Hindawi
Complexity
Volume 2020, Article ID 9518945, 18 pages
https://doi.org/10.1155/2020/9518945

WILEY | Hindawi

Research Article

Intelligent Defect Identification Based on PECT Signals and an Optimized Two-Dimensional Deep Convolutional Network

Baoling Liu,[1] **Jun He**,[2] **Xiaocui Yuan**,[1] **Huiling Hu**,[1] **Xuan Zeng**,[1] **Zhifang Zhu**,[1] **and Jie Peng**[1]

[1]*Jiangxi Engineering Research Center of High Power Electronics and Grid Smart Metering, Nanchang Institute of Technology, Nanchang, China*
[2]*State Grid Jiangxi Electric Power Research Institute, Beijing, China*

Correspondence should be addressed to Baoling Liu; bl_liu2009@hotmail.com

Received 13 July 2020; Revised 2 September 2020; Accepted 9 November 2020; Published 24 November 2020

Academic Editor: Kailong Liu

Accurate and rapid defect identification based on pulsed eddy current testing (PECT) plays an important role in the structural integrity and health monitoring (SIHM) of in-service equipment in the renewable energy system. However, in conventional data-driven defect identification methods, the signal feature extraction is time consuming and requires expert experience. To avoid the difficulty of manual feature extraction and overcome the shortcomings of the classic deep convolutional network (DCNN), such as large memory and high computational cost, an intelligent defect recognition pipeline based on the general Warblet transform (GWT) method and optimized two-dimensional (2-D) DCNN is proposed. The GWT method is used to convert the one-dimensional (1-D) PECT signal to a 2D grayscale image used as the input of 2D DCNN. A compound method is proposed to optimize the baseline VGG16, a well-known DCNN, from four aspects including reducing the input size, adding batch normalization layer (BN) after every convolutional layer(Conv) and fully connection layer (FC), simplifying the FCs, and removing unimportant filters in Convs so as to reduce memory and computational costs while improving accuracy. Through a pulsed eddy current testing (PECT) experiment considering interference factors including liftoff and noise, the following conclusion can be obtained. The time-frequency representation (TFR) obtained by the GWT method not only has excellent ability in terms of the transient component analysis but also is less affected by the reduction of image size; the proposed optimized DCNN can accurately identify defect types without manual feature extraction. And compared to the baseline VGG16, the accuracy obtained by the optimized DCNN is improved by 7%, to about 99.58%, and the memory and computational cost are reduced by 98%. Moreover, compared with other well-known DCNNs, such as GoogLeNet, Inception V3, ResNet50, and AlexNet, the optimized network has significant advantages in terms of accuracy and computational cost, too.

1. Introduction

Renewable energy source such as wind and tides are gradually replacing coal and oil as new energy sources because of their inexhaustible and pollution-free advantages. The large in-service equipment applied in the renewable energy system such as supports and pipes often suffers from various defects due to the harsh working environment, which is dangerous for the safe operation of the renewable energy system. So, it has important practical significance to explore an accurate, fast, and concise intelligent defect identification method applying the latest achievements of artificial intelligence into the time-series signal that are widely used in the field of renewable energy source, for example, pulsed eddy current testing, ultrasound testing, guided waves testing, and magnetic flux leakage testing.

With the advantages of excellent detection ability for surface and internal defect, a simple mechanism, and a low price, pulsed eddy current testing- (PECT-) based defect identification has always been a research hotspot in the field of nondestructive testing [1]. Defect identification methods using PECT can be classified as model-based and data-driven. Model-based methods usually establish a suitable electromagnetic model to analyze the defects, which is generally complicated and time consuming [2, 3]. Data-driven methods can build defect identification models with

the help of historical inspection data, which is suitable to complex systems for which it is difficult to establish accurate models through mechanism analysis [4, 5]. In recent years, the popularization of intelligent manufacturing and the development of information technology have provided new opportunities for data-driven defect identification methods and predicting remaining useful life [6, 7].

Conventional data-driven defect recognition includes the two steps of manual feature extraction and defect pattern recognition [8]. PECT signal feature extracted can be roughly divided into time-domain features, statistical features, frequency domain, and time-frequency domain features. Commonly used time-domain features, such as signal peaks, peak rising times, and zero-crossing times of differential signals, are simple but susceptible to noise and liftoff interference. Frequency-domain features, such as the frequency spectrum separating point [9] and specific frequency components or frequency bands [10], as extracted from the amplitude spectrum of a fast Fourier transform (FFT), often vary with the signal and are susceptible to noise. Statistical analysis methods, such as principal component analysis (PCA) [11] and independent component analysis (ICA) [12], have been applied to extract more effective time- or frequency-domain features. Time-frequency domain features generally adopt time-frequency analysis (TFA) methods, such as wavelet transform [13] and ensemble empirical mode decomposition (EEMD) [14], to transform a PECT signal to a two-dimensional space and then use such methods as PCA to extract a few components to form a feature vector. However, the manual feature extraction often requires great familiarity with the detection objects and signals and takes much trial and error to match the appropriate feature vector. In addition, the inevitable denoising before feature extraction would cause information loss. So, it is really a difficult task. The following pattern recognition is much simpler task. Machine learning methods, such as neural networks [11] and support vector machine [15], are used to establish the mapping relationship between features and defect patterns.

In recent years, with the development of artificial intelligence and machine learning, the deep convolutional neural network (DCNN) has become an important adaptive signal processing method [16]. Classical DCNN architectures consist of a series of convolutional layers (Convs) containing multiple filters, pooling layers, a rectified linear unit (ReLU), and fully connected layers (FCs) [17], where the Convs can automatically perform feature extraction through convolution operation, and the pattern recognition can completed by the FCs. So, DCNN can directly realize pattern recognition from a two-dimensional (2D) input signal and avoid the difficulties of manual feature extraction. Some scholars have carried out research on applying DCNN to PECT. For example, Cheon et al. adopted a single convolutional neural network (CNN) model to extract effective features for defect classification [18]. Saeed et al. adopted a pretrained DCNN and transfer learning to recognize artificial subsurface defects in a carbon fiber reinforced polymer (CFRP) sample, where the defect (i.e., input) signal was a 2D thermogram obtained from a pulsed-thermography setup [19]. Zhu et al. used a CNN to form a pattern recognizer to improve the identification accuracy of heat exchange tube defects using eddy current testing [20]. However, there is little research on how to directly realize the intelligent defect identification from a 1-D PECT signal with the application of DCNN. There are two possible reasons for this. First, PECT signal is 1D, while DCNN usually adopts a 2D convolutional layer for good feature extraction, which requires a 2D input [21]. Second, the parameters and the intermediate variables of DCNN need too much memory and computational effort [22], which restricts the application of DCNN to nondestructive testing, including PECT, because the inspection and maintenance of large-scale in-service equipment in the field of renewable energy source is often carried out using portable NDT equipment within a given time, and small memory and fast speed are as important as the accuracy for the signal processing methods.

Some scholars have carried out research on technical pipeline based on time-domain signals and DCNN in the field of bearing fault diagnosis. Time-frequency analysis (TFA) methods, such as the short-time Fourier transform (STFT) [23], synchrosqueezed transform (SST) [24], Wigner–Ville distribution (WVD) [25], and ensemble empirical mode decomposition (EEMD) [26], were used to transform time-series signals into 2D time-frequency representations (TFR) as the input of 2D DCNN. The above-mentioned TFR methods cannot adaptively configure important parameters according to signal characteristics so that the obtained TFRs have a certain degree of distortions, which will inevitably affect the accuracy of subsequent pattern recognition. Peng et al. proposed the parameterized time-frequency analysis method which can adaptively match the appropriate parameters of TFA methods for the signal model through a matching kernel function [27]. This method can more accurately express the time-frequency characteristics of transient signals [28, 29] and has been successfully used in low-signal-to-noise ratio, multicomponent, and weak-signal processing methods, such as micro-Doppler [30] and multiradar signals [31]. However, there is little research on its application to PECT.

On the contrary, in terms of reducing memory and computational costs of DCNN, scientists have conducted valuable research. For example, Li and Frankle et al. used the $L1$- and $L2$-norm to rank the importance of filters in the convolutional layers (Convs) and removed the unimportant ones [32, 33]. Liu et al. introduced a scaling factor for each filter (channel), which could automatically identify insignificant channels, and pruned afterwards [34]. Molchanov et al. treated network pruning as an optimization problem and proposed a criterion based on a Taylor expansion that approximated the change in the cost function [35]. Current research on network optimization methods is carried out on large databases such as Cifar and ImageNet, which usually contain thousands of image data. However, as far as nondestructive testing is concerned, it is difficult to build a large damage database even with the wide application of big data because defects are extremely rare in normally operating equipment. Moreover, in addition to reducing the parameters of the Convs and the FCs, how to reduce the

intermediate variables of the DCNN without causing the deteriorating accuracy is also worthy of further study.

In order to avoid the difficulty of manual feature extraction and overcome the shortcomings of DCNN that require large memory and computational cost, we propose a defect recognition pipeline based on PECT signal and an optimized DCNN to intelligently, quickly, and accurately identify defect. In the pipeline, the general Warblet transform (GWT) method, a kind of parameterized time-frequency method, is proposed to transform the PECT signal into a 2D TFR as the input of DCNN. A novel compound method is proposed to optimize VGG16 baseline architecture to obtain optimized DCNN which is used for feature extraction and pattern recognition of 2D TFRs. The pipeline was verified by PECT experiment considering such interference as lift-off and noise in terms of accuracy and computational cost.

The remainder of this article is organized as follows. Section 2 briefly introduces the self-made PECT equipment and the dataset used to verify the pipeline, which is introduced in Section 3. The pipeline is verified and analysed in Section 4. We discuss our conclusions in Section 5.

2. Self-Made PECT Equipment and Defects

2.1. Self-Made PECT Equipment. PECT technology uses pulse excitation with a certain duty cycle, which can be treated theoretically as a superposition of a series of harmonic components. Thus, PECT technology can potentially obtain better sensitivity for both surface and internal defect than conventional eddy current testing excited by a harmonic signal [36].

The data were obtained by the self-made PECT equipment, as shown in Figure 1, which consists mainly of the following: (a) PECT probe; (b) excitation signal generator; (c) amplifier and low-pass filtering; (d) data-acquisition card (DAC); (e) power module; and (f) detection signal. The PECT probe consists of a ferrite core, detection coil, and driver coil. The ferrite core concentrates more dense magnetic lines around the probe for better sensitivity and deeper detection. The detection and driver coils are coaxially wound with a ferrite core. The detection coil is inside and is wound with 1000 turns of copper wire. The driver coil has 500 turns of copper wire, is outside, and is excited by a square pulse with a frequency of 100 Hz, amplitude of 5 V, and duty cycle of 50%.

The pulse excitation signal is generated by the STM32 microcontroller through internal triggering. The excitation coil is excited by the pulse signal and induces a transient eddy current in the conductor specimen according to the Maxwell equations. Subsequently, the changing eddy current field generates an induced magnetic field above the test piece, which is picked up by the detection coil as a voltage signal. After amplification and low-pass filtering, the voltage signal enters the DAC with a sampling rate of 500 k/s. The power module can provide power to each part. The display and save function of the detection signal is programmed by LABVIEW software. And the detection signal, as shown in Figure 1(f), is a 1D time-series signal.

2.2. Introduction of Defect. A specimen with three kinds of artificial defects, i.e., surface defect, internal defect, and hidden defect, was used to verify the proposed defect identification pipeline, as shown in Figure 2. The two simulated surface defects were shown in Figure 2(a), where the first one was a crack with length 15 mm, width 1 mm, and depth 3 mm. The second one consisted of two cracks like the first one at a distance of 2 mm. Two simulated internal defects were two holes with diameters of 3 mm and 5 mm, respectively, and 2 mm distance from the surface, as shown in Figure 2(b). The artificial hidden defect consisting of a crack and hole in the same vertical position was used to simulate a very dangerous situation in which a serious internal defect was hidden by a slight surface defect. The sizes of crack and hole were shown in Figure 2(c). It is worth noting that the real internal defects are different from those shown in Figure 2. However, considering the difficulties in processing hidden internal defects, holes are often used to simulate various internal defects for research work and employee training.

In the experiment, the probe stood upright to collect data. The liftoff (the distance between the probe end face and the specimen surface) was set to 0.5 mm. The liftoff is a common interference during PECT [2]. To verify the robustness of the proposed method against this interference, the signal of #2 Crack with a liftoff of 1 mm was extracted. PECT signals were transformed to 2D TFRs through time-frequency analysis (TFA) and converted to grayscale images to form a dataset. Salt-and-pepper noise with amplitude 0.02 was added to imitate signals contaminated by noise that is another common interference [37]. Finally, 596 signals, including 298 surface defects, 198 internal defects, and 100 hidden defects, were obtained. Sixty percent of the dataset was randomly selected as the training set, and the remaining 40 percent was the test set. Table 1 displays information on the training and test sets.

3. Method

The proposed intelligent defect identification pipeline consists mainly of input signal processing module and network optimization module based on VGG16, as shown in Figure 3. In the input signal processing module, the GWT method transforms the 1D PECT signals to 2D TFRs, which are converted to grayscale images. Size compression is performed to save memory and increase the processing speed. In the network optimization module the VGG16 baseline architecture is improved from four aspects including reducing the input size, adding BNs, simplifying the FCs, and removing unimportant filters in Convs so as to reduce memory and computational costs while improving accuracy. In this section, the pipeline will be introduced.

3.1. Time Series Signal Processing

3.1.1. 2D TFR of PECT Signal Based on GWT Method. We propose to use the GWT method to obtain the TFR of PECT for two reasons. First, as a parameterized time-frequency analysis method, GWT constructs a matching kernel

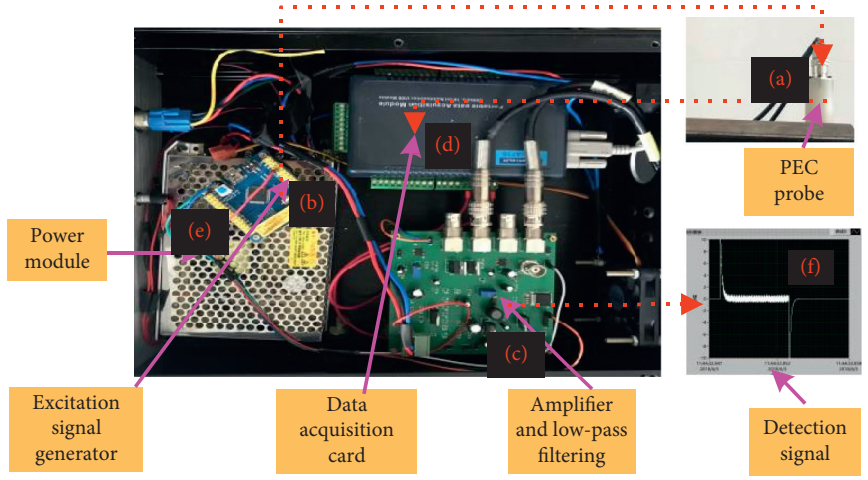


Figure 1: Overview of self-made experimental equipment: (a) PEC probe; (b) excitation signal generator; (c) amplifier and low-pass filtering; (d) DAC; (e) power module; (f) detection signal.

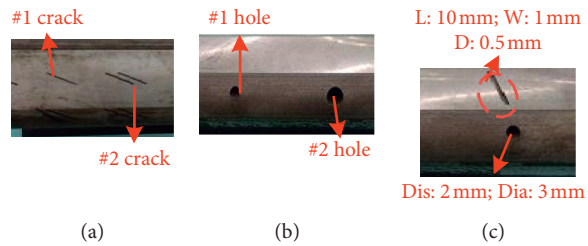


Figure 2: Diagram of three kinds of simulated defects: (a) surface defects; (b) internal defects; (c) hidden defect.

Table 1: Information on training and test sets.

	Surface defect			Internal defect		Hidden defect
	#1 crack	#2 crack	#2 crack (1 mm liftoff)	#1 hole	#2 hole	
Measured data	50	50	49	50	49	50
Noised data	50	50	49	50	49	50
Training set	180			120		60
Test set	118			78		40

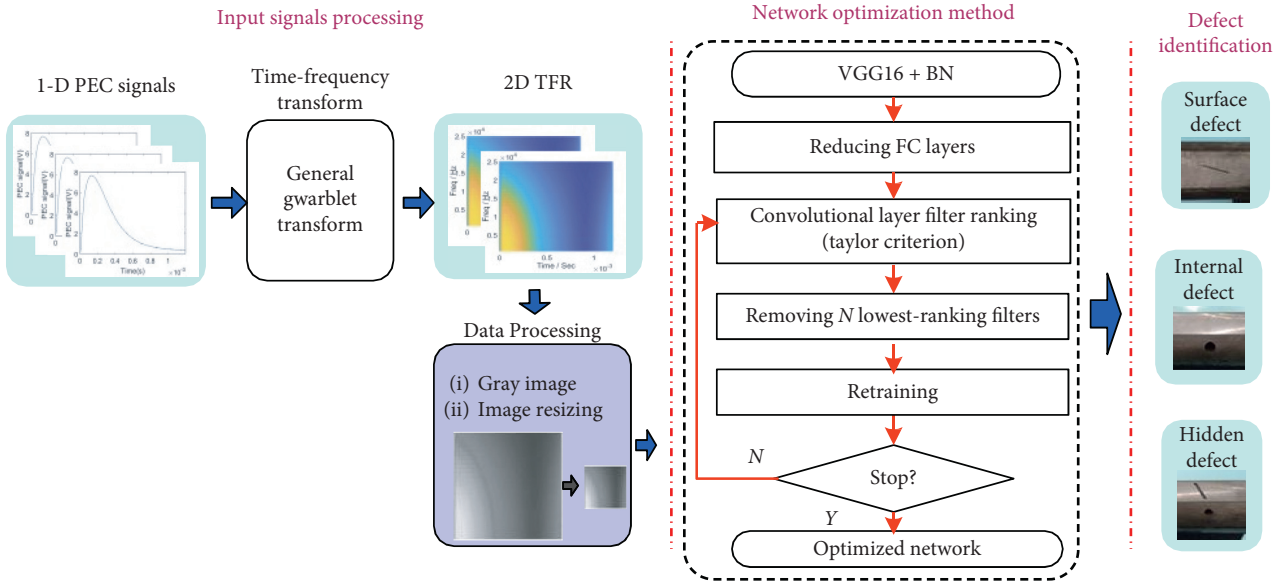


Figure 3: Technical pipeline of the proposed method.

function to adaptively match the appropriate parameters for the signal model, so the obtained 2-D TFR has excellent time-frequency resolution and energy concentration. Second, the PECT signal can theoretically be regarded as the superposition of a series of harmonic signals characterized by Fourier series, while the kernel function of the GWT method is also constructed by Fourier series. Thus, GWT can better approximate the PECT signal so as to accurately characterize the instantaneous frequency components contained in the PECT signal.

The GWT method can be defined as [38]

$$\mathrm{GWT}(t_0,\alpha,\beta,f,\omega;\sigma)=\int_{-\infty}^{+\infty}\overline{z}(\tau)g_\sigma^*(\tau-t_0)e^{-j\omega\tau}\mathrm{d}\tau,$$

$$\begin{cases}\overline{z}(\tau)=z(\tau)\Phi^R(\tau,\alpha,\beta,f)\Phi^R(\tau,t_0,\alpha,\beta,f),\\ \Phi^R(\tau,\alpha,\beta,f)=\exp\left[-j\left(\sum_{i=1}^{m}\frac{a_i}{f_i}\cos 2\pi f\tau_i+\sum_{i=1}^{m}\frac{\beta_i}{f_i}\sin 2\pi f_i\tau\right)\right],\\ \Phi^S(\tau,t_0,\alpha,\beta,f)=\exp\left[j2\pi\left(-\sum_{i=1}^{m}a_i\sin 2\pi f_it_0+\sum_{i=1}^{m}\beta_i\cos 2\pi f_it_0\right)\tau\right],\end{cases}\tag{1}$$

where $z(\tau)$ is a PECT signal. $g_\sigma^*(\tau-t_0)$ is a window function, and here, the Gaussian window function is adopted. $\Phi^R(\tau,\alpha,\beta,f)$ and $\Phi^S(\tau,t_0,\alpha,\beta,f)$ are, respectively, rotation and shifting operators that use Fourier kernel functions. m is the number of sine and cosine functions, and $\{f_1,f_2,\ldots,f_m\}$ is the corresponding harmonic frequency. $\{\alpha_1,\alpha_2,\ldots,\alpha_m\}$ and $\{\beta_1,\beta_2,\ldots,\beta_m\}$ are the undetermined coefficients of the kernel function.

Only a proper kernel function can make the GWT method obtain TFR with good energy concentration and a clear instantaneous frequency. We estimate the coefficients of the kernel function as follows [38].

Step 1: assume $\{\alpha_1,\alpha_2,\ldots,\alpha_m\}=\{0,0,\ldots,0\}_{1\times m}$ and $\{\beta_1,\beta_2,\ldots,\beta_m\}=\{0,0,\ldots,0\}_{1\times m}$, and bring them into equation (1) to obtain the initial TFR of the PECT signal.

Step 2: extract the location of the local maximum energy in TFR as the estimated instantaneous frequency $F_i(t)$.

Step 3: calculate the Fourier transform $F(n\omega_0)$ and Fourier coefficients $\{\alpha_1,\alpha_2,\ldots,\alpha_m\}$ and $\{\beta_1,\beta_2,\ldots,\beta_m\}$:

$$F(n\omega_0)=\sum_{\tau=0}^{\infty}F_i(\tau)\exp(-jn\omega_0\tau),\tag{2}$$

$$F(n\omega_0)=\begin{cases}\frac{1}{2}(\alpha_n+j\beta_n), & n<0,\\ \frac{1}{2}\alpha_0, & n=0,\\ \frac{1}{2}(\alpha_n-j\beta_n), & n>0.\end{cases}\tag{3}$$

Step 4: bring the Fourier coefficients into equation (1) to get a new TFR, and repeat steps (2) and (3) until the difference between the instantaneous frequencies obtained in two consecutive iterations is less than the preset threshold δ, at which time the iteration is terminated.

The TFR of a PECT signal from 1# crack obtained by the GWT method is shown in Figure 4. Considering that the PECT signal only responds at the moment when the pulse is triggered and then quickly decays, the TFA method is only performed on the signal segment at the moment of triggering.

To verify the effect of GWT, four other TFA methods were used to process the same signal. These are the short-time Fourier transform (STFT), ensemble empirical mode decomposition (EEMD), synchrosqueezed transform (SST), and smooth pseudo-Wigner-Ville distribution (PSWVD). The obtained TFRs and corresponding grayscale images are shown in Figures 5–8.

It can be seen from the figures that the TFRs obtained by GWT, STFT, and PSWVD are denser time-frequency spectra, while the TFRs of EEMD and SST only indicate a few time-frequency components. The spectra obtained by GWT and PSWVD have clearer profiles than STFT, which shows that these two TFA methods perform better in terms of instantaneous frequency analysis. However, the two methods produce peak-shaped distortions at the initial stage. This is because the PECT signal includes rich transient components at the moment of excitation, which inevitably causes cross-interference during the TFA process. GWT produces less distortion than PSWVD because of its better transient recognition ability.

3.1.2. Information Loss as Reducing Image Size. We will use the image entropy to evaluate the information loss generated when the TFRs obtained by the above methods are reduced in size.

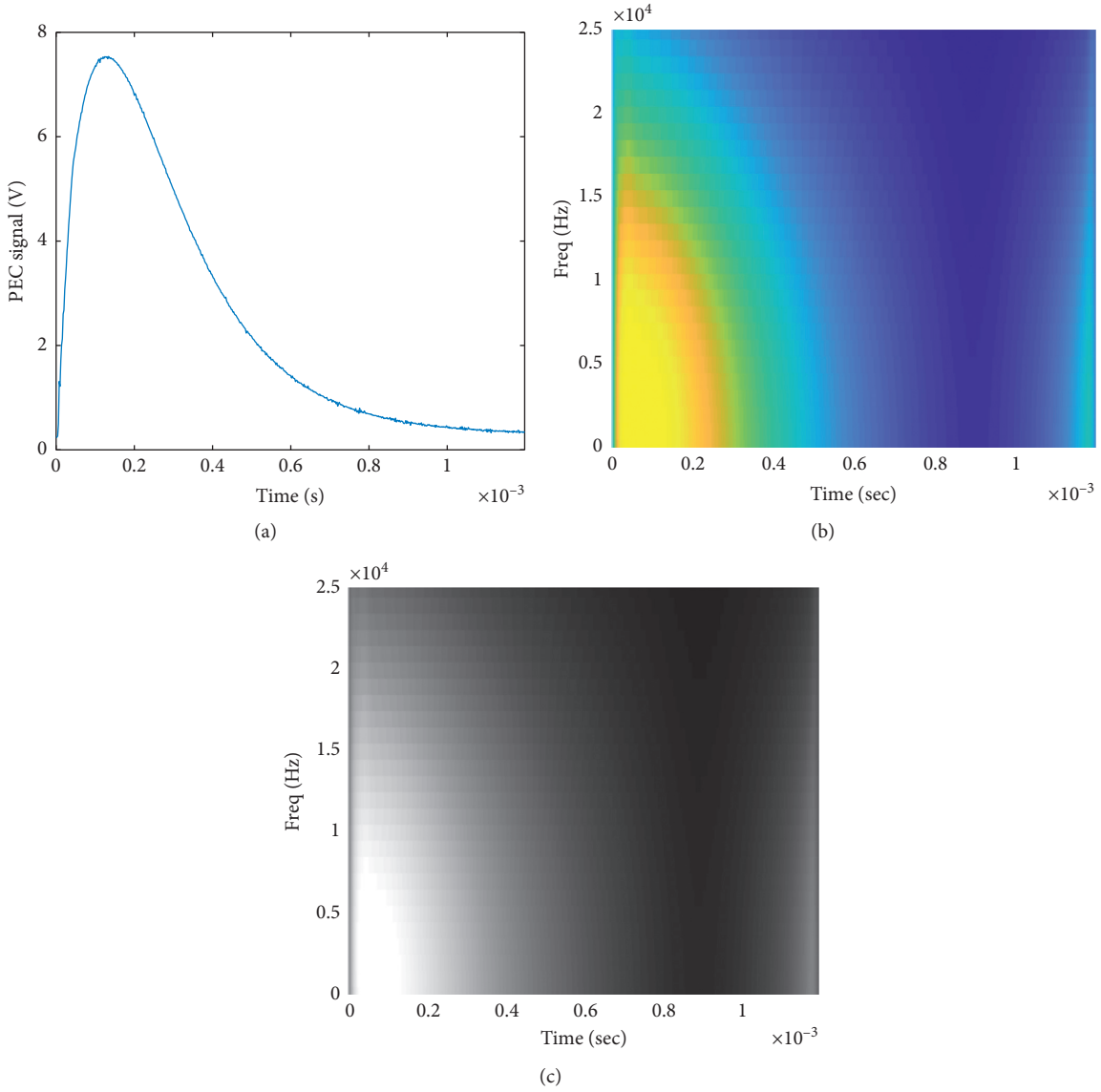


Figure 4: TFR obtained using the GWT method: (a) PEC signal; (b) TFR; (c) gray image.

Shannon introduced the concept of entropy to information theory to measure information [39]. In image processing, the 1D image entropy E may indicate the amount of information contained in an image through the aggregation characteristics of the gray distribution in the image, and this can be calculated as follows:

$$E = -\sum_{i=0}^{255} p_i \log p_i, \tag{4}$$

where p_i is the probability of the ith gray value appearing in the image, which can be obtained from the gray histogram.

Table 2 shows the image entropies and changes corresponding to several TFRs when the size is reduced from 224×224 to 32×32.

The values shown in Table 2 are in accordance with Figures 4–8. The grayscale images obtained by STFT, PSWVD, and GWT contain more information; hence, their information entropies are larger. TFRs obtained by EEMD and SST have only a few curves representing the frequency components, so their information entropies are smaller. When the signal is reduced from 224×224 to 32×32, it can be seen from Table 2 that the changes of information entropies of the first three are relatively small, such as 1.53% for the GWT method used in this article. However, the information entropies of EEMD and SST increase significantly. For example, the change of information entropy for EEMD reaches 168.39%, which means that the distribution of image information changes greatly, which may have a significant impact on defect recognition.

3.2. Network Optimization Based on VGG16

3.2.1. VGGNet. VGGNet was developed by the Visual Geometry Group of Oxford University and Google DeepMind in 2014 [40]. So far, VGGNet is still the most commonly used

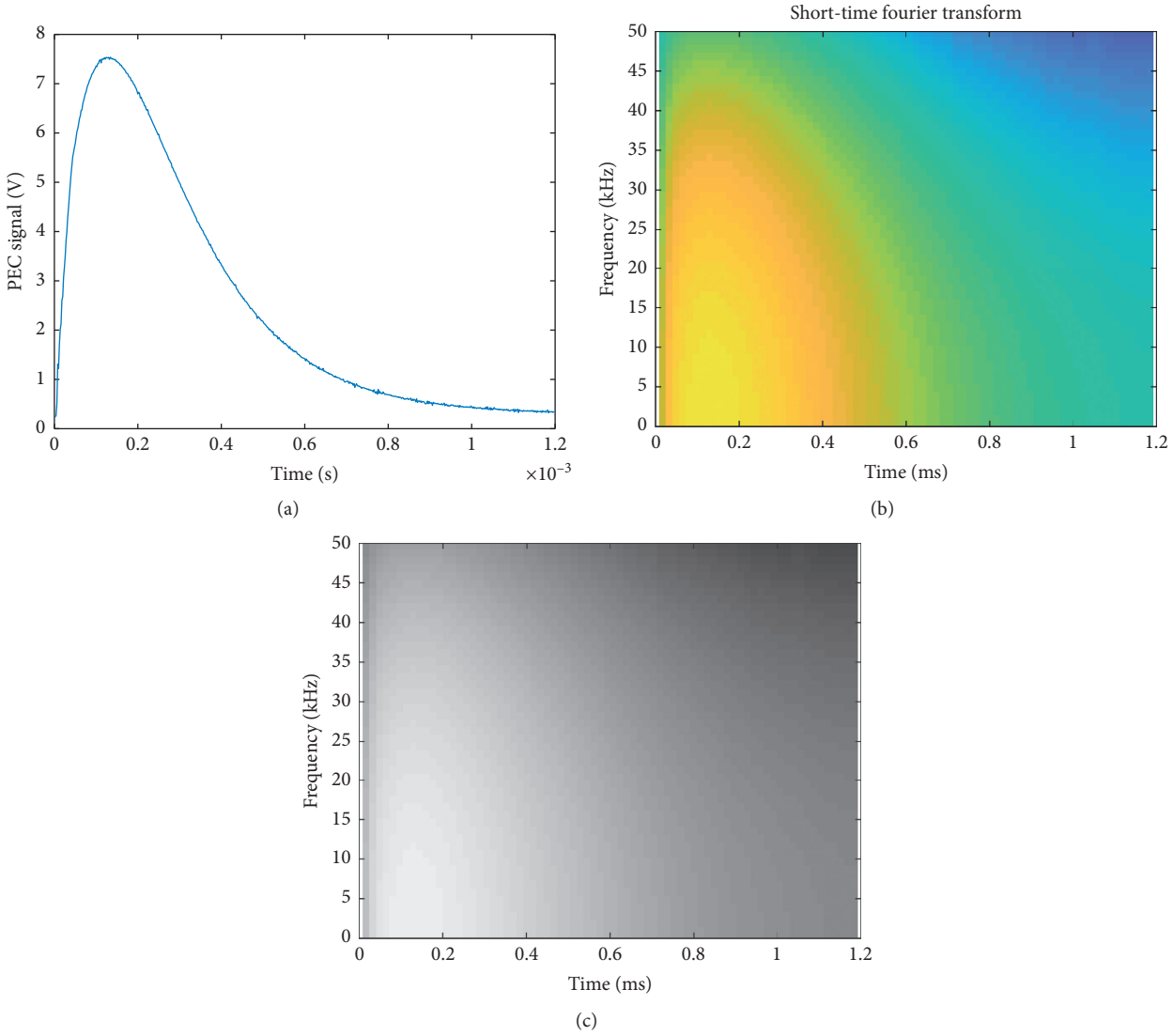


Figure 5: TFR obtained using the STFT method: (a) PEC signal; (b) TFR; (c) gray image.

pretraining architecture due to its excellent accuracy and feature extraction capabilities. Among them, VGG16 includes 5 Convs and three FCs and requires input of 224×224. Each convolutional layer is composed of multiple subconvolutional layers with small kernel and a pooling layer. The architecture increases the number of nonlinear mappings and improves the fitting ability of the network, which makes the VGG16 architecture more suitable to engineering applications where it is difficult to obtain a large number of samples. Moreover, the VGG16 architecture is simple and direct, so it is easy to implement network improvements.

3.2.2. Network Optimization Method. A wider and deeper architecture can improve the feature extraction and mapping capabilities of DCNN on input signals, but it has more parameters and requires more memory and computational effort. In fact, many channels (called filters in this article) in the Convs, especially deeper Convs, have very low or even zero weights and have not played the expected role [32]. Regarding FCs, previous research has proved that more layers and hidden nucleotides do not imply stronger mapping ability [8]. In terms of the input image, although larger size (or more pixels in the image) means more information is contained, more memory and computation cost are required due to more intermediate variables (such as feature maps).

We propose a network optimization method based on VGG16, which comprises the following four steps.

First, an adaptive pooling layer is adopted so that the VGG16 architecture accepts smaller input. In this article, the input signal is reduced from 224×224 to 32×32.

Second, a batch normalization layer (BN) is added after each convolutional layer (Conv) and fully connected layer (FC). BN refers to normalizing the data of each minibatch to a mean and variance of 0 and 1, respectively, when DCNN is trained by gradient descent. The BN alleviate the gradient disappearance (or explosion) phenomenon during DCNN

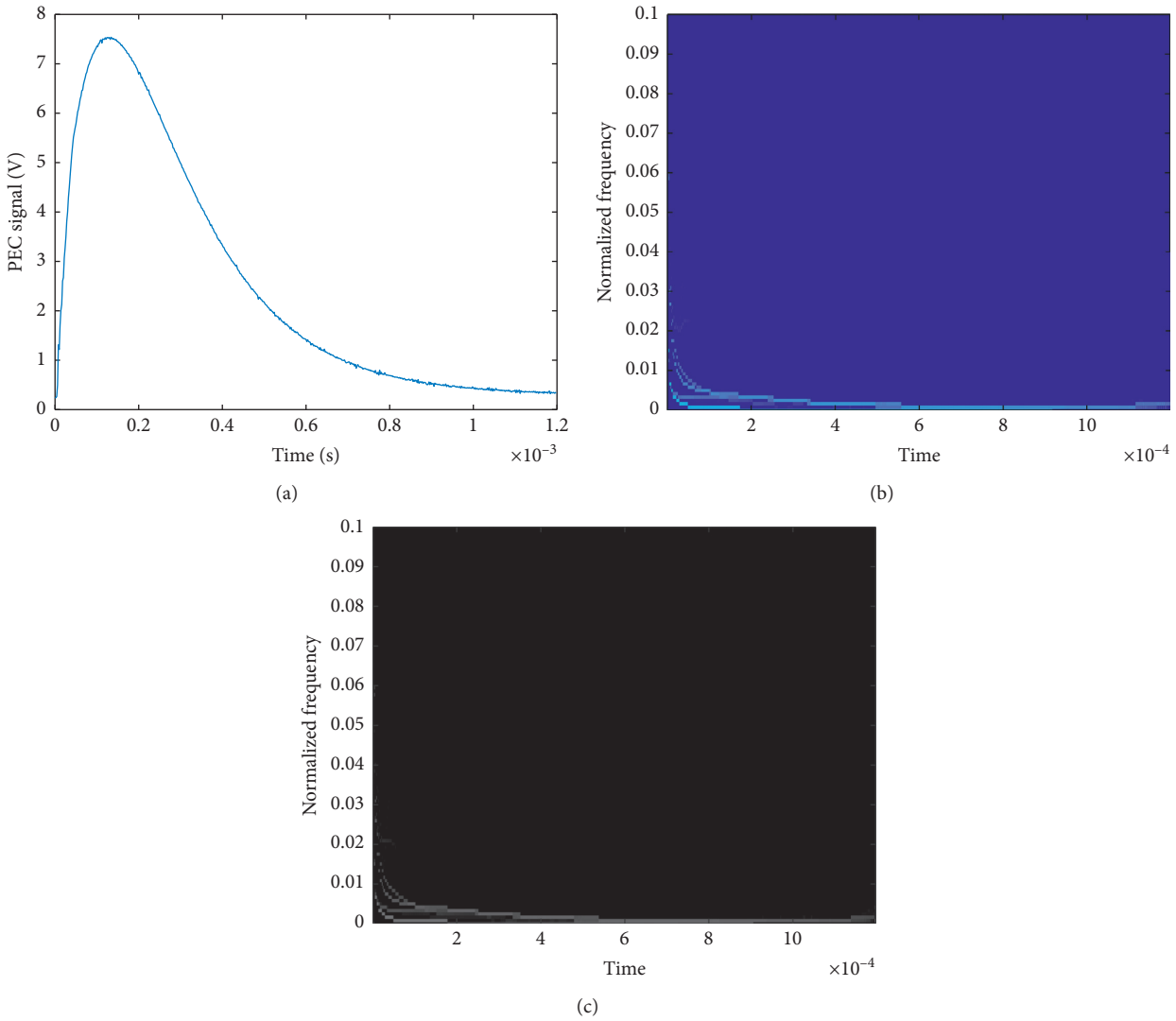


Figure 6: TFR obtained using the EEMD method: (a) PEC signal; (b) TFR; (c) gray image.

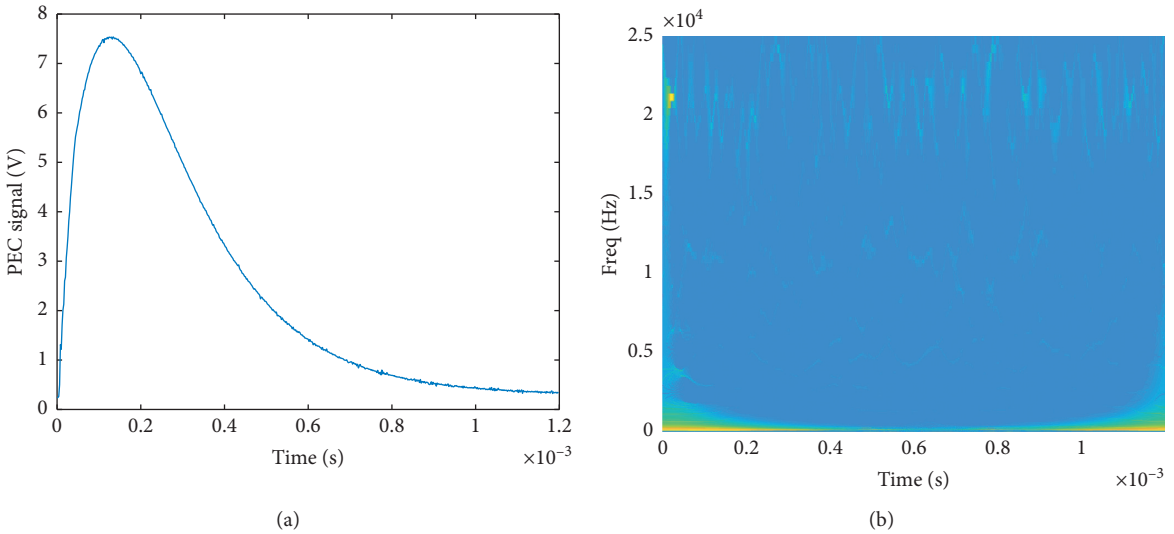
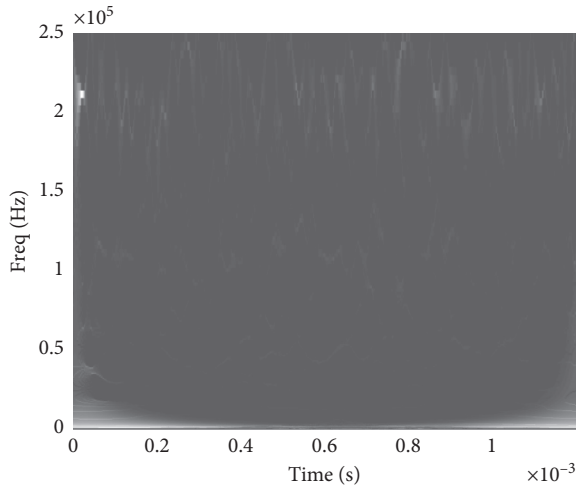


Figure 7: Continued.



(c)

Figure 7: TFR obtained using the SST method: (a) PEC signal; (b) TFR; (c) gray image.

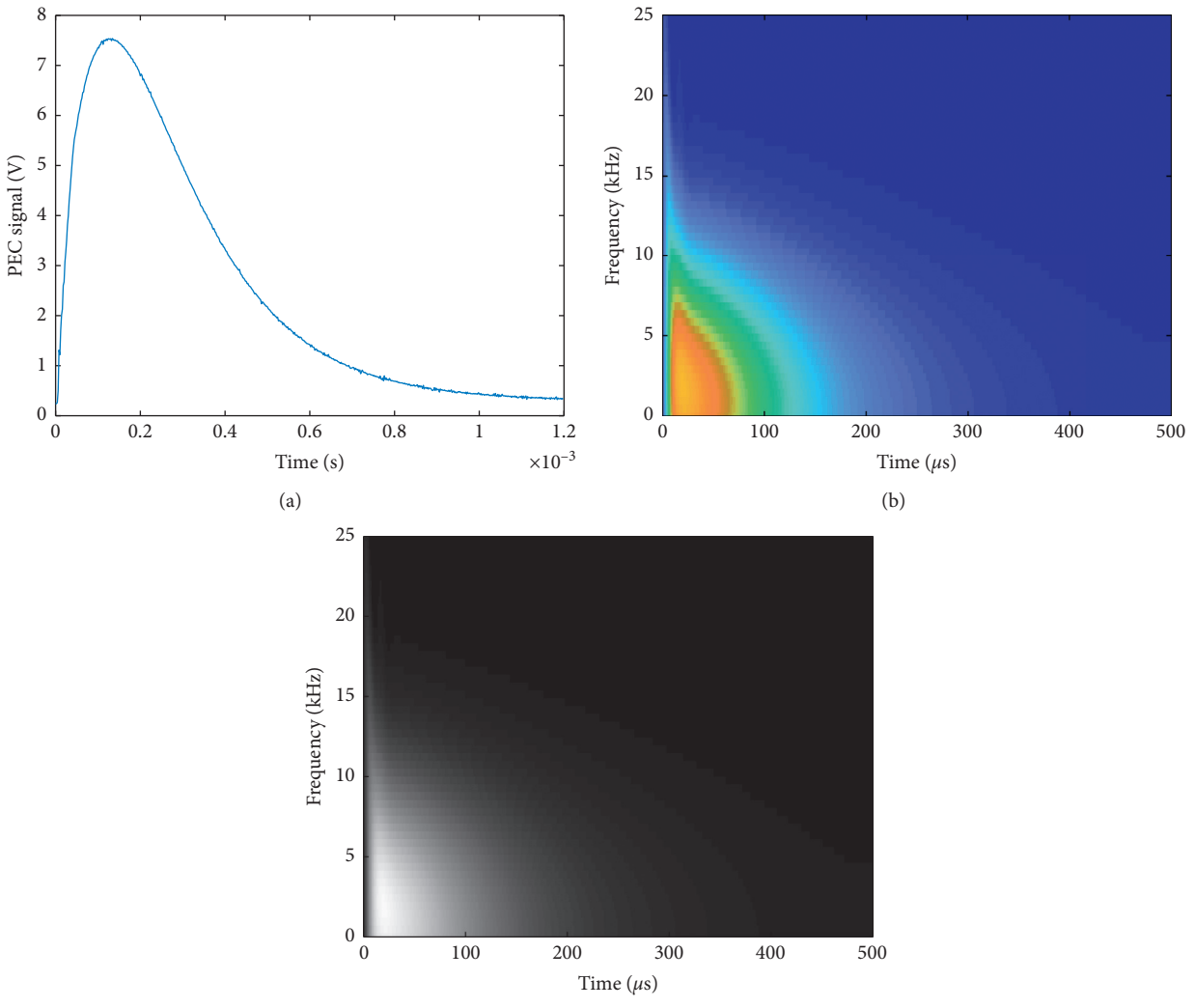


Figure 8: TFR obtained using the PSWVD method: (a) PEC signal; (b) TFR; (c) gray image.

Table 2: Image entropies obtained from TFRs of different sizes.

	STFT	EEMD	SST	PSWVD	GWT
224×224	6.42	1.00	1.58	4.43	6.97
32×32	6.47	2.69	2.98	5.25	7.07
Changes (%)	0.74	168.39	88.82	18.40	1.53

training, speeding up the model training and reducing the dependence on the initial parameters [41].

Third, the original three FCs are compressed to two, and the number of hidden neurons in each FC is reduced from 4096 to 512.

Finally, unimportant filters (channels) in the Convs are removed. The importance of each filter is evaluated based on the activation value (the output of the activation function). The evaluation criterion is the absolute value of the first-order term in the Taylor expansion of the objective function relative to the activation value. The biggest advantage of this method is the avoidance of additional calculation.

The principle of this criterion based on the Taylor expansion is as follows [35]. Assume that the cost of pruning operations can be described by

$$\left|\Delta C\left(h_i\right)\right| = \left|C\left(D, h_i = 0\right) - C\left(D, h_i\right)\right|, \tag{5}$$

where $C(D, h_i = 0)$ is the cost if output h_i is pruned and $C(D, h_i)$ is the cost if h_i is not pruned.

The first-order Taylor polynomial near $h_i = 0$ is used to approximate $C(D, h_i = 0)$, i.e.,

$$C\left(D, h_i = 0\right) = C\left(D, h_i\right) - \frac{\partial C}{\partial h_i} h_i + R_1\left(h_i = 0\right), \tag{6}$$

where $R_1(h_i = 0)$ is the first-order remainder and is neglected here. So, equation (5) can be written as follows:

$$\left|\Delta C\left(h_i\right)\right| = \left|C\left(D, h_i\right) - \frac{\partial C}{\partial h_i} h_i - C\left(D, h_i\right)\right| = \left|\frac{\partial C}{\partial h_i} h_i\right|. \tag{7}$$

Specifically, the kth filter of the lth convolutional layer is written as $z_l^{(k)}$, and the cost function generated by removing the filter is Θ_{TE}, which can be calculated as follows:

$$\Theta_{\mathrm{TE}}\left(z_l^{(k)}\right) = \left|\frac{1}{M} \sum_m \frac{\partial C}{\partial z_{l,m}^{(k)}} z_{l,m}^{(k)}\right|, \tag{8}$$

where M is the length of the vectorized feature map. In fact, $z_{l,m}^{(k)}$ is the activation of the kth filter in the lth convolutional layer. The partial derivative terms can be obtained from backpropagation. So, Θ_{TE} can be obtained without additional calculation. All of the filters will be sorted according to their Θ_{TE} values.

To avoid the performance degradation caused by removing a large number of filters at one time, we adopt multiple iterations of pruning and retraining to compress the network. The filters in all of the Convs are sorted according to the Taylor criterion, and the least important N filters are removed, where we set $N = 512$. The pruned network is retrained using the dataset shown in Table 1. After five iterations of pruning and training, the pruning operation is terminated. It is worth noting that the improved VGG16 architecture used in this article is quite different from the baseline VGG16 architecture due to improvements such as adding BN and reducing FCs. Thus, the architecture is trained by the dataset in Table 1 from scratch to obtain weights before iterations of pruning and retraining.

Figure 9 compares the baseline and optimized VGG16. The baseline VGG16 architecture requires 224×224 input signals, and the Convs directly perform convolution processing on the input signals to obtain the feature map. After the convolutions are completed, feature maps obtained by the last convolutional layer are fed into three FCs for mapping between feature maps and defect patterns. When the input signal is 224×224, the number of input neurons in the first FC layer is 25088. The numbers of hidden neurons in the first and second FC are both 4096. The number of hidden neurons in the third FC is three, which is equal to the number of defect types.

The input size in the optimized VGG16 architecture is 32×32. The BNs are added after every Conv and FC and the numbers of hidden neurons in the two FC layers are 512 and 3. If the jth filter in the ith convolutional layer $\mathrm{Con}_{i\mathrm{th}}$, $F_{\mathrm{Con}}(i, j)$, must be deleted, then the corresponding filter in the BN behind the Conv, $F_{\mathrm{BN}}(i, j)$, and the feature map $\mathrm{FM}(i, j)$, also must be deleted.

4. Results and Discussion

We verify the proposed intelligent defect recognition pipeline and analyze the effect of TFRs and network optimization method on defect identification in detail.

4.1. Effect of TFR on Defect Identification. To verify the effectiveness of the GWT method, four common-used TFA methods, i.e., STFT, PSWVD, EEMD, and SST, were also used to transform PECT signal in Table 1 into 2D TFR and formed the training set and test set. Five pretrained DCNN architectures, as shown in Table 3, were used to build end-to-end pattern recognizers to identify defects with the above 2D dataset. For simplicity, we do not repeat the principles of these methods, which are available in references.

The result of defect identification is shown in Figure 4. The optimizer used by five DCNNs was stochastic gradient descent with momentum (SGDM), whose main parameters, i.e., momentum, batch size, initial learning rate, and training epochs, were set as 0.9, 4, 0.0001, and 20, respectively. The loss function was the crossentropy function. In Table 4, the accuracy is the average test accuracy from five experiments. The time, used to indicate the computational cost of different algorithms, is the training time of an epoch in an experiment. In the following content, except for special

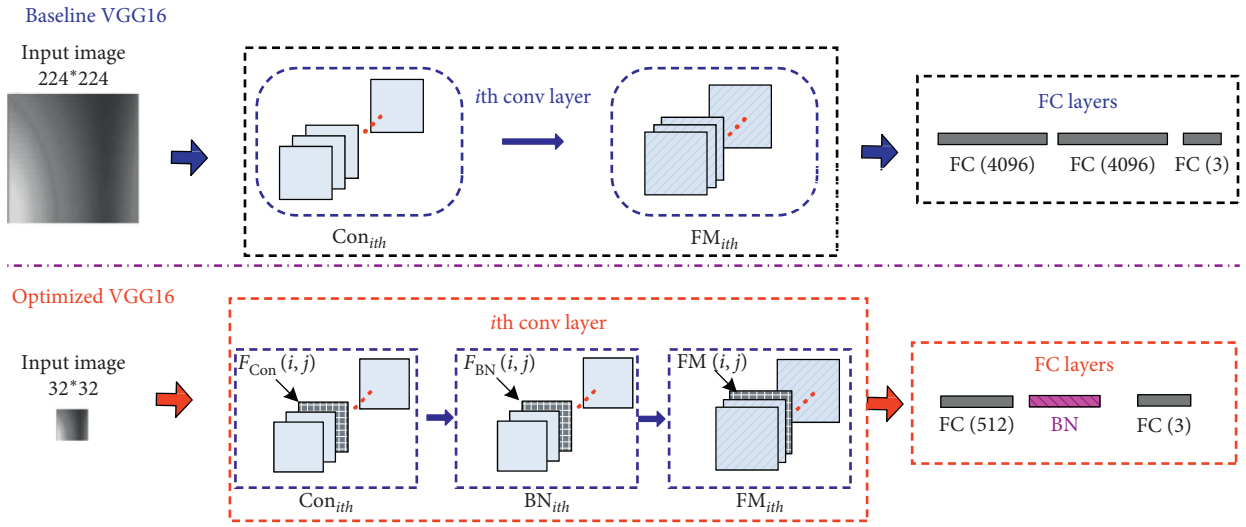


Figure 9: Comparison between baseline and pruned VGG16.

Table 3: Pretrained network parameters.

Network	Depth	Size (MB)	Parameters (millions)	Image input size
GoogLeNet	22	27	7.0	224 × 224
Inception V3	48	89	23.9	299 × 299
ResNet50	50	96	25.6	224 × 224
AlexNet	8	227	61.0	227 × 227
VGG16	16	515	138.0	224 × 224

Table 4: Identification accuracy and computational cost for different defect identification pipelines.

TFR	GoogLeNet		Inception V3		ResNet50		AlexNet		VGG16	
	Accuracy (%)	Time (s)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Time (s)	Accuracy (%)	Time (s)
STFT	74.12 ± 0.85	34.4	74.36 ± 1.67	74.36 ± 1.67	74.36 ± 1.67	74.36 ± 1.67	75.50 ± 1.57	14.5	85.16 ± 0.80	191.8
PSWVD	73.78 ± 0.89	35.6	71.42 ± 1.21	71.42 ± 1.21	71.42 ± 1.21	71.42 ± 1.21	89.92 ± 1.44	15.4	90.03 ± 1.80	193.4
EEMD	77.90 ± 1.00	36.7	73.52 ± 1.32	73.52 ± 1.32	73.52 ± 1.32	73.52 ± 1.32	80.42 ± 1.26	16.7	96.57 ± 0.48	196.7
SST	72.77 ± 0.52	37.4	74.37 ± 1.89	74.37 ± 1.89	74.37 ± 1.89	74.37 ± 1.89	80.33 ± 0.45	17.5	97.88 ± 0.35	208.5
GWT	78.16 ± 1.03	36.2	74.79 ± 1.04	74.79 ± 1.04	74.79 ± 1.04	74.79 ± 1.04	91.10 ± 0.31	14.1	92.46 ± 0.43	189.6

instructions, the same parameter settings are adopted. The processor was an Intel Core i5-7300U with a main frequency of 2.60 GHz. The algorithm was realized by Python software and ran on a single GPU.

It can be seen from Table 4 that GWT generally performed better in combination with all five pretrained DCNNs. The simple architectures of VGG16 and AlexNet obtained higher accuracy than the complex architectures of GoogLeNet, Inceptive3, and ResNet50 due to the small dataset. Figures 10–11 show the confusion matrices, accuracy, and loss function curves obtained during an experiment for two pipelines of TFR and DCNN. It can be indicated that besides the end-to-end pattern recognizer, TFRs also have a great impact on the identification accuracy. The defect identification pipeline composed of AlexNet and the other four TFRs except for STFT can achieve better accuracy. AlexNet cannot extract the features contained in the less clear TFR obtained by STFT. So, this leads to a larger deviation between the training accuracy and the validation accuracy, as shown in Figure 11(b)

The TFR obtained by the GWT method not only has excellent ability in terms of the transient component analysis but also is less affected by the reduction of image size, which not only guarantees high recognition accuracy but also is very helpful for reducing the memory and computational cost of DCNN.

4.2. Network Optimization Methods. As analysed above, simple architectures such as VGG16 and AlexNet are more suitable for applications with small datasets. However, VGG16 had a much higher memory and computational cost than AlexNet due to its deeper and wider architecture. Therefore, it is necessary to explore effective network

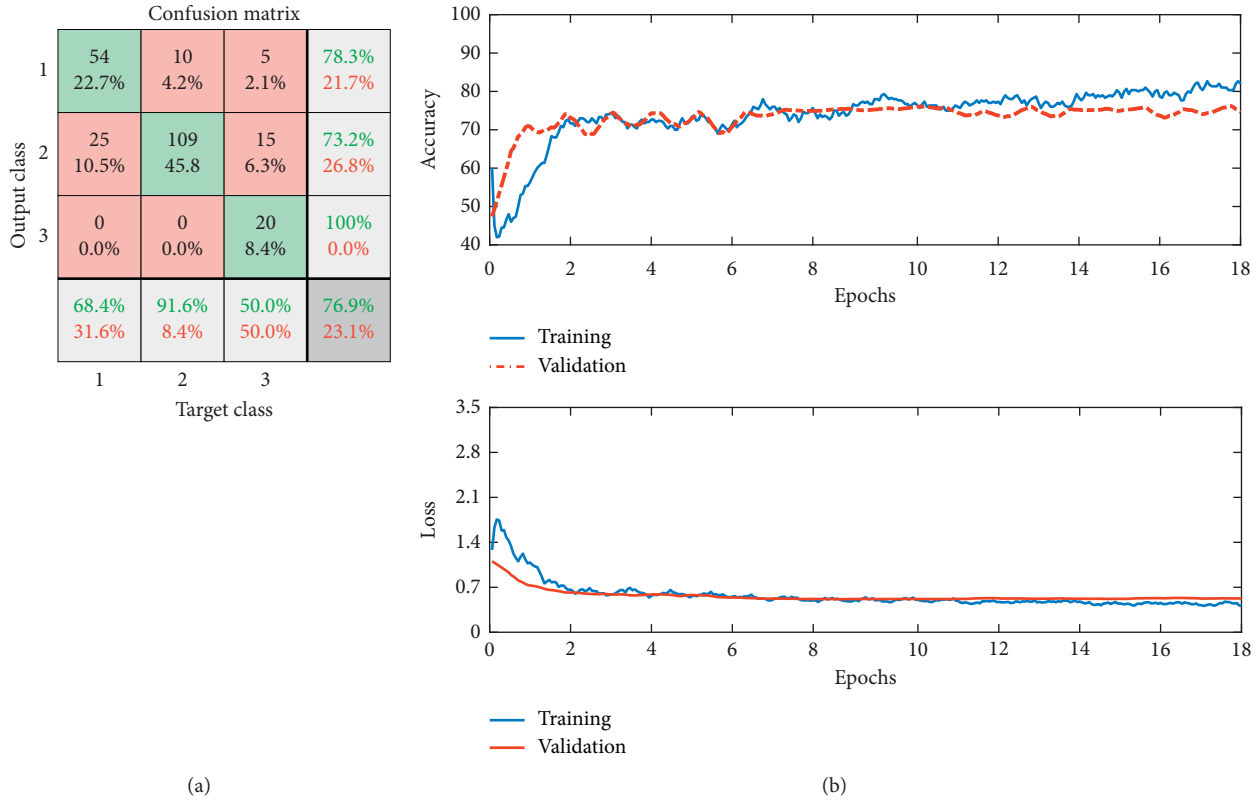


Figure 10: (a) Confusion matrix and (b) accuracy and loss curves for the combination of GWT and GoogLeNet.

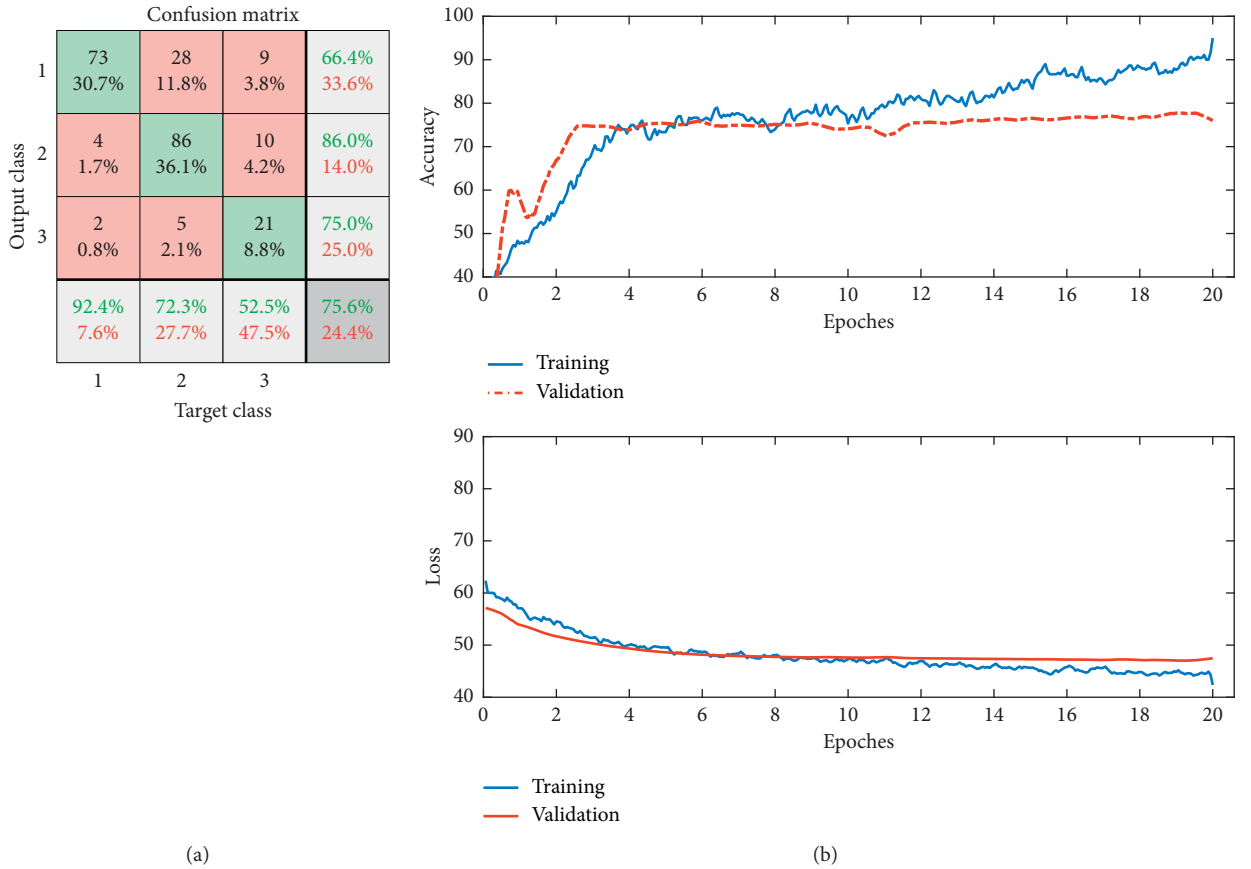


Figure 11: (a) Confusion matrix and (b) accuracy and loss curves for combination of STFT and AlexNet.

optimization methods to reduce the memory and computational of VGG16 while ensuring recognition accuracy, so as to be more suitable for engineering applications.

The proposed network optimization methods have four steps: reducing the input size, adding BNs, simplifying FCs, and removing unimportant filters in Convs. The following section will verify the proposed method in detail.

4.2.1. Reducing the Size of 2D Input. VGG16 can accept input signals of any size by adopting an adaptive pooling layer. In this article, the input signal is reduced from 224×224 to 32×32. In this section, the impact of using smaller input on memory is discussed, as shown in Table 5. It is worth noting that only the memory occupied by intermediate variables including the feature map from Convs and pooling layers (shown as ParamI in Table 5) and model parameters (ParamII in Table 5) are considered in this article. The format of ParamI is height × width × number of filters, and the format of ParamII is height × width × number of input filters × number of output filters. The feature map from the Convs and pooling layers are calculated by the following formula:

$$\begin{aligned} \mathrm{Con}_H &= \left(\frac{\mathrm{Con}_{H0} - f + 2 \times \mathrm{padding}}{\mathrm{stride}}\right) + 1, \\ \mathrm{Con}_W &= \left(\frac{\mathrm{Con}_{W0} - f + 2 \times \mathrm{padding}}{\mathrm{stride}}\right) + 1, \\ \mathrm{Pool}_H &= \left(\frac{\mathrm{Pool}_{H0} - f}{\mathrm{stride}}\right) + 1, \\ \mathrm{Pool}_W &= \left(\frac{\mathrm{Pool}_{W0} - f}{\mathrm{stride}}\right) + 1, \end{aligned} \tag{9}$$

where Con and Pool, respectively, represent the Convs and pooling layers. f is the Kernel size. The subscripts H and W, respectively, indicate the height and width of the feature map. Symbols with the subscript 0 represent the input feature map. The hyperparameters of the convolutional layer are as follows: f is 3, stride is 1, and padding is set to 1. The maximum pooling layer is adopted, where f is 2, stride is 2, and padding is 0.

It can be seen from Table 5 that when the input size is reduced from 224×224 to 32×32, the total memory is reduced by about 90%, where the intermediate variables are reduced by 98%. The model parameters are reduced by about 89%. Thus, using small-size input will save significant memory, which is of great significance to the engineering application of DCNN. The impact of using smaller input on accuracy is discussed in Section 4.2.2.

4.2.2. Adding BNs and Simplifying FCs. We discuss the effects of adding BNs and reducing FCs, by comparing the performance of VGG16 and three VGG16 based variants, as shown in Table 6, where VGG16 is the baseline VGG16 architecture, and VGG16I, VGG16II, and VGG16III are three kinds of VGG16 variants whose input sizes are all 32×32. Their main differences are whether to add BNs and simplify FCs.

The accuracies and computational costs of the four architectures have been compared in Table 7.

The following conclusions can be obtained from Table 7. First, by comparing VGG16 I and VGG16, it can be seen that reducing input size significantly reduces the computational cost, about 90%. However, the accuracy also decreases significantly, especially for EEMD and SST, whose accuracies decreased by over 30%. This may be because the information entropies of TFRs obtained by these two methods will change greatly as the size decreases, as shown in Table 2. The other three TFA methods are less affected by size changes, so their accuracy drops within 10%. Second, by comparing VGG16II and VGG16I, it can be seen that the adding BNs significantly improves the recognition accuracy, especially for the GWT method, whose accuracy reaches 96%. The computational cost increases slightly because the addition of the BN layers requires a small amount of computational cost. Third, by comparing VGG16III and VGG16II, it can be seen that simplifying FCs does not worsen the accuracy and further reduces the computational time by 40%. Table 7 verifies the effectiveness of the used GWT method for PEC signal processing again, especially in the case of small size.

Figure 12 contrasts the training accuracies, training loss functions, and test accuracies obtained by the above three improved VGG16 architectures, where curves 1, 2, and 3 in figures represent VGG16I, VGG16II, and VGG16III, respectively.

It can be seen that, for VGG16I without the BNs, i.e., curve 1 in Figure 12, neither the accuracy nor the loss function has been improved with the training process. However, for the VGG16II and VGG16III with BNs, i.e., curves 2 and 3 in Figure 12, their training accuracies increase from 40% to 100%, loss functions decrease from about 1 to nearly 0, and testing accuracies reach about 97% after five epochs of training. Figure 12 shows intuitively that the adding BNs is important in improving the accuracy and convergence speed of DCNN and that simplifying FCs will not degrade its performance.

4.2.3. Pruning Filters in Convs. We discuss the pruning filters in Convs based on the VGG16III architecture. Filter pruning is carried out in five iterations including pruning and retraining. The activation values of the feature maps generated by all of the filters in each Conv are calculated and ranked according to the Taylor criterion described by equations (5)–(8), and the 512 filters with the lowest contribution are removed. The training set shown in Table 1 is used to retrain the pruned architecture to maintain performance. Five iterations are performed, and a total of 2,560 filters are pruned.

Table 8 shows the numbers of remaining filters in every Conv and the testing accuracy after retraining in each iteration of pruning and retraining. It is worth noting that the VGG16III architecture is pretrained using the PECT dataset, as shown in Table 1 from scratch architecture, and the testing

Table 5: Effects of different sizes of input signals on memory.

Layer no.	224×224		32×32	
	ParamI	ParamII	ParamI	ParamII
Conv1_1	$224 \times 224 \times 64$	$(3 \times 3 \times 3) \times 64$	$32 \times 32 \times 64$	$(3 \times 3 \times 3) \times 64$
Conv1_2	$224 \times 224 \times 64$	$(3 \times 3 \times 64) \times 64$	$32 \times 32 \times 64$	$(3 \times 3 \times 64) \times 64$
Pool1	$112 \times 112 \times 64$	0	$16 \times 16 \times 64$	0
Conv2_1	$112 \times 112 \times 128$	$(3 \times 3 \times 64) \times 128$	$16 \times 16 \times 128$	$(3 \times 3 \times 64) \times 128$
Conv2_2	$112 \times 112 \times 128$	$(3 \times 3 \times 128) \times 128$	$16 \times 16 \times 128$	$(3 \times 3 \times 128) \times 128$
Pool2	$56 \times 56 \times 128$	0	$8 \times 8 \times 128$	0
Conv3_1	$56 \times 56 \times 256$	$(3 \times 3 \times 128) \times 256$	$8 \times 8 \times 256$	$(3 \times 3 \times 128) \times 256$
Conv3_2	$56 \times 56 \times 256$	$(3 \times 3 \times 256) \times 256$	$8 \times 8 \times 256$	$(3 \times 3 \times 256) \times 256$
Conv3_3	$56 \times 56 \times 2556$	$(3 \times 3 \times 256) \times 256$	$8 \times 8 \times 256$	$(3 \times 3 \times 256) \times 256$
Pool3	$28 \times 28 \times 256$	0	$4 \times 4 \times 256$	0
Conv4_1	$28 \times 28 \times 512$	$(3 \times 3 \times 256) \times 512$	$4 \times 4 \times 512$	$(3 \times 3 \times 256) \times 512$
Conv4_2	$28 \times 28 \times 512$	$(3 \times 3 \times 512) \times 512$	$4 \times 4 \times 512$	$(3 \times 3 \times 512) \times 512$
Conv4_3	$28 \times 28 \times 512$	$(3 \times 3 \times 512) \times 512$	$4 \times 4 \times 512$	$(3 \times 3 \times 512) \times 512$
Pool4	$14 \times 14 \times 512$	0	$2 \times 2 \times 512$	0
Conv5_1	$14 \times 14 \times 512$	$(3 \times 3 \times 512) \times 512$	$2 \times 2 \times 512$	$(3 \times 3 \times 512) \times 512$
Conv5_2	$14 \times 14 \times 512$	$(3 \times 3 \times 512) \times 512$	$2 \times 2 \times 512$	$(3 \times 3 \times 512) \times 512$
Conv5_3	$14 \times 14 \times 512$	$(3 \times 3 \times 512) \times 512$	$2 \times 2 \times 512$	$(3 \times 3 \times 512) \times 512$
Pool5	$7 \times 7 \times 512$	0	$1 \times 1 \times 512$	0
FC1	4096	$7 \times 7 \times 512 \times 4096$	512	$1 \times 1 \times 512 \times 512$
FC2	4096	4096×4096	3	512×3
FC3	3	4096×3	—	—
Total	15713283	134260416	288259	14974144
	149973699		15262403	

Table 6: VGG16 and three kinds of variants.

	VGG16	VGG16I	VGG16II	VGG16III
Size of input signal	224×224	32×32	32×32	32×32
BNs	No	No	Yes	Yes
FCs	4096, 4096, 3	4096, 4096, 3	4096, 4096, 3	512, 3

Table 7: Effect of adding BNs and reducing FCs.

TFR	VGG16		VGG16I		VGG16II		VGG16III	
	Accuracy (%)	Time (s)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Time (s)	Accuracy (%)	Time (s)
STFT	85.16 ± 0.80	191.8	85.16 ± 0.80	85.16 ± 0.80	92.28 ± 0.49	18.9	92.56 ± 0.43	10
PSWVD	90.03 ± 1.80	193.4	90.03 ± 1.80	90.03 ± 1.80	92.36 ± 0.51	16.7	92.80 ± 0.87	9.5
EEMD	96.57 ± 0.48	196.7	96.57 ± 0.48	96.57 ± 0.48	76.04 ± 0.66	17.4	79.06 ± 0.90	10.2
SST	97.88 ± 0.35	208.5	97.88 ± 0.35	97.88 ± 0.35	75.50 ± 1.19	20.1	76.80 ± 1.91	12.3
GWT	92.46 ± 0.43	189.6	92.46 ± 0.43	92.46 ± 0.43	96.02 ± 1.17	17.5	96.22 ± 0.34	10.8

accuracy before pruning is 96.6%. Other accuracies in Table 8 are averages of testing accuracies obtained from five experiments. Computational cost is the training time of an epoch in an experiment. The main parameters of the SGDM optimizer were set as momentum of 0.9, batch size of 4, LR of 10-5, and training epochs of 10. The loss function is the crossentropy function.

The following conclusions can be obtained from Table 8. First, the pruning filters did not cause the deterioration of network performance, but the accuracy increased by about 3%. Second, the computational cost was further reduced through the pruning filter, from 10.8 seconds to 3.96 seconds, because the corresponding intermediate operations can be reduced as some filters are removed. At last, compared to the VGG16 baseline architecture, the optimized VGG16 architecture obtained by the proposed method can reduce the computational cost by 98%. Moreover, it can also be calculated that the memory occupied by the optimized VGG16 was reduced by 98% in terms of intermediate variables and structural parameters, as shown in Table 5. Third, the deeper the Conv is, the higher the pruning rate (i.e., the percentage of removed filters in the initial filters) is, gradually increasing from 34% in the first Conv to 87.5% in the last. Finally, the learning rate plays an important role in improving network performance, as shown in Figure 13. When a small LR is used, e.g., LR = 0.00001, the adjustment

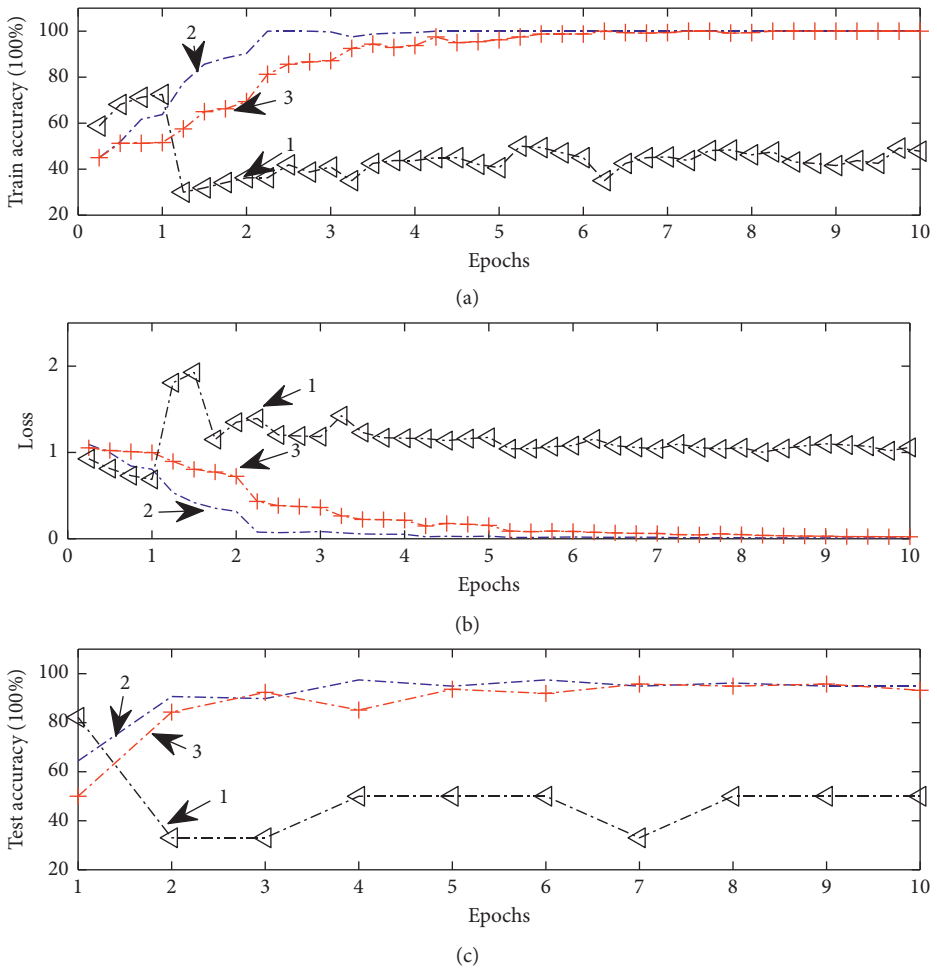


Figure 12: Comparison of training processes of three improved VGG16 architectures: (a) training accuracies, (b) training loss functions, and (c) test accuracies.

Table 8: Filter pruning based on Taylor criterion.

	Layer no.	Initial number of filters	First pruning iteration	Second pruning iteration	Third pruning iteration	Fourth pruning iteration	Fifth pruning iteration	Pruning rate (%)
Number of filters in convs	1	64	63	61	54	50	43	32.8
	4	64	63	55	48	39	34	46.8
	8	128	120	112	106	100	86	32.8
	11	128	126	119	108	96	78	39.1
	15	256	240	227	213	179	149	41.8
	18	256	243	228	205	169	146	42.9
	21	256	247	235	207	182	150	41.4
	25	512	468	425	353	294	222	56.6
	28	512	474	432	373	291	226	55.8
	31	512	476	414	367	295	228	55.4
	35	512	389	328	264	205	144	71.8
	38	512	407	310	217	166	94	81.6
	41	512	396	254	173	110	64	87.5
Time (s)	—	10.80	9.11	7.78	6.57	5.06	3.96	——
Accuracy (%)	—	96.60	97.30 ± 2.50	99.27 + 0.17	99.32 ± 0.41	99.60 ± 0.18	99.58 ± 0.27	——

is slight and a satisfying training effect can be obtained. The testing accuracy has increased and maintains about 99% after three epochs of training, as shown in Figure 13(a). However, when the LR is larger, e.g., LR = 0.0001, the adjustment fluctuates sharply, and it is difficult to obtain an ideal training effect, as shown in Figure 13(b).

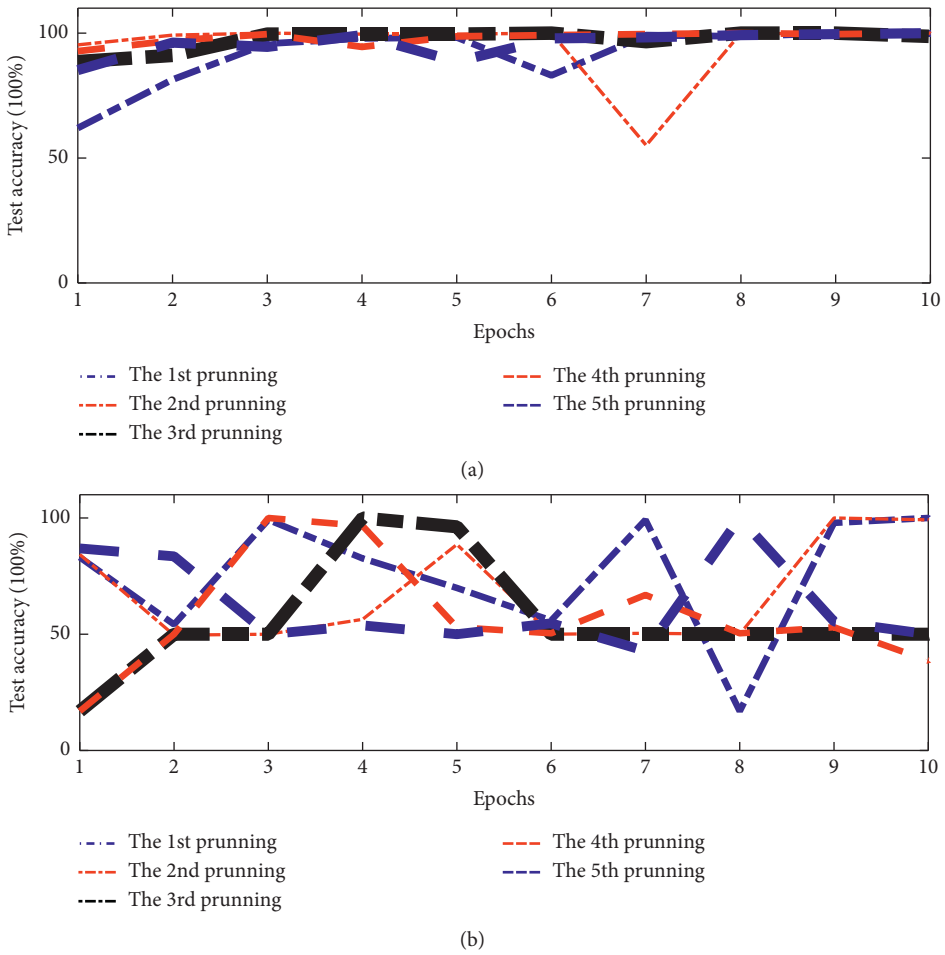


Figure 13: Effect of learning rate on accuracy: (a) LR = 0.00001; (b) LR = 0.0001.

In a word, the optimized VGG16 structure obtained by the proposed network optimization method not only reduces the computational effort significantly but also improves the accuracy. Moreover, VGG16 architecture is simple and straightforward, and the proposed network optimization method does not introduce additional calculations, so the proposed method has low complexity and is suitable for industrial applications.

5. Conclusion

We have proposed and verified an intelligent defect identification pipeline based on time series signals and optimized VGG16, which can accurately identify the defect type without manual feature extraction. By comparison to other TFA methods, the GWT method not only has a clearer resolution to the transient components contained in the PECT signal but also ensures that the image information is not lost when the TFR size is reduced to 2%. It has also been verified by experiments that the proposed optimized VGG16 can increase the accuracy by 7% and reduce the running time and memory by 98%. This provides an effective solution to the problems of large memory and high computational cost that restrict the application of DCNN in engineering applications. In the future work, other networks besides DCNN, such as fast recurrent Neural Network and long short term memory (LSTM) network, will be used in nondestructive testing. In addition, the influence of nonlinear interference, such as temperature and stress, on defect recognition will be further studied.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

The authors sincerely thank Professor John Brigham and Postdoctor Wang Qian of Durham University for their help in the pruning algorithm. This research was funded by Jiangxi Province Education Department (Grant nos. GJJ161104 and GJJ161122), Jiangxi Provincial Department of Science and Technology (Grant no. 20171BAB206037),

and National Natural Science Foundation of China (Grant nos. 61903176 and 62001202).

References

[1] A. Sophian, G. Tian, and M. Fan, "Pulsed eddy current non-destructive testing and evaluation: a review," *Chinese Journal of Mechanical Engineering*, vol. 30, no. 3, pp. 500–514, 2017.

[2] M. Fan, B. Cao, A. I. Sunny, W. Li, G. Tian, and B. Ye, "Pulsed eddy current thickness measurement using phase features immune to liftoff effect," *NDT & E International*, vol. 86, pp. 123–131, 2017.

[3] Y. Li, B. Yan, D. Li, Y. Li, and D. Zhou, "Gradient-field pulsed eddy current probes for imaging of hidden corrosion in conductive structures," *Sensors and Actuators A: Physical*, vol. 238, pp. 251–265, 2016.

[4] B. Liu, P. Huang, and D. Hou, "Application of Hilbert-Huang transform for defect recognition in pulsed eddy current testing," *Nondestructive Testing and Evaluation*, vol. 30, no. 3, pp. 232–251, 2015.

[5] G. Piao, J. Guo, T. Hu, Y. Deng, and H. Leung, "A novel pulsed eddy current method for high-speed pipeline inline inspection," *Sensors and Actuators A: Physical*, vol. 295, pp. 244–258, 2019.

[6] K. Liu, Q. Y. Shang, and W. D. Widanage, "A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery," *IEEE Transactions on Industrial Electronics*, vol. 99, 2020.

[7] Q. Ouyang, K. Z. Wang, and Y. G. LiXu, "Optimal charging control for lithium-ion battery packs: a distributed average tracking approach," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3430–3438, 2020.

[8] B. Liu, D. Hou, P. Huang et al., "An improved PSO-SVM model for online recognition defects in eddy current testing," *Nondestructive Testing and Evaluation*, vol. 28, no. 4, pp. 367–385, 2013.

[9] B. Yang, F. Luo, and Y. Zhang, "Quantification and classification of cracks in aircraft multi-layered structurefication and classification of cracks in aircraft multi-layered structure," *Chinese Journal of Mechanical Engineering*, vol. 42, no. 02, pp. 63–67, 2006.

[10] Y. He, M. Pan, and G. Tian, "Pulsed eddy current imaging and frequency spectrum analysis for hidden defect nondestructive testing and evaluation," *NDT & E International*, vol. 44, no. 4, pp. 344–352, 2011.

[11] J. A. Buck, P. R. Underhill, J. E. Morelli, and T. W. Krause, "Simultaneous multiparameter measurement in pulsed eddy current steam generator data using artificial neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 3, pp. 672–679, 2016.

[12] Y. Liu, S. Liu, H. Liu et al., "Pulsed eddy current data analysis for the characterization of the second-layer discontinuities," *Journal of Nondestructive Evaluation*, vol. 38, no. 1, p. 7, 2019.

[13] A. Benyahia, M. Zergoug, M. Amir et al., "Enhancement of pulsed eddy current response based on power spectral density after continuous wavelet transform decomposition," *International Journal of Computer and Information Engineering*, vol. 12, no. 2, pp. 116–119, 2018.

[14] B. Liu, P. Huang, X. Zeng, and Z. Li, "Hidden defect recognition based on the improved ensemble empirical decomposition method and pulsed eddy current testing," *NDT & E International*, vol. 86, pp. 175–185, 2017.

[15] D. J. Pasadas, P. Baskaran, H. G. Ramos et al., "Detection and classification of defects using ECT and multi-level SVM model," *IEEE Sensors Journal*, vol. 20, no. 5, pp. 2329–2338, 2019.

[16] P. Ma, H. Zhang, W. Fan et al., "A novel bearing fault diagnosis method based on 2D image representation and transfer learning-convolutional neural network," *Measurement Science and Technology*, vol. 30, no. 5, Article ID 055402, 2019.

[17] P. Ballester and R. M. Araujo, "On the performance of GoogLeNet and AlexNet applied to sketches," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA, Feburary 2016.

[18] S. Cheon, H. Lee, C. O. Kim, and S. H. Lee, "Convolutional neural network for wafer surface defect classification and the detection of unknown defect class," *IEEE Transactions on Semiconductor Manufacturing*, vol. 32, no. 2, pp. 163–170, 2019.

[19] N. Saeed, N. King, Z. Said, and M. A. Omar, "Automatic defects detection in CFRP thermograms, using convolutional neural networks and transfer learning," *Infrared Physics & Technology*, vol. 102, p. 103048, 2019.

[20] P. Zhu, Y. Cheng, P. Banerjee, A. Tamburrino, and Y. Deng, "A novel machine learning model for eddy current testing with uncertainty," *NDT & E International*, vol. 101, pp. 104–112, 2019.

[21] X. XiongJ. Wang et al., "A two-dimensional convolutional neural network optimization method for bearing fault diagnosis," *Proceeding of the CSEE*, vol. 39, no. 15, pp. 4558–4567, 2019.

[22] H. Hu, R. Peng, Y. W. Tai, and C. K. Tang, "Network trimming, a data-driven neuron pruning approach towards efficient deep architectures," 2016, http://arxiv.org/abs/1607.03250.

[23] Y. Zhang, K. Xing, R. Bai et al., "An enhanced convolutional neural network for bearing fault diagnosis based on time–frequency image," *Measurement*, vol. 157, Article ID 107667, 2020.

[24] D. Zhao, T. Wang, and F. Chu, "Deep convolutional neural network based planet bearing fault classification," *Computers in Industry*, vol. 107, pp. 59–66, 2019.

[25] H. Li, Q. Zhang, X. Qin et al., "K-SVD-based WVD enhancement algorithm for planetary gearbox fault diagnosis under a CNN framework," *Measurement Science and Technology*, vol. 31, no. 2, Article ID 025003, 2019.

[26] X. B. Jin, N. X. Yang, X. Y. Wang, Y. T. Bai, T. L. Su, and J. L. Kong, "Hybrid deep learning predictor for smart agriculture sensing based on empirical mode decomposition and gated recurrent unit group model," *Sensors*, vol. 20, no. 5, p. 1334, 2020.

[27] Y. Yang, Z. Peng, W. Zhang, and G. Meng, "Parameterised time-frequency analysis methods and their engineering applications: a review of recent advances," *Mechanical Systems and Signal Processing*, vol. 119, pp. 182–221, 2019.

[28] Y. Yang, W. Zhang, Z. Peng, and G. Meng, "Multicomponent signal analysis based on polynomial chirplet transform," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 9, pp. 3948–3956, 2013.

[29] Y. Yang, X. J. Dong, Z. K. Peng, W. M. Zhang, and G. Meng, "Vibration signal analysis using parameterized time-frequency method for features extraction of varying-speed rotary machinery," *Journal of Sound and Vibration*, vol. 335, pp. 350–366, 2015.

[30] X. Dong, S. Chen, G. Xing, Z. Peng, W. Zhang, and G. Meng, "Doppler frequency estimation by parameterized time-

frequency transform and phase compensation technique," *IEEE Sensors Journal*, vol. 18, no. 9, pp. 3734–3744, 2018.

[31] W. Lu, J. Xie, H. Wang et al., "Parameterized time-frequency analysis to separates," *Journal of Systems Engineering and Electronics*, vol. 28, no. 3, pp. 493–502, 2017.

[32] H. Li, A. Kadav, I. Durdanovic et al., "Pruning filters for efficient convnets," 2016, http://arxiv.org/abs/1608.08710.

[33] J. Frankle and M. Carbin, "The lottery ticket hypothesis, finding sparse; trainable neural networks," 2018, http://arxiv.org/abs/1803.03635.

[34] Z. Liu, J. Li, Z. Shen et al., "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2736–2744, Venice, Italy, October 2017.

[35] P. Molchanov, S. Tyree, T. Karras et al., "Pruning convolutional neural networks for resource efficient inference," 2016, http://arxiv.org/abs/1611.06440.

[36] M. Fan, P. Huang, B. Ye, D. Hou, G. Zhang, and Z. Zhou, "Analytical modeling for transient probe response in pulsed eddy current testing," *NDT & E International*, vol. 42, no. 5, pp. 376–383, 2009.

[37] X. Yin, Q. Zhang, H. Wang et al., "RBFNN-based minimum entropy filtering for a class of stochastic nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 65, no. 1, 2019.

[38] Y. Yang, Z. K. Peng, X. J. Dong et al., "General parameterized time-frequency transform," *IEEE Transactions on Signal Processing*, vol. 62, no. 11, pp. 2751–2764, 2014.

[39] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[40] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, http://arxiv.org/abs/1409.1556.

[41] S. Ioffe and C. Szegedy, "Batch normalization, accelerating deep network training by reducing internal covariate shift," 2015, http://arxiv.org/abs/1502.03167.