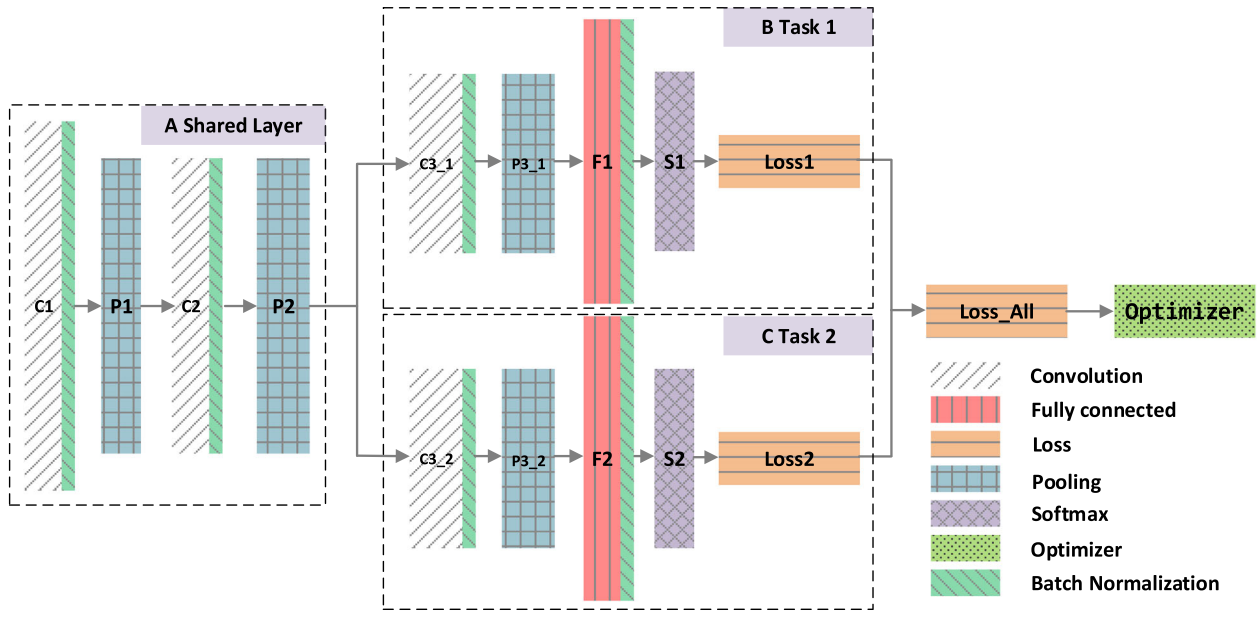



**Fig. 2** A example of deep learning model with 2 tasks

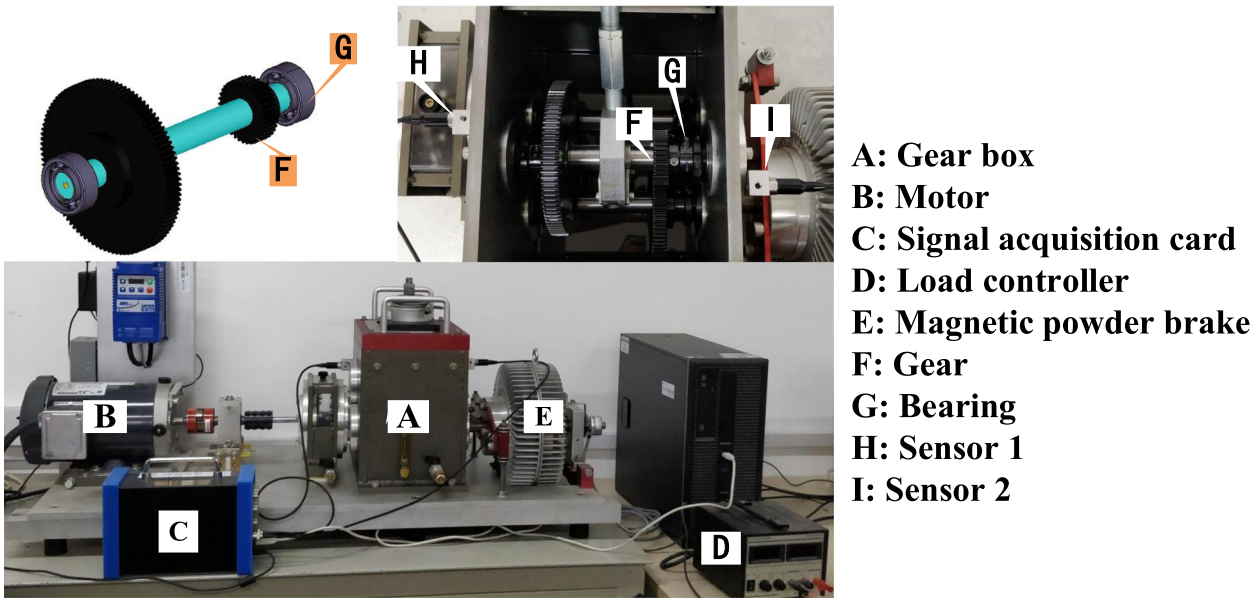



**Fig. 3** Drivetrain dynamics simulator system

network. The parameter optimization formula is shown as Eq. (10):

$$\theta^* = argmin_\theta L\left(f\left(x^i;\theta\right)\right) \tag{10}$$

Where $L(\cdot)$ ; $f(\cdot)$ are the values of the target function and the export function respectively; $\theta^*$ is the best parameter of the CNN; $\theta$ is all parameters obtained during CNN training and $x^i$ is the input value of the CNN.

The stochastic gradient descent (SGD) algorithm is often used in the shallow Back Propagation(BP) neural network, which can make the loss converge to the global optimum. However, in the proposed multi-task deep learning network [32], the SGD algorithm often falls into local optimum due to the super parameter selection deviation. Therefore, Adam (adaptive moments) algorithm [33] is utilized in this article. Adam is a commonly used algorithm. It will change the learning rate in the learning process to complete the optimization of the model's parameters. This process is completed by using the first and second-moment estimation of the gradient. Adam algorithm has many advantages in the process of model building. It can realize efficient calculation and is suitable for solving problems including high noise or sparse gradient. In this paper, it limits the range of learning rate after each iteration. Adam algorithm has strong robustness for the selection process of super parameters, so it plays a very good role in the training of neural network [34, 35] and the adaptive adjustment of parameters. In the Adam algorithm used in this paper, the parameter moment estimation index attenuation rate $\rho_1$ was set to 0.9 and $\rho_2$ was set to 0.999. Also, the parameter numerical stability constant $\varepsilon$ was set to $10^{-8}$.

**Table 1** The one-hot coding of faults

| Bearing | One-Hot | Gear | One-Hot |
|---|---|---|---|
| Normal | 10000 | Normal | 100000 |
| Ball | 01000 | Broken | 010000 |
| Combine | 00100 | Crack | 001000 |
| Inner | 00010 | Eccentric | 000100 |
| Outer | 00001 | Miss | 000010 |
| | | Wear | 000001 |

## Experiment and analysis

### Data description

A great quantity of data is used to train deep learning network, and the effect of the model is affected by the quality of training data. Therefore, the Drivetrain dynamics simulator system (DDS) (Fig. 3) is chosen to be the experimental equipment in this paper. We can replace the spare parts in the gear-box, such as gears and bearings, so as to simulate all kinds of single or compound faults in the gear-box. Table 1 shows the failure categories of gears and bearings and their one-hot codes.

**Table 2** Types of load

| Load | Current/A | Voltage/V |
|---|---|---|
| A | 0 | 0 |
| B | 0.37 | 4 |
| C | 0.56 | 6 |
| D | 0.75 | 8 |

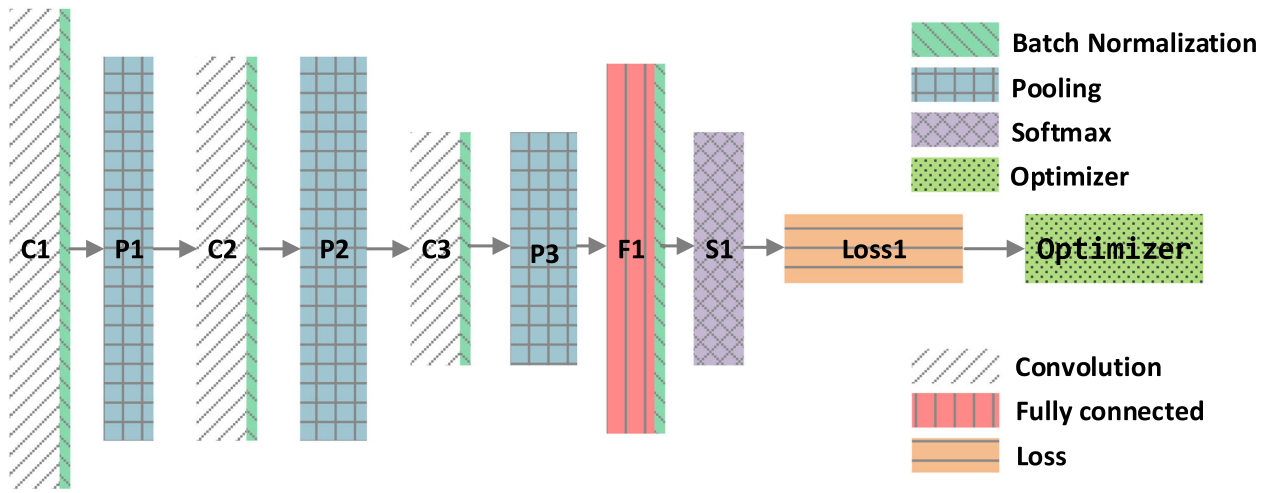



**Fig. 4** The model of single-label network

In order to make the experimental process more consistent with the real production process, we need to increase the diversity of samples. In the process of the experiment, we achieve this goal by adjusting the speed of the instrument and the working load. The load is changed by regulating the equipment E in Fig. 3 at the rear end of the workbench, and Table 2 shows the load type. Meanwhile, for the sake of making the data collected not only contain large spans but also contain similar speed data, we set four different speeds in the experiment, they are 1700RMP, 1800RMP, 3400RMP and 3800RMP. Changing the rotation speed is realized by adjusting the front end of the instrument to drive the motor.

The data set used in this article is the open bearing data set provided by CWRU. Sensors are installed on both sides of the gear-box. Also, the acceleration sensor is used to collect the vibration signal of DDS. To make the production environment more truly, the method of metal tapping is used to carry out artificial noise pollution during the experiment. The amount of noise pollution is about 5% of the total amount collected.

In this article, we analysis the vibration signals that were collected in the experiment. We set the sampling time as 20s, the sampling frequency of the acceleration sensor is 20KHz, and the sensors can collect time-domain signals of two channels at each working state. At last, 960 vibration signal files (4 different loads *4 different speeds *30 different types of compound failures *2 (Left and right channels)) were collected in the experiment, and 409,600 signal points are contained in each signal file. In the preprocessing stage of experimental data, the original vibration signal was stochastically divided, then we segmented 409600 points of the collected signals into 200 [1,2000] time-domain signals. 75% of the collected data was selected as the training set and the rest 25% as the test set stochastically, then we acquired the time-domain signals. However, the literatures [12, 16] show that using the original signals directly to train network will not get a n

**Table 3** TABLE IV. Network parameters of the baseline model

| Layertype | Kernel size | Stride | Kernel channel size | Output Size | Batch Normalization |
|---|---|---|---|---|---|
| Batch Input | \ | \ | \ | 1000*1 | \ |
| Convolution C1 | 32*1 | 2 | 32 | 485*32 | Yes |
| Pooling P1 | 2*1 | 2 | 32 | 242*32 | \ |
| Convolution C2 | 16*1 | 1 | 64 | 227*64 | Yes |
| Pooling P2 | 2*1 | 2 | 64 | 113*64 | \ |
| Convolution C3 | 8*1 | 1 | 32 | 106*32 | Yes |
| Pooling P3 | 2*1 | 2 | 32 | 53*32 | \ |
| Fully-connected F1 | 100 | \ | 1,Dropout = 0.8 | 100*1 | Yes |
| Softmax S1 | 30 | \ | \ | 30*1 | \ |

**Table 4** Network parameters of the proposed model

| Layertype | Kernel size | Stride | Kernel channel size | Output Size | Batch Normalization |
|---|---|---|---|---|---|
| Batch Input | \ | \ | \ | 1000*1 | \ |
| Convolution C1 | 32*1 | 2 | 32 | 485*32 | Yes |
| Pooling P1 | 2*1 | 2 | 32 | 242*32 | \ |
| Convolution C2 | 16*1 | 1 | 64 | 227*64 | Yes |
| Pooling P2 | 2*1 | 2 | 64 | 113*64 | \ |
| Convolution C3_1 | 8*1 | 1 | 32 | 106*32 | Yes |
| Pooling P3_1 | 2*1 | 2 | 32 | 53*32 | \ |
| Convolution C3_2 | 8*1 | 1 | 32 | 106*32 | Yes |
| Pooling P3_2 | 2*1 | 2 | 32 | 53*32 | \ |
| Fully-connected F1 | 100 | \ | 1,Dropout = 0.8 | 100*1 | Yes |
| Fully-connected F2 | 100 | \ | 1,Dropout = 0.8 | 100*1 | Yes |
| Softmax S1 | 5 | \ | \ | 5*1 | \ |
| Softmax S2 | 6 | \ | \ | 6*1 | \ |

results. Besides, it was also found that when feed the original signals into the network, the accuracy of the network is only 30%, and the loss can not be converged. So, in this paper, we used Fast Fourier transformation (FFT) to transform the raw temporal signals, and made it as the input of the network. Frequency domain signal length is 1000 points. At last, we got 144,000 training data and 48,000 test data.

### Experimental setup

Baseline model: At present, no one has solved the problem of multi-target fault diagnosis by using multi-task deep learning method. Consequently, in this article, we compared it with the previous deep learning model which contains a single label. We used the same input data to train the two models. Meanwhile, the labels had been transformed into the one-hot encoding vectors. The length of these vectors is 30. Figure 4 shows the detailed framework map of the network, and Table 3 lists the detailed parameters of the model. The activation function is ReLU, and the pool type is Max pooling.

The proposed model: The multi-task deep learning model used in this paper is divided into two parts. One of them is a shared layer which contains two layers of convolution layer and pooling layer. The other is a network which includes a convolutional layer, a pooling layer, a fully connected layer and a softmax layer for task 1 and task 2. The activation function is ReLU, and the pooling type is max pooling. In order to eliminate the bad influence of singular samples and accelerate the convergence speed of the model, we use the batch normalization strategy after each convolutional layer and fully connected layer. Table 4 lists the detailed structures of the convolutional and pooling layers. The experiments were implemented by Tensorflow framework which is exploited by Google.

### Experiment I: performance under cross validation

In the experiment, we used 4-fold cross-validation to test the performance of the method. The total number of iterations was set as 5000, the batch-size of training was set as 600, and test the accuracy every 1000 times. We recorded the diagnosis results of gear fault, bearing fault and their combined accuracy which both the gear and bearing fault are diagnosed correctly. The results are shown in Table 5.

Also, we used the same way to evaluate the baseline method. Due to the baseline method is a common single-label model which has one output, as a result, only the combined accuracy can be recorded. Table 6 shows the experiment results.

**Table 5** The accuracy of the 4-fold cross validation on proposed model

| Experiment | Bearing accuracy | Gear accuracy | Joint accuracy |
|---|---|---|---|
| 1 | 98.856% | 97.808% | 96.119% |
| 2 | 97.962% | 97.275% | 96.025% |
| 3 | 98.152% | 97.546% | 95.872% |
| 4 | 97.853% | 96.987% | 95.414% |

**Table 6** The accuracy of the 4-fold cross validation on baseline model

| Experiment | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Joint Accuracy | 92.785% | 92.669% | 93.017% | 92.561% |

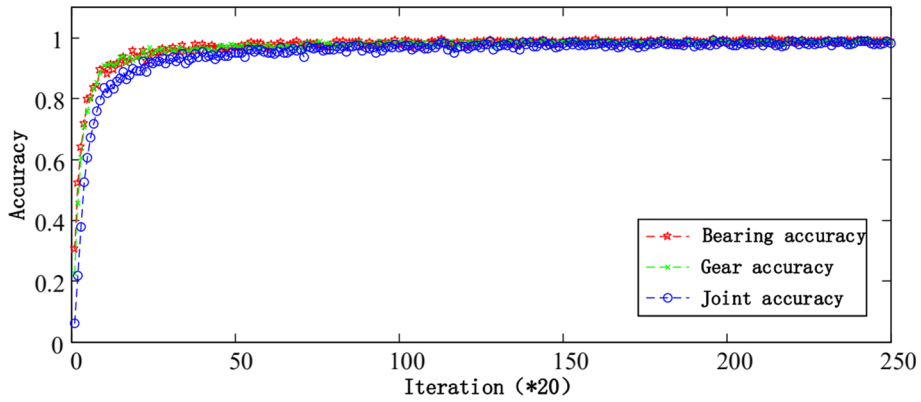



**Fig. 5** The training accuracy curves of bearing, gear and joint training with BN

By observing Tables 5 and 6, we can find that the baseline model is worse than the proposed method, the method in this paper improves the accuracy by about 2%.

#### Experiment II: performance under batch normalization

Parameters of the model will vary with the increase of iteration when the CNN model is trained. For each layer, the varies in the number of iterations also change the distribution of its input data. Therefore, the learning of model parameters will become slow because the parameters of the model have to adapt to the iterative distribution of the input constantly. If the distribution of input in each layer is fixed, it will be easier to update the parameters in the model. Such as setting the input to Gaussian distribution with a variance of 1 and a mean of 0. This standardization of each layer of input is called batch normalization (BN) [36]. The output of the BN layer is:

$$\hat{y}^{l(ij)} = \frac{y^{l(ij)} - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{11}$$

$$Z^{l(ij)} = \gamma^{l(i)} \hat{y}^{l(ij)} + \beta^{l(i)} \tag{12}$$

In the formula above, $\mu_B = E\left[y^{l(ij)}\right], \sigma_B^2 = Var\left[y^{l(ij)}\right]$, where $z^{l(ij)}$ is the outcome of nerve cell,$\varepsilon$ is a minimal constant added to stabilize the result, $y^{l(i)}$ and $\beta^{l(i)}$ represent the scale and displacement parameters to be studied respectively.

Model training often takes a long time while using deep learning methods. In this paper, the accuracy of the training set was recorded every 20 times and the test was recorded every 200 times. When executing the code, we will compare the three accuracy rates to the set threshold after every 200 tests, then update the threshold and save the better model. To save the best model, considering that 48000 samples of all test sets need to be input during the test, one test will take a long time. If the test is changed every 20 times, the training time of the whole model will be greatly increased. At the same time, the accuracy rate is very low in the early stage, because when setting every 20 tests, it is easy to find that the model training of 20 times does not iterate all the training sets. Considering the

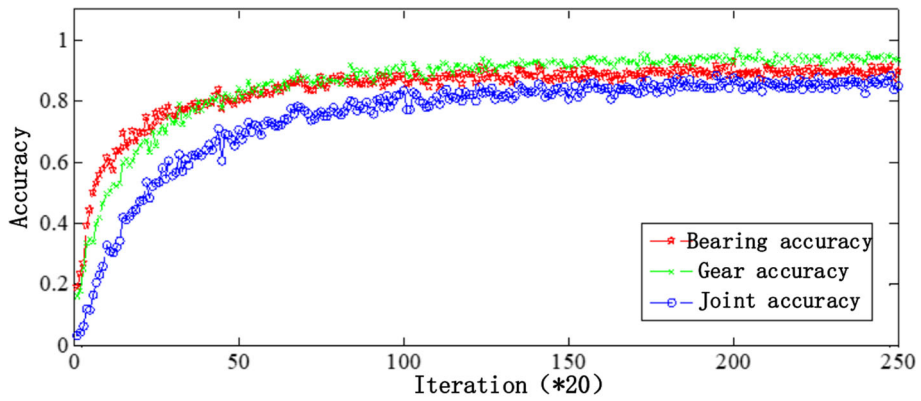



**Fig. 6** The training accuracy curves of bearing, gear and joint training without BN

**Table 7** Data set splited by load and speed

| Data Set | Split method | Train set | | Test set | | Total number of samples |
|---|---|---|---|---|---|---|
| | | Signal | Sample size | Signal | Sample size | |
| A | Load | B,C,D | 144000 | A | 48000 | 192000 |
| B | Load | A,C,D | 144000 | B | 48000 | 192000 |
| C | Load | A,B,D | 144000 | C | 48000 | 192000 |
| D | Load | A,B,C | 144000 | D | 48000 | 192000 |
| E | Speed | 1800,3400,3800 | 144000 | 1700 | 48000 | 192000 |
| F | Speed | 1700,3400,3800 | 144000 | 1800 | 48000 | 192000 |
| G | Speed | 1700,1800,3800 | 144000 | 3400 | 48000 | 192000 |
| H | Speed | 1700,1800,3400 | 144000 | 3800 | 48000 | 192000 |

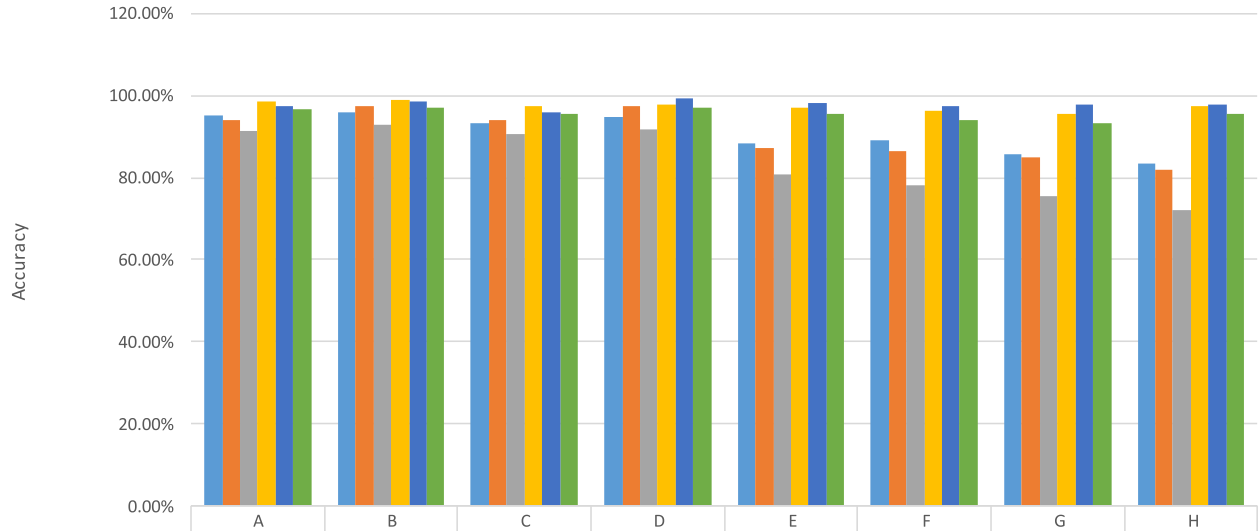
sample number and the size of the batch size of the training set, each test model carries out a complete training set data weight as much as possible iterative update. This is also the reason for the great difference in the number of records.

In the experiment, if we record too many times, the whole line graph will be very dense, which is not convenient to observe. Figures 5 and 6 show the accuracy curve of the model training stage with or without BN. It is shown in the figure that the joint accuracy of 100 iterations in the training phase is 80% and reaches 90% after 80 more iterations by using BN. However, the accuracy of training 5000 times without BN layer is not up to 90%. It is concluded that the BN layer can accelerate the speed of network convergence.

**Experiment III: experiments with different speed and load**

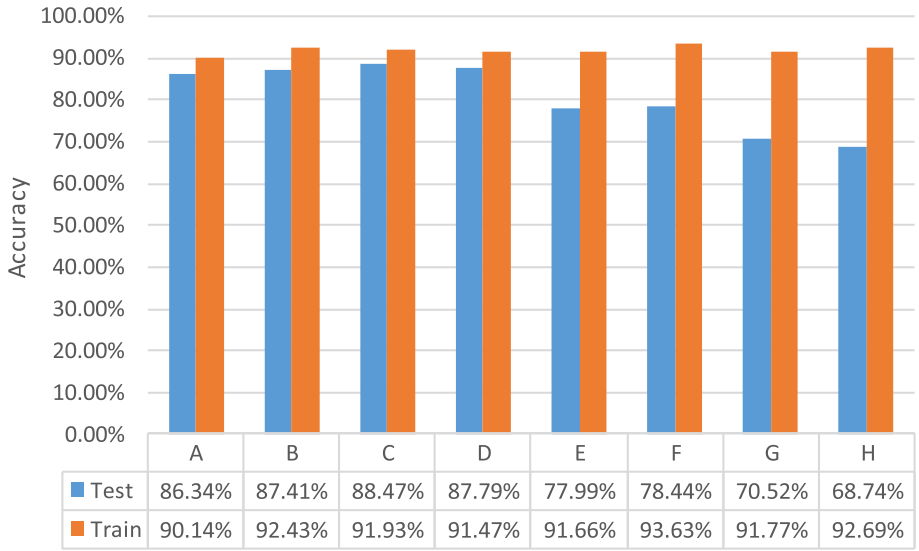
In practice, data missing often occurs under specific conditions, so we divided the data by load and speed. These three segmentation methods were proposed because many articles use random percentage segmentation to evaluate the model. However, in real production, working conditions are not always the same, which can lead to data loss situations. Therefore, in order to study the robustness of the model better, speed and load segmentation methods were added in this paper. The following is the specific method of data set segmentation.

1. Split on speed: Select samples under three speeds as train data, and samples under the other one speed as test data.




| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| Test set Bearing | 95.42% | 95.85% | 93.26% | 95.01% | 88.46% | 89.15% | 85.72% | 83.63% |
| Test set Gear | 94.26% | 97.52% | 94.17% | 97.48% | 87.15% | 86.35% | 84.95% | 82.02% |
| Test set Joint | 91.26% | 92.85% | 90.75% | 91.95% | 80.79% | 78.11% | 75.43% | 71.98% |
| Training set Bearing | 98.48% | 99.16% | 97.46% | 97.75% | 97.03% | 96.41% | 95.48% | 97.33% |
| Training set Gear | 97.33% | 98.73% | 96.02% | 99.48% | 98.45% | 97.35% | 97.79% | 97.83% |
| Training set Joint | 96.69% | 97.13% | 95.46% | 97.04% | 95.76% | 94.16% | 93.38% | 95.67% |

**Fig. 7** The results of the proposed model




| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| Test | 86.34% | 87.41% | 88.47% | 87.79% | 77.99% | 78.44% | 70.52% | 68.74% |
| Train | 90.14% | 92.43% | 91.93% | 91.47% | 91.66% | 93.63% | 91.77% | 92.69% |

**Fig. 8** The results of the baseline model

2. Split on load: Select samples under three loads as train data, and samples under the other one load as test data.

The split data is processed by the method mentioned above and the new data set is shown in Table 7.

Looking at Figs. 7 and 8, we can see that the model can get better precision while the data is divided under different loads, approximately 91.8±1%.

It can be seen that the diagnostic results of the proposed method are better than the baseline method obviously. However, when the samples are divided at different speeds, the serious overfitting occurred in both baseline method and the proposed method in this paper. For the latter method, the output of the test data was about 76.4±4.4%. And with the increase in speed, the test result is getting worse and worse.

#### Experiment IV: experiments on strengthening generalization ability by using triplet thought

When we divided the data set by speed and feed it into the network, the method in this paper had serious overfitting.

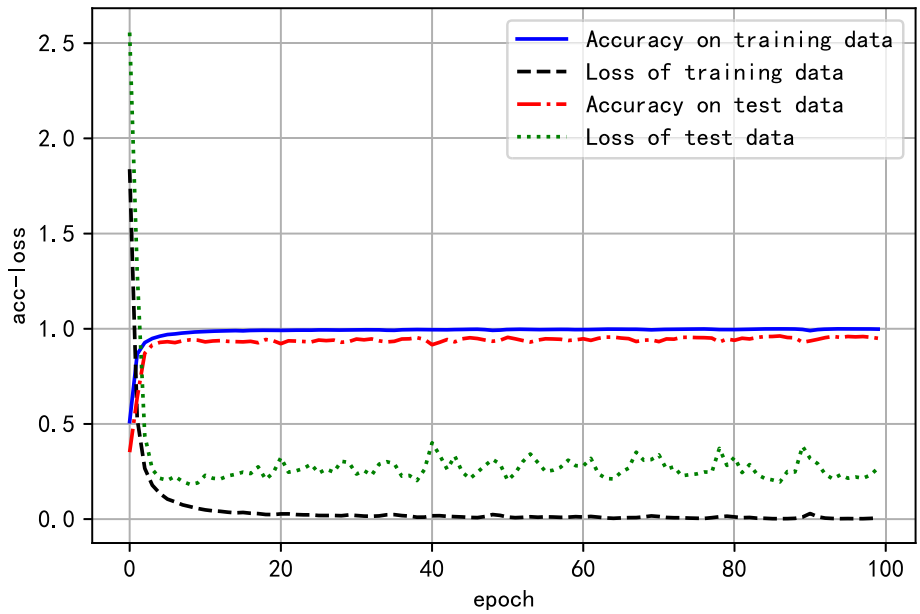



**Fig. 9** An experiment with triple loss as loss function

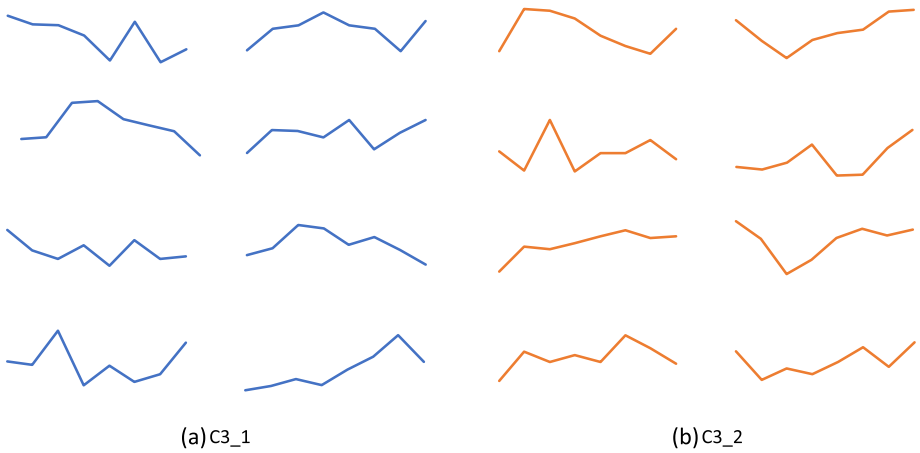



**Fig. 10** Visualization of convolutional kernels on C3_2 and C3_2

As a result, the classification of the test set was very poor. To alleviate the problem which the generalization ability of the model was not strong when the data was segmented by speed, this paper used triplet loss function as a criterion for updating model parameters.

Triplet loss can measure the distribution of features in high-dimensional space so that the distance between the same faults is very close, and the distance between different faults is very far. We can see (In Fig. 9) the output result of the test set can achieve

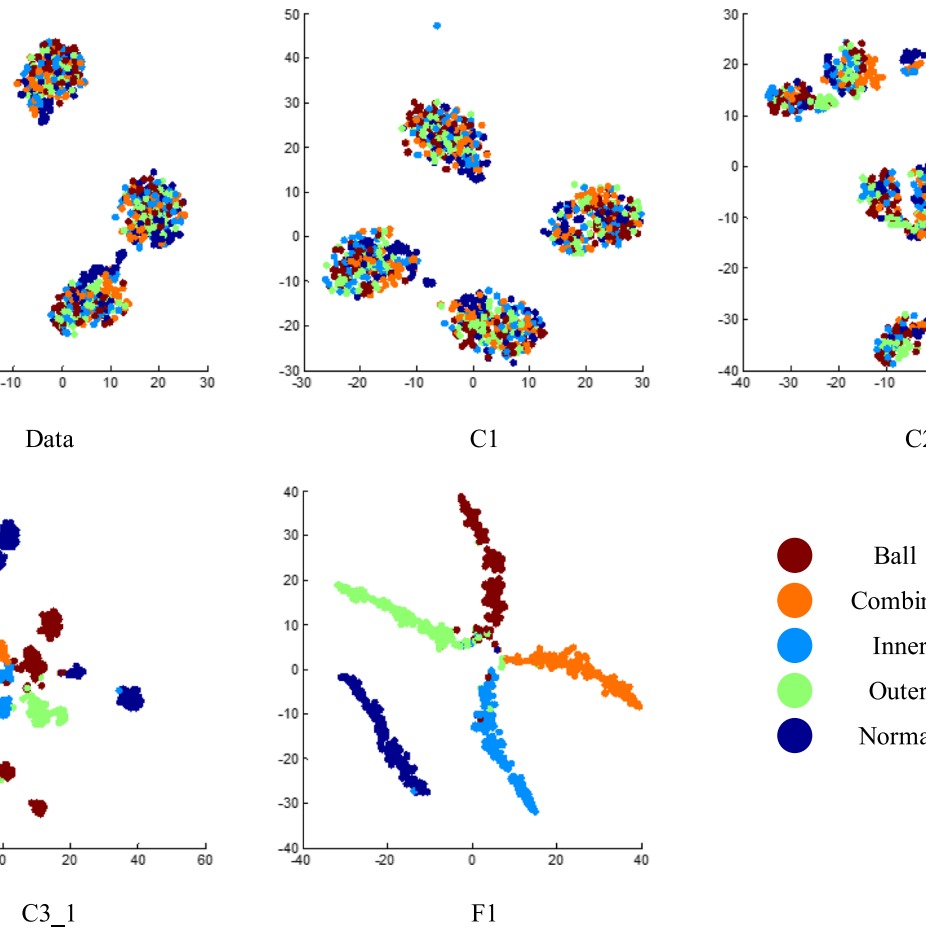



**Fig. 11** Experimental classification results of bearing fault diagnosis task

90.13%, far higher than the previous 76.4±4.4% and no overfitting.

### Network visualizations

It is generally believed that a deep learning model is a mysterious object, and it is not easy to understand its operation principle and internal structure. In this paper, we used the convolution kernel and feature visualization to analyze the internal situation of the proposed deep learning method.

First of all, to verify that for different tasks, the proposed model can extract their characteristics well, we visualized the convolution kernels of C3_1 and C3_2 in the network structure. Figure 10 is the output results. Obviously, the results show that the parameters learned by the convolution kernel of the two tasks are very different.

Furthermore, some test samples were selected and T-SNE method was used to observe the characteristic distribution of the output of each layer. At last, the characteristic distributions of gear and bearing diagnosis were achieved. By observing Figs. 11 and 12, the convolution (C1 and C2) of the shared layer cannot effectively extract features or classify faults. Besides, the results of convolution blocks C3_1 and C3_2 indicate the convolution in the separate task layer can distinguish different fault types. We can draw a conclusion that the deep learning method based on multi-task can get a better diagnosis effect for the multi-task problem.

## Conclusions

The proposed model is the first application of the multi-task deep learning model in the fault diagnosis of bearing and gear. In order to test and verify the performance of the proposed model in practical application, the data we collected were segmented in different means to imitate the data under specific working conditions. BN is used for accelerated network training. Meanwhile, we set up a single label network for comparative analysis and the following conclusions are drawn:

Batch normalization can significantly reduce the training time of the network. When all data types exist, the proposed model can adaptively extract the characteristics

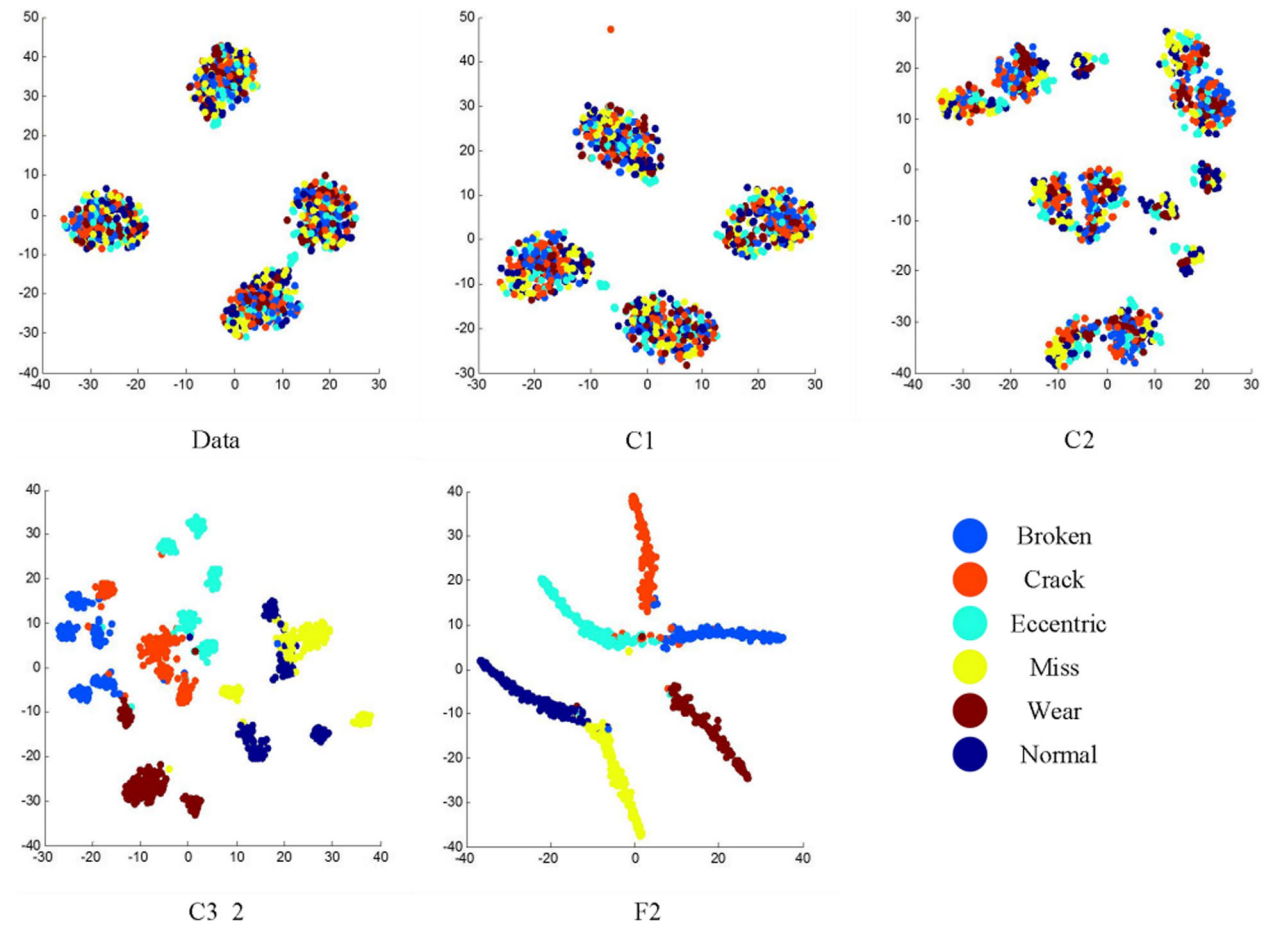



**Fig. 12** Experimental classification results of gear fault diagnosis task

of all tasks. Diagnostic precision of bearing fault can achieve 98.35±0.5%, diagnostic precision of gear fault can achieve 97.4±0.4%, and the joint precision can still maintain good results, which can achieve 95.76±0.35%. The accuracy of the model is about 2.5-3% higher than that of the baseline model. After introducing triplet loss as the loss function, the overfitting problem can be solved, and the model diagnosis accuracy reaches 90.13%.

When the training phase is missing data under specific load and the test phase uses the data under the specific load, the joint accuracy is 91.8±1%. When the training phase is missing velocity data, the combined accuracy in the testing phase is 76.4±4.4% by using the missing velocity data for model testing. The same overfitting occurs in the baseline model. The higher the speed, the lower the accuracy of the test phase.

Our model can accurately diagnose gear and bearing faults when the data is sufficient. In the case of loss of load data, this method has high diagnostic accuracy. However, the model shows severe overfitting without speed data. Therefore, it will be meaningful work to optimize the generalization performance of the model in the absence of velocity data.

**Acknowledgements**
This research is supported financially by National Natural Science Foundation of China (Grant No.51505234,51405241,51575283).

**Authors' contributions**
Xiaoping Zhao, Kaiyang Lv, Zhongyang Zhang, Yonghong Zhang and Yifei Wang conceived and designed the study. Yifei Wang and Zhongyang Zhang performed the simulations. Xiaoping Zhao, Kaiyang Lv and Yonghong Zhang conduct the experiment and confirm the conclusion. Xiaoping Zhao, Kaiyang Lv, Zhongyang Zhang and Yifei Wang wrote the paper. All authors reviewed and edited the manuscript. All authors read and approved the final manuscript.

**Availability of data and materials**
The raw data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

**Competing interests**
The authors declare that they have no competing interests.

**Author details**
[1]School of Computer and Software, Nanjing University of Information Science and Technology, No.219 Ningliu Road, 210044 Nanjing, China. [2]Network Monitoring Center of Jiangsu Province, Nanjing University of Information Science and Technology No.219 Ningliu Road, 210044 Nanjing, China. [3]School of Automation, Nanjing University of Information Science and Technology No.219 Ningliu Road, 210044 Nanjing, China.

Received: 14 April 2020 Accepted: 29 September 2020
Published online: 12 October 2020

**References**

1. Chen Y, Zhang N, Zhang Y, Chen X, Wu W, Shen XS Energy Efficient Dynamic Offloading in Mobile Edge Computing for Internet of Things. In: IEEE Transactions on Cloud Computing. https://doi.org/10.1109/TCC.2019.2898657
2. Li S, Zhao S, Yang P, Andriotis P, Xu L, Sun Q (2019) Distributed consensus algorithm for events detection in cyber-physical systems. IEEE Internet Things J 6(2):2299–2308
3. Li S, Zhao S, Yuan Y, Sun Q, Zhang K (2018) Dynamic security risk evaluation via hybrid Bayesian risk graph in cyber-physical social systems. IEEE Trans Comput Soc Syst 5(4):1133–1141
4. Xu X, Li Y, Huang T, Xue Y, Peng K, Qi L, Dou W (2019) An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks. J Netw Comput Appl 133:75–85
5. Shao H, Jiang H, Zhang X, Niu M (2015) Rolling bearing fault diagnosis using an optimization deep belief network. Meas Sci Technol 26(11):115002
6. Wang X, Li Y, Rui T, Zhu H, Fei J (2015) Bearing fault diagnosis method based on Hilbert envelope spectrum and deep belief network. J Vibroengineering 17(3):1295–1308
7. Chen Z, Li C, Sánchez R-V (2015) Multi-layer neural network with deep belief network for gearbox fault diagnosis. J Vibroengineering 17(5):2379–2392
8. Li C, Sanchez R-V, Zurita G, Cerrada M, Cabrera D, Vásquez RE (2015) Multimodal deep support vector classification with homologous features and its application to gearbox fault diagnosis. Neurocomputing 168:119–127
9. Sun W, Shao S, Zhao R, Yan R, Zhang X, Chen X (2016) A sparse auto-encoder-based deep neural network approach for induction motor faults classification. Measurement 89:171–178
10. Pang S, Yang X, Zhang X (2016) Aero engine component fault diagnosis using multi-hidden-layer extreme learning machine with optimized structure. Int J Aerosp Eng 2016. https://doi.org/10.1155/2016/1329561
11. Shaheryar A, Yin X-C, Hao H-W, Ali H, Iqbal K (2016) A denoising based autoassociative model for robust sensor monitoring in nuclear power plants. Sci Technol Nucl Installations 2016. https://doi.org/10.1155/2016/9746948
12. Yang Z-X, Wang X-B, Zhong J-H (2016) Representational learning for fault diagnosis of wind turbine equipment: A multi-layered extreme learning machines approach. Energies 9(6):379
13. Zhang S, Luo J (2018) A sparse autoencoder algorithm based on spectral envelope curve and its application in gearbox fault diagnosis[J]. Zhendong yu Chongji/J Vib Shock 37(4):249–256
14. Liu H, Li L, Ma J (2016) Rolling bearing fault diagnosis based on STFT-deep learning and sound signals[J]. Shock Vib 2016. https://doi.org/10.1155/2016/6127479
15. Wang L, Zhao X, Pei J, Tang G (2016) Transformer fault diagnosis using continuous sparse autoencoder. SpringerPlus 5(1):1–13
16. Lu C, Wang Z-Y, Qin W-L, Ma J (2017) Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification. Signal Process 130:377–388
17. Pan J-S, Tsai P-W, Huang H-C (2017) Advances in intelligent information hiding and multimedia signal processing. In: Conference Proceedings IIH-MSP, Springer. p 4
18. Wang L-H, Zhao X-P, Wu J-X, Xie Y-Y, Zhang Y-H (2017) Motor fault diagnosis based on short-time fourier transform and convolutional neural network. Chin J Mech Eng 30(6):1357–1368
19. Verstraete D, Ferrada A, Droguett EL, Meruane V, Modarres M (2017) Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings. Shock Vib 2017. https://doi.org/10.1155/2017/5067651
20. Zhao X, Wu J, Zhang Y, Shi Y, Wang L (2018) Fault diagnosis of motor in frequency domain signal by stacked de-noising auto-encoder. Cmc-Computers Mater Contin 57(2):223–242
21. Haldar NAH, Li J, Reynolds M, Sellis T, Yu JX (2019) Location prediction in large-scale social networks: an in-depth benchmarking study. The VLDB Journal 28(5):623–648
22. Li S, Choo K-KR, Sun Q, Buchanan WJ, Cao J (2019) IoT forensics: Amazon echo as a use case. IEEE Internet of Things J 6(4):6487–6497
23. LeCun Y, Bengio Y, et al (1995) Convolutional networks for images, speech, and time series. Handb Brain Theory Neural Netw 3361(10):1995
24. Li S, Tryfonas T, Russell G, Andriotis P (2016) Risk assessment for mobile systems through a multilayered hierarchical Bayesian network. IEEE Trans Cybern 46(8):1749–1759
25. Qi L, He Q, Chen F, Dou W, Wan S, Zhang X, Xu X (2019) Finding all you need: web APIs recommendation in web of things through keywords search. IEEE Trans Comput Soc Syst 6(5):1063–1072
26. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. pp 1097–1105
27. Hinton GE, Salakhutdinov RR (2009) Replicated softmax: an undirected topic model. In: Advances in Neural Information Processing Systems. pp 1607–1614

28. Qi L, Chen Y, Yuan Y, et al. (2020) A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems[J]. World Wide Web 23(2):1275–1297
29. Xu X, Zhang X, Gao H, et al. (2019) BeCome: Blockchain-enabled computation offloading for IoT in mobile edge computing[J]. IEEE Trans Ind Inform 16(6):4187–4195
30. Xu X, Chen Y, Zhang X, Liu Q, Liu X, Qi L (2019) A blockchain-based computation offloading method for edge computing in 5G networks. Softw Pract Experience. https://doi.org/10.1155/2017/5067651
31. Xu X, He C, Xu Z, et al., Wan S, Bhuiyan MZA (2019) Joint optimization of offloading utility and privacy for edge computing enabled IoT[J]. IEEE Internet of Things J 7(4):2622–2629
32. Li J, Cai T, Deng K, et al. (2020) Community-diversified influence maximization in social networks[J]. Inf Syst:101522
33. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. Conference paper at ICLR 2015
34. Xu X, Liu Q, Luo Y, Peng K, Zhang X, Meng S, Qi L (2019) A computation offloading method over big data for IoT-enabled cloud-edge computing. Futur Gener Comput Syst 95:522–533
35. Xu X, Xue Y, Qi L, Yuan Y, Zhang X, Umer T, Wan S (2019) An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. Futur Gener Comput Syst 96:89–100
36. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167. https://arxiv.org/abs/1502.03167


## Publisher's Note