Journal of *Marine Science and Engineering*

MDPI

Article

On the Sparse Gradient Denoising Optimization of Neural Network Models for Rolling Bearing Fault Diagnosis Illustrated by a Ship Propulsion System

Shuangzhong Wang [1,†], Ying Zhang [2,*,†], Bin Zhang [2], Yuejun Fei [3], Yong He [3], Peng Li [4] and Mingqiang Xu [5]

[1] Logistics Engineering College, Shanghai Maritime University, Shanghai 201306, China
[2] College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China
[3] East China Sea Center of Standard & Metrology (Technology), SOA, Shanghai 201306, China
[4] East China Sea Forecasting Center, SOA, Shanghai 200136, China
[5] CTTIC Big Data (Shanghai) Technology Co., Ltd., Shanghai 201901, China
* Correspondence: yingzhang@shmtu.edu.cn
† These authors contributed equally to this work.

Abstract: The drive rolling bearing is an important part of a ship's system; the detection of the drive rolling bearing is an important component in ship-fault diagnosis, and machine learning methods are now widely used in the fault diagnosis of rolling bearings. However, training methods based on small batches have a disadvantage in that the samples which best represent the gradient descent direction can be disturbed by either other samples in the opposite direction or anomalies. Aiming at this problem, a sparse denoising gradient descent (SDGD) optimization algorithm, based on the impact values of network nodes, was proposed to improve the updating method of the batch gradient. First, the network is made sparse by using the node weight method based on the mean impact value. Second, the batch gradients are clustered via a distribution-density-based clustering method. Finally, the network parameters are updated using the gradient values after clustering. The experimental results show the efficiency and feasibility of the proposed method. The SDGD model can achieve up to a 2.35% improvement in diagnostic accuracy compared to the traditional network diagnosis model. The training convergence speed of the SDGD model improves by 2.16%, up to 17.68%. The SDGD model can effectively solve the problem of falling into the local optimum point while training a network.

Keywords: neural networks; sparse denoising; gradient optimization; rolling bearings fault diagnosis

Citation: Wang, S.; Zhang, Y.; Zhang, B.; Fei, Y.; He, Y.; Li, P.; Xu, M. On the Sparse Gradient Denoising Optimization of Neural Network Models for Rolling Bearing Fault Diagnosis Illustrated by a Ship Propulsion System. *J. Mar. Sci. Eng.* **2022**, *10*, 1376. https://doi.org/10.3390/jmse10101376

Academic Editor: Leszek Chybowski

Received: 7 August 2022
Accepted: 19 September 2022
Published: 26 September 2022

1. Introduction

With the rapid development of the global economy, maritime transportation logistics have become an important lifeline to the global economy. The safe and reliable operation of the power transmission equipment of large marine ships is critical to the marine logistic enterprise. In order to meet the needs of production, marine powertrain equipment usually needs to work for a long time or even under a state of overload, which leads to the most common failure of the rolling bearing system in the powertrain [1]. Once these failures occur, they can cause huge economic losses. These accidents make people realize the importance of fault diagnosis in marine powertrain equipment. If the faults can be found and repaired in time, according to the operation data of the equipment, the loss can be effectively retrieved [2].

As shown in Figure 1, the internal power and transmission system inside a ship includes the engine and a large number of rotational parts, most of which include rolling bearings, and the frictional wear of these bearings occurs continuously with mechanical movement. These bearing faults account for a considerable proportion of all powertrain failures [3].

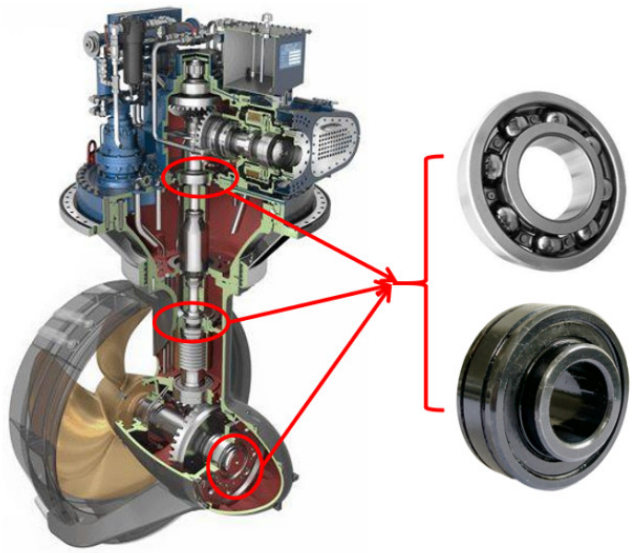


Figure 1. Rolling bearings in the marine powertrain.

Model-based diagnosis methods are the most commonly used in the field of fault diagnosis [4]. By analyzing the operation process of the equipment, a fault diagnosis model is established for performing effective fault diagnosis. However, the wear and aging of rolling bearings often have a non-linear or uncertain correspondence and are always accompanied by the failure of other components. Multiple faults may occur simultaneously and interact with each other, so it is difficult to build the fault diagnosis model accurately [5].

Data-driven fault diagnosis methods can effectively demonstrate the operation information and the fault status from a large amount of historical data [6]. Based on the obtained historical data, a data-driven diagnosis model can be built to provide an effective diagnosis result [7]. Commonly used data-driven fault diagnosis methods include the artificial neural network (ANN) algorithm [8], autoencoder [9], and Bayesian network [10].

Feature extraction is an important step for data-driven fault diagnosis based on neural networks. Frequency domain feature extraction is the most commonly used method [11]. Hu et al. [12] applied kernel principal component analysis (KPCA) and wavelet packet decomposition methods to extract features from continuous, raw time signals and feed the features to a weighted limit learning machine for classification and diagnosis. A feature extraction approach based on the sparse filtering of the raw data was proposed in [13]. However, some diagnosis networks can extract data features themselves. The convolutional operation is the most commonly used method to extract data features [14,15]. In order to make the diagnosis network more adaptable to multiple fault types, Cai et al. [16] investigated deep Bayesian networks to model the dynamic processes of faults, as well as Markov chains for different fault types, including transient faults, intermittent faults, and permanent faults. The network-based methods have some disadvantages. During the training process of diagnosis networks, they often fall into local optimal points. Tao et al. [17] addressed the fact that adding a genetic element to the updating process of the gradients can improve the convergence speed effectively as well as the accuracy of the diagnosis network.

There are numerous research results on how to alleviate diagnosis networks from falling into local optimal points and improve the convergence speed. Stochastic gradient descent (SGD) was first applied to solve the problem, where one sample is randomly selected in each training epoch to represent the updated gradient values of all samples. The SGD method is sensitive to gradient noise and tends to fall into local optimal points [18]. Then, full-batch gradient descent (FGD) was proposed, where the gradients of all samples are obtained during each training epoch, and then take the mean value as the updating gradient. The FGD method is highly undesirable due to the iteration efficiency being too low when the sample size is huge [19]. The mini-batch gradient descent (MGD) method combines the advantages of both SGD and FGD [20]. The number of samples taken during

each iteration is fixed, and the mean value of these sample gradients is used as the updating value. This method is currently the most widely used one [21].

In order to achieve faster convergence speeds for the network during the training process, a series of improvements have been made based on stochastic gradient descent algorithms. By adding a momentum factor, the historic gradient changing information can be integrated to improve the optimization efficiency [22]. The momentum updating strategies include the classical momentum algorithm (CM) and the Nesterov accelerated gradient algorithm (NAG) [23,24]. In order to reduce the continuous accumulation of the gradient variance during random sampling in the iterative process, researchers have proposed the stochastic variance reduction gradient algorithm (SVRG), which gives a general framework for variance reduction algorithms [25]. The subsequent variance reduction algorithms that emerged successively, mostly improved upon this version, such as VR-SGD and stochastic primal-dual coordinates (SPDC) [26,27]. The accelerated gradient methods reserve a corresponding gradient value for each sample. During the iteration process, samples are drawn in turn and replaced the former gradients with new ones [28]. The stochastic average gradient algorithm (SAG), augmented Lagrange-stochastic gradient algorithm (AL-SGA), and point-SAGA were derived based on different updating strategies [29–31]. The adaptive learning rate adjustment algorithm allows for the flexible online adjustment of the learning rate. It can reduce oscillations and speed up convergence during the training process. According to the different adjustment strategies, some scholars proposed the adaptive gradient (Adagrad) [32] and adaptive moment estimation (Adam) [33] algorithms for the adaptive adjustment of learning rate.

However, it has been pointed out that large-scale training datasets have the problem of sample redundancy [34]. The noise and imbalanced distribution of sample gradients are not fully considered. Somehow, falling into a local optimal point is still a serious issue.

Based on the above analysis, a new sparse denoising gradient descent (SDGD) network model was proposed to improve the updating method of the batch gradient. The algorithm proposed in this paper can obviously improve the convergence speed of the neural network and effectively solve the problem of falling into local optimal points during network model training.

The main contributions of the work can be summarized as follows.

(1) A sparse method based on mean impact values is proposed. The network nodes are weighted based on the mean impact value sparse method, and then the key nodes of the network are marked. The network parameters can be effectively "sparsed".

(2) The density-based spatial clustering of applications with noise (DBSCAN) method was used to cluster and denoise the batch gradients. The parameters after denoising can obviously improve the convergence speed of the training and effectively solve the problem of falling into local optimal points.

The rest of this paper is organized as follows. Section 2 provides the preliminaries. Section 3 gives a specific description of the proposed SDGD optimization algorithm. Section 4 gives a case study. Section 5 draws a conclusion and presents the future work.

2. Preliminaries

2.1. Fault Diagnosis of Marine Machinery

For modern maritime ships, effective machinery fault diagnosis is essential [35]. The existing data-driven intelligent recognition methods that are widely used mainly collect historical data through a large number of sensors to establish an intelligent recognition model, so that it has effective recognition ability. Yan et al. performed fault diagnosis by means of vibration signals collected from marine blowers [36]. Firstly, the time domain vibration signal was extracted by the EEMD method, and then the diagnostic model was constructed by the back propagation neural network (BPNN). The EEMD-BPNN intelligent diagnostic method has excellent diagnostic results for marine blowers. Xu et al. [37] combined expert knowledge and data-driven methods effectively. The model was first set up by an expert system and then optimized using historical data. Thus, a belief, rule-based

expert system for the fault diagnosis of marine diesel engines was established. Xu et al. [38] proposed an approach that fused multiple data-driven models for the fault diagnosis of a marine diesel engine. The method can effectively fuse the predictions of individual models, and the results are often more reliable than single data-driven diagnosis models. Tan et al. [39] proposed a one-class SVM-based approach for the diagnosis of the main propulsion system in a ship's operation. Compared with traditional methods, this method can effectively reduce the amount of data required. A model based on neural networks was developed to detect the state of ship engine systems [40]. Operational data from the machine were first collected, and then the diagnostic network was trained. The diagnostic modeling of long- and short-term memory (LSTM) networks, combined with automatic encoders, has also been used more often in the diagnosis of marine machinery [41,42]. LSTM has an advantage with time-series input signals.

Compared with other data-based fault diagnosis methods in the industry, the diagnostic approach for maritime ships is generally consistent. First, the historical data signals are collected via sensors. Then, feature extraction of the signal is performed. Finally, the model of the diagnosis network is established [43,44]. However, these approaches are essentially rudimentary applications of neural networks and fail to consider some practical problems in the data-training efficiency of maritime ship environments. How to increase the speed of training convergence efficiency, effectively improving the diagnosis accuracy, and avoid falling into local optimum points during the training process needs to be addressed urgently.

2.2. Overview of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Density-based spatial clustering of applications with noise (DBSCAN) is a density-based clustering algorithm. The three important definitions in DBSCAN: density reachable, and density connection, are shown in Figure 2.

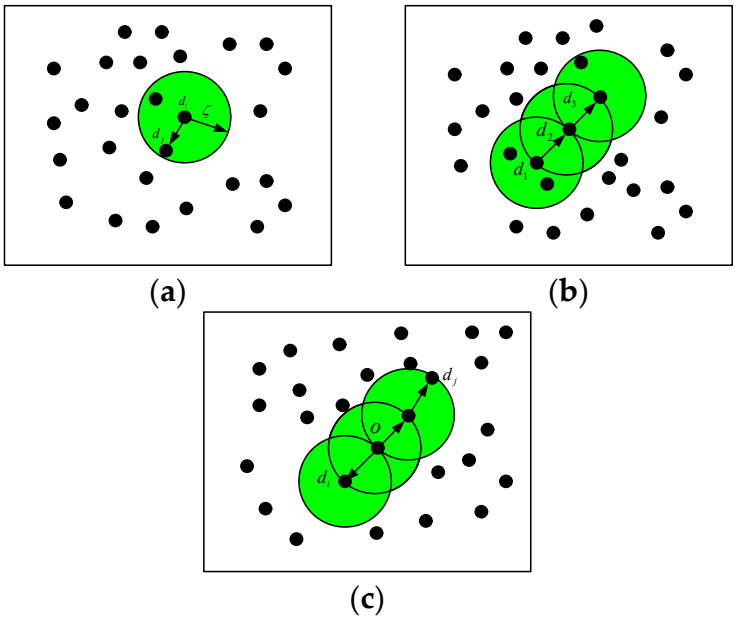


Figure 2. DBSCAN (**a**) directly density reachable, (**b**) density reachable, and (**c**) density connected.

Assume a sample set, $D = (d_1, d_2, \ldots, d_B)$, where B is the number of the samples. The DBSCAN algorithm is executed as follows: First, the neighborhood, ζ, and the minimum number of density points, MinPts, are set. Second, select a random point, d_i, from the sample set. Next, determine whether d_i is a core object. If the data d_i is a core object, all "density reachable" points of d_i in the sample data set are found and a new bunch with all other density reachable points is created. Third, a cluster is obtained based on the "density connected" of all points in the bunch. Following the above steps, all the data points in the

sample data set are processed to obtain the final clustering result: $D' = (d'_1, d'_2, \ldots, d'_{B'})$, as well as the noise. The above clustering process is noted as Equation (1).

$$(d'_i | i = 1, 2 \ldots B') = f_d(d_i | i = 1, 2 \ldots B), B' \leq B \tag{1}$$

3. The Proposed SDGD Model

We propose an optimization method to select valuable nodes based on the mean impact value of the nodes in the network. The gradients of the valuable nodes were clustered and denoised using the DBSCAN method. Considering the information of each sample's updating gradient inside a batch as a vector, it is assumed that they obey a certain distribution. In this way, we conduct clustering on this gradient information and remove the noisy gradients, which can effectively avoid the drawbacks of the original method of averaging by simply summation.

3.1. Keys Nodes Sparse Method Based on Mean Impact Value

Studies related to network sparsity have shown that only few of the nodes in a multi-layer multi-node network have a critical impact on the propagation of the subsequent layers. The mean impact value (MIV) method can quantify the impact weight of each neural node in the network propagation. For a neural network, the sample dataset is $X = (X_1, X_2, \ldots, X_N)$, where $X_n = (x_n^1, x_n^2, \ldots, x_n^l, \ldots, x_n^{L_1})$. The hth layer of the network has L_h nodes. The input of the hth layer corresponding to the input of X_n is $U_n^h = (u_n^1, u_n^2, \ldots, u_n^l, \ldots, u_n^{L_h})$, and the output of the layer is $V_n^h = (v_n^1, v_n^2, \ldots, v_n^l, \ldots, v_n^{L_h})$. The key nodes sparse method based on MIV is described as follows.

At the hth layer, the self-increment and self-subtraction operation is carried out on the input of its lth node.

$$U_{n,\pm\delta}^{h_l} = (u_n^1, u_n^2, \ldots, (1 \pm \delta)u_n^l, \ldots, u_n^{L_h}) \tag{2}$$

where, $U_{n,\pm\delta}^{h_l}$ is propagated from the hth layer to the output layer. The nonlinear iterative process transmitted forward from the hth layer is denoted as $\hat{Y} = F_h(U^h)$, where ψ_h is the nonlinear activation function of the hth layer. The new output obtained is noted as

$$\begin{aligned} \hat{Y}_{\pm}^{h_l} &= \sum_{j=h}^{H} \psi_j [\omega_j U_{n,\pm\delta}^{h_l} + b_j] \\ &= \sum_{j=h}^{H} \psi_j [\sum_{i=1}^{L_j} \omega_{ij} u_{n,\pm\delta}^i + b_{ij}] \\ &= F_h(U_{n,\pm\delta}^{h_l}) \end{aligned} \tag{3}$$

Therefore,

$$\begin{aligned} IV_l^h &= \left\| \hat{Y}_+^{h_l} - \hat{Y}_-^{h_l} \right\|_1 \\ &= \left\| \sum_{j=h}^{H} \psi_j [\sum_{i=1}^{L_j} \omega_{ij} u_{n,+\delta}^i + b_{ij}] - \sum_{j=h}^{H} \psi_j [\sum_{i=1}^{L_j} \omega_{ij} u_{n,-\delta}^i + b_{ij}] \right\|_1 \\ &= \left\| F_h(U_{n,+\delta}^{h_l}) - F_h(U_{n,-\delta}^{h_l}) \right\|_1 \end{aligned} \tag{4}$$

where IV_l^h is the impact value of the lth neuron in hth layer for the diagnosis network output. Similarly, the impact values of all other neurons in the hth layer can be derived as Equation (5).

$$IV^h = [IV_1^h, IV_2^h, \ldots, IV_l^h, \ldots, IV_{L_h}^h] \tag{5}$$

Each element in IV^h is the impact value of the corresponding neuron in the hth layer of the output. MIV^h can be obtained by summing up all the elements together.

$$MIV^h = \frac{1}{L_h}\sum_{i=1}^{L_h} IV_i^h \tag{6}$$

The value obtained by dividing IV_l^h with MIV^h can be considered as the relative impact value of the corresponding node. Based on the obtained relative impact values, the nodes in this layer can be effectively sparsed. Before the batch internal gradients update, the impact value threshold of the node can be set to θ. The nodes in each layer are filtered according to θ. If the impact value of the corresponding node is greater than θ, the node is marked as a key node, otherwise this node is sparsed. The connection values between key nodes are updated using DBSCAN. The remaining connection values are obtained by BSGD. By this method, the complexity of the algorithm operations is greatly reduced while the gradient updating is effectively performed.

3.2. Clustering Noise Reduction Method Based on Distribution Density

It is generally accepted that within each batch, the distribution density of its superior gradients will be greater than the distribution density of the other noisy updating values. The values outside the distribution edge of the intra-batch gradient clustering are treated as noise. In the process of training the diagnosis network, the loss function of the network is defined as Equation (7). B is the sample size of the batch.

$$\begin{aligned} J_b|b = 1,2\ldots B \quad &= \tfrac{1}{2}\sum_{l=1}^{L_H} e_l^2 + \lambda\sum(\omega)^2 \\ &= \tfrac{1}{2}\sum_{l=1}^{L_H} (y_l - \hat{y}_l)^2 + \lambda\sum(\omega)^2 \end{aligned} \tag{7}$$

where $\sum(\omega)^2$ denotes the L_2-normalization of all parameters of the network, and λ is the weight coefficient of the normalization term. The updating formulas for the weights $\nabla\omega_{ij}^b$ based on DBSCAN are as follows.

$$\nabla\omega'_{ij} = \frac{1}{B'}\sum_{n=1}^{B'} d'_n \tag{8}$$

$$(d'_n|n = 1,2\ldots B') = f_d(d_n|n = 1,2\ldots B) \tag{9}$$

$$\begin{aligned} d_n \quad &= \tfrac{\partial J_n}{\partial\omega_{ij}}|n = 1,2\ldots B \\ &= \frac{\partial\left(\frac{1}{2}[\sum_{j=1}^{H}\psi_j(\sum_{i=1}^{L_j}\omega_{ij}x_n^i + b_{ij}) - y_n]^2 + \lambda\sum_{j=1}^{H}\sum_{i=1}^{L_j}[\omega_{ij}]^2\right)}{\partial\omega_{ij}} \end{aligned} \tag{10}$$

The updating process of $\nabla b'_{ij}$ based on DBSCAN is similar and will not be detailed here.

3.3. Sparse Denoising Gradient Descent Optimization Algorithm

This section will introduce the sparse gradient denoising optimization algorithm in detail. Compared with the original algorithm's strategy of summing up the gradients in the batch and averaging them, this algorithm denoises the gradients of the samples within the batch using the DBSCAN-clustering method and then updating the gradients.

The structure diagram of the algorithm is shown in Figure 3. The key nodes of the network are marked by the MIV-sparse algorithm during each iteration, and the connection weights between the key nodes are clustered and denoised by the DBSCAN algorithm. Then the network weights are updated until the network training stopping condition is

satisfied. The trained diagnosis network is used to diagnose the test data and gives the diagnosis results.

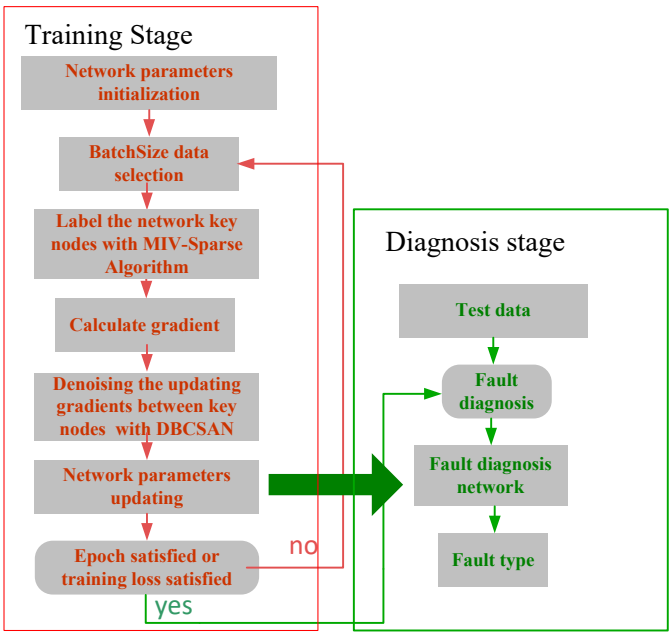


Figure 3. Structure diagram of SDGD optimized diagnosis network.

The training flow for the sparse denoising gradient decent optimization algorithm is shown in Figure 4.

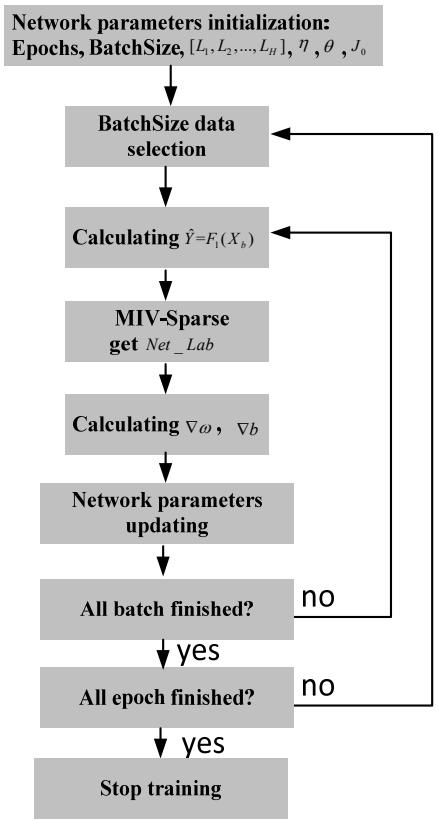


Figure 4. Training flow for the sparse denoising gradient decent optimization algorithm.

The process of the sparse denoising gradient descent optimization algorithm is described as follows.

(1) Setting network parameters: $[L_1, L_2, \ldots, L_H]$. BatchSize = B, learning rate η, impact value threshold θ, and the network connection parameters are randomly initialized. The training loss stop condition threshold J_0 and training epochs are set;

(2) The samples in the batch are sequentially fed into the network for forward propagation to obtain the predicted output value corresponding to that sample, and J_b is calculated. According to the loss J_b, the backward derivative is derived and the gradient updated values $\nabla\omega_{ij}^b = \frac{\partial J_b}{\partial \omega_{ij}}$, $\nabla b_{ij}^b = \frac{\partial J_b}{\partial b_{ij}}$ of all nodes of the network corresponding to each sample are calculated.

(3) According to Algorithm 1, all nodes of the network are marked and *Net_Lab* is obtained. Gradients of connection weights between key nodes are updated using the DBSCAN method $\nabla\omega'_{ij} = f_d(\nabla\omega_{ij}^b | b = 1, 2 \ldots B)$, $\nabla b'_{ij} = f_d(\nabla b_{ij}^b | b = 1, 2 \ldots B)$. The results are obtained by summing up and taking the average. $\omega_{ij}(k+1) = \omega_{ij}(k) - \eta \cdot \nabla\omega'_{ij}$, $b_{ij}(k+1) = b_{ij}(k) - \eta \cdot \nabla b'_{ij}$

Algorithm 1: MIV-Sparse algorithm.

-B: batch size;
-H: number of network layers;
-$[L_1, L_2, \ldots, L_H]$: node numbers per network layer;
-U^h: input of the hth layer;
-$\hat{Y} = F_h(U^h)$: the nonlinear iterative process transmitted from the hth layer backward;
-Input: network: $[L_1, L_2, \ldots, L_H]$, impact value threshold θ, training dataset
-Output: spared network: *Net_Lab*

1. ***for*** $n = 1 \ldots B$ ***do*** //samples traversal
2. ***for*** $i = 1 \ldots H$ *do* //transmission between input and output layers
3. ***for*** $k = 1 \ldots L_i$ ***do*** //node traversal per layer
4. $U_{n,\pm\delta k}^{h_i} = \left(u_n^1, u_n^2, \ldots, (1 \pm \delta)u_n^k, \ldots, u_n^{L_i}\right)$ //self-increment and self-subtraction operation
5. Forward Propagation: $\hat{Y}_{n,\pm k}^{h_i} = F_i(U_{n,\pm\delta k}^{h_i})$
6. $IV_{n,k}^{h_i} = \|\hat{Y}_{n,+k}^{h_i} - \hat{Y}_{n,-k}^{h_i}\|_1 = \|F_i(U_{n,+\delta k}^{h_i}) - F_i(U_{n,-\delta k}^{h_i})\|_1$
7. ***end for*** //quit node traversal
8. $IV_n^{h_i} = [IV_{n,1}^{h_i}, IV_{n,2}^{h_i}, \ldots, IV_{n,k'}^{h_i}, \ldots, IV_{n,L_i}^{h_i}]$ //impact value vector of the ith layer
9. Summation comparison: $IV_n^{h_i} = IV_n^{h_i} / \frac{1}{L_i} \sum_{k=1}^{L_i} IV_{n,k}^{h_i}$
10. ***end for*** //quit layer traversal
11. ***end for*** //quit samples traversal
12. ***for*** $i = 1 \ldots H$ ***do*** //marked the key nodes
13. $IV^{h_i} = \frac{1}{B} \sum_{n=1}^{B} IV_n^{h_i}$
14. All elements within IV^{h_i} are compared with θ. The network node greater than θ is marked as *1*, otherwise it is marked as *0*, and spare network *Net_Lab* is obtained.
15. ***end for***

The Pseudo code and flow chart of SDGD algorithm are shown in Algorithm 2 and Figure 5, respectively. During the forward propagation of the network, the key nodes of the net are marked with the MIV-sparse method. According to the loss function, J_b of each sample is calculated. Finally, gradients of the connection weights between the key nodes are updated using the DBSCAN method.

Algorithm 2: Sparse denoising gradient decent optimization algorithm

-J_b: loss function;
-$\nabla\omega$: connection weight gradient;
-∇b: connection bias gradient;
-IV: impact value of nodes;
-MIV: mean impact value;
-Input: network parameters $[L_1, L_2, \ldots, L_H]$, $BatchSize = B$, learning rate η, impact value threshold θ, training loss stop condition threshold J_0, training epochs;
-Output: the trained network Net_{tra}
1. ***for*** $k = 1 \ldots$ epochs ***do*** //Cycle epochs times
2. ***for*** $b = 1 \ldots$ B ***do*** //All samples traversal
3. Calculating $\hat{Y} = F_1(X_b)$ //Forward propagation
4. Calculating loss function J_b;
5. ***if*** $J_b \leq J_0$ //Whether the stopping condition is satisfied
6. ***yes:***stop training;
7. ***else***
8. $\nabla\omega_{ij}^b = \frac{\partial J_b}{\partial \omega_{ij}}$, $\nabla b_{ij}^b = \frac{\partial J_b}{\partial b_{ij}}$; //Calculate the corresponding gradient update value for each sample
9. get Net_Lab //Net_Lab is obtained according to Algorithm 1
10. $\nabla\omega'_{ij} = f_d(\nabla\omega_{ij}^b | b = 1, 2 \ldots B)$, $\nabla b'_{ij} = f_d(\nabla b_{ij}^b | b = 1, 2 \ldots B)$ //Calculating gradient update values by DBSCAN method;
11. $\omega_{ij}(k+1) = \omega_{ij}(k) - \eta \cdot \nabla\omega'_{ij}$, $b_{ij}(k+1) = b_{ij}(k) - \eta \cdot \nabla b'_{ij}$ //Updating network connection weights
12. ***end if***
13. ***end for***
14. ***end for***

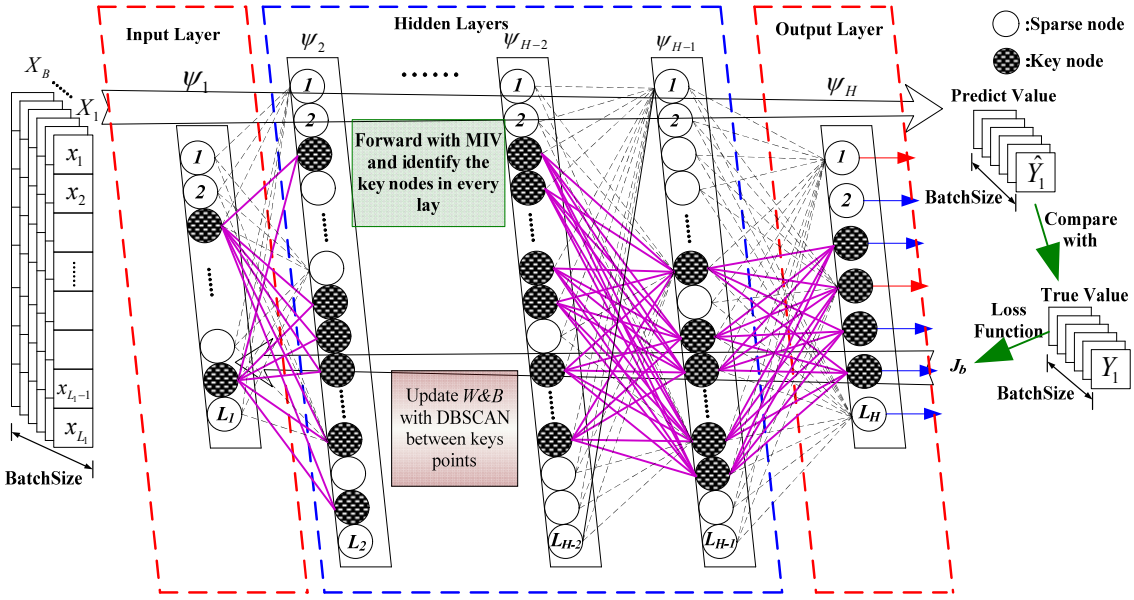


Figure 5. SDGD algorithm flow chart.

The input data for each batch in Figure 5 are $X_1, X_2, \ldots, X_B$ and $X_1 = (x_1, x_2, \ldots, x_l, \ldots, x_{L1})$. The network consists of three pieces: the input layer, the hidden layers, and the output layer. The white neurons in the network layer indicate less important nodes after MIV filtering. The black neurons indicate key nodes. The neuronal connections between all key nodes are indicated by the purple connecting lines, which are updated with the SDGD algorithm during the gradient update. The rest of the connection lines are connected by dashed lines. The red arrows in the output layer indicate that the output value is positive at that location, while blue is negative. The input data are first propagated forward, indicated by the large green arrow, to give the predicted output $\hat{Y}$. The loss value, J, is calculated after a comparison with the real output. The network weights are then updated by back-propagating through the large red arrow. $W\&B$ in the figure indicates the network weights.

The network structure in Figure 5 is an abstract representation. For a convolutional-type network, the input and hidden layers correspond to the convolutional and pooling layers. While the output layer corresponds to the flattened and fully connected layers as well as the output layer.

4. Validation Experiments

In order to demonstrate the effectiveness of the proposed method for fault diagnoses, two fault diagnosis experiments were conducted to verify it. The advantages of the proposed method are mainly reflected in its ability to improve the convergence rate of the existing algorithms in the training process, avoid local optimization, and prevent oversaturation. It represents an improvement in the training process. The existing data-driven intelligent diagnosis methods can be divided into two categories. One is the network model, with a strong feature extraction ability, such as convolutional neural networking and its deformation. The other is to process the signal first, extract the characteristics of the signal, and then send them to the network model, such as DNN and SVM. Therefore, in this part of the experimental verification, we selected several widely used network structures in the field of fault diagnosis: Resnet, random CNN, SAE, and DNN, based on feature extraction of the original data for comparative experimental analysis.

The diagnostic object of case one is a "motor drive system bearings dataset" publicly available at Case Western Reserve University. In this case, we use a fault diagnosis method based on Resnet random-CNN and -SAE. The experimental results show that the proposed algorithm in the manuscript has significantly improved the diagnostic accuracy and training convergence speed of the network. The object of case two is the rolling bearing dataset published by Xi'an Jiaotong University. In this case, a fault diagnosis neural network based on time-frequency domain feature extraction is built. The experimental results show that the proposed algorithm significantly improved the solving of the local optimal point trap.

4.1. Case One: Study of Accuracy & Convergence Speed of Fault Diagnosis Model

4.1.1. Dataset Preparation and Parameter Settings

This experiment case uses the open-source rolling bearing fault dataset from Case Western Reserve University as the experimental dataset [45]. The experimental object is a motor drive system. The vibration acceleration signal of the device is collected by installing sensors at different locations in the drive system. The collection location includes the drive end, fan end, and base. The load on the motor can change during operation. The load change range is 0–3 hp. The states of the bearing after failure are simulated by manually setting different levels of damage at various locations on the bearing. The details of the three different degrees and the three different locations of damage are shown in Table 1. They combine with each other for a total of nine different faults. The data selected for this experiment were collected by drive end at a 0 hp load. They are ball defect I (BDI), ball defect II (BDII), ball defect III (BD III), inner race defect I (IRI), inner race defect II (IRII), inner race defect III (IRIII), outer race defect I (ORI), outer race defect II (ORII), and outer race defect III (ORIII), respectively. The sampling frequency of the vibration sensor was 12 kHz and the motor speed was set to 1772 rpm. It can be calculated that there are about 400 sampling points per circle. The sample status information is shown in Table 1. The samples are processed using a k-fold cross validation method in order to ensure that the experiment results are non-accidental.

Table 1. Data information for the nine types of fault states.

Fault Type	Fault Diameter (inch)	Label	Sample Size
BDI	0.007	1	1000
BDII	0.014	2	1000
BDIII	0.021	3	1000
IRI	0.007	4	1000
IRII	0.014	5	1000
IRIII	0.021	6	1000
ORI	0.007	7	1000
ORII	0.014	8	1000
ORIII	0.021	9	1000

The acceleration signals for the nine types of faults on the drive end are shown in Figure 6.

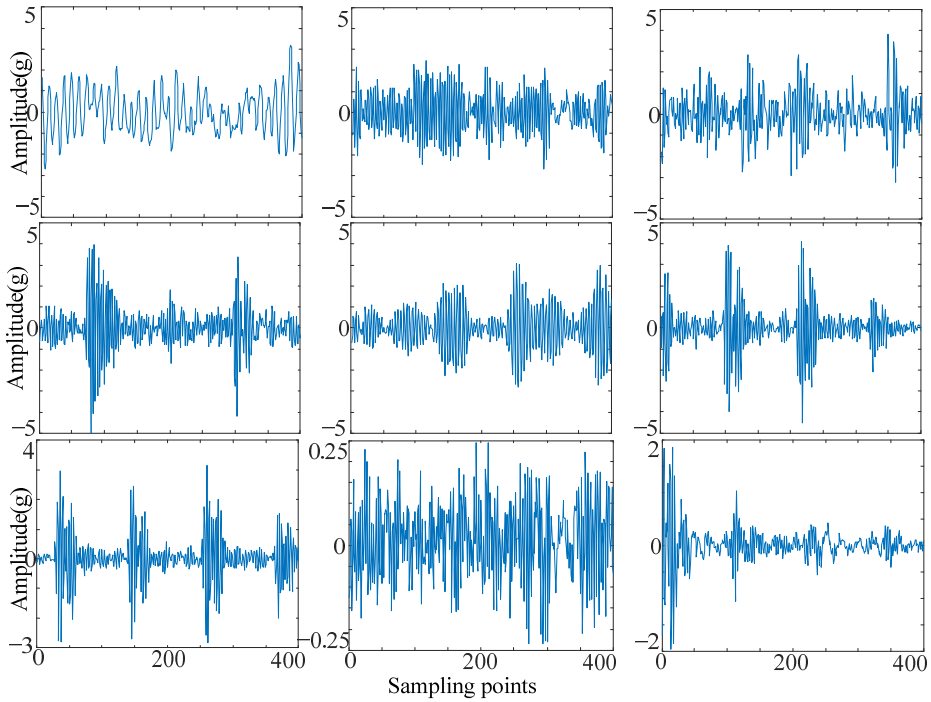


Figure 6. Acceleration signals for the nine types of faults on the drive end.

In the process of the experimental validation, after many experiments and analyses, we have given a range for the selection of the parameters. First, for the two parameters, ζ and *MinPts*, of DBSCAN, we know that, for the minimum distance, ζ, if the value is set too large, the denoising effect will not be achieved; if it is too small, the total number of categories will be too large. Therefore, after several choices, we decided to select the maximum standard deviation of all samples within the batch as the basis. The best results are achieved when set to 0.3–0.4 times. For the parameter *MinPts*, the aim is to achieve the best possible filtering effect, and we want the valid samples to be clustered together as much as possible. Therefore, we assume that the gradient information is useful for most of the samples, and when its multiplicity is set to 1, it is the normal MGD algorithm. During the experiments we found that, if the parameter was set too large, the clustering result was similar to MGD. If it was too small, there will be many clusters, and the denoising performance will be poor. Therefore, we suggest 0.3–0.5 times the BatchSize to speed up clustering and effectively achieve the effect of noise removal. It has little impact on the iteration rate of the network. The connection values for the top 50% of the network are set to use the proposed algorithm in the paper for gradient updating.

4.1.2. Comparative Experiments Based on RESNET

Resnet-18 networks are usually used to process image datasets of 255*255*3 using 3 channels. Conventional Resnet-18 has four Res blocks, and each Res block is composed of four convolution layers. Since the dimensions of the input data used in the experiment RE 400*1, we have simplified the network structure and removed two Res blocks (Conv4 and Conv5) to speed up the convergence speed. The network parameters are set as per Table 2. The input data to the network are the 1D raw vibration signal data, shown in Figure 5, and the input data for the network in case one is a time-series 400-dimensional input data, as shown in Figure 6.

Table 2. Resnet network settings.

Layer Number	Layer Name	Core/Pool Size	Output Shape
1	Conv1	[5,5]	[400,5]
2	Pooling1	4(stride = 1)	[100,5]
3	Conv2	$\begin{bmatrix} 6,5 \\ 6,5 \end{bmatrix} \times 2$	$[100,5^4]$
4	Conv3	$\begin{bmatrix} 6,5 \\ 6,5 \end{bmatrix} \times 2$	$[100,5^8]$
5	Output AvergePooling+FC+Softmax 9		

The input data for the CNN and SAE are also the 1D raw vibration signal data, shown in Figure 5, and the shape of the data is 400*1. The input data for the DNN in case two are 10*1 dimensional feature data processed in the frequency domain, with reference to [12].

It can be seen that the (10 times) average accuracy of the SDGD-Resnet method is nearly the same when compared to that of the common Resnet diagnosis method from Table 3.

Table 3. Accuracy comparison.

No.	RESNET (Acc:%)	SDGD-RESNET (Acc:%)
1	99.44	99.78
2	99.56	99.67
3	99.72	99.44
4	99.67	99.56
5	99.44	99.50
6	99.50	99.61
7	99.78	99.78
8	99.78	99.72
9	99.61	99.34
10	99.38	99.78
Mean accuracy	99.58	99.62

We have visualized the results of a particular run as an example. The visualization results of the proposed method, with the usage of t-distributed stochastic neighbor embedding (t-SNE) technology, are shown in Figure 7. It is observed that some categories, such as type 8, are separated from other categories. However, type 1, type 2, and type 3 overlap. Moreover, the confusion matrix for the proposed method for the test set is calculated as exhibited. It can be seen that most of the fault types can be identified clearly. The results are consistent with the visual inspection results.

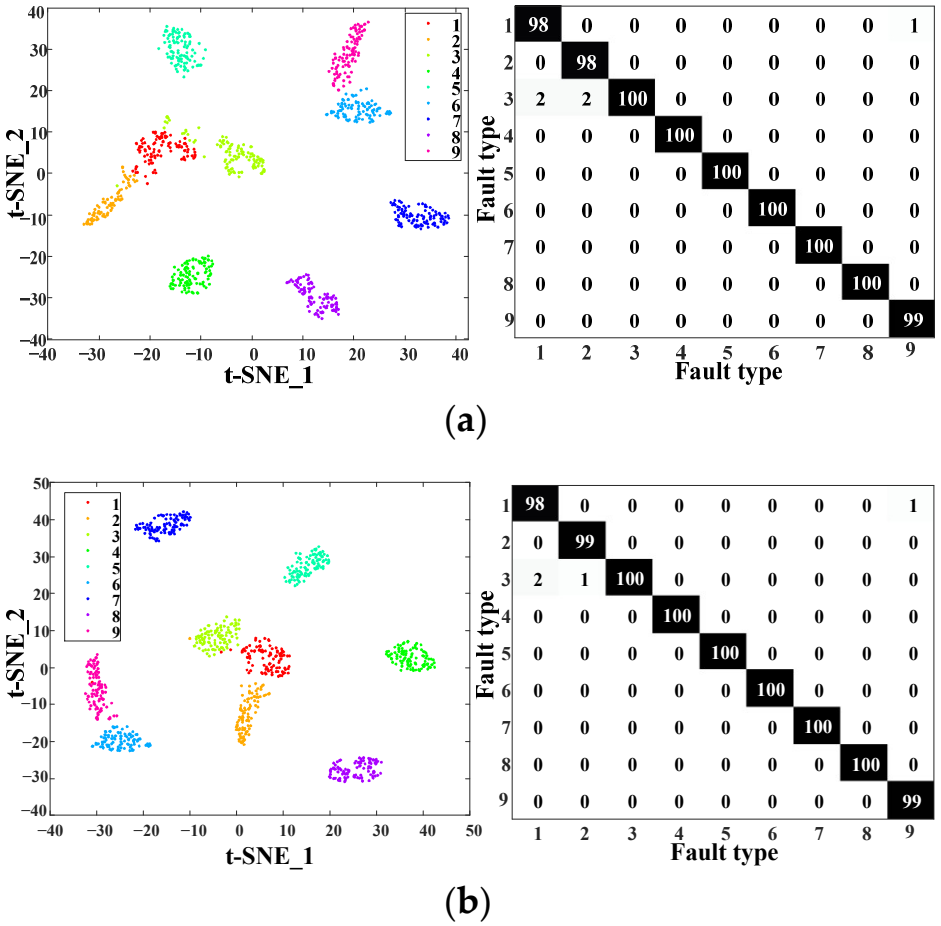
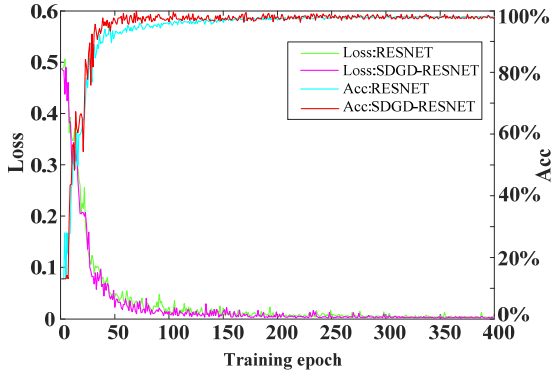


Figure 7. (**a**) Visualization and prediction error matrix of RESNET (**b**) Visualization and prediction error matrix of SDGD-RESNET.

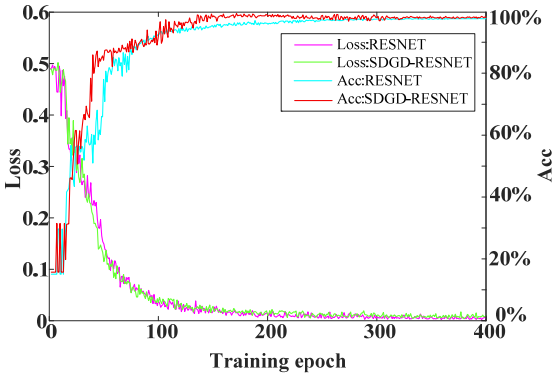
The convergence speed of the proposed algorithm (green line) is significantly better than that of the common RESNET diagnosis method, denoted as the purple line. Correspondingly, the diagnostic accuracy of the proposed algorithm (red line) improved faster than that of the common RESNET method, denoted as the blue line. With larger batchsize settings, as illustrated in Figure 8d, the network's accuracy improves significantly faster than the conventional networks. Moreover, Table 4 demonstrates that SDGD-RESNET outperforms RESNET by a maximum of 14.89% in the impact on convergence speed, which shows that SDGD-RESNET is better than RESNET on convergence performance while maintaining a comparable diagnostic accuracy to RESNET.

Table 4. Convergence speed comparison.

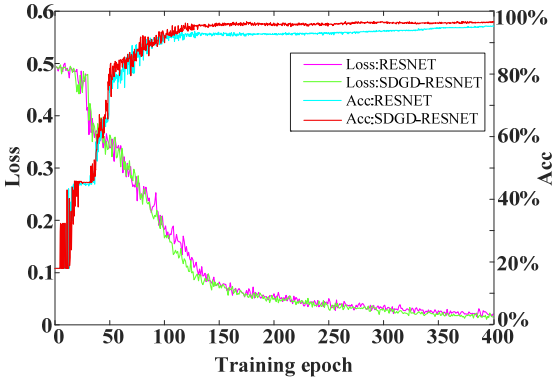
BatchSize	SDGD-Resnet (Epochs)	Resnet (Epochs)	Loss-Aim	Improvement Index (%)
50	39	45	0.05	13.33
100	80	94	0.05	14.89
200	125	135	0.05	7.41
350	478	559	0.05	14.49



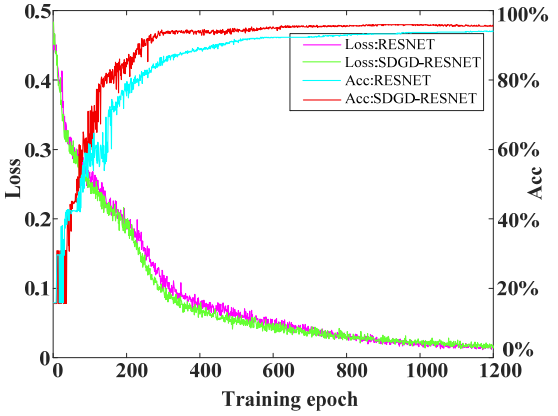
(**a**)



(**b**)



(**c**)



(**d**)

Figure 8. Convergence speed comparison (**a**) BatchSize = 50 (**b**) BatchSize = 100 (**c**) BatchSize = 200 (**d**) BatchSize = 350.

4.1.3. Comparative Experiments Based on Random CNN

Faults 1, 2, 4, 5, 7, and 8 were selected as the diagnostic types. The network parameters are set as per Tables 5 and 6, referring to [11].

Table 5. Convolutional network settings.

Parameters	First Convolution Layer	Second Convolution Layer
Number of filters	5	4
Size of filter	16	18
Stride	1	1

Table 6. Pooling Layer.

Parameters	First Pooling Layer	Second Pooling Layer
Pooling size	5	4
Stride	1	1

Table 7 shows a comparison of the results for 10 rounds. As can be seen, the proposed SDGD algorithm can improve the diagnostic accuracy of the network by 2.35% to 99.13%.

Table 7. Accuracy comparison.

No.	CNN (Acc:%)	SDGD−CNN (Acc:%)
1	97.33	99.17
2	97.00	99.33
3	96.83	99.00
4	96.86	98.83
5	96.50	99.17
6	95.86	99.17
7	96.33	99.33
8	97.57	99.00
9	96.50	99.00
10	97.00	99.33
Mean accuracy	96.78	99.13

The visualization results are shown in Figure 9.

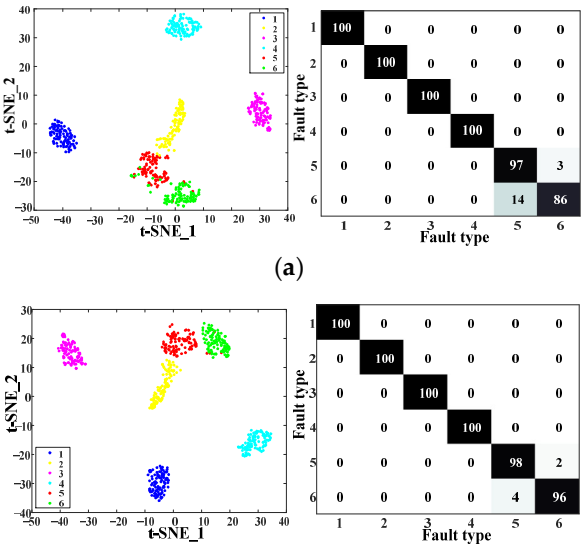


Figure 9. (**a**) Visualization and prediction error matrix of CNN. (**b**) Visualization and prediction error matrix of SDGD-CNN.

The comparison of the convergence speed between the proposed algorithm and the common CNN-based algorithm are shown in Figure 10.

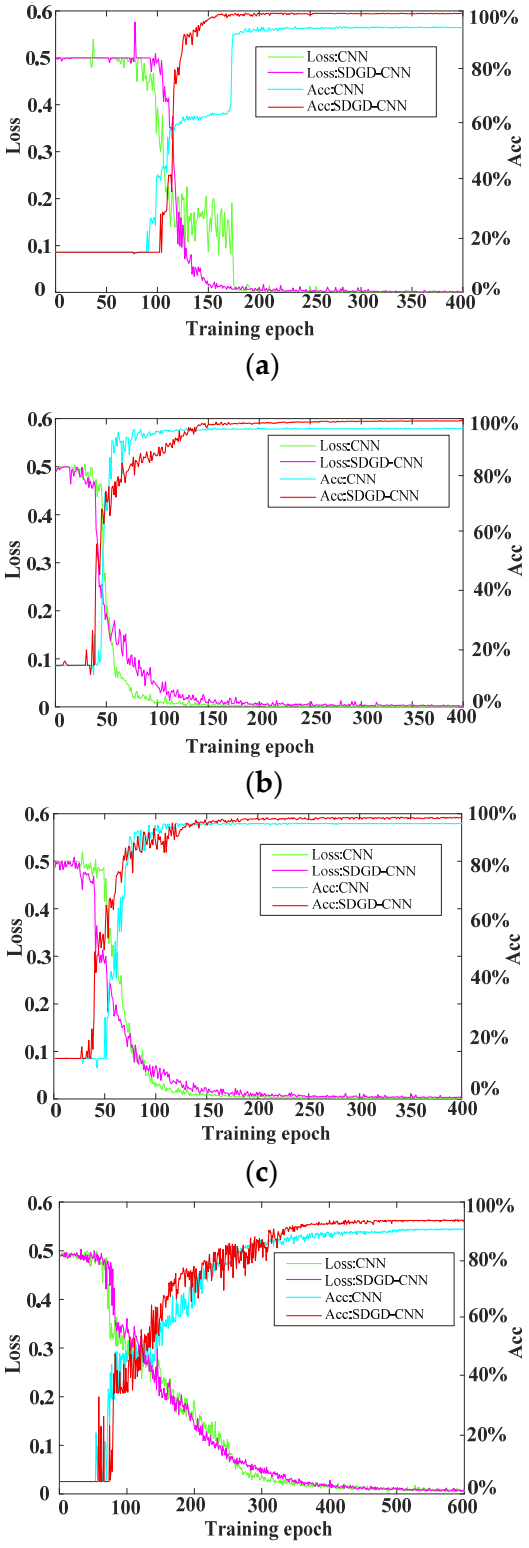


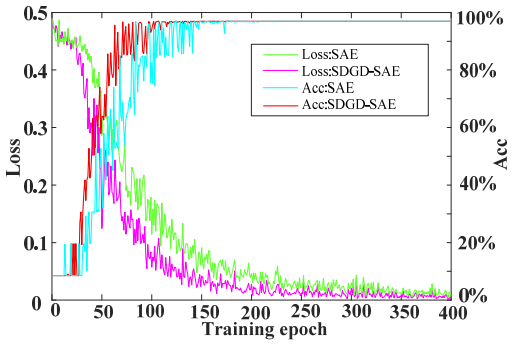
Figure 10. Convergence speed comparison (**a**) BatchSize = 40 (**b**) BatchSize = 50 (**c**) BatchSize = 100 (**d**) BatchSize = 200.

The diagnostic accuracy comparisons are shown in Table 8.

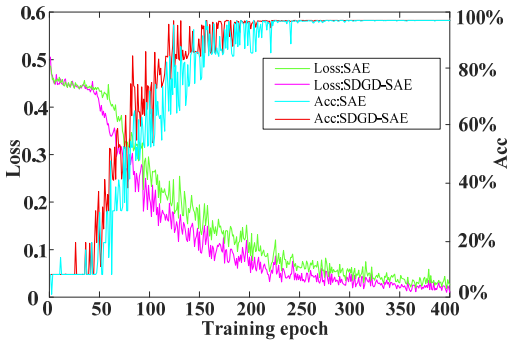
Table 8. Diagnostic accuracy comparison.

BatchSize	SDGD-CNN (Acc:%)	CNN (Acc:%)	Improvement Index (%)
40	99.13	96.78	2.35
50	98.45	96.29	2.16
100	98.17	94.71	3.46
200	94.71	92.01	2.7

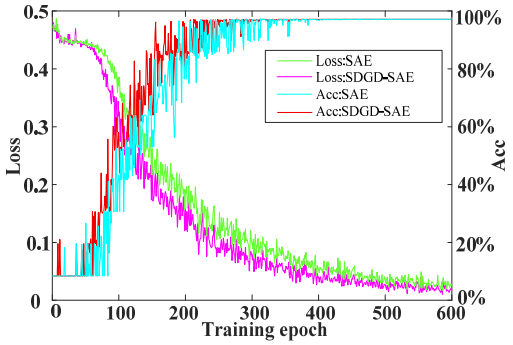
In Figure 11, the diagnostic accuracy of the proposed algorithm is denoted by the red line and the common CNN method by the blue line. After comparing the two curves, it is clear that the SDGD algorithm can improve the diagnostic accuracy of the network. When the batchsize chosen is too large, this will lead to decreasing training efficiency. As can be seen from the figures, the declines in loss are comparable for both. Although the loss decreases at the same level, the SDGD algorithm can obviously enhance the network's generalization ability, improving the diagnosis accuracy of the network, and reducing the saturation degree of the network. In Figure 10a, the CNN sometimes falls into a local optimum during training, while SDGD effectively solves this problem.



(**a**)



(**b**)



(**c**)

Figure 11. Convergence speed comparison (**a**) BatchSize = 40 (**b**) BatchSize = 50 (**c**) BatchSize = 100.

4.1.4. Comparative Experiments Based on Sparse Auto-Encoder (SAE)

A total of nine fault types were used as the inputs for this experiment. The SAE hidden layer parameters were set to 512/400/300/200/100. The sparse method of SDGD-SAE in this paper has strong superiority over the traditional SAE sparse method. Table 9 shows a comparison of the results for 10 rounds. It can be seen that the (10 times) average accuracy of the SDGD-SAE method is nearly the same when compared with the common SAE method.

Table 9. Accuracy comparison.

No.	SAE (Acc:%)	SDGD-SAE (Acc:%)
1	96.56	97.33
2	96.44	96.44
3	96.00	96.89
4	96.22	97.00
5	96.56	97.33
6	97.11	97.00
7	96.67	96.67
8	95.78	96.11
9	97.00	97.21
10	97.56	96.89
Mean accuracy	96.59	96.89

The comparison of the convergence speed between SDGD-SAE and SAE is shown in Figure 11. Table 10 shows that SDGD-SAE improves the convergence speed by up to 17.68% over SAE.

Table 10. Convergence speed comparison.

BatchSize	SDGD-SAE (Epochs)	SAE (Epochs)	Loss-Aim	Improvement Index (%)
40	163	198	0.05	17.68
50	236	273	0.05	13.55
100	393	449	0.05	12.47

4.1.5. Computational Cost

The resource configuration of the experimental platform: Window 10 system, Matlab2019a platform, Cpu @ 2.3GHz, RAM 8GB.

When the batchsize is 40, SDGD-ResNet takes 0.39 s per round, while ResNet takes 0.36 s; SDGD-CNN takes 0.25 s per round, while CNN takes 0.22 s; SDGD-SAE takes 0.05 s per round, while SAE takes 0.02 s. It can be seen that, as the computational complexity increases, the computational time consumption increases slightly.

4.2. Case Two: Study of Local Optimal Trap of Fault Diagnosis Model Training

4.2.1. Dataset Preparation and Parameter Settings

The data set for the rolling bearings from Xi'an Jiaotong University (XJTU-SY) is used in this case [46]. The vibration acceleration signals were collected in the horizontal and vertical directions of the experimental test bench, respectively. When the vibration signal vibration amplitude exceeds 10 g, it is determined that the test experiment has failed. When a failure occurs, according to the bearing failure location, the fault types are divided into inner ring defect (ID), outer race defect (OR), and cage defect (CD). The experimental conditions are set as follows: speed 2250 rpm, sampling frequency 25.6 kHz, sampling duration 1.28s, and sampling interval 1 min. When selecting 880 consecutive horizontal vibration signal sampling points as one sample, 37 samples can be obtained for each sampling. The experimental data set is shown in Table 11. The acceleration signals for the three types of faults are shown in Figure 12.

Table 11. Data information on three types of fault states.

Fault Type	Label	Train Sample Size	Test Sample Size
ID	100	265	105
OR	010	266	104
CD	001	269	101

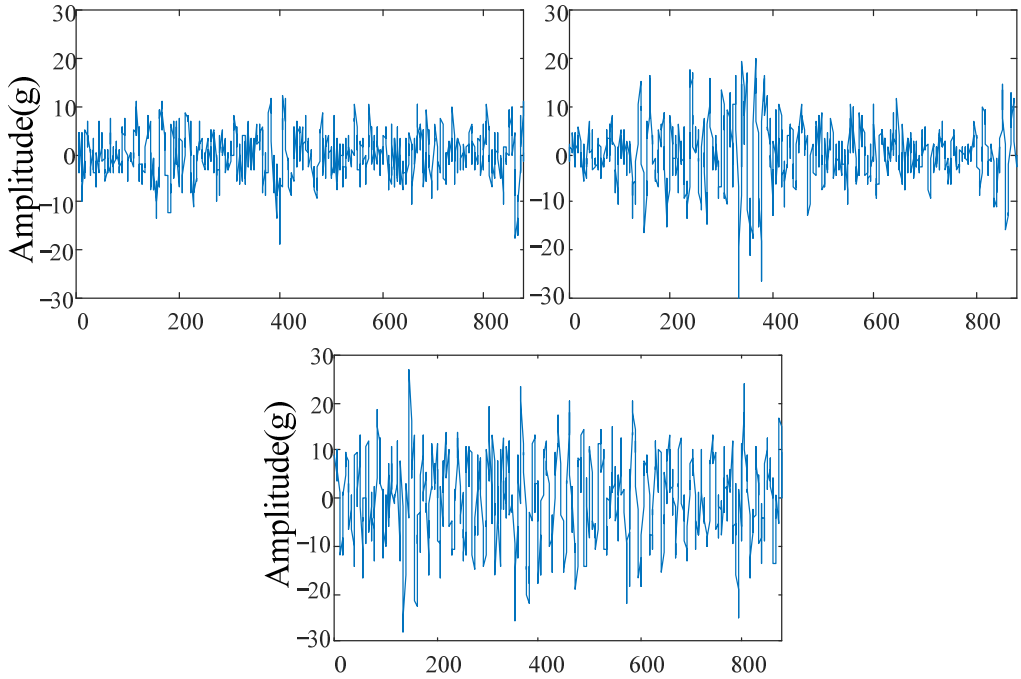
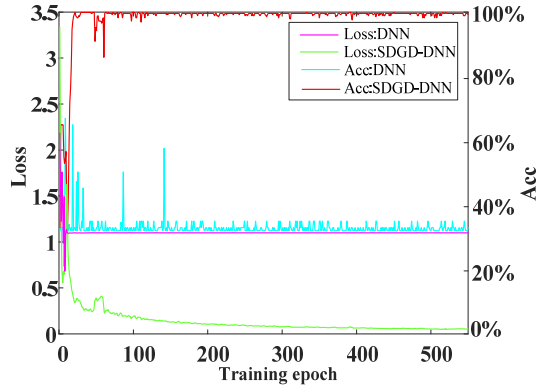


Figure 12. Acceleration signals for the three types of faults.

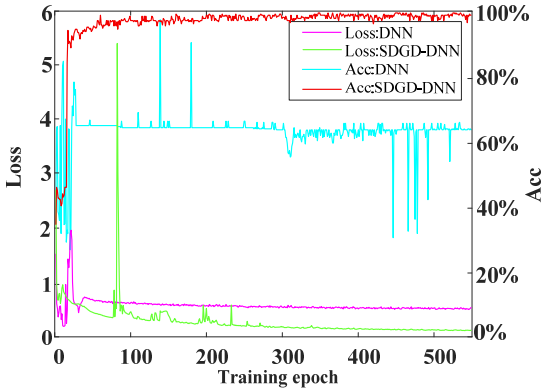
The setting ranges for parameters ζ and *MinPts* in DBSCAN are recommended as follows. ζ is set between 0.3~0.4 times (of the largest standard deviation of all the batch samples) and *MinPts* is set to 0.3~0.5 times that of the batch size. This has little impact on the iteration rate of the network. The connection values for the top 50% of the network are set to use the proposed algorithm in the paper for gradient updating. The experiments are simulated in the Matlab platform. The initial values of the diagnostic network can be assigned by a random initialization algorithm. The simulation platform can control the random values generated by setting a random seed with the 'rng()' function.

A fault diagnosis neural network with time-frequency domain features was built. The initialization parameters of the network are controlled by setting the random seed. Compared with common diagnosis networks, which often fall into local optimal points during training, the proposed method can effectively avoid this situation. Referring to [12], 10 statistical features in the time-frequency domain are extracted for training the network, including: Absolute mean, Variance, Crest, Clearance factor, Kurtosis, Crest factor, Skewness, Pulse factor, Root mean square, Shape factor. The network parameters are set as follows: the network input layer shape is 15, the hidden layer shape is 31 + 30, and the output layer shape is 3.

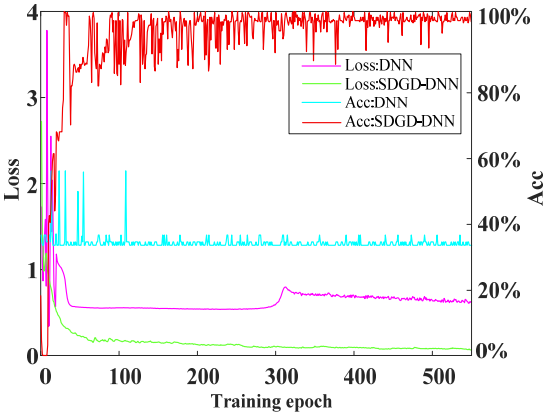
In Figure 13, the red line and blue line are the accuracy rates of DNN and SDGD-DNN during training, respectively, and the green line and purple line are the training losses for them.



(**a**)



(**b**)



(**c**)

Figure 13. Convergence speed comparison (**a**) rng(4) (**b**) rng(200) (**c**) rng(258).

From the experimental accuracy, it can be seen that the proposed optimization algorithm can effectively solve the problem of the network falling into local optimal points during the training process and cannot continue to improve the network accuracy under certain initialization parameters. As shown in the three experimental results in Table 12, with the same initialization network parameters, the proposed algorithm can effectively avoid falling into local optimal points and achieve a test accuracy of as much as 99%.

Table 12. Accuracy comparison.

Random Seed	rng (4)	rng (200)	rng (258)
DNN accuracy	33%	63.67%	65%
SDGD-DNN accuracy	99.33%	99%	98.33%

4.3. Experimental Analysis

Case one aims to reveal the improvement of the SDGD algorithm concerning model convergence speed and the prevention of over fitting. We plan to verify the Resnet, Random-CNN, and SAE network models. Case Two mainly aims to reveal the ability of the SDGD algorithm to effectively avoid falling into local optimum during model training. Compared with the model used in case one, the DNN-based network model can be effectively trapped into local optimization by setting random seeds. In Section 4.2.1, we pointed out that DNNs can easily fall into local optimization by designing random seeds. However, the Resnet-based, Random-CNN-based and SAE-based network models find it difficult to effectively find and reproduce the local optimum during their training process. Therefore, case two will choose a different network model structure from case one to implement validation. The comparisons between the performances in terms of diagnostic accuracy, convergence speed, and local optimal point trap involved in the experiments are given below. A table comparing the experiments based on case one above is shown in Table 13.

Table 13. Methods comparison.

Method	With SDGD	Fault Types	Acc:%	Acc Improve:%	Convergence Speed Improve:%	Avoid Local Optimal Trap
Resnet	N Y	9 (CRWU)	99.58 99.62	0.04	7.41–14.89	N
CNN	N Y	6 (CRWU)	96.78 99.13	2.35	0–5.71	Y
SAE	N Y	9 (CRWU)	96.59 96.89	0.3	12.47–17.68	N
DNN	N Y	3 (XJT)	53.89 98.89	33.33-66.33	-	Y

Note:"-": The DNN loss did not meet the training target, so they were not counted.

(1) Diagnostic accuracy

As can be seen from Table 13, for different diagnostic models, the proposed algorithm enables an improvement in the convergence speed. The accuracy of the modules has also improved to some extent. The main reasons for this improvement can be attributed to the following two aspects. First, based on published research, it is known that there is an amount of redundant structure in neural networks, so the MIV-based method can effectively filter out the redundant weights. Thus, the network's generalization capability becomes enhanced. Therefore, the SDGD model allows for the improvement of diagnostic capabilities for different network structures.

(2) Convergence speed & Local optimal point trap

When performing weight updating, the DBSCAN method is used to cluster and denoise the batch gradients. This gradient updating method effectively removes gradient noise, thus, allowing the model to converge quickly, as we can see in the comparison of the experimental results in case one. The experimental results show large convergence speed improvements for various models and for different batch sizes. At the same time, conventional optimization methods often fall into local optimization points due to gradient noise. The proposed SDGD model is effective in removing the noise interference, so that in case two, it can effectively avoid the situation of falling into the local optimal point trap.

5. SDGD's Application on a Ship Engine System

The structure of the application of SDGD on a ship engine system is shown in Figure 14. The collected historical data were used to train the SDGD-based diagnosis network until the stop condition was satisfied. Then, the data collected in real-time were fed into the trained network to give a diagnosis result. At the same time, the collected new data were also sent to the network for further training, which is used to update the network weights

in real-time. With the fast convergence property of SDGD, the real-time training of the network is faster, and the real-time performance is guaranteed.

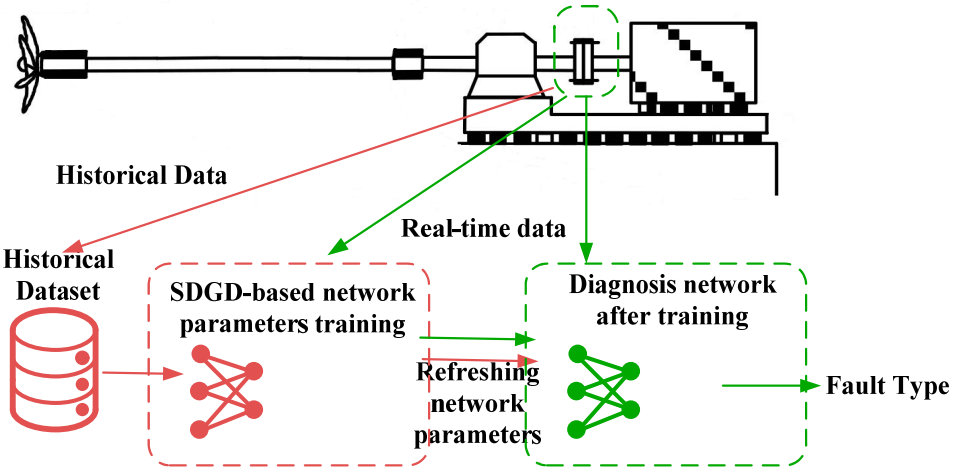


Figure 14. Model figure of sparse gradient denoising optimized diagnosis network.

The SDGD optimized neural network model consists of three main components: the establishment of a historical database, the training of the diagnostic model, and real-time fault diagnosis.

(1) Establishment of a historical database. The data are obtained by installing vibration or stress sensors at different positions on the marine powertrain to obtain data on its different fault states. Also, information on the operation of the equipment needs to be recorded. After the data have been collected, the acquired data are classified and stored. Tagging the different fault states and building a database to provide data support for the training of the network.

(2) Training of diagnostic model. After the collection of historical data, a local network model is built. The model is efficiently trained using the SDGD optimization algorithm. After the training stop condition is met, the network parameters are saved. During real-time diagnosis, the newly collected fault data can also be used to train the network so that the network parameters can be updated in real-time.

(3) Real-time fault diagnosis. The parameters, after refreshing, are used as the parameters for the diagnostic network. The vibration data from the equipment are collected by sensors and sent to a local computer via a communication network. Once a fault has occurred, the device immediately alarms.

In the experimental verification part of the previous section, case one involves the SKF and NTN bearings; The SY bearings were involved in case two. These kinds of bearings are widely used in marine power mechanical systems. The proposed diagnostic network has good applicability to the above different data sets. In the real-time diagnosis stage of the model, the new fault data will dynamically update the data set in real-time, and the new data and historical data will participate in the learning and training of the network model together, which makes the diagnosis network have good adaptability when facing new fault types.

In these experiments, we are simulating a real-time system when conducting the model tests. The model runs at *ms* level during testing, so we can assume that this is a real-time system. We split all the data sets into training and test sets. When we train the network with the training set, this corresponds to the training part of the red box in Figure 14. During testing, the data from the test set can be considered as data collected by the device in real-time and are sent to the network for real-time fault diagnosis. Due to practical constraints, we are still a long way from actually conducting such experiments. However, this assumption is entirely reasonable. This kind of simulation has clear practical significance.

The above mechanism makes the proposed method feasible for practical application in the rolling bearing fault diagnosis system of ocean ships.

6. Conclusions

In order to improve the performance of neural network-based fault diagnosis methods, this paper proposes the SDGD method. The interference of gradient noise with network convergence is effectively removed, while the sparsity of the network structure is also well considered at the same time. Based on this method, the Resnet-, random CNN-SAE-, and DNN-based fault diagnosis methods were established, respectively. The improved diagnosis network based on SDGD is compared with the classical algorithm. It was verified that the proposed algorithm can effectively accelerate the convergence speed of the diagnosis network while ensuring diagnosis accuracy and can effectively solve the problem of the diagnosis network falling into the local optimal points.

Although some satisfactory results have been obtained with the proposed method, the limitations of this work should be soberly recognized. First, we found that the computational speed of the clustering algorithm can have a large impact on the convergence speed of the diagnostic model. For example, if the DBSCAN parameters are not set correctly, the calculation time will significantly increase. Therefore, knowing how to cluster effectively and improve the clustering speed and, thus, further accelerate the convergence speed are areas that need to be further investigated.

Moreover, the MIV approach proved to be effective over the course of the study but lacks theoretical support. It is worth investigating whether the removed neurons may also have an effect on the network. This will give the MIV method its theoretical basis when selecting the number of key nodes. In addition, the interpretability principle of the MIV-based sparse method proposed in this paper requires further research.

Author Contributions: S.W. and Y.Z. wrote the manuscript. S.W. wrote the algorithmic program and performed the experiments; Y.Z. built the methodology, conceived and supervised the research and experiments, contributed as the lead author of the article; B.Z. and Y.Z. gave the review and editing; Y.F. analyzed the data; Y.H. performed the investigation; P.L. audited the data; M.X. performed the validation and visualization. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (no. 61673259); supported by Shanghai "Science and Technology Innovation Action Plan" Hong Kong, Macao and Taiwan Science and Technology Cooperation Project (no.21510760600); and also supported by Capacity Building Project of Local Colleges and Universities of Shanghai (no. 21010501900).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available in [45,46].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, Y.; Chen, Z.; Tang, L.; Zhai, W. Skidding dynamic performance of rolling bearing with cage flexibility under accelerating conditions. *Mech. Syst. Signal Processing* **2021**, *150*, 107257. [CrossRef]
2. Wang, S.; Zhang, Y. Multi-Level Federated Network Based on Interpretable Indicators for Ship Rolling Bearing Fault Diagnosis. *J. Mar. Sci. Eng.* **2022**, *10*, 743. [CrossRef]
3. Hou, J.; Wu, Y.; Ahmad, A.; Gong, H.; Liu, L. A Novel Rolling Bearing Fault Diagnosis Method Based on Adaptive Feature Selection and Clustering. *IEEE Access* **2021**, *9*, 99756–99767. [CrossRef]
4. Wang, Y.; Ding, X.; Zeng, Q.; Wang, L.; Shao, Y. Intelligent Rolling Bearing Fault Diagnosis via Vision ConvNet. *IEEE Sens. J.* **2021**, *21*, 6600–6609. [CrossRef]
5. Zheng, H.; Yang, Y.; Yin, J.; Li, Y.; Wang, R.; Xu, M. Deep Domain Generalization Combining A Priori Diagnosis Knowledge Toward Cross-Domain Fault Diagnosis of Rolling Bearing. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–11. [CrossRef]
6. Zhang, Y.; Zhang, Z.; Chen, L.; Wang, X. Reinforcement Learning-Based Opportunistic Routing Protocol for Underwater Acoustic Sensor Networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 2756–2770. [CrossRef]
7. Zhang, Y.; Liu, Q. On IoT intrusion detection based on data augmentation for enhancing learning on unbalanced samples. *Future Gener. Comput. Syst. Int. J. Escience* **2022**, *133*, 213–227. [CrossRef]

8. Kong, X.G.; Fu, Y.; Wang, Q.B.; Ma, H.; Wu, X.; Mao, G. A High Generalizable Feature Extraction Method Using Ensemble Learning and Deep Auto-Encoders for Operational Reliability Assessment of Bearings. *Neural Processing Lett.* **2020**, *51*, 383–406. [CrossRef]
9. Ye, Z.; Zhang, Q.; Shao, S.; Niu, T.; Zhao, Y. Rolling Bearing Health Indicator Extraction and RUL Prediction Based on Multi-Scale Convolutional Autoencoder. *Appl. Sci.* **2022**, *12*, 5747. [CrossRef]
10. Chen, G.; Liu, M.; Chen, J. Frequency-temporal-logic-based bearing fault diagnosis and fault interpretation using Bayesian optimization with Bayesian neural networks. *Mech. Syst. Signal Processing* **2020**, *145*, 106951. [CrossRef]
11. Xia, M.; Li, T.; Xu, L.; Liu, L.; de Silva, C.W. Fault Diagnosis for Rotating Machinery Using Multiple Sensors and Convolutional Neural Networks. *IEEE-ASME Trans. Mechatron.* **2018**, *23*, 101–110. [CrossRef]
12. Hu, Q.; Qin, A.; Zhang, Q.; He, J.; Sun, G. Fault Diagnosis Based on Weighted Extreme Learning Machine With Wavelet Packet Decomposition and KPCA. *IEEE Sens. J.* **2018**, *18*, 8472–8483. [CrossRef]
13. Lei, Y.; Jia, F.; Lin, J.; Xing, S.; Ding, S.X. An Intelligent Fault Diagnosis Method Using Unsupervised Feature Learning Towards Mechanical Big Data. *IEEE Trans. Ind. Electron.* **2016**, *63*, 3137–3147. [CrossRef]
14. Zhang, X.; Chen, G.; Hao, T.; He, Z. Rolling bearing fault convolutional neural network diagnosis method based on casing signal. *J. Mech. Sci. Technol.* **2020**, *34*, 2307–2316. [CrossRef]
15. Islam, M.; Kim, J. Automated bearing fault diagnosis scheme using 2D representation of wavelet packet transform and deep convolutional neural network. *Comput. Ind.* **2019**, *106*, 142–153. [CrossRef]
16. Cai, B.; Liu, Y.; Xie, M. A Dynamic-Bayesian-Network-Based Fault Diagnosis Methodology Considering Transient and Intermittent Faults. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 276–285. [CrossRef]
17. Tao, C.; Wang, X.; Gao, F.; Wang, M. Fault diagnosis of photovoltaic array based on deep belief network optimized by genetic algorithm. *Chin. J. Electr. Eng.* **2020**, *6*, 106–114. [CrossRef]
18. Liang, N.; Huang, G.; Saratchandran, P.; Sundararajan, N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Trans. Neural Netw.* **2006**, *17*, 1411–1423. [CrossRef]
19. Nesterov, Y. Gradient methods for minimizing composite functions. *Math. Program.* **2013**, *140*, 125–161. [CrossRef]
20. Ai, X.; Sheng, V.; Li, C. A MBGD enhancement method for imbalance smoothing. *Multimed. Tools Appl.* **2022**, *81*, 24225–24243. [CrossRef]
21. Zhang, Y.; Li, P.; Wang, X. Intrusion Detection for IoT Based on Improved Genetic Algorithm and Deep Belief Network. *IEEE Access* **2019**, *7*, 31711–31722. [CrossRef]
22. Paul, R.; Friedman, Y.; Cohen, K. Accelerated Gradient Descent Learning Over Multiple Access Fading Channels. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 532–547. [CrossRef]
23. Li, X.; Wang, W.; Zhu, S.; Xiang, W.; Wu, X. Generalized Nesterov Accelerated Conjugate Gradient Algorithm for a Compressively Sampled MR Imaging Reconstruction. *IEEE Access* **2020**, *8*, 157130–157139. [CrossRef]
24. Qu, G.; Li, N. Accelerated Distributed Nesterov Gradient Descent. *IEEE Trans. Autom. Control* **2020**, *65*, 2566–2581. [CrossRef]
25. Yu, T.; Liu, X.; Dai, Y.; Sun, J. Stochastic Variance Reduced Gradient Methods Using a Trust-Region-Like Scheme. *J. Sci. Comput.* **2021**, *87*, 1–24. [CrossRef]
26. Shang, F.; Zhou, K.; Liu, H.; Cheng, J.; Tsang, I.W.; Zhang, L.; Tao, D.; Jiao, L. VR-SGD: A Simple Stochastic Variance Reduction Method for Machine Learning. *IEEE Trans. Knowl. Data Eng.* **2020**, *32*, 188–202. [CrossRef]
27. Koppel, A.; Zhang, K.; Zhu, H.; Basar, T. Projected Stochastic Primal-Dual Method for Constrained Online Learning With Kernels. *IEEE Trans. Signal Processing* **2019**, *67*, 2528–2542. [CrossRef]
28. Li, H.; Fang, C.; Yin, W.; Lin, Z. Decentralized Accelerated Gradient Methods With Increasing Penalty Parameters. *IEEE Trans. Signal Processing* **2020**, *68*, 4855–4870. [CrossRef]
29. Li, H.; Zheng, L.; Wang, Z.; Yan, Y.; Feng, L.; Guo, J. S-DIGing: A Stochastic Gradient Tracking Algorithm for Distributed Optimization. *IEEE Trans. Emerg. Top. Comput. Intell.* **2022**, *6*, 53–65. [CrossRef]
30. Wang, Z.; Li, H. Edge-Based Stochastic Gradient Algorithm for Distributed Optimization. *IEEE Trans. Netw. Sci. Eng.* **2020**, *7*, 1421–1430. [CrossRef]
31. Liu, H.; Wang, J.; Meng, X. Hierarchical maximum likelihood generalized extended stochastic gradient algorithms for bilinear-in-parameter systems. *Optim. Control Appl. Methods* **2022**, *43*, 402–417. [CrossRef]
32. Traore, C.; Pauwels, E. Sequential convergence of AdaGrad algorithm for smooth convex optimization. *Oper. Res. Lett.* **2021**, *49*, 452–458. [CrossRef]
33. Su, S.; Kek, S. An Improvement of Stochastic Gradient Descent Approach for Mean-Variance Portfolio Optimization Problem. *J. Math.* **2021**, *2021*, 1–10. [CrossRef]
34. Wu, Z.; Li, S.; Chen, C.; Hao, A.; Qin, H. Deeper Look at Image Salient Object Detection: Bi-Stream Network With a Small Training Dataset. *IEEE Trans. Multimed.* **2022**, *24*, 73–86. [CrossRef]
35. Tan, H.; Zhang, D.; Tian, H.; Jiang, D.; Guo, L.; Wang, G.; Lin, Y. Multi-label classification for simultaneous fault diagnosis of marine machinery: A comparative study. *Ocean Eng.* **2021**, *239*, 109723. [CrossRef]
36. Yan, H.; Hu, H.; Jiang, W. A Novel Fault Diagnosis Method for Marine Blower with Vibration Signals. *Pol. Marit. Res.* **2022**, *29*, 77–86. [CrossRef]
37. Xu, J.; Zhao, Z.; Xu, B.; Yang, J.; Chang, L.; Yan, X.; Wang, G. Machine learning-based wear fault diagnosis for marine diesel engine by fusing multiple data-driven models. *Knowl. -Based Syst.* **2020**, *190*, 105324. [CrossRef]

38. Xu, J.; Yan, P.; Sheng, X.; Yuan, C.; Xu, D.; Yang, J. A Belief Rule-Based Expert System for Fault Diagnosis of Marine Diesel Engines. *IEEE Trans. Syst. Man Cybern. -Syst.* **2020**, *50*, 656–672. [CrossRef]
39. Tan, H.; Niu, Y.; Tian, H.; Hou, L.; Zhang, J. A one-class SVM based approach for condition-based maintenance of a naval propulsion plant with limited labeled data. *Ocean Eng.* **2019**, *193*, 106592. [CrossRef]
40. Karatuğ, Ç.; Arslanoğlu, Y. Development of condition-based maintenance strategy for fault diagnosis for ship engine systems. *Ocean Eng.* **2022**, *256*, 111515. [CrossRef]
41. Han, P.; Ellefsen, A.; Li, G.; Holmeset, F.T.; Zhang, H. Fault Detection with LSTM-Based Variational Autoencoder for Maritime Components. *IEEE Sens. J.* **2021**, *21*, 21903–21912. [CrossRef]
42. Velasco-Gallego, C.; Lazakis, I. RADIS: A real-time anomaly detection intelligent system for fault diagnosis of marine machinery. *Expert Syst. Appl.* **2022**, *204*, 117634. [CrossRef]
43. Velasco-Gallego, C.; Lazakis, I. Development of a time series imaging approach for fault classification of marine systems. *Ocean Eng.* **2022**, *263*, 112297. [CrossRef]
44. Zhang, R.; Zhang, Y.; Tan, X.; Liu, X.; Li, Q. Fault Diagnosis for Marine Main Engines using Improved Semi-Supervised Locally Linear Embedding. *J. Syst. Simul.* **2021**, *33*, 7383–7388. [CrossRef]
45. Loparo, K. Case Western Reserve University Bearing Data Centre Website. 2012. Available online: https://engineering.case.edu/bearingdatacenter (accessed on 14 September 2022).
46. Wang, B.; Lei, Y.; Li, N.; Li, N. A Hybrid Prognostics Approach for Estimating Remaining Useful Life of Rolling Element Bearings. *IEEE Trans. Reliab.* **2020**, *69*, 401–412. [CrossRef]