This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

Deep Residual Shrinkage Networks for Fault Diagnosis

Minghang Zhao, Shisheng Zhong, Xuyun Fu, Baoping Tang, and Michael Pecht, *Fellow Member, IEEE*

***Abstract*—This paper develops new deep learning methods, namely, deep residual shrinkage networks, to improve the feature learning ability from highly noised vibration signals and achieve a high fault diagnosing accuracy. Soft thresholding is inserted as nonlinear transformation layers into the deep architectures to eliminate unimportant features. Moreover, considering that it is generally challenging to set proper values for the thresholds, the developed deep residual shrinkage networks integrate a few specialized neural networks as trainable modules to automatically determine the thresholds, so that professional expertise on signal processing is not required. The efficacy of the developed methods is validated through experiments with various types of noise.**

***Index Terms*—Deep learning, deep residual networks, fault diagnosis, soft thresholding, vibration signal.**

I. Introduction

ROTATING machines are integral to manufacturing, power supply, transportation, and aerospace industries. However, because these rotating machines operate under harsh working environments, failures unavoidably occur in their mechanical transmission systems and can result in accidents and economic losses. Accurate fault diagnosis of mechanical transmission systems can be used to schedule maintenance and extend service time as well as ensure human safety [1]-[3].

The existing fault diagnosis algorithms for mechanical transmission systems can be classified into two categories, i.e., signal analysis-based methods and machine learning-powered methods [4]. In general, signal analysis-based fault diagnosis methods identify faults by detecting the fault-related vibration components or characteristic frequencies. However, for large rotating machines, the vibration signals are often composed of many different vibration components, including meshing of gears and rotation of shafts and bearings. Further, when faults are in their early stage, the fault-related components are often weak and can easily be overwhelmed by other vibration components and harmonics. As a result, it is often difficult for the traditional signal analysis-based fault diagnosis methods to identify the fault-related vibration components and characteristics frequencies.

Machine learning-powered fault diagnosis methods, on the other hand, are able to diagnose faults without identifying the fault-related components and characteristics frequencies. A number of statistical parameters (e.g., kurtosis, root mean square, energy, and entropy) can be extracted to represent the health states, and then a classifier (e.g., multi-class support vector machines, one-hidden-layer neural networks, and naïve Bayes classifier) can be trained to diagnose the faults. However, the extracted statistical parameters are often not discriminative enough to distinguish the faults, which can lead to low diagnostic accuracy. As a consequence, finding a discriminative feature set has become a long-standing challenge for machine learning-powered fault diagnosis [5].

In recent years, deep learning methods [6], which refer to the machine learning methods with multiple levels of nonlinear transformations, have become a useful tool in vibration-based fault diagnosis. To replace the traditional statistical parameters, deep learning methods automatically learn features from raw vibration signals, which can yield higher diagnostic accuracy. A variety of deep learning methods have been used in machine fault diagnosis [7]-[14]. For example, Ince et al. [7] employed a 1-dimensional convolutional neural network (ConvNet) to learn features from current signals for real-time motor fault diagnosis. Shao et al. [9] applied a convolutional deep belief network for fault diagnosis of electric locomotive bearings. However, parameter optimization is often a difficult task for traditional deep learning methods. The gradients of the error function, which have to be back-propagated layer by layer, gradually become inaccurate after flowing through a number of layers. As a result, the trainable parameters in the beginning layers (i.e., the layers close to the input layer) cannot be optimized effectively.

Deep residual networks (ResNets) are an attractive variant of ConvNets, which use identity shortcuts to ease the difficulty of parameter optimization [15]. In ResNets, the gradients not only are back-propagated layer by layer, but also directly flow back to the beginning layers through identity shortcuts [16]. Due to the improved parameter optimization, ResNets have been applied for fault diagnosis in a few recent papers [17]-[20]. For example, Ma at al. [17] used a ResNet with demodulated time-frequency features to diagnose planetary gearboxes under

Manuscript received July 16, 2019; revised September 4, 2019; accepted September 21, 2019. This work was supported by the Key National Natural Science Foundation of China under Grant U1533202. Paper no. TII-19-3183. *(Corresponding author: Minghang Zhao)*

M. Zhao, S. Zhong, and X. Fu are with the School of Naval Architecture and Ocean Engineering, Harbin Institute of Technology at Weihai, Weihai 264209, China (e-mail: zhaomh@hit.edu.cn; zhongss@hit.edu.cn; fuxuyun@hit.edu.cn).

B. Tang is with the State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing 400044, China (e-mail: bptang@cqu.edu.cn).

M. Pecht is with the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD 20742, USA (e-mail: pecht@umd.edu).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

nonstationary running conditions. Zhao at al. [18] used ResNets to fuse multiple sets of wavelet packet coefficients for fault diagnosis. The advantages of ResNets over classical ConvNets have been validated in these papers.

Vibration signals collected from large rotating machines, such as wind turbines, manufacturing machines, and heavy trucks, often contain large amounts of noise. The feature learning ability of ResNets often decreases when dealing with highly noised vibration signals. The convolutional kernels used in ResNets, which serve as local feature extractors, may fail to detect the fault-related features due to the interference of noise. In such a case, the learned high-level features at the output layer are often not discriminative enough to correctly classify the faults. Therefore, it is necessary to develop new deep learning methods for vibration-based fault diagnosis of rotating machines under strong background noise.

This paper develops two deep residual shrinkage networks (DRSNs), i.e., a DRSN with channel-shared thresholds (DRSN-CS) and a DRSN with channel-wise thresholds (DRSN-CW), to improve the feature learning ability of ResNets from highly noised vibration signals, with the ultimate goal of yielding high diagnostic accuracy. The major contributions are summarized as follows:

- Soft thresholding (i.e., a popular shrinkage function) is inserted into the deep architecture as nonlinear transformation layers, in order to effectively eliminate the noise-related features.
- The thresholds are adaptively determined using specially designed sub-networks so that each piece of vibration signal can have its own set of thresholds.
- Two kinds of thresholds, namely, channel-shared thresholds and channel-wise thresholds, are considered in soft thresholding, which is the reason for the terms DRSN-CS and DRSN-CW.

The remainder of this paper is arranged as follows. Section II provides a brief overview of classical ResNets and a detailed elaboration on the developed DRSN-CS and DRSN-CW. Experimental comparisons are given in Section III, and the conclusions are given in Section IV.

II. Theory of the Developed DRSNs

As stated in Section I, as a potentially effective way to learn discriminative features from highly noised vibration signals, the integration of deep learning methods and soft thresholding is considered in this study. Accordingly, this section focuses on developing two improved variants of ResNets, i.e., DRSN-CS and DRSN-CW, by presenting the theoretical backgrounds and essential ideas in detail.

A. Basic Components

Both the ResNets and the developed DRSNs have some basic components that are the same as the traditional ConvNets, including the convolutional layer, rectifier linear unit (ReLU) activation function, batch normalization (BN), global average pooling (GAP), and cross-entropy error function. The concepts of these basic components are introduced as follows.

The convolutional layer is the key component that makes a ConvNet different from traditional fully connected neural networks. The convolutional layer can greatly reduce the amount of parameters that need to be trained. This is achieved by using convolutions instead of matrix multiplications, in which the convolutional kernels in the convolutional layers can have much fewer parameters than the transformation matrices in the fully connected layers. Further, with fewer trainable parameters, it will be less likely for the deep learning methods to suffer overfitting, so that it can be easier to yield a relatively high accuracy on testing datasets. The convolution between the input feature map and a convolutional kernel, which follows by adding a bias term, can be expressed by

$$y_j = \sum_{i \in M_j} x_i * k_{ij} + b_j \tag{1}$$

where x_i is the ith channel of the input feature map, y_j is the jth channel of the output feature map, k is the convolutional kernel, b is the bias, and M_j is a collection of channels that are used for calculating the jth channel of the output feature map [20]. The convolution can be repeated a number of times to obtain the output feature map.

Figure 1 shows the convolution process. As shown in Figs. 1(a)-(b), the feature map and convolutional kernel are, in fact, 3-dimensional (3D) tensors. In this study, the 1-dimensional (1D) vibration signals are taken as input, so that the height of the feature map and convolutional kernel always equals to one. As shown in Fig. 1(c), the convolutional kernel slides on the input feature map so that a channel of the output feature map can be obtained. In each convolutional layer, there is often more than one convolutional kernel, so that the output feature map can have more than one channel.

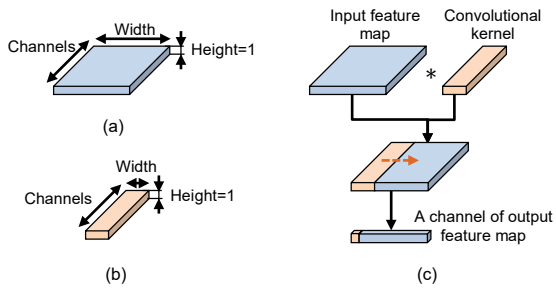


Fig. 1. Illustrations of (a) a feature map, (b) a convolutional kernel, and (c) the convolution process.

BN is a feature normalizing technique that is inserted into deep learning architectures as a trainable process [21]. The purpose of BN is to reduce internal covariant shift, in which the distribution of features often continuously changes over the training iterations. In such a condition, the parameters in the convolutional layers have to be continuously updated to adapt to the changed distributions, which increases the training difficulty. BN normalizes the features to a fixed distribution (with an average value of zero and a standard deviation of one) in the first step, and then adjusts the features to a desirable distribution which is learned in the training process. The process of BN is expressed by

$$\mu = \frac{1}{N_{\text{batch}}} \sum_{n=1}^{N_{\text{batch}}} x_n \tag{2}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

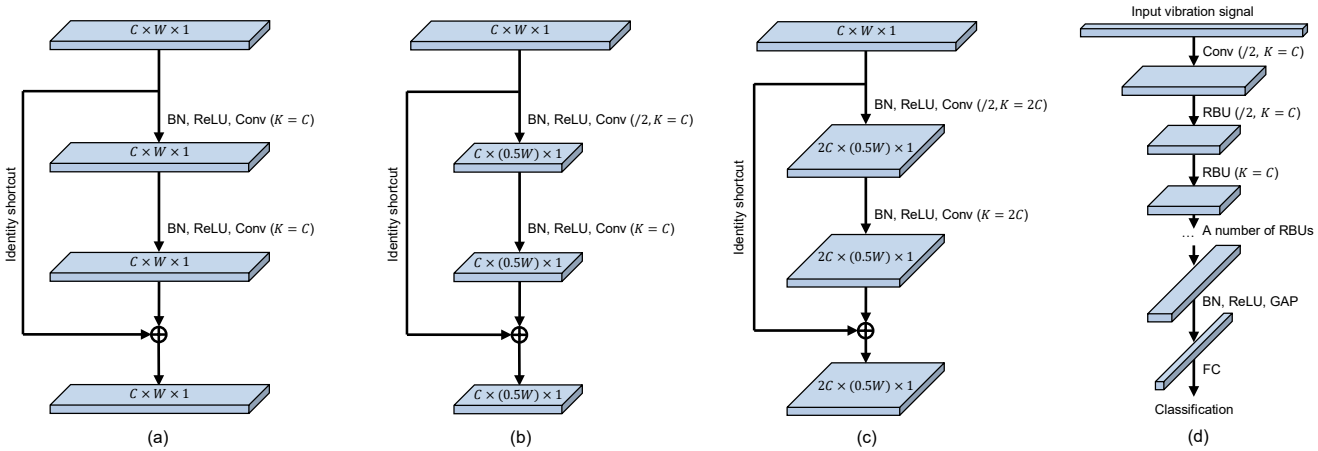


Fig. 2. Three kinds of RBUs, including (a) an RBU in which the output feature map is the same size as the input feature map, (b) an RBU with a stride of 2, in which the width of the output feature map is reduced to half of that of the input feature map, and (c) an RBU with a stride of 2 and a doubled number of convolutional kernels, in which the number of channels of the output feature map is doubled. (d) The overall architecture of a ResNet. "/2" means to move the convolutional kernel with a stride of 2 to reduce the width of the output feature map. *C*, *W*, and 1 are the indicators of the number of channels, width, and height of the feature map, respectively. *K* is the number of convolutional kernels in the convolutional layer.

$$\sigma^2 = \frac{1}{N_{\text{batch}}} \sum_{n=1}^{N_{\text{batch}}} (x_n - \mu)^2 \tag{3}$$

$$\hat{x}_n = \frac{x_n - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{4}$$

$$y_n = \gamma \hat{x}_n + \beta \tag{5}$$

where x_n and y_n represent the input and output features of the nth observation in a mini-batch, respectively. γ and β are two trainable parameters to scale and shift the distributions. ϵ is a constant which is close to zero.

Activation functions are generally an essential component of neural networks and are used for nonlinear transformations. In the past decades, a variety of activation functions have been developed, including sigmoid, tanh, and ReLU. The ReLU activation function has attracted much attention recently because it can be effective in preventing gradient vanishing. The derivative of the ReLU activation function is either one or zero, which is helpful to keep the range of features unchanged when flowing among the layers. A ReLU activation function is expressed by

$$y = \max(x, 0) \tag{6}$$

where x and y are the input and output of the ReLU activation function, respectively [6].

GAP is an operation that calculates a mean value from each channel of the feature map [22]. In general, it is used before the final output layer. GAP can reduce the number of weights to be used in the fully connected output layer, so that it will be less likely for the deep neural networks to encounter overfitting. GAP can also address the shift variant problem, so that the features learned by deep neural networks will not be influenced by the changing of locations of faulty impulses.

Cross-entropy error is often used as the objective function to be minimized in multi-class recognition tasks [6]. Compared to the conventional squared mean error, cross-entropy error often leads to a higher training efficiency because it is less likely for the gradient of cross-entropy error, with respect to weights, to vanish to zeros. To calculate the cross-entropy error, a softmax function has to be applied to enforce the features to the range of (0, 1). The softmax function is expressed by

$$y_j = \frac{e^{x_j}}{\sum_{i=1}^{N_{\text{class}}} e^{x_i}} \tag{7}$$

where x and y are the input and output feature maps of the softmax function, respectively; i and j are the indexes of the neurons at the output layer; and N_{class} is the number of classes. Here, y_j can be regarded as a predicted probability of an observation belonging to the jth class. Then, the cross-entropy error per observation is expressed by

$$E = -\sum_{j=1}^{N_{\text{class}}} t_j \log(y_j) \tag{8}$$

where t is the targeted output and t_j is the actual probability of an observation belonging to the jth class. After calculating the cross-entropy error, a gradient descent algorithm can be applied to optimize the parameters, and the deep neural networks can be fully trained after a number of iterations.

B. Architecture of the Classical ResNet

ResNets are a newly emerging deep learning method that have attracted much attention in recent years [15]. Residual building units (RBUs) are the basic components. As shown in Fig. 2(a), an RBU is composed of two BNs, two ReLUs, two convolutional layers, and an identity shortcut. The identity shortcut is the part that makes a ResNet superior to the general ConvNets. The gradients of cross-entropy error are back-propagated layer-by-layer in the general ConvNet. With the use of identity shortcuts, the gradients can flow effectively to the earlier layers, which are close to the input layer, so that the parameters can be updated more efficiently. Figures 2(b)-(c) show RBUs that result in different sizes of output feature maps. The motivation for reducing the width of the output feature map is to reduce the calculation amount in the following layers, and the motivation for increasing the number of channels of the output feature map is to facilitate the integration of different features to be discriminative features. Figure 2(d) shows the overall architecture of a ResNet, which consists of an input layer, a convolutional layer, a number of RBUs, a BN, a ReLU, a GAP, and an output fully connected (FC) layer, and is used as the baseline to be further improved in this study.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

C. Design of Fundamental Architectures for DRSNs

In this subsection, the motivations for developing DRSNs are introduced, and the architectures of the two developed DRSNs (i.e., DRSN-CS and DRSN-CW) are elaborated in detail.

1) Theoretical background

In the past 20 years, soft thresholding has often been used as a key step in many signal denoising methods [23],[24]. In general, the raw signal is transformed to a domain in which the near-zero numbers are unimportant, and then soft thresholding is applied to convert the near-zero features to zeros. For example, as a classical signal denoising method, wavelet thresholding is often composed of three steps: wavelet decomposition, soft thresholding, and wavelet reconstruction. To ensure a good performance in signal denoising, a key task in wavelet thresholding is to design a filter that can transform useful information to very positive or negative features, and noise information to near-zero features. However, designing such a filter requires much expertise on signal processing and has always been a challenging issue. Deep learning provides a new way to address this issue. Instead of artificially designing filters by experts, deep learning enables the filters to be learned automatically using a gradient descent algorithm. As a result, the integration of soft thresholding and deep learning can be a promising way to eliminate noise-related information and construct highly discriminative features. The function of soft thresholding can be expressed by

$$y = \begin{cases} x - \tau & x > \tau \\ 0 & -\tau \le x \le \tau \\ x + \tau & x < -\tau \end{cases} \tag{9}$$

where x is the input feature, y is the output feature, and τ is the threshold, i.e., a positive parameter. Instead of setting the negative features to zero in the ReLU activation function, soft thresholding sets the near-zero features to zeros, so that useful negative features can be preserved.

The process of soft thresholding is shown in Fig. 3(a). It can be observed that the derivative of output on input is either one or zero, which is effective in preventing gradient vanishing and exploding problems, as shown in Fig. 3(b). The derivative can be expressed by

$$\frac{\partial y}{\partial x} = \begin{cases} 1 & x > \tau \\ 0 & -\tau \le x \le \tau \\ 1 & x < -\tau \end{cases} \tag{10}$$

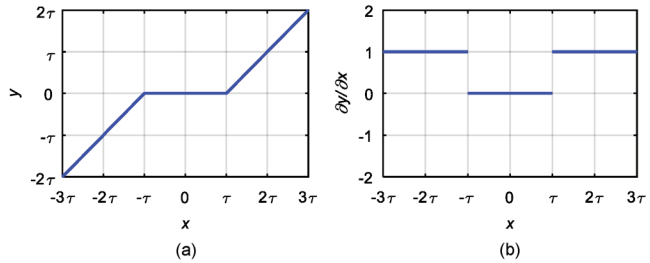


Fig. 3. Illustration of (a) soft thresholding and (b) its derivative.

In classical signal denoising algorithms, it is often difficult to set a proper value to the threshold. Besides, the optimal value varies from case to case. Aiming at this problem, the thresholds used in the developed DRSNs are automatically determined in the deep architectures, in order to avoid the trouble of artificial operation. The ways of determining thresholds in the developed DRSNs are introduced in the subsequent sections.

2) Architecture of the Developed DRSN-CS

The developed DRSN-CS is a variant of ResNet that uses soft thresholding to remove noise-related features. Soft thresholding is inserted as a nonlinear transformation layer into the building unit. Moreover, the value of the threshold can be learned in the building unit, which is introduced below.

As shown in Fig. 4(a), the building unit entitled "residual shrinkage building unit with channel-shared thresholds (RSBU-CS)" is different from the RBU in Fig. 2(a) in that the RSBU-CS has a special module for estimating the threshold to be used in soft thresholding. In the special module, GAP is applied to the absolute values of the feature map x to get a 1D vector. Then, the 1D vector is propagated into a two-layer FC network to obtain a scaling parameter, which is similar to [25].

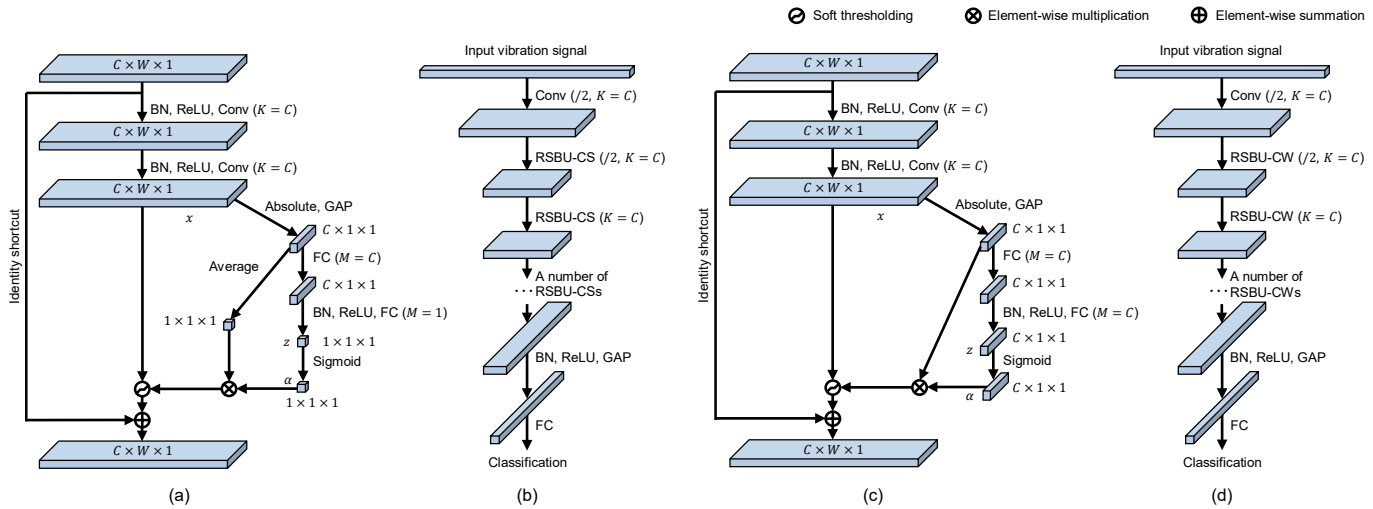


Fig. 4. (a) A building unit entitled RSBU-CS, (b) an overall architecture of DRSN-CS, (c) a building unit entitled RSBU-CW, and (d) an overall architecture of DRSN-CW, where K is the number of convolutional kernels in the convolutional layer; M is the number of neurons in the FC network; and C, W, and 1 in $C \times W \times 1$ are the indicators of the number of channels, width, and height of the feature map, respectively. x, z, and α are the indicators of the feature maps to be used when determining thresholds.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

A sigmoid function is then applied at the end of the two-layer FC network, so that the scaling parameter is scaled to the range of (0, 1), which can be expressed by

$$\alpha = \frac{1}{1 + e^{-z}} \tag{11}$$

where z is the output of the two-layer FC network in the RSBU-CS, and α is the corresponding scaling parameter. After that, the scaling parameter α is multiplied by the average value of $|x|$ to get the threshold. This arrangement is motivated by the fact the threshold for soft thresholding not only needs to be positive, but also cannot be too large. If the threshold is larger than the largest absolute value of the feature map, the output of soft thresholding will be zeros. In summary, the threshold used in the RSBU-CS is expressed by

$$\tau = \alpha \cdot \underset{i,j,c}{\text{average}} |x_{i,j,c}| \tag{12}$$

where τ is the threshold and i, j, and c are the indexes of width, height, and channel of the feature map x, respectively. The thresholds can be kept in a reasonable range, so that the output of soft thresholding will not be all zeros. RSBU-CSs with a stride of 2 and a doubled number of channels can be constructed similarly to the RBUs in Figs. 2(b)-(c).

The brief architecture of the developed DRSN-CS is shown in Fig. 4(b), which is similar to the classical ResNet in Fig. 2(d). The only difference is that the RSBU-CSs are used as the building units instead of the RBUs. A number of RSBU-CSs are stacked in the DRSN-CS, so that the noise-related features can be gradually reduced. Another advantage of the developed DRSN-CS is that the thresholds are automatically learned in the deep architecture rather than manually set by experts, so that professional knowledge of signal processing is not required when implementing the developed DRSN-CS.

3) Architecture of the developed DRSN-CW

The developed DRSN-CW is another variant of ResNet, and is different from the DRSN-CS in that an individual threshold is applied to each channel of the feature map, which is introduced below. A residual shrinkage building unit with channel-wise thresholds (RSBU-CW) is shown in Fig. 4(c). The feature map x is reduced to a 1D vector using an absolute operation and a GAP layer, and then propagated into a two-layer FC network. The second layer in the FC network has more than one neuron, and the number of neurons is equal to the number of channels of the input feature map. The output of the FC network is scaled to the range of (0, 1) using

$$\alpha_c = \frac{1}{1 + e^{-z_c}} \tag{13}$$

where z_c is the feature at the cth neuron, and α_c is the cth scaling parameter. After that, the thresholds are calculated by

$$\tau_c = \alpha_c \cdot \underset{i,j}{\text{average}} |x_{i,j,c}| \tag{14}$$

where τ_c is the threshold for the cth channel of the feature map and i, j, and c are the indexes of width, height, and channel of the feature map x, respectively. Similar to the DRSN-CS, the thresholds can be positive and kept in a reasonable range, thereby preventing the output features from being all zeros.

The overall architecture of the developed DRSN-CW is shown in Fig. 4(d). A number of RSBU-CWs are stacked so that discriminative features can be learned through a variety of nonlinear transformations with soft thresholding as shrinkage functions to eliminate the noise-related information.

III. Experimental Results

The developed DRSNs were implemented using TensorFlow 1.0, which is a machine learning toolkit released by Google that can run on the graphic processing units (GPUs) for acceleration. Experiments were conducted on a computer with an i7-6700 central processing unit and a NVIDIA GeForce GTX 1070 GPU. The experimental results are discussed in this section.

A. Experimental Data Collection

As shown in Fig. 5, a drivetrain diagnostic simulator was used for experimental data collection. The simulator was mainly composed of a motor, a two-stage planetary gearbox, a two-stage fixed-axis gearbox, and a programmable magnetic brake. An acceleration sensor was mounted at the input side of the planetary gearbox. Vibration signals were collected at a sampling frequency of 12800 Hz. As summarized in Table I, eight health conditions in the planetary gearbox were considered in this study, including one healthy condition, three bearing faults, and four gear faults.

For each health condition, three different rotating speeds (20 Hz, 30 Hz, and 40 Hz) and three torsional loads (1 lb·ft, 6 lb·ft, and 18 lb·ft) were considered in the experiments. Under each specific rotating speed and torsional load, 400 observations were collected, so that each health condition had $3 \times 3 \times 400 = 3600$ observations. Each observation was a 0.16-second signal and had 2048 data points. It is notable that such short signals were used to make the fault diagnosis task more challenging, in order to validate the efficacy of the developed DRSNs. In real applications, long signals with more data points can be used. To validate the efficacy of the developed DRSNs in diagnosing machine faults with different background noise, white Gaussian noise, Laplacian noise, and pink noise were added into each signal to yield signal-to-noise ratios (SNRs) from 5 dB to –5 dB, respectively. Specifically, the noise addition was performed on the raw vibration signals. Afterwards, the noised vibration signals were kept unchanged during the optimization of deep learning models. It is also notable that each noise was generated independently, so that the added noises were different for the vibration signals.

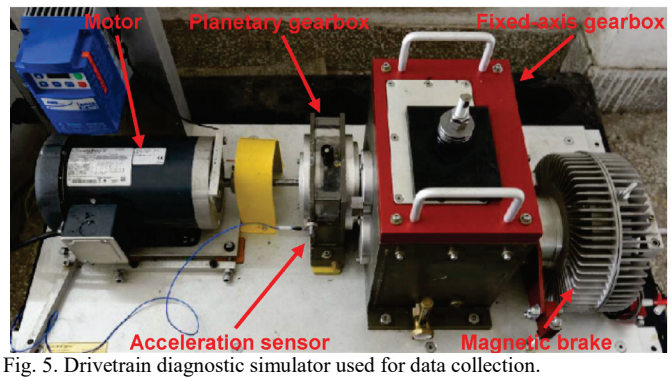


Fig. 5. Drivetrain diagnostic simulator used for data collection.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

TABLE I
SUMMARY OF EIGHT HEALTH STATES OF PLANETARY GEARBOX CONSIDERED IN THE EXPERIMENTS

Category	Description	Label
1	No fault in the bearings and gears	H
2	Inner race fault on a rolling bearing	F1
3	Outer race fault on a rolling bearing	F2
4	Ball fault on a rolling bearing	F3
5	Root crack fault on a gear	F4
6	Surface pitting fault on a gear	F5
7	Tooth broken fault on a gear	F6
8	Tooth missing fault on a gear	F7

B. *Hyperparameter Setup*

Experiments were conducted under a scheme of 10-fold cross-validation. Specifically, the dataset was equally divided into 10 subsets; 1 subset was used as the test set, and the other 9 subsets were used as the training set in each experiment; the experiments were repeated 10 times, so that each subset has a chance to be used as the test set. Furthermore, the initialization and choice of hyperparameters in the developed deep learning methods are introduced clearly as follows.

The architecture-related hyperparameters are used to define the structure of the neural networks, including the number of layers, the number of convolutional kernels, the size of convolutional kernels, and so forth. Because no consensus has been reached as to how to set these hyperparameters, this study sets them according to the popular recommendations [18]-[20]. The architecture-related hyperparameters are summarized in Table II. A CBU refers to a convolutional building unit, which is different from an RBU in that a CBU does not use the identity

TABLE II
ARCHITECTURE-RELATED HYPERPARAMETERS OF THE CONVNET, RESNET, DRSN-CS, AND DRSN-CW IN THE EXPERIMENTS

Number of components	Output size	ConvNet	ResNet	DRSN-CS	DRSN-CW
1	1 × 2048 × 1	Input	Input	Input	Input
1	4 × 1024 × 1	Conv(4, 3, /2)	Conv(4, 3, /2)	Conv(4, 3, /2)	Conv(4, 3, /2)
1	4 × 512 × 1	CBU(4, 3, /2)	RBU(4, 3, /2)	RSBU-CS(4, 3, /2)	RSBU-CW(4, 3, /2)
3	4 × 512 × 1	CBU(4, 3)	RBU(4, 3)	RSBU-CS(4, 3)	RSBU-CW(4, 3)
1	8 × 256 × 1	CBU(8, 3, /2)	RBU(8, 3, /2)	RSBU-CS(8, 3, /2)	RSBU-CW(8, 3, /2)
3	8 × 256 × 1	CBU(8, 3)	RBU(8, 3)	RSBU-CS(8, 3)	RSBU-CW(8, 3)
1	16 × 128 × 1	CBU(16, 3, /2)	RBU(16, 3, /2)	RSBU-CS(16, 3, /2)	RSBU-CW(16, 3, /2)
3	16 × 128 × 1	CBU(16, 3)	RBU(16, 3)	RSBU-CS(16, 3)	RSBU-CW(16, 3)
1	16	BN, ReLU, GAP	BN, ReLU, GAP	BN, ReLU, GAP	BN, ReLU, GAP
1	8	FC	FC	FC	FC

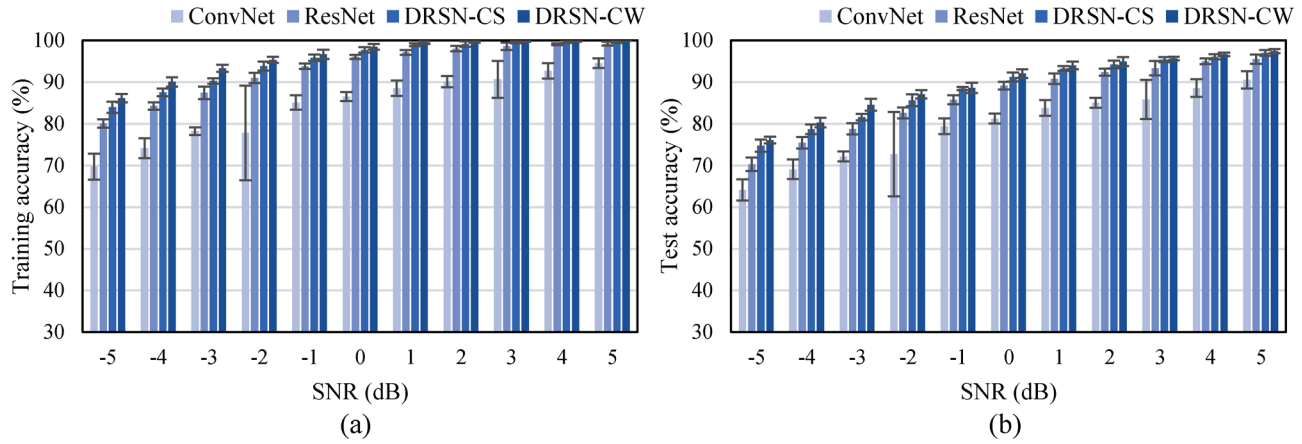


Fig. 6. Training and test accuracies of ConvNet, ResNet, DRSN-CS, and DRSN-CW for fault diagnosis with Gaussian noise.

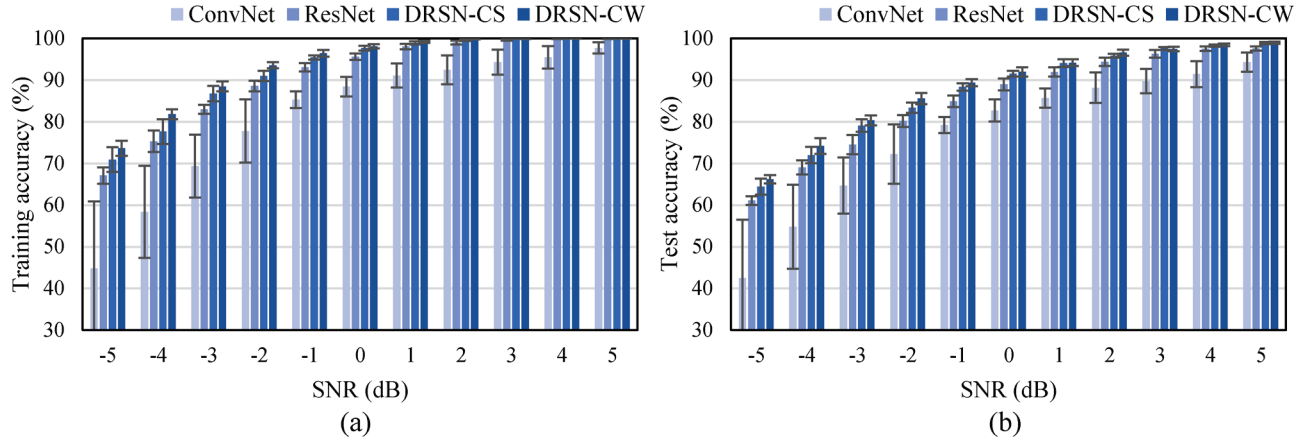


Fig. 7. Training and test accuracies of ConvNet, ResNet, DRSN-CS, and DRSN-CW for fault diagnosis with Laplacian noise.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

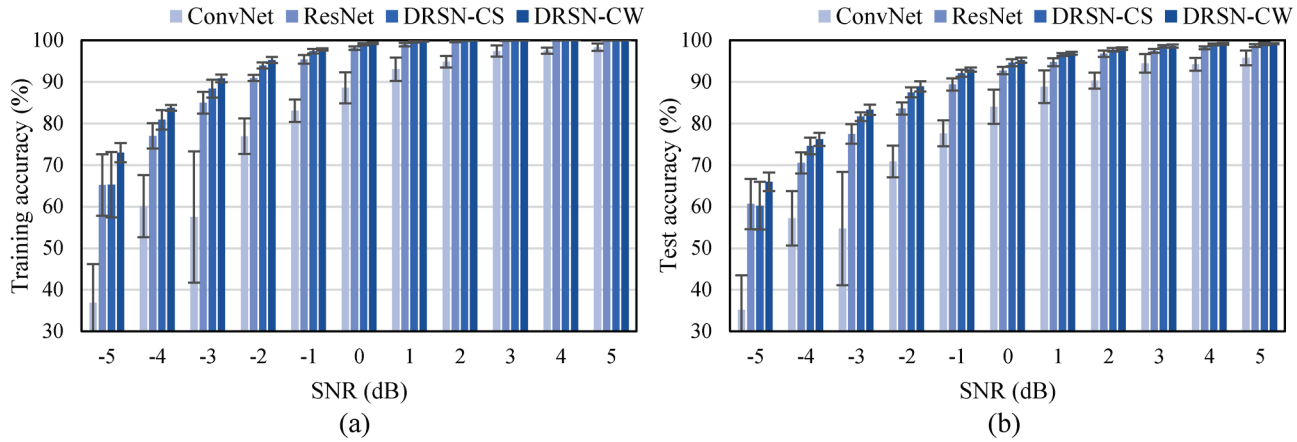


Fig. 8. Training and test accuracies of ConvNet, ResNet, DRSN-CS, and DRSN-CW for fault diagnosis with pink noise.

shortcut. The first and second numbers in the bracket are the number and width of convolutional kernels, respectively. The "/2" in some brackets indicates that the width of the feature map is reduced by moving the convolutional kernels with a stride of 2. The output sizes of the feature maps in different layers are shown in the second column of Table II, which are either in the 3D form of channels × width × height or in the 1D form of a vector. The 3D feature maps are reduced to 1D vectors after the GAP. In the end, the FC output layer has 8 neurons, which equals the number of considered classes (i.e., 1 healthy condition and 7 faulty conditions).

The optimization-related hyperparameters are used to define the training process. The training rate is 0.1 in the first 40 epochs, 0.01 in the subsequent 40 epochs, and 0.001 in the final 20 epochs, so that the parameters can be updated in larger steps at the beginning and slighted updated at the end, which follows the setup in [20]. Momentum is a training strategy to accelerate training using the updates in the previous step. The coefficient of the momentum is set to 0.9 following the recommendation in [15]. L2 regularization is used to reduce the effect of overfitting and yield a higher test accuracy [6]. L2 regularization adds a penalty term in the objective function to push the weights towards zero. In this way, it is unlikely that the absolute values of the weights will be optimized to be very large, and the output of the deep neural networks will not change a lot after multiplying with the weights when dealing with similar inputs. The coefficients of the penalty term are set to 0.0001, which is kept the same as classical ResNets [15]. The mini batch refers to the group of randomly selected observations that are fed into the deep architecture. The time consumption can be reduced compared with the condition in which one observation is fed in each time. The size of the mini batch is set to 128 in order to be consistent with [20].

C. *Experimental Comparison with the ConvNet and ResNet*

The detailed results of the ConvNet, ResNet, DRSN-CS, and DRSN-CW with different types of noise under different SNRs are provided in Figs. 6-8. The average accuracies of the results in Figs. 6-8 are given in Table III. Further, the accuracies without manually added noise are provided in Table IV, and the computational time for the model optimization is summarized in Table V.

TABLE III
AVERAGE ACCURACIES OF THE RESULTS IN FIGS. 6-8 (%)

Method	Training accuracy	Test accuracy
ConvNet	82.06 ± 4.29	77.62 ± 4.22
ResNet	91.96 ± 1.05	86.25 ± 1.34
DRSN-CS	93.61 ± 0.98	88.55 ± 1.02
DRSN-CW	94.84 ± 0.55	89.57 ± 0.87

TABLE IV
AVERAGE ACCURACIES WITHOUT MANUALLY ADDED NOISE (%)

Method	Training accuracy	Test accuracy
ConvNet	99.79 ± 0.86	96.67 ± 1.60
ResNet	99.99 ± 0.01	99.54 ± 0.24
DRSN-CS	100.00 ± 0.01	99.65 ± 0.08
DRSN-CW	100.00 ± 0.00	99.70 ± 0.16

TABLE V
COMPUTATIONAL TIME OF THE CONSIDERED METHODS (SECOND)

Method	Time
ConvNet	961.81
ResNet	978.04
DRSN-CS	1585.33
DRSN-CW	1511.11

A nonlinear unsupervised dimension reduction method, i.e., t-distributed stochastic neighbor embedding [26], is then used to visualize the high-level features at the final GAP layer in 2D spaces. Although the visualization in 2D spaces suffers some errors due to the information loss in dimension reduction, the purpose of 2D visualization is to provide an intuitive idea of whether these high-level features are discriminative or not. As shown in Figs. 9(a)-(b), the testing observations under different health conditions are highly mixed together in the classical ConvNet and ResNet. The observations of some health conditions (e.g., F6) are distributed in a few different areas because the vibration signals are collected under different operating conditions and have different characteristics. The ConvNet and ResNet fail to project them into the same area. In contrast, as shown in Figs. 9(c)-(d), the observations under the same health conditions are mostly grouped in the same area and are also basically separable from the observations under the other health conditions in the DRSN-CS and DRSN-CW.

The training and test errors of the considered deep learning methods are plotted in Fig. 10. Both the training and test errors of the ResNet are obviously lower than the ConvNet, which validates that the use of identity shortcuts can facilitate

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

parameter optimization and result in a more accurate trained model. More significantly, there are lower training and test errors for the developed DRSN-CS and DRSN-CW than the classical ResNet. The reason is that the integration of soft thresholding as shrinkage functions in the deep architectures can reduce the noise-related features, so that the high-level features at the final layer can become more discriminative.

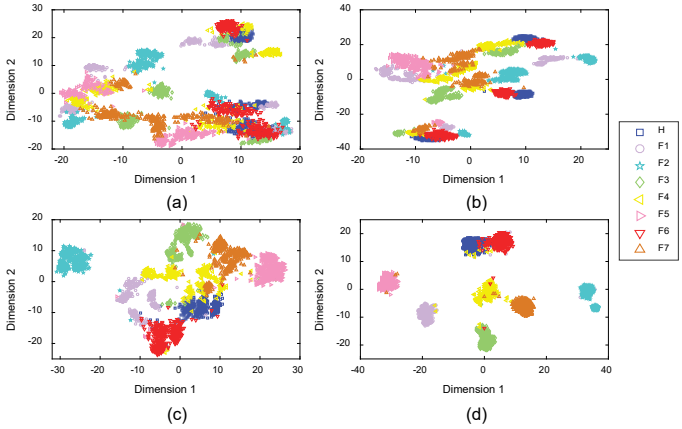


Fig. 9. 2D visualizations of high-dimensional features at the final GAP layer of testing observations in (a) the ConvNet, (b) the ResNet, (c) the DRSN-CS, and (d) the DRSN-CW, when SNR = 5 dB (see Fig. 6).

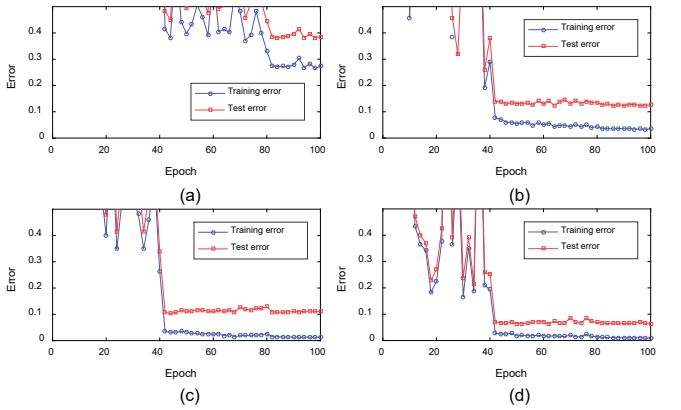


Fig. 10. Training and test errors obtained from (a) the ConvNet, (b) the ResNet, (c) the DRSN-CS, and (d) the DRSN-CW, when SNR = 5dB (see Fig. 6).

D. Comparison between the DRSN-CS and DRSN-CW

As indicated in Table III, the developed DRSN-CW yields improvements of 1.23% and 1.02% in terms of average training and test accuracy, respectively, when compared to the DRSN-CS. As shown in Figs. 9(c)-(d), some pairs of health conditions (e.g., H and F4, H and F6, F4 and F7) suffer severe overlaps in the DRSN-CS, while only one pair of health conditions (i.e., H and F6) has a certain level of overlapping in the DRSN-CW. In other words, the health conditions in the DRSN-CW are more separable than those in the DRSN-CS. Moreover, as shown in Figs. 10(c)-(d), the test errors of the DRSN-CW are reduced to a level (i.e., around 0.07) which is obviously lower than that of DRSN-CS (i.e., around 0.11).

A direct reason for the higher accuracy of the DRSN-CW compared to the DRSN-CS is that different channels of the feature maps often contain different amounts of noise-related information. Accordingly, the developed DRSN-CW can apply different thresholds to shrink the features in different channels of the feature map, and is more flexible than the DRSN-CS, in which a universal threshold is applied to all the channels of the feature map. As a consequence, the developed DRSN-CW can be more effective in eliminating noise-related information and can yield higher accuracy than the DRSN-CS.

The computational time of the DRSN-CS and the DRSN-CW is summarized in Table V. It can be observed that the DRSN-CS takes more time than the DRSN-CW because the DRSN-CS has one more calculation step (i.e., an averaging operation) than the DRSN-CW in each building unit (see Figs. 4a and 4c). In the future, the architectures of the DRSN-CS and the DRSN-CW should be optimized to reduce their computational time.

IV. Conclusions

It is a significant task to improve the feature learning ability of deep learning methods when they are applied to machinery fault diagnosis tasks with highly noised vibration signals. This paper develops two new variants of deep learning methods, i.e., a deep residual shrinkage network with channel-shared thresholds (DRSN-CS) and a deep residual shrinkage network with channel-wise thresholds (DRSN-CW). These methods integrate soft thresholding as trainable shrinkage functions inserted into the deep architectures to enforce the unimportant features to be zeros, so that the learned high-level features can become more discriminative. The thresholds are set using inserted modules (i.e., specially designed sub-networks), so that professional expertise on signal processing is not needed.

The efficacy of the developed DRSNs in improving diagnostic accuracy has been validated through experimental comparisons with the conventional deep learning methods. The developed DRSN-CS and DRSN-CW not only outperformed the classical ConvNet by yielding improvements of 10.93% and 11.95%, respectively, but also outperformed the classical ResNet by yielding improvements of 2.30% and 3.32%, respectively, in terms of average test accuracy under various types and amounts of artificially inserted noise. As a consequence, the integration of soft thresholding as trainable shrinkage functions in deep learning methods can effectively improve the discriminative feature learning ability from highly noised vibration signals.

The developed DRSN-CW method's slight improvement in performance (1.02%) over DRSN-CS, in terms of the overall average test accuracy, is because different channels of a feature map often contain different amounts of noise-related features. Accordingly, the developed DRSN-CW allows each channel of the feature map to have its own threshold, which is more flexible than the DRSN-CS, in which all the channels of a feature map use the same threshold. As a result, the DRSN-CW has a higher feature learning ability and diagnostic performance than the DRSN-CS.

The developed DRSNs are not only applicable to fault diagnosis tasks using vibration signals, but also to pattern recognition tasks in a variety of fields when dealing with various kinds of signals that interfere with noise, such as acoustic signals, visual signals, and current signals.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

REFERENCES

[1] X. Jin, F. Cheng, Y. Peng, W. Qiao, and L. Qu, "Drivetrain gearbox fault diagnosis: Vibration- and current-based approaches," *IEEE Ind. Electron. Mag.*, vol. 24, no. 6, pp. 56–66, 2018.

[2] Y. Wang, P.W. Tse, B. Tang, Y. Qin, L. Deng, and T. Huang, "Kurtogram manifold learning and its application to rolling bearing weak signal detection," *Measurement*, vol. 127, pp. 533–545, 2018.

[3] F. Cong, J. Chen, G. Dong, and M. Pecht, "Vibration model of rolling element bearings in a rotor-bearing system for fault diagnosis," *J. Sound Vib.*, vol. 332, no. 8, pp. 2081–2097, 2013.

[4] Y. Lei, J. Lin, M. J. Zuo, and Z. He, "Condition monitoring and fault diagnosis of planetary gearboxes: A review," *Measurement*, vol. 48, pp. 292–305, 2014.

[5] R. Liu, B. Yang, E. Zio, and X. Chen, "Artificial intelligence for fault diagnosis of rotating machinery: A review," *Mech. Syst. Signal Process.*, vol. 108, pp. 33–47, 2018.

[6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[7] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-D convolutional neural networks," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7067–7075, 2016.

[8] W. Sun, R. Zhao, R. Yan, S. Shao, and X. Chen, "Convolutional discriminative feature learning for induction motor fault diagnosis," *IEEE Trans. Ind. Inform.*, vol. 13, no. 3, pp. 1350–1359, 2017.

[9] H. Shao, H. Jiang, H. Zhang, and T. Liang, "Electric locomotive bearing fault diagnosis using a novel convolutional deep belief network," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2727–2736, 2018.

[10] R. Liu, G. Meng, B. Yang, C. Sun, and X. Chen, "Dislocated time series convolutional neural architecture: An intelligent fault diagnosis approach for electric machine," *IEEE Trans. Ind. Inform.*, vol. 13, no. 3, pp. 1310–1320, 2017.

[11] C. Sun, M. Ma, Z. Zhao, and X. Chen, "Sparse deep stacking network for fault diagnosis of motor," *IEEE Trans. Ind. Inform.*, vol. 14, no. 7, pp. 3261–3270, 2018.

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

[13] R. Chen, X. Huang, L. Yang, X. Xu, X. Zhang, and Y. Zhang, "Intelligent fault diagnosis method of planetary gearboxes based on convolution neural network and discrete wavelet transform," *Comput. Ind.*, vol. 106, pp. 48–59, 2019.

[14] Y. Han, B. Tang, and L. Deng, "An enhanced convolutional neural network with enlarged receptive fields for fault diagnosis of planetary gearboxes," *Comput. Ind.*, vol. 107, pp. 50–58, 2019.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Seattle, WA, USA, Jun. 27–30, 2016, pp. 770–778.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision—ECCV 2016* (Lecture Notes in Computer Science 9908), B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham, Switzerland: Springer, 2016, pp. 630–645.

[17] S. Ma, F. Chu, and Q. Han, "Deep residual learning with demodulated time-frequency features for fault diagnosis of planetary gearbox under nonstationary running conditions," *Mech. Syst. Signal Process.*, vol. 127, pp. 190–201, 2019.

[18] M. Zhao, M. Kang, B. Tang, and M. Pecht, "Multiple wavelet coefficients fusion in deep residual networks for fault diagnosis," *IEEE Trans. Ind. Electron.*, vol. 66, no. 6, pp. 4696–4706, 2019.

[19] W. Zhang, X. Li, and Q. Ding, "Deep residual learning-based fault diagnosis method for rotating machinery," *ISA Transactions*, to be published.

[20] M. Zhao, M. Kang, B. Tang, and M. Pecht, "Deep residual networks with dynamically weighted wavelet coefficients for fault diagnosis of planetary gearboxes," *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4290–4300, 2018.

[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, Lille, France, Jul. 7–9, 2015, pp. 448–456.

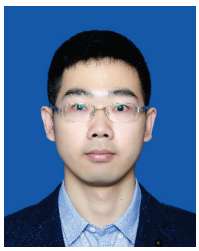
[22] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Represen.*, Banff, Canada, Apr. 14–16, 2014.

[23] D. L. Donoho, "De-noising by soft-thresholding," *IEEE Trans. Inf. Theory*, vol. 41, no. 3, pp. 613–627, 1995.

[24] K. Isogawa, T. Ida, T. Shiodera, and T. Takeguchi, "Deep shrinkage convolutional neural network for adaptive noise reduction," *IEEE Signal Process. Lett.*, vol. 25, no. 2, pp. 224–228, 2018.

[25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 18–23, 2018, pp. 7132–7141.

[26] L. J. P. van der Maaten and G. E. Hinton, "Visualizing high-dimensional data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.



Minghang Zhao was born in Shandong, China, in June 1991. He received the B.E. and Ph.D. degrees in mechanical engineering from Chongqing University, Chongqing, China, in 2013 and 2018, respectively. He is currently a lecturer with the School of Naval Architecture and Ocean Engineering, Harbin Institute of Technology at Weihai, Weihai, China.

He was previously a visiting research scholar with the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD, USA, from 2016 to 2017. His research interests include machine learning (especially deep neural networks) powered fault diagnosis, prognostics, and health management of mechanical and electrical systems.

Shisheng Zhong received the M.E. degree in mechanical engineering from Harbin Institute of Technology, Harbin, China, and the Ph.D. degree in mechanical engineering from Huazhong University of Science and Technology, Wuhan, China, in 1992 and 1995, respectively.

He is currently a Professor and Ph.D. Supervisor of mechanical engineering with the School of Naval Architecture and Ocean Engineering, Harbin Institute of Technology at Weihai, Weihai, China. His main research interests include intelligent manufacturing, prognostics and health management, and maintenance, repair, and overhaul.

Dr. Zhong is also a vice chairman of the Machinery Industry Automation Branch of the Chinese Mechanical Engineering Society.

Xuyun Fu received the B.E., M.E., and Ph.D. degrees in mechanical engineering from Harbin Institute of Technology, Harbin, China, in 2003, 2007, and 2010, respectively.

He is currently an Associate Professor of mechanical engineering with the School of Naval Architecture and Ocean Engineering, Harbin Institute of Technology at Weihai, Weihai, China. His main research interests include intelligent operation and maintenance, prognostics and health management, aeroengine health management and maintenance decision.

Baoping Tang received the M.Sc. and Ph.D. degrees in mechanical engineering from Chongqing University, Chongqing, China, in 1996 and 2003, respectively.

He is currently a Professor and Ph.D. Supervisor with the College of Mechanical Engineering, Chongqing University, Chongqing, China. More than 150 papers has been published in his research career. His main research interests include wireless sensor networks, mechanical and electrical equipment security service and life prediction, and measurement technology and instruments.

Dr. Tang was the recipient of the National Scientific and Technological Progress 2nd Prize of China in 2004 and the National Invention 2nd Prize of China in 2015.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2943898, IEEE Transactions on Industrial Informatics

Michael Pecht (S'78–M'83–SM'90–F'92) received the B.S. degree in acoustics, M.S. degree in electrical engineering and engineering mechanics, and Ph.D. degree in engineering mechanics from the University of Wisconsin at Madison, Madison, WI, USA, in 1982.

He is the Founder of the Center for Advanced Life Cycle Engineering, University of Maryland, College Park, MD, USA, where he is also a Chair Professor. He has been leading a research team in the area of prognostics.

Dr. Pecht is a Professional Engineer and a Fellow of the American Society of Mechanical Engineers. He was the recipient of the IEEE Undergraduate Teaching Award and the International Microelectronics Assembly and Packaging Society William D. Ashman Memorial Achievement Award for his contributions in electronics reliability analysis. He served as the Chief Editor of the IEEE TRANSACTIONS ON RELIABILITY for eight years and an Associate Editor for the IEEE TRANSACTIONS ON COMPONENTS AND PACKAGING TECHNOLOGY.