Mechanical Systems and Signal Processing 110 (2018) 349–367

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

Deep normalized convolutional neural network for imbalanced fault classification of machinery and its understanding via visualization

Feng Jia, Yaguo Lei *, Na Lu, Saibo Xing

State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China

ARTICLE INFO

Article history:
Received 28 July 2017
Received in revised form 9 March 2018
Accepted 12 March 2018
Available online 30 March 2018

Keywords:
Deep learning
Convolutional neural network
Imbalanced classification
Visualization
Intelligent fault diagnosis

ABSTRACT

Deep learning has attracted attentions in intelligent fault diagnosis of machinery because it allows a deep network to accomplish the tasks of feature learning and fault classification automatically. Among deep learning models, convolutional neural networks (CNNs) are able to learn features from mechanical vibration signals and thus several studies have applied CNNs in intelligent fault diagnosis of machinery. However, these studies suffer from the following weaknesses. (1) The imbalanced distribution of machinery health conditions is not considered. (2) What CNNs have learned is not clear. Therefore, in this paper, a framework called deep normalized convolutional neural network (DNCNN) is proposed for imbalanced fault classification of machinery to overcome the first weakness. Meanwhile, neuron activation maximization (NAM) algorithm is developed to handle the second weakness. To verify the proposed methods, three bearing datasets containing single faults and compound faults are constructed with different imbalanced degrees. The classification accuracies of the three datasets demonstrate that DNCNN is able to deal with the imbalanced classification problem more effectively than the commonly used CNNs. By analyzing the kernels of the convolutional layers of DNCNN via NAM algorithm, we find that these kernels act as filters and they become complex when the layers go deeper. This result may help us understand what DNCNN has learned in intelligent fault diagnosis of machinery.

1. Introduction

In modern industries, machinery becomes more automatic and sophisticated than ever before. Incipient faults in any location of the machinery could produce chain reaction and lead to its damage [1–3]. Aiming to inspect the health conditions of the machinery comprehensively, massive signals are acquired by the mounted sensors after the long-time collection [4]. Since intelligent fault diagnosis methods are able to process these massive signals and recognize the health conditions of the machinery automatically, lots of efforts have been made to study these methods. Asr et al. [5] designed a feature extraction method using empirical mode decomposition and fed the extracted features into non-naive Bayesian classifier for intelligent fault diagnosis of rotating machinery. A symbolic aggregate approximation framework was proposed by Georgoulas et al. [6] to extract features from bearing signals and the nearest neighbor classifier was employed to classify the faults. Xiong et al.

* Corresponding author.
E-mail address: yaguolei@mail.xjtu.edu.cn (Y. Lei).

https://doi.org/10.1016/j.ymssp.2018.03.025
0888-3270/

[7] considered that the vibration signals of bearings present multifractal properties, and thus they applied multifractal detrended fluctuation analysis to extract multifractal features for intelligent fault diagnosis of bearings. Based on the dynamic characteristics of gearbox signals, Li et al. [8] applied symbolic dynamic entropy features to extract features of the gearbox signals and applied support vector machine to recognize the health conditions. Considering that the amplitudes of the carrier rotating frequency and the difference spectrum would change when a fault occurs in gearboxes, Lei et al. [9] designed two features for the gearboxes specifically and fed these features into relevance vector machine to recognize the health conditions of the gearboxes.

Through literature review, it can be found that the prior studies of intelligent fault diagnosis have two main steps: feature extraction and fault classification. In the step of feature extraction, the researchers should first analyze the signals collected from machinery and understand the properties of the signals, and then design the suitable features according to the specific diagnosis issue. Such feature designing processes make full use of human knowledge in signal processing techniques and diagnostic expertise, but consume much human labor. In order to change this situation, it would be desirable to use advanced artificial intelligence techniques to accomplish the tasks of feature learning and fault classification automatically. Therefore, deep learning is introduced into the intelligent fault diagnosis of machinery [10–12].

Among the deep learning models [13,14], convolutional neural networks (CNNs) are suitable to learn features from mechanical vibration signals because of their ability in handling the periodic signals. Therefore, the researchers have applied CNNs in intelligent fault diagnosis of machinery. For instance, Janssens [15] applied a CNN to learn features from the spectra of bearings and classify their health conditions. Based on multi-source data, Lee et al. [16] used CNNs to diagnose the faults in semiconductor manufacturing processes. Guo et al. [17] proposed a CNN based method for intelligent fault diagnosis of bearings and showed its advantages compared with the diagnosis methods using manual features. Wang et al. [18] presented an adaptive deep convolutional neural network for feature learning and fault classification of bearings. Ding et al. [19] first transformed the vibration signals into wavelet packet image and then used a CNN to recognize the health conditions of bearings. Through the analysis of these studies, it can be found that the intelligent fault diagnosis methods based on CNNs have two weaknesses as follows.

(1) The imbalanced distribution of machinery health conditions is not considered. In real cases, the machinery works under normal condition in most of its operating phases, and the faults seldom happen during the operation [20–22]. Consequently, the data samples of the machinery faults are more difficult to collect than the data samples of the normal condition. Thus, the data samples of different machinery health conditions follow a long tail distribution, i.e. the data samples of the normal condition are abundant while the data samples of the faults are relatively scarce. The imbalanced distribution of the data samples forces CNNs to be biased towards the majority health conditions [23]. As a result, the characteristics of the minority health conditions are learned inadequately, leading to their misclassification.
(2) What CNNs have learned is not clear. The reason why deep learning models are applied in intelligent fault diagnosis is that they can accomplish the feature learning and fault classification of machinery simultaneously, which releases us from the tough work of manually designing feature extraction algorithms. However, deep learning models are always treated as a "black box" in the field of intelligent fault diagnosis and few papers attempt to discover and analyze the patterns in these models. CNNs also suffer from the same dilemma. Although intelligent fault diagnosis methods based on CNNs have achieved good results, it is not easy to understand how CNNs learn features automatically. Intuitively, one way to understand the feature learning process is to analyze the weight matrices of a neural network quantitatively. The reported results may afford us some inspirations. In our previous work [24], we tried to explore the interpretation of a shallow neural network in mechanical feature learning, and the results indicated that the weight matrix of the network is viewed as Gabor-like filters. Following this work, we can directly visualize the kernels (The weight matrix of a convolutional layer in CNNs is called kernels) of the first layer of a CNN since the kernels are connected to the input signals. But it is hard to visualize the kernels in a deeper layer because of the indirect effect on the inputs. Thus, how to solve this problem needs to be further studied.

To overcome the first weaknesses, this paper proposes a framework called deep normalized convolutional neural network (DNCNN) for imbalanced fault classification of machinery. In DNCNN, firstly, normalized layers based on Rectified linear units (ReLU) [25] and weight normalization strategy [26] are used for the effective training of DNCNN. Secondly, weighted softmax loss is developed to deal with the balanced and imbalanced fault classification of machinery adaptively. The proposed DNCNN is validated by three bearing datasets with different imbalanced degrees. By comparing with the commonly used CNNs, the superiority of DNCNN is verified in imbalanced fault classification of machinery.

To overcome the second weakness, this paper proposes a neuron activation maximization (NAM) algorithm. By using the NAM algorithm, we can visualize the kernels of the convolutional layers of DNCNN to understand its feature learning process. The visualization results show that the kernels in the first convolutional layer are the filters with single peak characteristics and the kernels in the second convolutional layer are the filters with more complex properties. Therefore, DNCNN attempts to learn filters in the convolutional layers automatically. With the help of these filters, DNCNN is able to retain the important components of the signals and suppress the useless aspects for classification. This result may help us understand how DNCNN learns the features from vibration signals.

The rest of this paper is organized as follows. Section 2 briefly introduces CNNs and the imbalanced classification problem. Section 3 is dedicated to describing the proposed DNCNN and NAM algorithm. In Section 4, the effectiveness of the proposed methods is validated using three bearing datasets, one of which is a balanced dataset and the others are moderately and highly imbalanced datasets. Conclusions are drawn in Section 5.

2. CNNS and the imbalanced classification problem

2.1. Introduction to CNNs

CNNs are a kind of feed-forward neural networks, where the connectivity between its layers is inspired by the animal visual cortex [27,28]. CNNs mainly have three kinds of layers: the convolutional layer, the pooling layer and the fully connected layer. These layers are used to accomplish the tasks of feature learning and classification. In this paper, the introduced CNNs are the 1-D CNNs because the inputs of the CNNs are 1-D vibration signals.

(1) Convolutional layer. The parameters of a convolutional layer are a set of kernels. During the forward pass, each kernel is convolved across the input vector of the convolutional layer, producing a feature vector. Concretely, the convolution process of the convolutional layer l is shown in Fig. 1(a). The convolutional layer l uses a set of kernels $\boldsymbol{k}^l \in \Re^{J\times D\times H}$ to learn feature vectors, where J indicates the number of the kernels and $D \times H$ is the depth and height of a kernel. The jth output feature vector $\boldsymbol{x}_j^l$ is obtained by

$$\boldsymbol{x}_j^l = \sigma(\boldsymbol{u}_j^l) \tag{1}$$

$$\boldsymbol{u}_j^l = \boldsymbol{k}_j^l * \boldsymbol{x}^{l-1} + \boldsymbol{b}_j^l = \sum_d \boldsymbol{k}_{j,d}^l * \boldsymbol{x}_d^{l-1} + \boldsymbol{b}_j^l \tag{2}$$

where $\boldsymbol{u}_j^l$ is the linear activation, $\sigma(\cdot)$ is the activation function, $\boldsymbol{x}_d^{l-1}$ is the dth feature vector in the previous layer $l-1$, $d = 1,2,\ldots,D$, $j = 1,2,\ldots,J$, $\boldsymbol{k}_{j,d}^l$ is an H-dimension vector, and $\boldsymbol{b}_j^l$ is the bias vector.

(2) Pooling layer. The pooling layer is used to conduct a form of down-sampling. In this paper, the form of down-sampling is max pooling [29] and the stride size of the pooling layer equals the pooling size. The max pooling serves to reduce the sizes of the feature vectors and the parameters of CNNs, decrease training time and memory requirements, and control the overfitting. In the pooling layer l, the max pooling is conducted by

$$\boldsymbol{x}_j^l = \mathrm{down}(\boldsymbol{x}_j^{l-1}, s) \tag{3}$$

where down$(\cdot)$ represents the down-sampling function of max pooling, $\boldsymbol{x}_j^l$ is the output feature vector of the pooling layer, $\boldsymbol{x}_j^{l-1}$ is the feature vector in the previous layer $l-1$, and s is the pooling size. For better understanding, Fig. 1(b) shows a max pooling process when s equals 2.

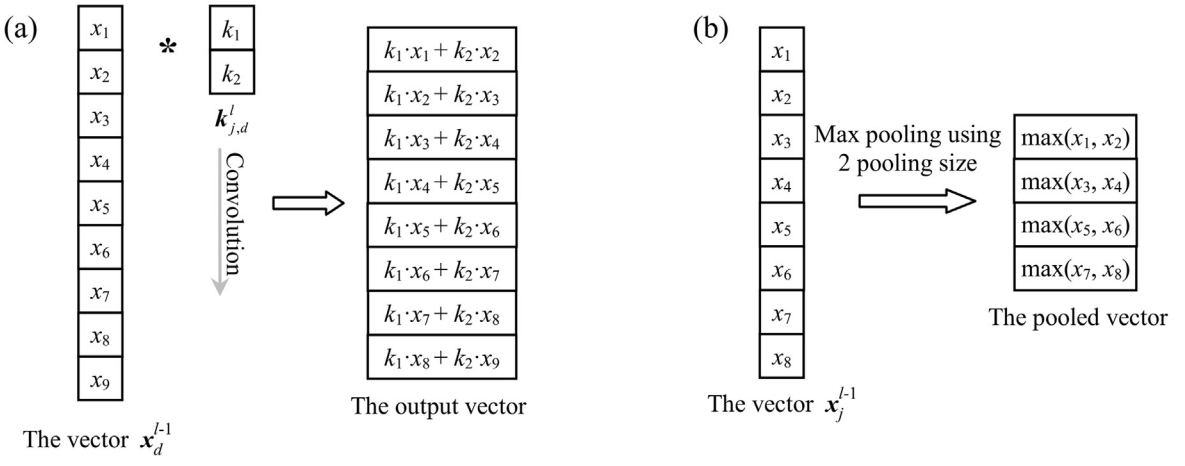

Fig. 1. (a) Illustration of the convolution process in the convolutional layer, and (b) illustration of the pooling process in the pooling layer.

(3) Fully connected layer. After the pooling layer, the learned feature vectors are flattened into one vector and this vector is used as the input of the fully connected layers. The output $\boldsymbol{x}^l$ of the lth fully connected layer is obtained by

$$\boldsymbol{x}^l = \sigma(\boldsymbol{u}^l) \tag{4}$$

$$\boldsymbol{u}^l = (\boldsymbol{w}^l)^{\mathrm{T}} \boldsymbol{x}^{l-1} + \boldsymbol{b}^l \tag{5}$$

where $\boldsymbol{u}^l$ is the linear activation, $\boldsymbol{x}^{l-1}$ is the output vector of the previous layer $l-1$, $\boldsymbol{w}^l \in \Re^{M\times N}$ is the weight matrix of the fully connected layer, M is the dimension of $\boldsymbol{x}^{l-1}$, N is the dimension of $\boldsymbol{x}^l$, and $\boldsymbol{b}^l \in \Re^N$ is the bias vector.

2.2. Imbalanced classification problem of CNNs

In the imbalanced classification, the samples of majority health conditions are abundant and would be over-represented. The samples of minority health conditions are scarce and would be under-represented. Thus, the trained classifier will achieve poor classification accuracies for these scarce samples. CNNs cannot deal with this problem appropriately because of their loss function. For multi-class classification, the loss function used in CNNs is the softmax loss [30] and it can be described as follows. Given a training set $\{\boldsymbol{x}^{(q)}, \boldsymbol{y}^{(q)}\}_{q=1}^{Q}$, where $\boldsymbol{x}^{(q)}$ is the qth sample, Q is the number of samples, $\boldsymbol{y}^{(q)}$ is the health condition and $\boldsymbol{y}^{(q)} \in \{1, 2, \ldots, C\}$. The prediction $p_c^{(q)}$ of $\boldsymbol{x}^{(q)}$ belonging to the cth health condition can be calculated as

$$p_c^{(q)} = \exp\left((\boldsymbol{w}_c^L)^{\mathrm{T}} \boldsymbol{x}^{(q),L-1}\right) \Big/ \sum_{c=1}^{C} \exp\left((\boldsymbol{w}_c^L)^{\mathrm{T}} \boldsymbol{x}^{(q),L-1}\right) \tag{6}$$

where L indicates the maximum of l, namely the last layer of the CNN, and $\boldsymbol{x}^{(q),L-1}$ is the output of the layer $L-1$.

Based on Eq. (6), predictions are turned into a probability distribution over the health conditions. Such probabilistic predictions $p_c^{(q)}$ are used to compute the softmax loss

$$\ell = -\frac{1}{Q}\sum_{q=1}^{Q}\sum_{c=1}^{C} 1\{\boldsymbol{y}^{(q)} = c\} \log(p_c^{(q)}) \tag{7}$$

where $1\{\cdot\}$ denotes the indicator function returning 1 if the condition is true, and 0 otherwise. In Eq. (7), the softmax loss tries to minimize the overall classification errors during the training processes and it assumes that the misclassification errors of different health conditions share the equivalent importance. As a result, the errors of the majority health conditions affect the training processes of CNNs dominantly and the errors of the minority health conditions are neglected to some extent, which makes CNNs tend to favor the majority health conditions and misclassify the minority health conditions.

3. DNCNN for imbalanced fault classification of machinery

In this section, we first describe the proposed DNCNN for imbalanced fault classification of machinery, where normalized layers help to improve the training of DNCNN and the weighted softmax loss helps to deal with imbalanced classification problem. Then, the NAM algorithm is developed to analyze the properties of the kernels in DNCNN.

3.1. The proposed DNCNN

3.1.1. Normalized layers

The normalized layers are constructed based on ReLU [25] and weight normalization strategy [26], where ReLU is applied to avoid the gradient vanishing problem and weight normalization strategy is applied to improve the optimizability of the parameters of DNCNN.

In intelligent fault diagnosis of machinery, many studies still use sigmoid function as the activation function of CNNs [15–19]. However, one shortcoming of sigmoid function is the gradient vanishing problem [31], which may make the training of CNNs slow to converge. ReLU is a new nonlinear activation function [25] and it is described as

$$\sigma_{\mathrm{r}}(z) = \begin{cases} 0, & \text{if } z \leqslant 0 \\ z, & \text{if } z > 0 \end{cases} \tag{8}$$

Since the derivative value of ReLU is constant as 1 or 0, it is able to avoid the gradient vanishing problem and encourage sparse activations. Moreover, in the training processes of a CNN, the practical success of back propagation (BP) algorithm depends on the curvature of the optimized objective. If the condition number of the Hessian matrix is low at the optimum, pathological curvature will occur in the training process [26]. We applied weight normalization strategy to overcome this problem, which improves the convergence of the optimization process of DNCNN. In DNCNN, normalized layers contain the normalized convolutional layer and the normalized fully connected layer, and they are described as below.

For convenience, we use C, P and F to represent the convolutional layer, the pooling layer and the fully connected layer, respectively. Based on Eqs. (1) and (2), the original convolutional layer can be concisely rewritten as

$$\boldsymbol{x}_j^{\mathrm{C},l} = \sigma_{\mathrm{r}}\left(\boldsymbol{k}_j^{\mathrm{C},l} * \boldsymbol{x}^{\mathrm{P},l-1} + \boldsymbol{b}_j^{\mathrm{C},l}\right) \tag{9}$$

where $\boldsymbol{k}_j^{\mathrm{C},l}$ is the kernel, C represents the convolutional layer, P represents the pooling layer, and l indicates the layer number.

We normalize the kernels of the original convolutional layer and get the normalized convolutional layer

$$\boldsymbol{x}_j^{\mathrm{C},l} = \sigma_{\mathrm{r}}\left(\gamma_j^{\mathrm{C},l}\frac{\boldsymbol{k}_j^{\mathrm{C},l} * \boldsymbol{x}^{\mathrm{P},l-1}}{\left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}} + \boldsymbol{b}_j^{\mathrm{C},l}\right) \tag{10}$$

where $\gamma_j^{\mathrm{C},l}$ is the gradient-based learnable scaling parameter and $\|\cdot\|_{\mathrm{F}}$ is Frobenius norm. The optimization of $\gamma_j^{\mathrm{C},l}\frac{\boldsymbol{k}_j^{\mathrm{C},l} * \boldsymbol{x}^{\mathrm{P},l-1}}{\left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}}$ in Eq. (10) can be divided into two parts. The first part is the optimization of $\gamma_j^{\mathrm{C},l}$ and the second part is the optimization of $\frac{\boldsymbol{k}_j^{\mathrm{C},l}}{\left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}}$. The first part is beneficial for the smooth optimization of the normalized convolutional layer, and thus $\gamma_j^{\mathrm{C},l}$ is a learnable parameter to restore the representation power of the network [32]. The second part is beneficial for the normalization of the kernel $\boldsymbol{k}_j^{\mathrm{C},l}$ [26].

To obtain the gradients of $\gamma_j^{\mathrm{C},l}$ and $\boldsymbol{k}_j^{\mathrm{C},l}$, we compute the sensitivities at the normalized convolutional layer l. Let ℓ denotes the loss function of the network, and the sensitivities can be described as

$$\boldsymbol{\delta}_j^{\mathrm{C},l} = \frac{\partial \ell}{\partial \boldsymbol{u}_j^{\mathrm{C},l}} = \sigma_{\mathrm{r}}'(\boldsymbol{u}_j^{\mathrm{C},l}) \circ \mathrm{up}(\boldsymbol{\delta}_j^{\mathrm{P},l+1}) \tag{11}$$

where $\boldsymbol{u}_j^{\mathrm{C},l} = \gamma_j^{\mathrm{C},l}\frac{\boldsymbol{k}_j^{\mathrm{C},l} * \boldsymbol{x}^{\mathrm{P},l-1}}{\left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}} + \boldsymbol{b}_j^{\mathrm{C},l}$, $\circ$ denotes element-wise multiplication, $\boldsymbol{\delta}_j^{\mathrm{P},l+1}$ is the sensitivities in the pooling layer $l+1$, $\mathrm{up}(\cdot)$ denotes an up-sampling operation, and σ_{r}' is the derivative function of σ_{r}.

Thus, the gradients of $\gamma_j^{\mathrm{C},l}$ and $\boldsymbol{k}_j^{\mathrm{C},l}$ in Eq. (10) can be obtained by

$$\frac{\partial \ell}{\partial \gamma_j^{\mathrm{C},l}} = \sum_{d,h}\left(\sum_u (\boldsymbol{\delta}_j^{\mathrm{C},l})_u (\boldsymbol{p}_i^{\mathrm{P},l-1})_u \circ \boldsymbol{k}_{j,d}^{\mathrm{C},l} / \left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}\right) \tag{12}$$

$$\frac{\partial \ell}{\partial \boldsymbol{k}_{j,d}^{\mathrm{C},l}} = \frac{\gamma_j^{\mathrm{C},l}}{\left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}}\sum_u (\boldsymbol{\delta}_j^{\mathrm{C},l})_u (\boldsymbol{p}_d^{\mathrm{P},l-1})_u - \frac{\gamma_j^{\mathrm{C},l}}{\left\|\boldsymbol{k}_j^{\mathrm{C},l}\right\|_{\mathrm{F}}^2}\frac{\partial \ell}{\partial \gamma_j^{\mathrm{C},l}}\boldsymbol{k}_{j,d}^{\mathrm{C},l} \tag{13}$$

where $(\boldsymbol{p}_d^{\mathrm{P},l-1})_u$ is the segment in $\boldsymbol{x}_d^{\mathrm{P},l-1}$ that is multiplied element-wise by $\boldsymbol{k}_{j,d}^{\mathrm{C},l}$ during convolution in order to compute the element at u in $\boldsymbol{x}_d^{\mathrm{C},l}$. The gradient of $\boldsymbol{b}_j^{\mathrm{C},l}$ in the normalized convolutional layer is the same as that in the original layer, and it is given in Ref. [33].

Similarly, the normalized fully connected layer can be denoted as

$$\boldsymbol{x}_n^{\mathrm{F},l} = \sigma_{\mathrm{r}}(\boldsymbol{u}_n^{\mathrm{F},l}) = \sigma_{\mathrm{r}}\left(\gamma_n^{\mathrm{F},l}\frac{(\boldsymbol{w}_n^{\mathrm{F},l})^{\mathrm{T}}\boldsymbol{x}^{\mathrm{F},l-1}}{\left\|\boldsymbol{w}_n^{\mathrm{F},l}\right\|_2} + \boldsymbol{b}_n^{\mathrm{F},l}\right) \tag{14}$$

where $\boldsymbol{w}_n^{\mathrm{F},l} \in \Re^M$, F represents the fully connected layer, and $n = 1, 2, \ldots, N$.

The gradients of $\gamma_n^{\mathrm{F},l}$ and $\boldsymbol{w}_n^{\mathrm{F},l}$ are given as

$$\frac{\partial \ell}{\partial \gamma_n^{\mathrm{F},l}} = \sum_{m=1}^{M}\left(\boldsymbol{x}_n^{\mathrm{F},l-1}(\boldsymbol{\delta}_n^{\mathrm{F},l})^{\mathrm{T}} \circ \boldsymbol{w}_n^{\mathrm{F},l} / \left\|\boldsymbol{w}_n^{\mathrm{F},l}\right\|_2\right) \tag{15}$$

$$\frac{\partial \ell}{\partial \boldsymbol{w}_n^{\mathrm{F},l}} = \frac{\gamma_n^{\mathrm{F},l}}{\left\|\boldsymbol{w}_n^{\mathrm{F},l}\right\|_2}\boldsymbol{x}_n^{\mathrm{F},l-1}(\boldsymbol{\delta}_n^{\mathrm{F},l})^{\mathrm{T}} - \frac{\gamma_n^{\mathrm{F},l}}{\left\|\boldsymbol{w}_n^{\mathrm{F},l}\right\|_2^2}\frac{\partial \ell}{\partial \gamma_n^{\mathrm{F},l}}\boldsymbol{w}_n^{\mathrm{F},l} \tag{16}$$

where the sensitivity $\boldsymbol{\delta}_n^{\mathrm{F},l}$ is $(\boldsymbol{w}_n^{\mathrm{F},l+1})^{\mathrm{T}}\boldsymbol{\delta}^{\mathrm{F},l+1} \circ \sigma_{\mathrm{r}}'(\boldsymbol{u}_n^{\mathrm{F},l})$.

Based on these gradients, the parameters of the normalized layers can be well updated using the back propagation algorithm.

3.1.2. Weighted softmax loss for imbalanced classification problem

In Section 2.2, we have analyzed the imbalanced classification problem of CNNs. We can handle this problem by a simple idea: The losses of the minority health conditions can be weighted based on the imbalanced degree of the dataset in order to highlight their effects on the training processes of CNNs. Thus, we develop a class weight strategy to improve the softmax loss, namely the weighted softmax loss.

Given a training set $\{\boldsymbol{x}^{(q)}, \boldsymbol{y}^{(q)}\}_{q=1}^{Q}$, where $\boldsymbol{x}^{(q)}$ is the qth data sample and $\boldsymbol{y}^{(q)} \in \{1, 2, \ldots, C\}$ is the target class representing a health condition of machinery. We count the sample numbers of each health condition, and the number n_c of the cth health condition can be calculated as

$$n_c = \sum_{q=1}^{Q} 1\{\boldsymbol{y}^{(q)} = c\}. \tag{17}$$

The set of n_c measures the imbalanced degree for each health condition and indicates the imbalanced distribution of the dataset. In our strategy, the loss of the minority health condition should be given a big class weight based on the imbalanced degree and the loss of the majority health condition should not be affected. Thus, the class weight for each health condition can be calculated as

$$v_c = \frac{\max\{n_c\}_{c=1}^{C}}{n_c} \tag{18}$$

where $\max\{\cdot\}$ represents the maximum value of the set. In Eq. (18), the class weight v_c is adaptively calculated following the imbalanced distribution of the dataset, and it equals 1 for each health condition when the dataset is balanced.

Using the class weights, the weighted softmax loss is denoted as

$$\ell_{\text{wsl}} = -\frac{1}{Q}\sum_{q=1}^{Q}\sum_{c=1}^{C} v_c 1\{\boldsymbol{y}^{(q)} = c\} \log(p_c^{(q)}) \tag{19}$$

where $p_c^{(q)} = \exp\left((\boldsymbol{w}_c^{\text{F},L})^{\text{T}} \boldsymbol{x}^{(q),L-1}\right) / \sum_{c=1}^{C} \exp\left((\boldsymbol{w}_c^{\text{F},L})^{\text{T}} \boldsymbol{x}^{(q),L-1}\right)$.

The gradients of the weighted softmax loss with respective to the parameter $\boldsymbol{w}_c^{\text{F},L}$ can be calculated by

$$\frac{\partial \ell_{\text{wsl}}}{\partial \boldsymbol{w}_c^{\text{F},L}} = -\frac{1}{Q}\sum_{q=1}^{Q}[\boldsymbol{x}^{(q),L-1}(v_c 1\{\boldsymbol{y}^{(q)} = c\} - p_c^{(q)})] \tag{20}$$

The parameters of the weighted softmax loss can be updated using Eq. (20). Since the weighted softmax loss is extended by the original softmax loss, it can be used for both balanced and imbalanced classification.

3.1.3. DNCNN construction

The architecture of DNCNN is illustrated in Fig. 2. DNCNN consists of the normalized convolutional layers, the pooling layers and the normalized fully connected layers. For convenience, we still use C, P and F to represent the normalized convolutional layer, the pooling layer and the normalized fully connected layer, respectively. It is noted that the kernel size of a convolutional layer is represented by $J \times D \times H$, where J indicates the number of the kernels, D indicates the depth of a kernel

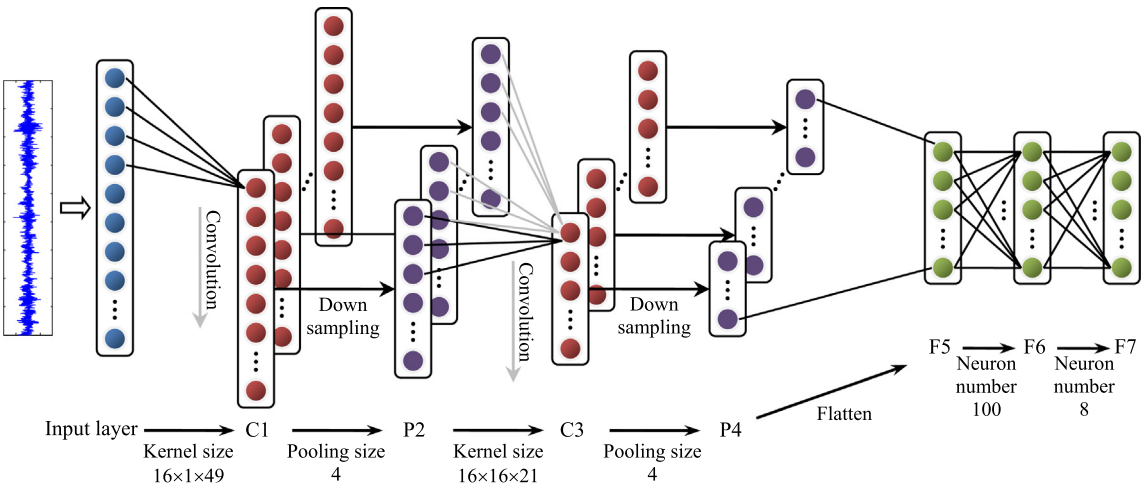

Fig. 2. The architecture of DNCNN.

and H indicates the height of a kernel. When given vibration signals, DNCNN first uses the normalized convolutional layer C1 to learn features. Then the pooling layer P2 is applied to down sample the features in C1. By repeating such processes, C3 and P4 are employed to learn the high-level features. Next, the features in P4 are flattened into a vector and it is input into the layer F5. Finally, F5, F6 and F7 are used to recognize the health conditions of machinery. The details of the parameters of DNCNN for one sample are summarized in Table 1.

3.2. Neuron activation maximization algorithm

As we know, deep learning models including CNNs are always treated as a black box and our understanding of how these networks work has lagged behind. In this paper, we attempt to find a way to understand the feature learning process of CNNs. As discussed in the introduction, only the kernels in the first convolutional layer of a CNN can be directly visualized because of the direct connection to the input signals. How to discover the patterns of the kernels of a deep convolutional layer becomes a problem.

The activity of our brains to a heard story may give us some enlightenment. Our sensory cortex will light up if the story is about the delicious food and our motor cortex will get active if the story is about motion [34]. In DNCNN, a kernel is viewed as a neuron that processes the information. We can find an input signal to maximize the activation of the neuron. Since the neuron is excited as much as possible, the signal and the neuron are closely related. By analyzing the pattern of the signal, we can know what the neuron detects. Now, the problem of discovering the patterns of the kernels can be converted into an optimization problem of finding a signal that maximizes their activations. Thus, we develop the NAM algorithm. After the training process of DNCNN, we can calculate the activation of the learned $\boldsymbol{k}_j^{C,l}$ as follows.

$$\boldsymbol{x}_j^{C,l} = f_{\boldsymbol{k}_j^{C,l}}(\boldsymbol{x}) \tag{21}$$

where $f_{\boldsymbol{w}_j^{C,l}}(\cdot)$ is the trained function of DNCNN between $\boldsymbol{k}_j^{C,l}$ and an input signal $\boldsymbol{x}$. We use the L1 norm of the activation, namely A_j, as the activation index to measure the activation degree of $\boldsymbol{k}_j^{C,l}$.

$$A_j = \left\|\boldsymbol{x}_j^{C,l}\right\|_1 = \left\|f_{\boldsymbol{k}_j^{C,l}}(\boldsymbol{x})\right\|_1 \tag{22}$$

NAM algorithm aims to find a suitable $\boldsymbol{x}$ maximizing A_j. Thus, the optimization problem can be defined as

$$\arg\max_{\boldsymbol{x}} \left\|f_{\boldsymbol{k}_j^{C,l}}(\boldsymbol{x})\right\|_1 \tag{23}$$

To use the gradient descent methods, we convert the optimization problem in Eq. (23) into a minimization problem. In addition, L2 norm regular is introduced in the minimization problem to penalize the large values and avoid the overfitting phenomenon. The alternative optimization problem can be rewritten as

$$\arg\min_{\boldsymbol{x}} -\left\|f_{\boldsymbol{k}_j^{C,l}}(\boldsymbol{x})\right\|_1 + \beta\|\boldsymbol{x}\|_2 \tag{24}$$

where β is the regular parameter controlling the weight between the activation degree and the penalty term.

To solve the problem in Eq. (24), L-BFGS is utilized. L-BFGS is an adaptive gradient descent method and free of learning rate selection. Based on this algorithm, the update process of $\boldsymbol{x}$ can be described as

$$\boldsymbol{x}^{(\tau+1)} = \boldsymbol{x}^{(\tau)} - \eta^{(\tau)}\boldsymbol{H}^{(\tau)}\nabla\ell^{(\tau)} \tag{25}$$

where τ means the τth interval of the update process, $\boldsymbol{H}$ is the inverse of Hessian matrix, $\nabla\ell$ equals $\partial\left(-\left\|f_{\boldsymbol{k}_j^{C,l}}(\boldsymbol{x})\right\|_1 + \beta\|\boldsymbol{x}\|_2\right)\Big/\partial\boldsymbol{x}$, and η is the step size. The computation of $\boldsymbol{H}^{(\tau)}$ is detailed in Ref. [35], and η can be automatically determined by

Table 1
The details of the architecture of DNCNN.

Layer	Parameter name	Parameter size	Output size
Input layer	/	/	1200×1
C1	Kernels	$16 \times 1 \times 49$	$16 \times 1152 \times 1$
P2	Max pooling size	4	$16 \times 288 \times 1$
C3	Kernels	$16 \times 16 \times 21$	$16 \times 268 \times 1$
P4	Max pooling size	4	$16 \times 67 \times 1$
F5	/	/	1072×1
F6	Weight matrix	1072×100	100×1
F7	Weight matrix	100×8	8×1

$$\eta^{(\tau)} = \arg\min_{\eta>0} J(\boldsymbol{x}^{(\tau)} - \eta \boldsymbol{H}^{(\tau)} \nabla \ell^{(\tau)}) \tag{26}$$

After the optimization, we can analyze the signal $\boldsymbol{x}$ and study the properties of $\boldsymbol{k}_j^{C,l}$.

4. Experimental verification

We use the bearing datasets involving both single faults and compound faults to verify the performance of DNCNN. The vibration signals of the bearings were collected from a test bench with a sampling frequency of 12.8 kHz. The test bench [36] consisted of two supporting pillow blocks, a hydraulic motor, a hydraulic cylinder, and a hydraulic radial load application system. The bearings were installed in the hydraulic motor-driven mechanical system and the loads were added by the hydraulic cylinder. Accelerometers were mounted to acquire the vibration signals. Eight health conditions of the bearings were simulated: normal condition (N), faults occurred on the roller (RF), faults occurred on the inner race (IF), faults occurred on the outer race (OF), compound faults occurred on the outer race and the roller (ORF), compound faults occurred on the outer race and the inner race (OIF), compound faults occurred on the inner race and the roller (IRF), and compound faults occurred on the outer race, the roller and the inner race (ORIF). There are 1092 data samples for each health condition, and each data sample contains 1200 data points. Fig. 3 gives the data samples for each health condition of bearings.

As shown in Table 2, three datasets (Dataset A, B and C) are constituted with different imbalanced data degrees. In Dataset A, 50% of samples of each bearing health condition are used for training, and the rest samples are used for testing. Since the numbers of training samples are same for each health condition, Dataset A is a balanced dataset. In real cases, the fault samples are harder to collect than the normal samples, and the compound fault samples are harder to collect than the single fault samples. Thus, we reduce the training samples of the single and compound fault samples from Dataset A and constitute the Dataset B to simulate the imbalanced classification. In Dataset B, the percents of training samples of single faults are 30%, and the percents of training samples of compound faults are 20% or 10%. In order to facilitate the comparisons, the percents of testing samples are still 50%. In Dataset C, more percents of training samples for different health conditions are reduced, as shown in Table 2. Dataset B is considered as a moderately imbalanced dataset and Dataset C is considered as a highly imbalanced dataset.

4.1. Diagnosis results of DNCNN for balanced data classification

In this section, we focus on the performance of the proposed DNCNN for balanced data classification, and thus Dataset A is considered. For comparisons, the CNN using sigmoid function (S-CNN) and the CNN using ReLU (R-CNN) are used for classification, and they share the same architecture as DNCNN. The learning rates of the three methods are 0.01. The training epochs of S-CNN, R-CNN and DNCNN are 1500, 150 and 150, respectively.

In the training processes, we record the training errors and the testing errors in each epoch. Fig. 4 shows the losses and the gradients of the three methods. In Fig. 4(b), the distributions of the gradients of S-CNN shrink gradually from the layer F7 to the layer C1 because of the gradient vanishing problem caused by the sigmoid function. The gradients of the layer C1 are so small that the kernels of this layer cannot update well, making the training errors of S-CNN decrease slowly in Fig. 4(a). After 600 epochs, overfitting occurs in the training process of S-CNN. Because of using ReLU, the gradient vanishing problem does not exist in the training processes of R-CNN and DNCNN in Fig. 4(d) and (f). Thus, the losses of R-CNN and DNCNN converge much quickly. As shown in Fig. 4(c) and (e), with the help of weight normalization strategy, the training errors and

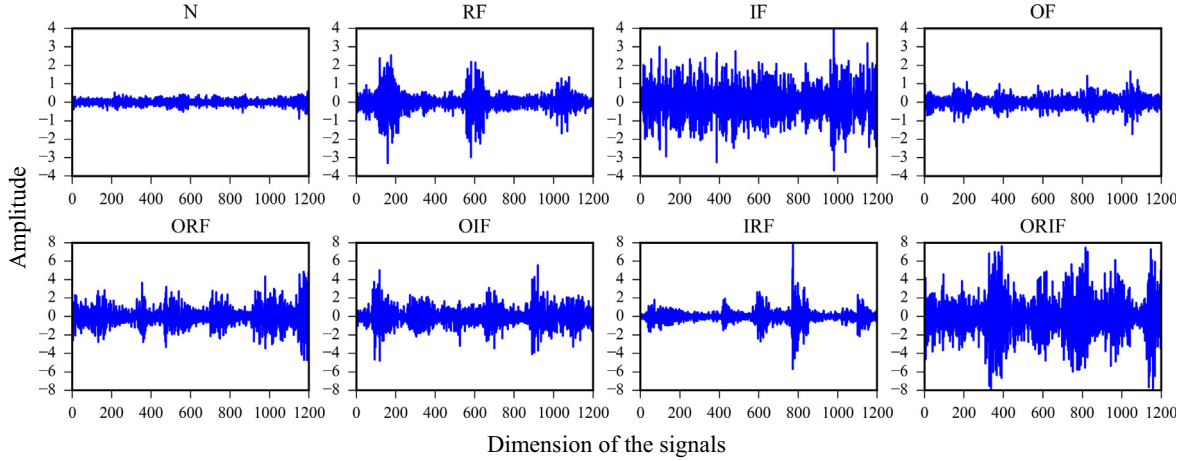

Fig. 3. Typical vibration signals for each health condition of bearings.

Table 2
The description of the datasets.

Health conditions	The percent of training samples			The percent of testing samples
	Dataset A	Dataset B	Dataset C	Dataset A/B/C
N	50%	50%	50%	50%
RF	50%	30%	20%	50%
IF	50%	30%	20%	50%
OF	50%	30%	20%	50%
ORF	50%	20%	5%	50%
OIF	50%	20%	5%	50%
IRF	50%	20%	5%	50%
ORIF	50%	10%	2%	50%

Notes: Dataset A is balanced, Dataset B is moderately imbalanced, and Dataset C is highly imbalanced.

testing errors of DNCNN are lower than those of R-CNN, which means weight normalization strategy is able to improve the convergence of the optimization process of DNCNN. These results verify the effectiveness of the normalized layers in DNCNN.

Fig. 5 shows the testing accuracies of Dataset A using the three methods. It is noted that 10 trials are carried out for each experiment to reduce the effects of the randomness. In Fig. 5, it can be seen that DNCNN achieves the best accuracy of 99.22% with 0.22% standard deviation, and S-CNN achieves the lowest accuracy of 96.48% with 0.19% standard deviation. The performance of R-CNN is between DNCNN and S-CNN, and it obtains an accuracy of 97.59% with a standard deviation of 0.64%.

To detail the classification results of each health condition, we plot the confusion matrices of the testing accuracies for Dataset A. In Fig. 6(a)–(c), it is seen that the single faults are more easily recognized and achieve higher accuracies than the compound faults. In Fig. 6(a), some samples of OIF are misclassified as ORIF and some samples of ORIF are misclassified as ORF and OIF. Thus, the accuracies of OIF and ORIF using S-CNN are only 87.8% and 93.6%, respectively. Compared with S-CNN, R-CNN classifies OIF and ORIF with the accuracies of 92.2% and 96%, respectively, as shown in Fig. 6(b). In Fig. 6(c), it is seen that each accuracy of DNCNN in the confusion matrix is higher than the accuracies of R-CNN and S-CNN. These results indicate the superiority of DNCNN in the balanced classification.

CNNs are able to learn features from the vibration signals automatically, and t-distributed stochastic neighbor embedding (t-SNE) [37] is utilized to visualize the learned features. We use t-SNE to map the learned features in the layer P4 into the two-dimension features. The mapped features of the three methods are shown in Fig. 7(a)–(c), respectively. In Fig. 7(a), the features of the compound faults of S-CNN are overlapped slightly and do not cluster well, which matches with the confusion matrix displayed in Fig. 6(a). In Fig. 7(b), the features obtained by R-CNN cluster better than those by S-CNN, but some samples of OIF and ORIF still mix with each other. For DNCNN, it is seen that the features of the same bearing health condition gather closely and the features of the different bearing health conditions separate well in Fig. 7(c). Therefore, DNCNN is able to learn better features from the vibration signals than S-CNN and R-CNN.

4.2. Diagnosis results of DNCNN for imbalanced data classification

In this section, we use the proposed DNCNN for imbalanced data classification, and thus Dataset B and C are considered. Similarly, S-CNN and R-CNN are also used for classification. The testing accuracies for Datasets B and C are displayed in Fig. 5. Compared with the accuracies for Dataset A, the accuracies of the three methods for Dataset B decrease because fewer data samples are used for training these methods. The accuracy of S-CNN is 89.59% with a standard deviation of 0.56%, the accuracy of R-CNN is 95.52% with a standard deviation of 0.698%, and the accuracy of DNCNN is 98.19% with a standard deviation of 0.696%. Since the weighted softmax loss is used in DNCNN to handle the imbalanced classification, DNCNN obtains the highest accuracy in the three methods. Compared with Dataset B, Dataset C is a more imbalanced dataset, which means the data samples used for training are further reduced. The accuracy of S-CNN decreases from 89.59% to 74.46%, and the accuracy of R-CNN decreases from 95.52% to 88.2%. The accuracies of the two methods drop greatly. In contrast, the accuracy of DNCNN is 95.52%, only decreasing by 2.67%. Therefore, the proposed DNCNN performs better than S-CNN and R-CNN in imbalanced fault classification.

To detail the imbalanced classification results of each health condition, we plot the number of the training samples and the confusion matrices of the testing accuracies for Datasets B and C. Fig. 8(a) displays the number of the training samples of each health condition, which shows the imbalanced data distribution of Dataset B. Fig. 8(b)–(d) show the confusion matrices of the three methods. It is seen that the majority health conditions are more easily recognized and achieve higher accuracies than the minority health conditions. Since the training samples of ORIF are the least, S-CNN misclassifies 36.5% of the samples of ORIF as other health conditions. 78.1% of the samples of ORIF are correctly classified by R-CNN, and 17.9% of the samples of ORIF are misclassified as OIF. DNCNN correctly classifies 95.4% of the samples of ORIF, and misclassifies 4% of the samples as OIF.

As shown in Fig. 9(a), Dataset C is the highly imbalanced one. 546 samples of the normal condition are used for training in Dataset C, whereas only 22 samples of ORIF are used for training. Fig. 9(b)–(d) show the confusion matrices of the three methods. It can be seen that S-CNN performs the worst in the three methods and all the samples of ORIF are misclassified

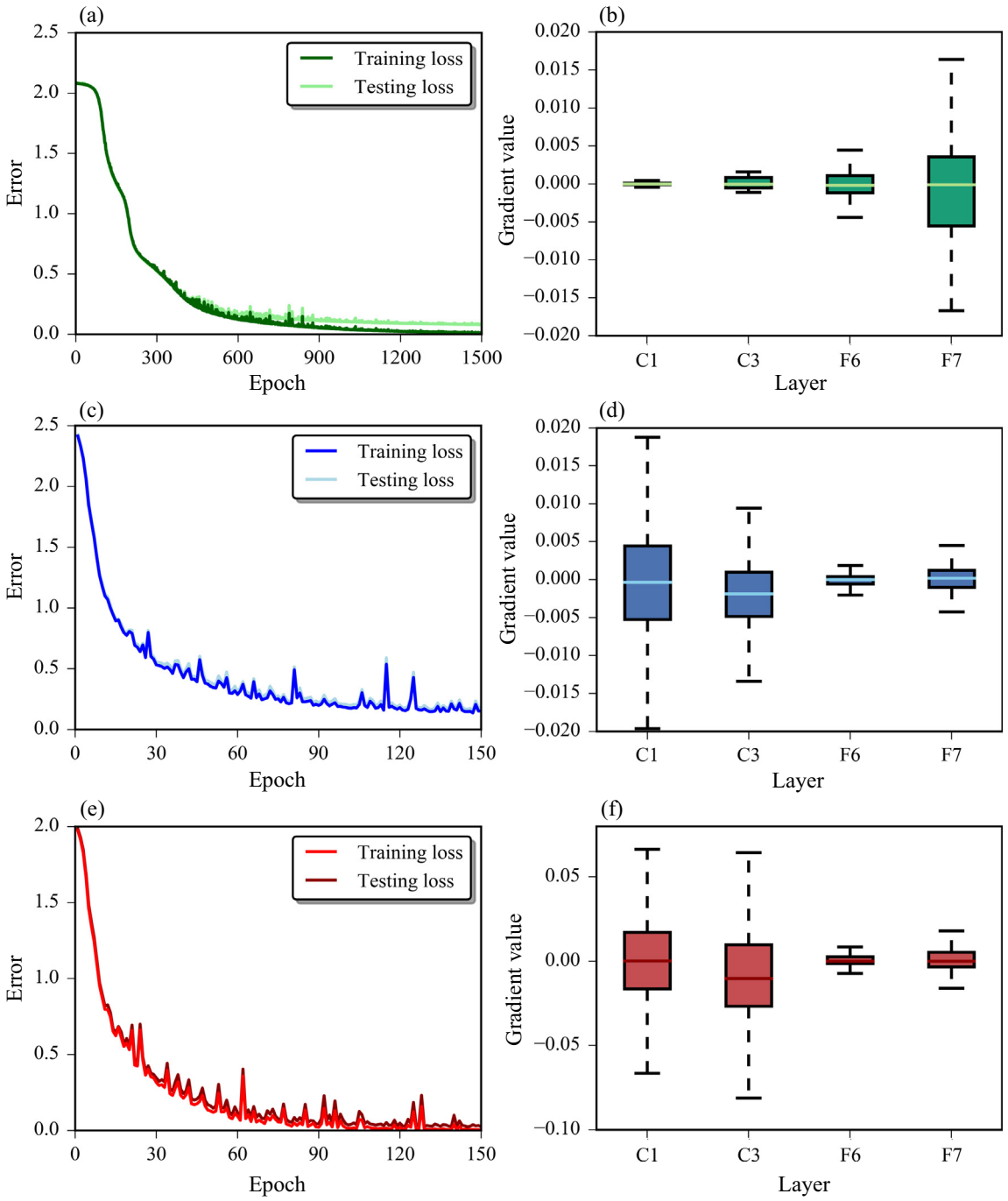

Fig. 4. The results of Dataset A: (a) The losses of S-CNN, (b) the box plot of the gradients of S-CNN, (c) the losses of R-CNN, (d) the box plot of the gradients of R-CNN, (e) the losses of DNCNN, and (f) the box plot of the gradients of DNCNN.

as other health conditions. Compared with the confusion matrix of Dataset B, the accuracy of ORIF using R-CNN drops from 78.1% to 28.2%, which means that R-CNN cannot deal with the highly imbalanced classification well. In contrast, DNCNN uses the weighted softmax loss to highlight the effects of minority health conditions in the training processes. Thus, the accuracy of ORIF achieves 80.4% and the accuracies of the other compound faults are over 94%. These results demonstrate the benefits of the weighted softmax loss in highly imbalanced classification.

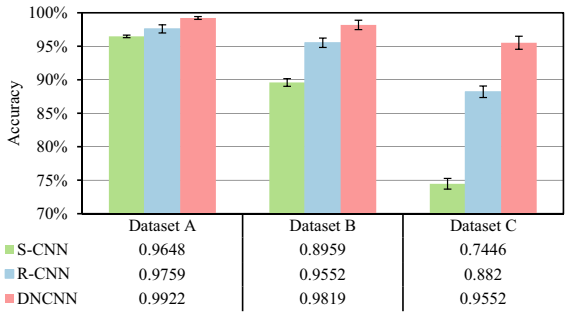

Fig. 5. Testing accuracies of the three datasets using the three methods.

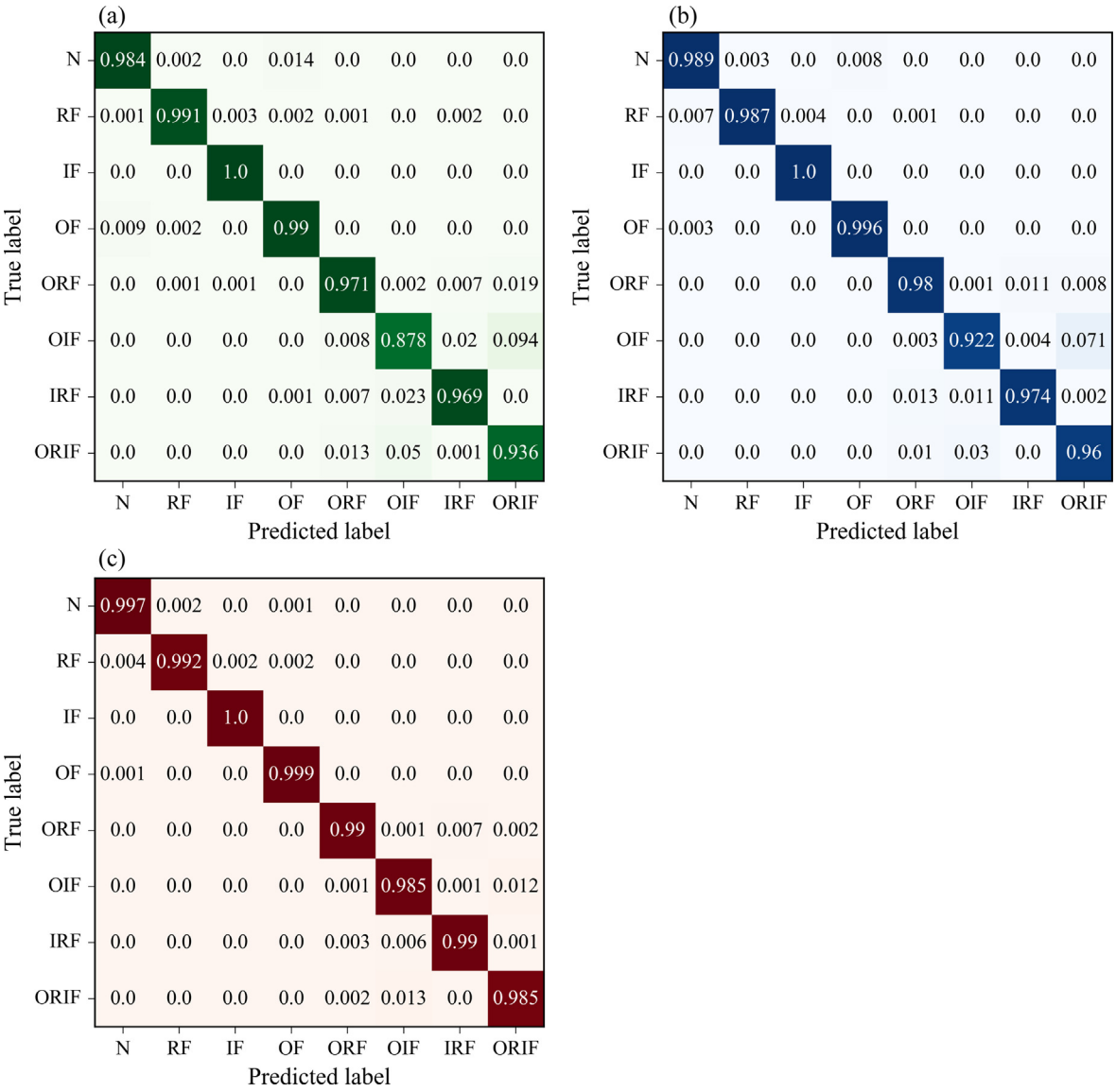

Fig. 6. The results of Dataset A: (a) The confusion matrix of S-CNN, (b) the confusion matrix of R-CNN, and (c) the confusion matrix of DNCNN.

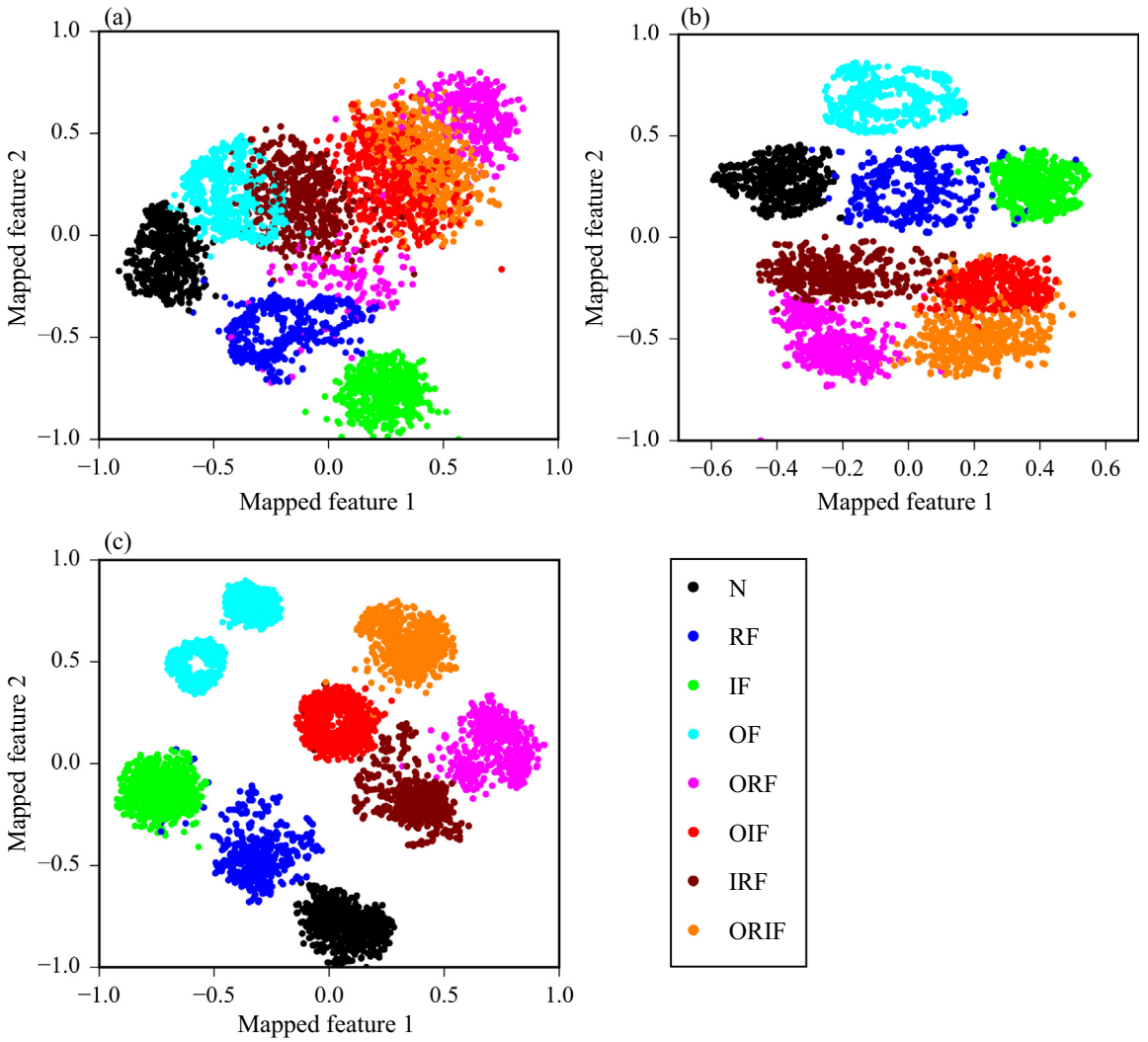

Fig. 7. The visualization of the learned features for Dataset A: (a) S-CNN, (b) R-CNN, and (c) DNCNN.

4.3. Understanding of the feature learning process of DNCNN

We show that the DNCNN is able to learn features from bearing vibration signals in Fig. 7(c). However, it does not make sense how DNCNN learns these features. Consequently, we study the kernels of the normalized convolutional layers of DNCNN using NAM algorithm.

The first normalized convolutional layer of DNCNN is C1, and we can directly display the kernels of this layer in the time domain and the frequency domain. Fig. 10 shows that the kernels of the layer C1 are a set of filters actually. Although a few of these filters present similar properties, most of them have different frequency bands concerning low-frequency or high-frequency information of the input vibration signals. Thus, in DNCNN, the kernels of the layer C1 act as the band-pass filters to remove the unwanted information from the input signal and preserve the necessary information to recognize bearing health conditions.

We also use NAM algorithm to analyze the kernels of the layer C1. The NAM algorithm aims to find an optimal signal that maximizes the activation of each kernel, as shown in Fig. 11. Since the input dimension of DNCNN is 1200, the optimal signals also have the same dimensions. To show these signals clearly, we only display part of the signals in the time domain. It is seen that the patterns of the optimal signals in Fig. 11(a) are consistent with those of the kernels in Fig. 10(a), and the frequency components of the optimal signals in Fig. 11(b) are the main components concerned by the kernels in Fig. 10(b). The results show that the optimal signals have almost the same properties as the kernels of the layer C1, which demonstrates the effectiveness of the developed NAM algorithm. Thus, we can get the properties of the kernels of a deeper convolutional layer by analyzing the signals that maximizes the activations of these kernels.

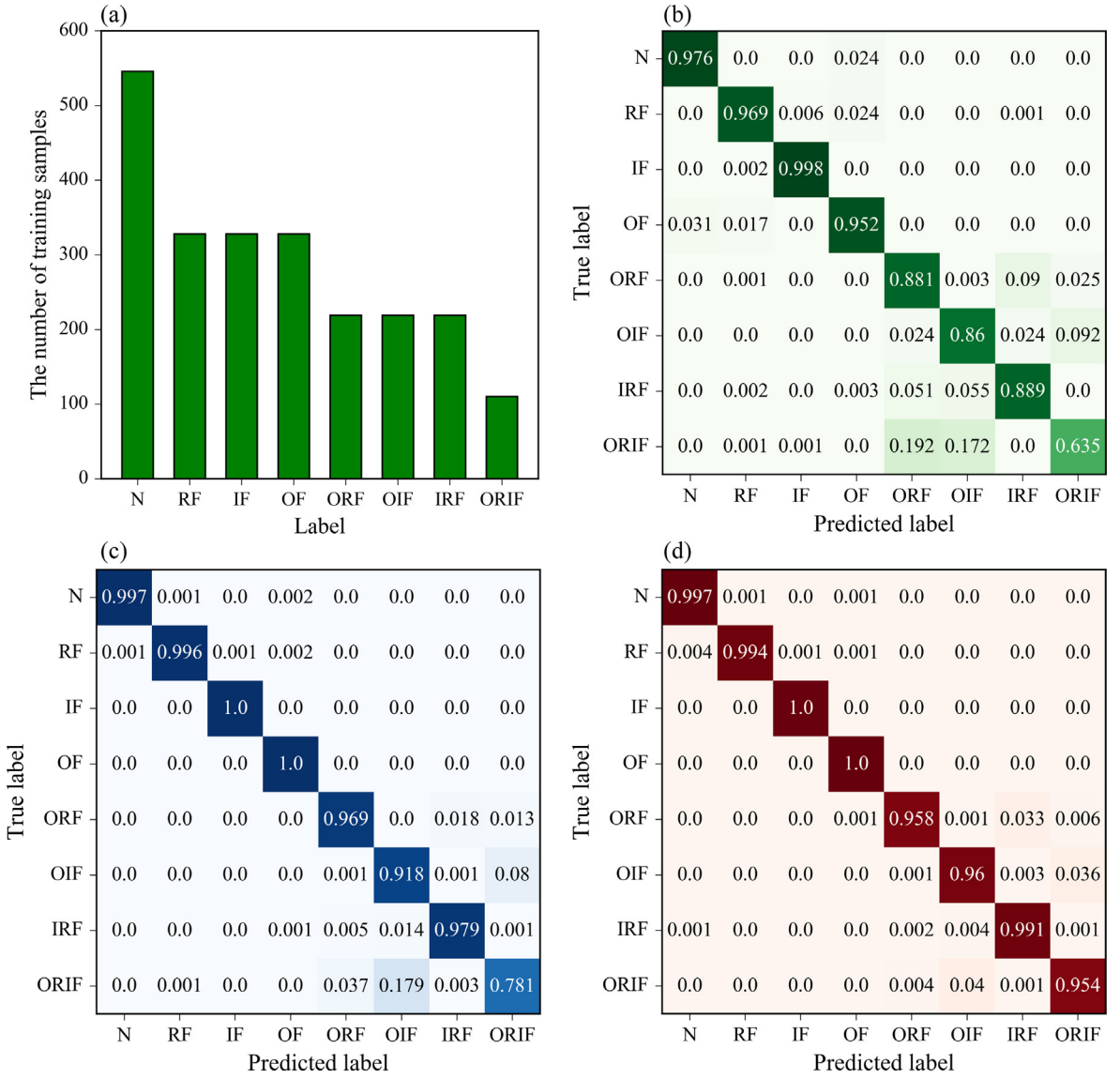

Fig. 8. (a) The number of the training samples of Dataset B, (b) the confusion matrix of S-CNN, (c) the confusion matrix of R-CNN, and (d) the confusion matrix of DNCNN.

The second normalized convolutional layer of DNCNN is C3. Since we cannot directly analyze the kernels of the layer C3, NAM algorithm is applied to obtain the optimal signals for these kernels, as shown in Fig. 12. In the time domain, the optimal signals of the layer C3 present complex properties. For instance, the first kernel acts as a single impulse and the twelfth kernel is like a set of fault impulses. In the frequency domain, different from the kernels of the layer C1 having single peak characteristics, the kernels of the layer C3 concern more parts of the frequency information. These results indicate that the kernels of the layer C3 are a set of filters that preserve more information for classification than those of the layer C1. As a result, the kernels of the layer C1 of DNCNN are the filters passing through single-band components of the input signals, and the kernels of the layer C3 are the filters passing through multi-band components of the input signals. In other words, DNCNN attempts to learn different filters automatically, and the filters become complex when the layers go deeper. Each filter can remove unrelated information to construct effective features for the recognition of the bearing health conditions. Thus, the feature learning process of DNCNN can be regarded as a signal processing process of the vibration signals.

In NAM algorithm, we use the activation index A_j in Eq. (22) to measure the activation degree of a kernel and try to find a signal $\boldsymbol{x}$ that maximizes the index. Since the activation of the kernel is excited mostly, the pattern of the signal is what the kernel likes to see. Alternately, we can use the activation index to find which kernel likes to process the signal of a given health condition. As we know, the bigger the activation index of a kernel is, the fewer components the signal are filtered

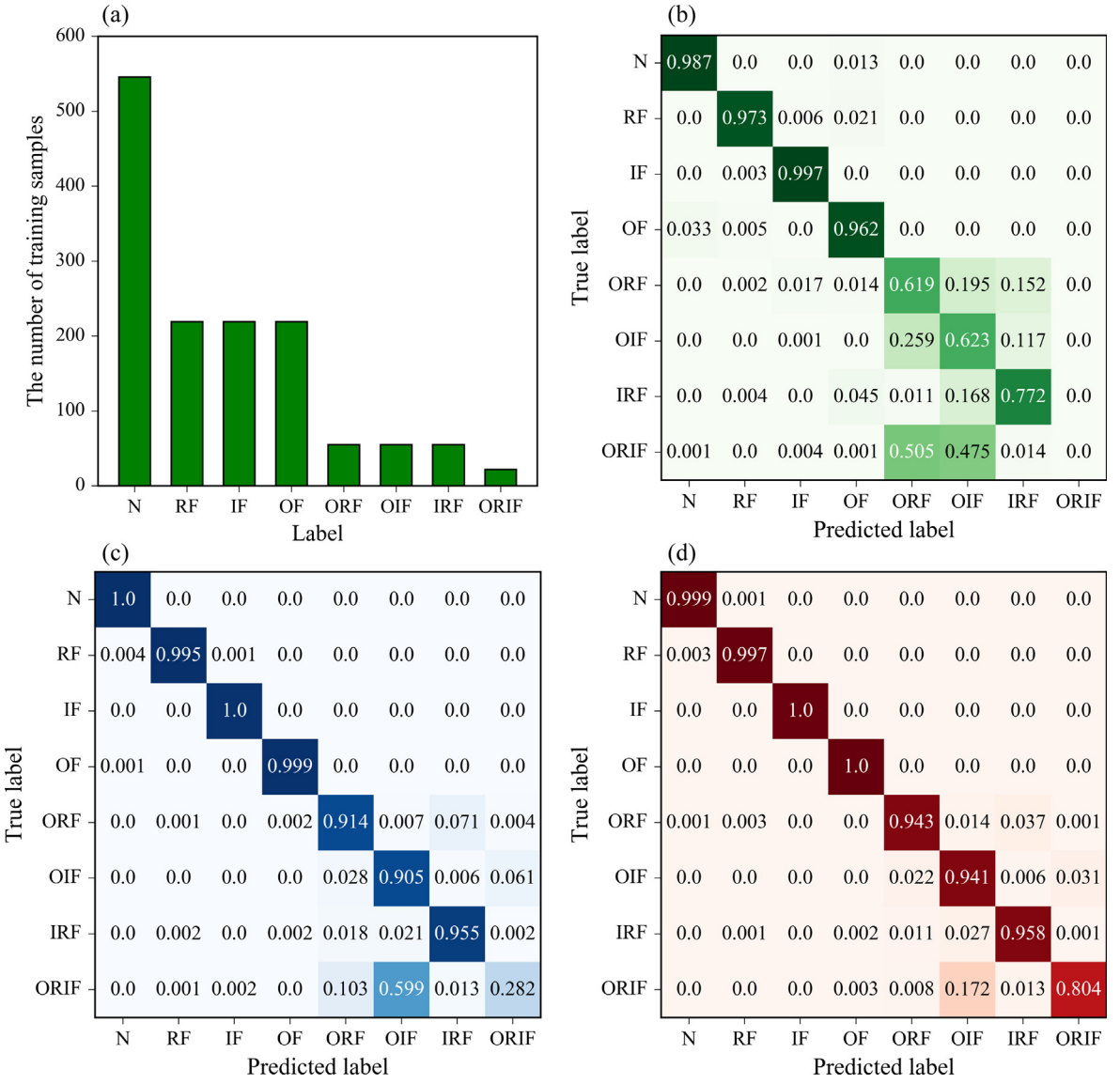

Fig. 9. (a) The number of the training samples of Dataset C, (b) the confusion matrix of S-CNN, (c) the confusion matrix of R-CNN, and (d) the confusion matrix of DNCNN.

by a kernel. Therefore, for each health condition, we calculate the value of the index A_j for every kernels. Fig. 13(a) shows the first three kernels that are most activated for each health condition in the layer C1. It can be seen that the first three kernels that are most activated for RF is the 7th, 9th and 12th kernels. By checking the 7th, 9th and 12th kernels in Fig. 10, it can be found that the high-frequency components of RF are mostly concerned in the layer C1. By this way, we can inspect what components the kernels concern for the signals of each health condition. Moreover, although the three most activated kernels of RF and IF are same in the layer C1, the three most activated kernels of RF and IF are different in the layer C3. As a result, DNCNN can extract different information from the health conditions and thus it learns the discriminative features. Based on the results in Fig. 13, we can understand what information is removed or preserved by the kernels for each health condition.

4.4. Discussions and future work

(1) In this paper, DNCNN using the normalized layers and the weighted softmax loss is proposed for imbalanced fault classification of machinery. The normalized layer is able to improve the training processes of DNCNN so that DNCNN can learn better features and obtain higher accuracies than S-CNN and R-CNN, as verified in Section 4.1. The weighted

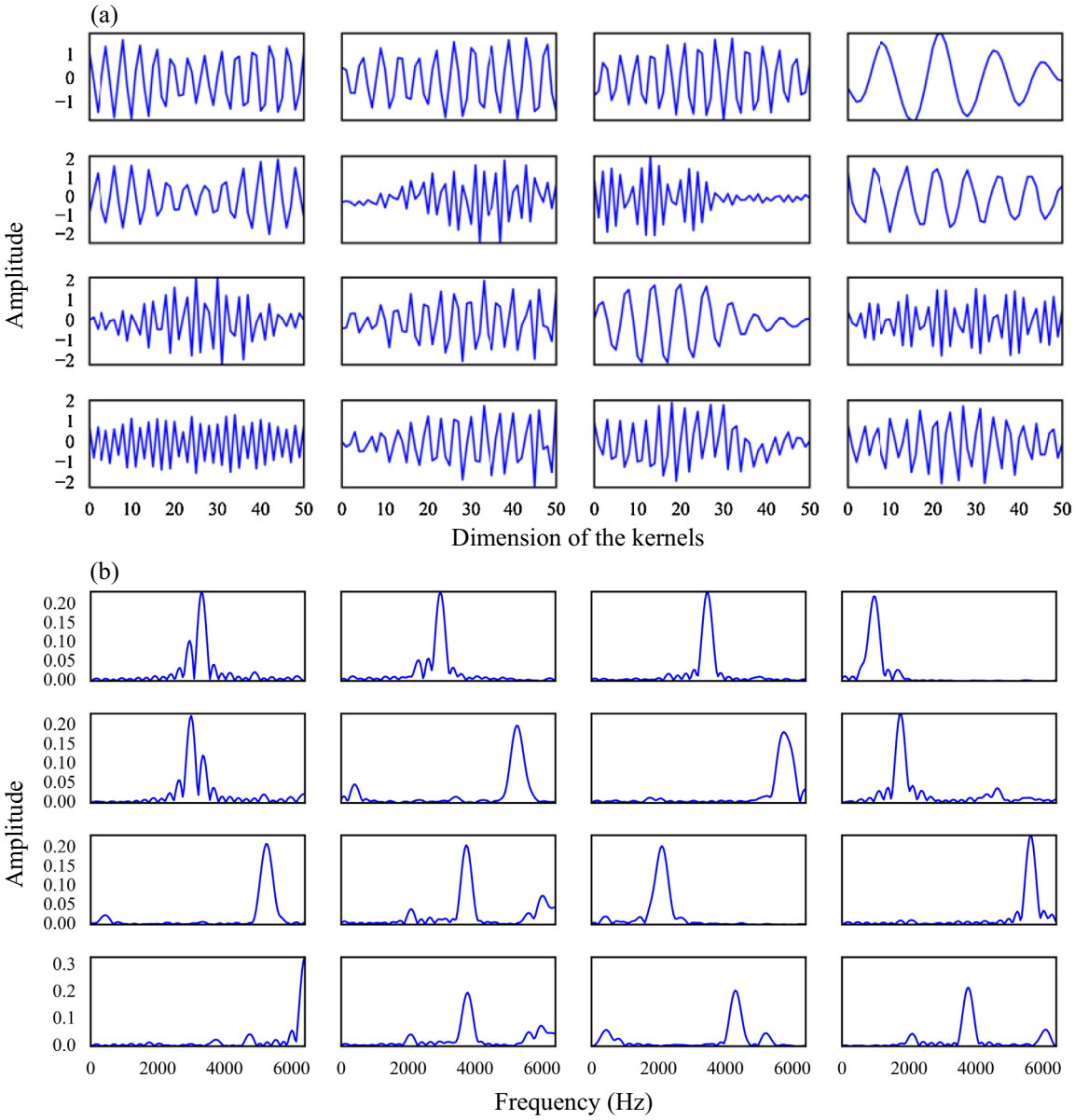

Fig. 10. The kernels of the layer C1 of DNCNN: (a) in the time domain, and (b) in the frequency domain.

softmax loss can help DNCNN deal with the imbalanced classification problem more effectively than S-CNN and R-CNN. For instance, Dataset C is highly imbalanced and the training samples of ORIF are limited. The testing accuracy of this health condition using DNCNN is over 80%. Whereas the testing accuracies of ORIF using S-CNN and R-CNN are only 0 and 28.2%, respectively. Thus, DNCNN shows its superiority in imbalanced fault classification of machinery.

(2) Deep learning models have attracted attentions in the field of intelligent fault diagnosis of machinery since they are able to learn features from vibration signals automatically. But few papers further investigate what these CNNs have learned from the vibration signals. Thus, we attempt to handle this question preliminarily. We analyze the kernels of the normalized convolutional layers of DNCNN by NAM algorithm. Not surprisingly, we can find out some patterns in these kernels, as shown in Figs. 10–13. The kernels in the first convolutional layer are a set of simple filters that have single peak characteristics. The kernels in the second convolutional layer are a set of complex filters that have multiple peak characteristics. Such phenomena, to some extent, demonstrates that DNCNN tries to learn multi-stage filters to process the vibration signals automatically. These filters retain the important components of the input signals for classification and suppress the useless aspects. In future work, we will specifically analyze the properties of each kernel, which may help us understand deep learning models more comprehensively.

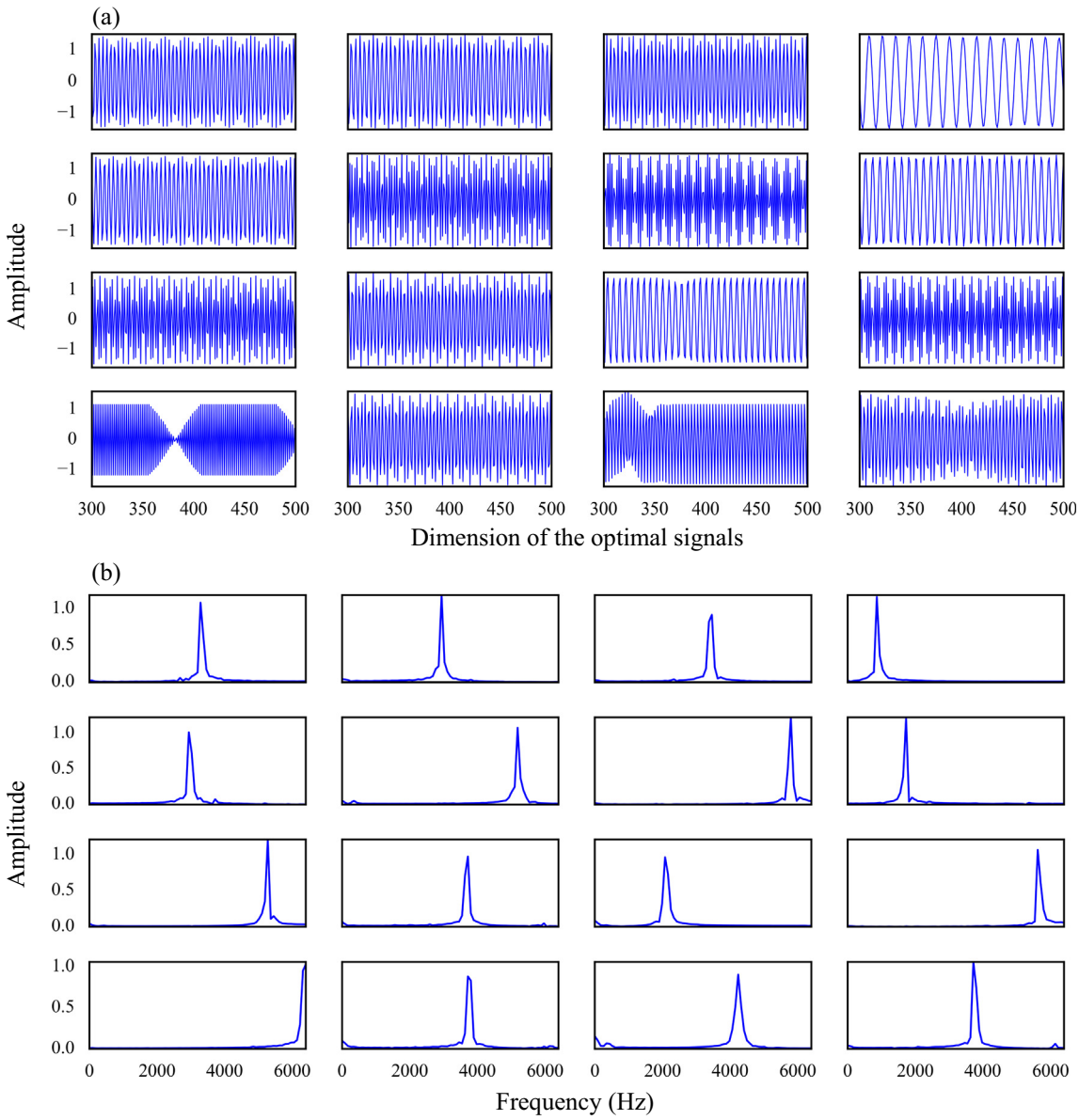

Fig. 11. The optimal signals that maximize the activations of the kernels in the layer C1 of DNCNN: (a) in the time domain, and (b) in the frequency domain.

(3) We do not pay attention on the parameter selection of DNCNN in this paper and thus the architecture of DNCNN is selected in advance. Although the cross validation and other methods are used for architecture selection in some studies, these methods are only guided by classification accuracies and cannot give us more insights on the criterions of the architecture selection. Thus, how to select the architecture of a CNN is still an open issue. Actually, we can find some clues to deal with this problem based on the visualization results of DNCNN. For instance, J, H and L are the main parameters of DNCNN. J means how many filters should be used in a layer, H means how long a filter should be applied, and L determines how many stages the filters will be used. Intuitively, the bigger J means more kinds of filters used in a layer, the longer H means the better performance of a filter and the larger L indicates that more stages of filters are used in a CNN. However, the larger these parameters are, the more the computational resources are required. The visualization of the kernels may enlighten us on building the relationship between the architecture selection of a CNN and the parameter selection of a digital filter. We will focus on this topic in future work.

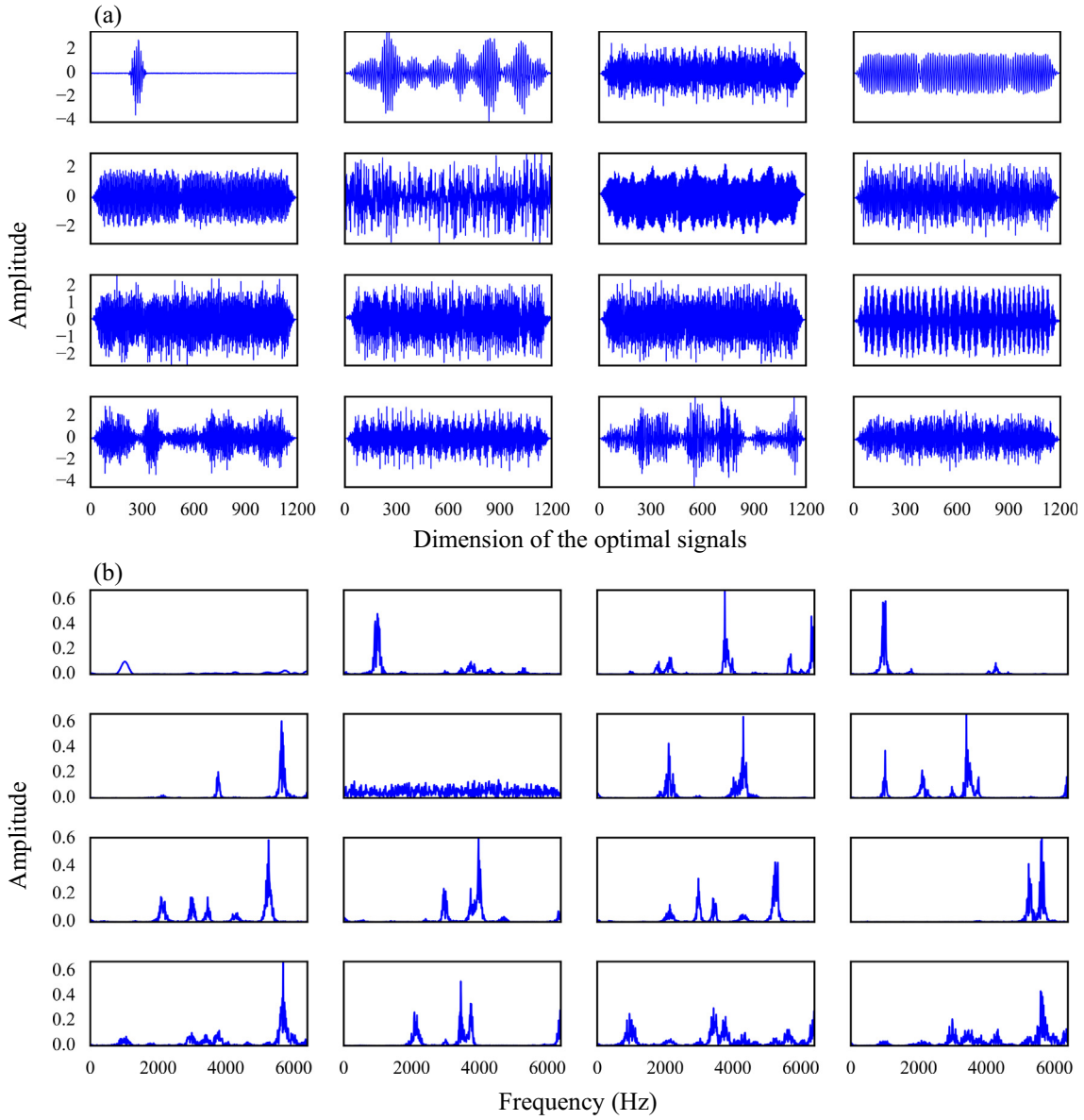

Fig. 12. The optimal signals that maximize the activations of the kernels in the layer C3 of DNCNN: (a) in the time domain, and (b) in the frequency domain.

5. Conclusions

A framework called DNCNN is proposed for imbalanced fault classification of machinery. In this framework, ReLU and weight normalization strategy are applied to construct normalized layers to improve the training processes and the weighted softmax loss is developed to deal with the imbalanced classification problem. Three bearing datasets with different imbalanced degrees are used to verify the proposed DNCNN. The classification results show that DNCNN not only learns better features than the commonly used CNNs, but also deals with the imbalanced classification problem more effectively. Furthermore, the NAM algorithm is developed to explore the properties of the kernels of DNCNN. The results show that the kernels of the normalized convolutional layers act as filters and they become complex when the layers go deeper, which may help us understand what DNCNN have learned from the vibration signals.

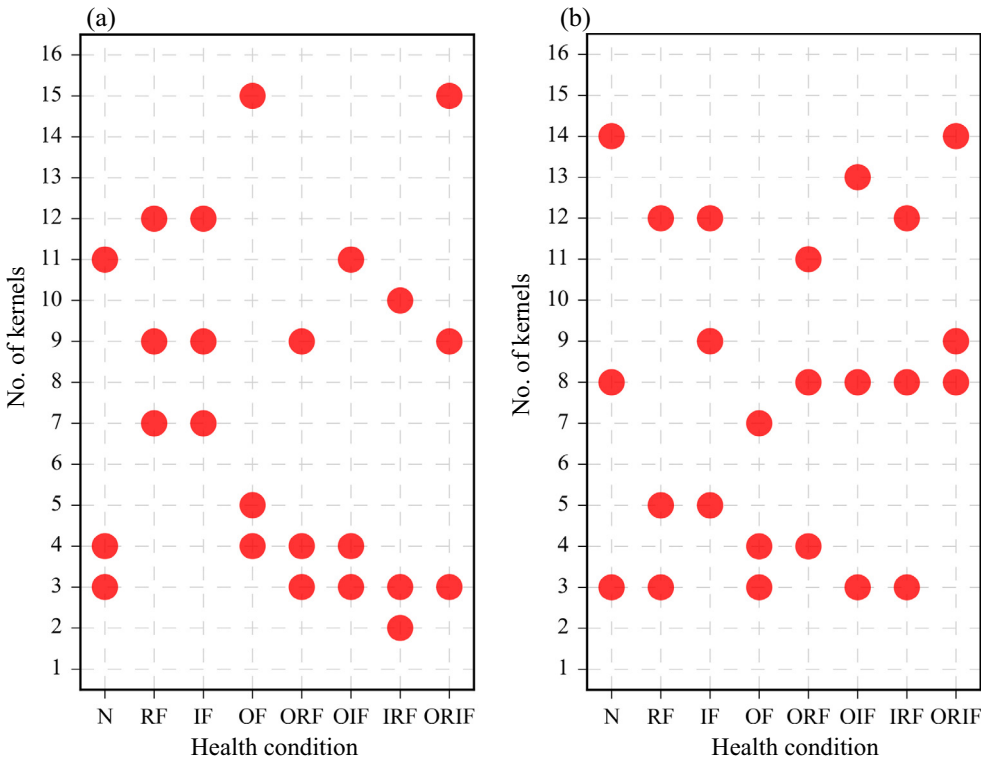

Fig. 13. The first three kernels that are most activated for each health condition: (a) the kernels in the layer C1 of DNCNN, and (b) the kernels in the layer C3 of DNCNN.

Acknowledgements

This research was supported by National Natural Science Foundation of China (U1709208 and 51421004), National Program for Support of Top-notch Young Professionals, and Visiting Scholar Foundation of the State Key Lab. of Traction Power in Southwest Jiaotong University (TPL1703).

References

[1] K. Worden, W.J. Staszewski, J.J. Hensman, Natural computing for mechanical systems research: a tutorial overview, Mech. Syst. Sig. Process. 25 (2011) 4–111.
[2] A. Glowacz, W. Glowacz, Z. Glowacz, J. Kozik, Early fault diagnosis of bearing and stator faults of the single-phase induction motor using acoustic signals, Measurement 113 (2018) 1–9.
[3] L. Guo, N. Li, F. Jia, Y. Lei, J. Lin, A recurrent neural network based health indicator for remaining useful life prediction of bearings, Neurocomputing 240 (2017) 98–109.
[4] J.B. Ali, N. Fnaiech, L. Saidi, B. Chebel-Morello, F. Fnaiech, Application of empirical mode decomposition and artificial neural network for automatic bearing fault diagnosis based on vibration signals, Appl. Acoust. 89 (2015) 16–27.
[5] M.Y. Asr, M.M. Ettefagh, R. Hassannejad, S.N. Razavi, Diagnosis of combined faults in rotary machinery by non-naive bayesian approach, Mech. Syst. Sig. Process. 85 (2017) 56–70.
[6] G. Georgoulas, P. Karvelis, T. Loutas, C.D. Stylios, Rolling element bearings diagnostics using the symbolic aggregate approximation, Mech. Syst. Sig. Process. 60 (2015) 229–242.
[7] Q. Xiong, W. Zhang, T. Lu, G. Mei, S. Liang, A fault diagnosis method for rolling bearings based on feature fusion of multifractal detrended fluctuation analysis and alpha stable distribution, Shock Vib. 2016 (2016) 1–12.
[8] Y. Li, Y. Yang, G. Li, M. Xu, W. Huang, A fault diagnosis scheme for planetary gearboxes using modified multi-scale symbolic dynamic entropy and mRMR feature selection, Mech. Syst. Sig. Process. 91 (2017) 295–312.
[9] Y. Lei, Z. Liu, X. Wu, N. Li, W. Chen, J. Lin, Health condition identification of multi-stage planetary gearboxes using a mRVM-based method, Mech. Syst. Sig. Process. 60 (2015) 289–300.
[10] H. Shao, H. Jiang, H. Zhao, F. Wang, A novel deep autoencoder feature learning method for rotating machinery fault diagnosis, Mech. Syst. Sig. Process. 95 (2017) 187–204.
[11] W. Zhang, G. Peng, C. Li, Y. Chen, Z. Zhang, A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals, Sensors 17 (2017) 425.
[12] M. Ma, C. Sun, X. Chen, Discriminative deep belief networks with ant colony optimization for health status assessment of machine, IEEE Trans. Instrum. Meas. 66 (2017) 3115–3125.
[13] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.
[14] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Networks 61 (2015) 85–117.
[15] O. Janssens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle, S. Van Hoecke, Convolutional neural network based fault detection for rotating machinery, J. Sound Vib. 377 (2016) 331–345.

[16] K.B. Lee, S. Cheon, C.O. Kim, A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes, IEEE Trans. Semicond. Manuf. 30 (2017) 135–142.
[17] X. Guo, L. Chen, C. Shen, Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis, Measurement 93 (2016) 490–502.
[18] F. Wang, H. Jiang, H. Shao, W. Duan, S. Wu, An adaptive deep convolutional neural network for rolling bearing fault diagnosis, Meas. Sci. Technol. (2017), https://doi.org/10.1088/1361-6501/aa6e22.
[19] X. Ding, Q. He, Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis, IEEE Trans. Instrum. Meas. 66 (2017) 1926–1935.
[20] W. Mao, L. He, Y. Yan, J. Wang, Online sequential prediction of bearings imbalanced fault diagnosis by extreme learning machine, Mech. Syst. Sig. Process. 83 (2017) 450–473.
[21] I. Martin-Diaz, D. Morinigo-Sotelo, O. Duque-Perez, R.J. Romero-Troncoso, Early fault detection in induction motors using AdaBoost with imbalanced small data and optimized sampling, IEEE Trans. Ind. Appl. (2017), https://doi.org/10.1109/TIA.2016.2618756.
[22] H. Malik, S. Mishra, Proximal support vector machine (PSVM) based imbalance fault diagnosis of wind turbine using generator current signals, Energy Proc. 90 (2016) 593–603.
[23] Y. Tang, Y.Q. Zhang, N.V. Chawla, S. Krasser, SVMs modeling for highly imbalanced classification, IEEE Trans. Syst., Man, Cybern. Part B (Cybernetics) 39 (2009) 281–288.
[24] Y. Lei, F. Jia, J. Lin, S. Xing, S.X. Ding, An intelligent fault diagnosis method using unsupervised feature learning towards mechanical big data, IEEE Trans. Ind. Electron. 63 (2016) 3137–3147.
[25] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning, 2010, pp. 807–814.
[26] T. Salimans, D.P. Kingma, Weight normalization: a simple reparameterization to accelerate training of deep neural networks, in: Advances in Neural Information Processing Systems, 2016, pp. 901–909.
[27] T.N. Sainath, B. Kingsbury, G. Saon, H. Soltau, A. Mohamed, G. Dahl, B. Ramabhadran, Deep convolutional neural networks for large-scale speech tasks, Neural Networks 64 (2015) 39–48.
[28] D.H. Hubel, T.N. Wiesel, Receptive fields and functional architecture of monkey striate cortex, J. Physiol. 195 (1968) 215–243.
[29] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, Recent advances in convolutional neural networks, arXiv preprint arXiv:1512.07108, 2015.
[30] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. (2012) 1097–1105.
[31] M.A. Nielsen, Neural Networks and Deep Learning, Determination Press, 2015.
[32] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167, 2015.
[33] J. Bouvrie, Notes on Convolutional Neural Networks, 2006.
[34] A.G. Huth, W.A. de Heer, T.L. Griffiths, F.E. Theunissen, J.L. Gallant, Natural speech reveals the semantic maps that tile human cerebral cortex, Nature 532 (2016) 453–458.
[35] D.C. Liu, J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. 45 (1989) 503–528.
[36] Y. Lei, Z. He, Y. Zi, Application of a novel hybrid intelligent method to compound fault diagnosis of locomotive roller bearings, J. Vib. Acoust.-Trans. the ASME 130 (2008) 034501.
[37] L. Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (2008) 2579–2605.