applied sciences

MDPI

Article

SDD-CNN: Small Data-Driven Convolution Neural Networks for Subtle Roller Defect Inspection

Xiaohang Xu [1], Hong Zheng [1,*], Zhongyuan Guo [1], Xiongbin Wu [1] and Zhaohui Zheng [2]

[1] School of Electronic Information, Wuhan University, Wuhan 430072, China; xuxiaohang@whu.edu.cn (X.X.); guozhongyuan1993@outlook.com (Z.G.); xbwu@whu.edu.cn (X.W.)
[2] Department of Public Courses, Wuhan Railway Vocational College of Technology, Wuhan 430205, China; zhengzhaohui@sina.cn
* Correspondence: zh@whu.edu.cn

Received: 13 January 2019; Accepted: 27 March 2019; Published: 31 March 2019

Abstract: Roller bearings are some of the most critical and widely used components in rotating machinery. Appearance defect inspection plays a key role in bearing quality control. However, in real industries, bearing defects are usually extremely subtle and have a low probability of occurrence. This leads to distribution discrepancies between the number of positive and negative samples, which makes intelligent data-driven inspection methods difficult to develop and deploy. This paper presents a small data-driven convolution neural network (SDD-CNN) for roller subtle defect inspection via an ensemble method for small data preprocessing. First, label dilation (LD) is applied to solve the problem of an imbalance in class distribution. Second, a semi-supervised data augmentation (SSDA) method is proposed to extend the dataset in a more efficient and controlled way. In this method, a coarse CNN model is trained to generate ground truth class activation and guide the random cropping of images. Third, four variants of the CNN model, namely, SqueezeNet v1.1, Inception v3, VGG-16, and ResNet-18, are introduced and employed to inspect and classify the surface defects of rollers. Finally, a rich set of experiments and assessments is conducted, indicating that these SDD-CNN models, particularly the SDD-Inception v3 model, perform exceedingly well in the roller defect classification task with a top-1 accuracy reaching 99.56%. In addition, the convergence time and classification accuracy for an SDD-CNN model achieve significant improvement compared to that for the original CNN. Overall, using an SDD-CNN architecture, this paper provides a clear path toward a higher precision and efficiency for roller defect inspection in smart manufacturing.

Keywords: subtle defect inspection; small data driven; convolution neural network; semi-supervised data augmentation; roller bearings

1. Introduction

As an important basic part of the machinery industry, the roller bearing has been widely used in many fields of national industries, such as aerospace, transportation, and machinery manufacturing. The roller bearing has the function of supporting and reducing friction and is one of the most indispensable and vulnerable parts in rotating machinery [1]. The quality of the roller surface directly affects the performance and service life of mechanical equipment. Therefore, accurately and efficiently inspecting the surface defects of rollers is of great significance in improving the yield of rollers, the stable operation of machinery, and the production of enterprises.

At this stage, there are several ways to inspect the surface defects of rollers: (1) Manual detection. As a traditional detection method, it has a low efficiency and is easily affected by subjective judgment, resulting in an unstable detection accuracy. With the development of industrial automation, manual testing has been gradually eliminated in modern production; (2) nondestructive testing schemes based

on ultrasound, eddy current, etc. Such methods have the disadvantages of a complicated operation and slow detection speed and are difficult to promote; (3) detection methods based on traditional machine vision (MV). Such methods have made significant progress and yield reliable results in many cases [2], but specific preprocessing approaches are required to extract representative features with expert knowledge [3]. However, the wide variety of rollers and the short cycle of product upgrades all require the feature representation to be redesigned from scratch; (4) the deep-learning-based fault diagnosis method for spectrum signals. As a fast and effective data analysis method, deep learning has been widely used by researchers and has caused breakthroughs in image recognition, computer vision, and other fields [4]. Compared with the traditional MV method, the deep-learning-based method can dig out the deep structure of data and automatically learn the representative features. Z. Chen et al. [5] adopted a method based on deep neural networks (DNN), including the deep Boltzmann machine (DBM), deep belief network (DBN), and stacked autoencoder (SAE), to diagnose and classify the frequency spectrum signals of rollers. W. Zhang et al. [6] constructed their own convolutional neural network (CNN) architecture for spectral signal detection. Similar work also includes models deployed by researchers such as H. Shao et al. [7], X. Li et al. [8], and M. Gan et al. [9]. Such advanced methods have achieved a high detection accuracy, but they still cannot meet the requirements of some enterprises for visual detection.

Therefore, using the deep-learning-based method to visually detect the surface defects of rollers is a problem that is unavoidable and worth studying. At present, this method still has many difficulties and challenges. As a data-driven method, the deep learning model often requires massive amounts of data for training to achieve a stable and accurate classification performance. However, in actual production, the appearance of roller defects is very subtle and the probability of occurrence is low, making the number of positive and negative samples extremely unbalanced. Therefore, obtaining a high-performance classifier by training a small amount of data is critical.

This paper proposes a small data-driven convolutional neural network (SDD-CNN) for roller surface defect inspection and classification. First, a label dilation (LD) method is adopted to solve the problem of imbalance in the number of samples between categories. Second, a semi-supervised data augmentation (SSDA) method is proposed for the problem of insufficient defect samples. This method first trains a coarse network to generate a characteristic response intensity map for each sample. Then, pseudorandom cropping is performed on each sample according to the probability value of the intensity map to achieve more accurate data expansion. Third, four state-of-the-art CNN architectures are used to train the sample images of the roller surface, obtain a high-performance classifier, and finally achieve the visual detection of roller surface defects based on deep learning.

2. Related Works

The next generation of industry, namely Industry 4.0, holds the promise of increased flexibility in manufacturing along with higher customization, better quality, and improved productivity [10]. With the widespread deployment of sensors and the Internet of Things, the need to handle big manufacturing data has become highly desirable. In this sense, deep learning plays a critical role in processing and analyzing these big data [11]. In recent years, deep-learning-based algorithms and applications have made remarkable progress in the industrial field. This has effectively promoted the development of intelligent manufacturing and accelerated the arrival of Industry 4.0.

There are three main types of deep-learning-based applications in intelligent manufacturing: (1) For product quality inspection. CNNs, originally designed for image analysis, are a good fit for automatic defect identification in surface integration detection. D. Weimer et al. [12] designed a deep CNN architecture and realized the automatic extraction of features in industrial detection through the optimization of hyperparameters. R. Ren et al. [13] proposed a general method based on deep learning to realize automatic surface inspection. Similar applications include the work of Y.J. Cha et al. [14] and J.K. Park et al. [15]; (2) for mechanical fault assessment. In this kind of application, a CNN integrates feature learning and fault diagnosis into one model. This has been applied in many product

types, including gearboxes [16,17], generators [18], and rotors [19]; (3) predictive prognosis for system maintenance. In this kind of application, a deep recurrent neural network (RNN) is used to model the historical data of the system state and predict the possible various states to guide the producer to maintain the system and realize the strategy of intelligent maintenance. R. Zhao et al. [20] designed a gated recurrent unit network based on local features to express local feature sequences, and realized health monitoring on three machines. Y. Wu et al. [21] realized the estimation of the remaining service life of aircraft turbofan engines by exploring a long short-term memory (LSTM) network. Similar applications also include the work of R. Zhao et al. [22].

In the selection of a network architecture, many deep-learning-based models can be found in the applications of intelligent industry. Specifically, these include: (1) CNN. This type of network is deployed in the above work [12–19]; (2) the restricted Boltzmann machine (RBM) and its variants. P. Wang et al. [23], J. Deutsch et al. [24], and W. Zhang et al. [25] used a DBN to monitor the operating state of a system; (3) AE and its variants. W. Sun et al. [26], Z. Yang et al. [27], and L. Wang et al. [28] explored unsupervised feature extraction by SAE; (4) RNN and its variants. This kind of network was adopted in the above work [20–22].

3. Materials and Methods

3.1. Overview of Proposed Method

As shown in Figure 1, the SDD-CNN-based roller defect inspection approach proposed in this paper includes raw image acquisition, roller ring region expansion, surface sample acquisition, small dataset preprocessing, CNN model training, and classification. Details are as follows:

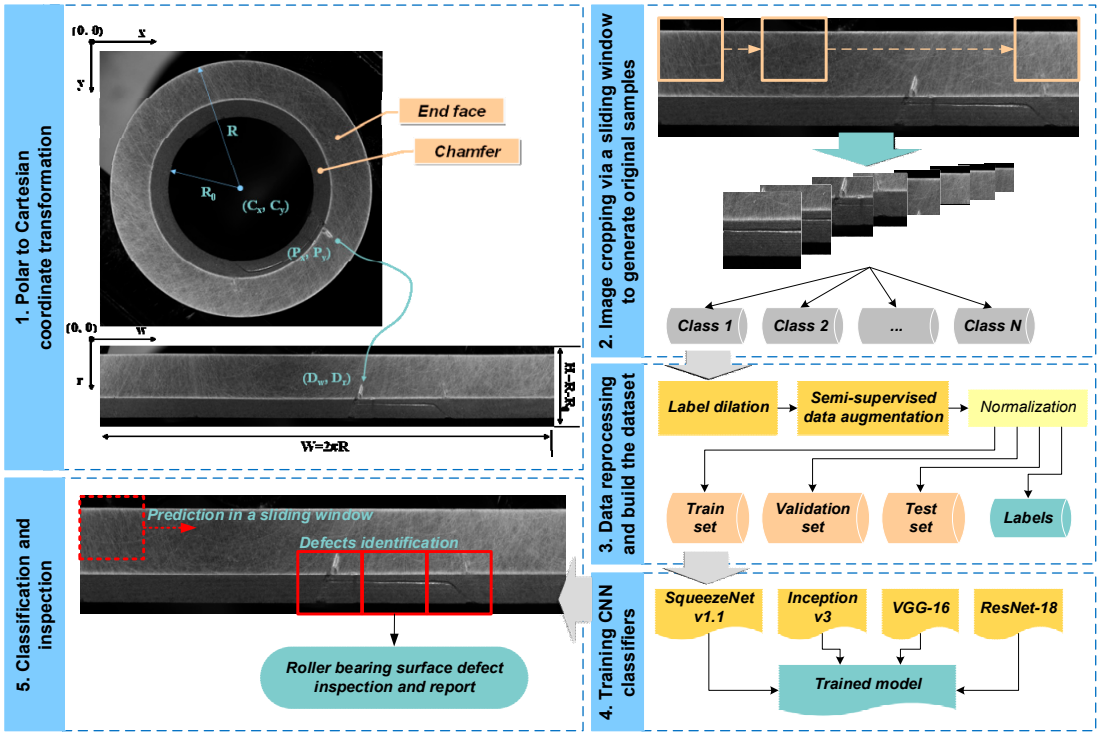


Figure 1. Framework and flowchart of proposed small data-driven convolution neural network (SDD-CNN) for roller defect inspection.

(1) Capture raw roller images using a monocular camera and a customized light source system. This paper obtains 194 raw images with 2592 × 1944 pixel resolutions. To improve the robustness of the classifier, these images cover three different lighting conditions by adjusting the brightness of the light source system: perfect brightness (PB), 1.2PB, and 0.8PB [4];

(2) To facilitate the inspection algorithm, the ring-shape image of the roller is transformed into a rectangle by using a Polar-to-Cartesian (P2C) coordinate transformation [2]. Let (C_x, C_y) denote the center of the roller; R and R_0 denote the outer radius and inner radius of the roller, respectively; and (P_x, P_y) and (D_w, D_r) represent the coordinates of the corresponding points before and after the transformation, respectively. Then, the P2C transformation can be described as

$$\begin{cases} x = C_x + (r + R_0) * \cos(\theta) \\ y = C_y - (r + R_0) * \sin(\theta) \end{cases} \quad (1)$$

where $\theta = 2\pi w/W$. To ensure the quality of the resulting image, the bilinear interpolation method is adopted to obtain a proper pixel value;

(3) Use a sliding window to crop the rectangular roller image into a small image suitable for CNN training and prediction (with a resolution of 256 × 256). Then, screen and label the cropped image [29]. Figure 2 displays examples of the 11 categories of roller surface samples. In this paper, 830 surface samples were obtained, and the distribution of different categories is listed in Table 1;

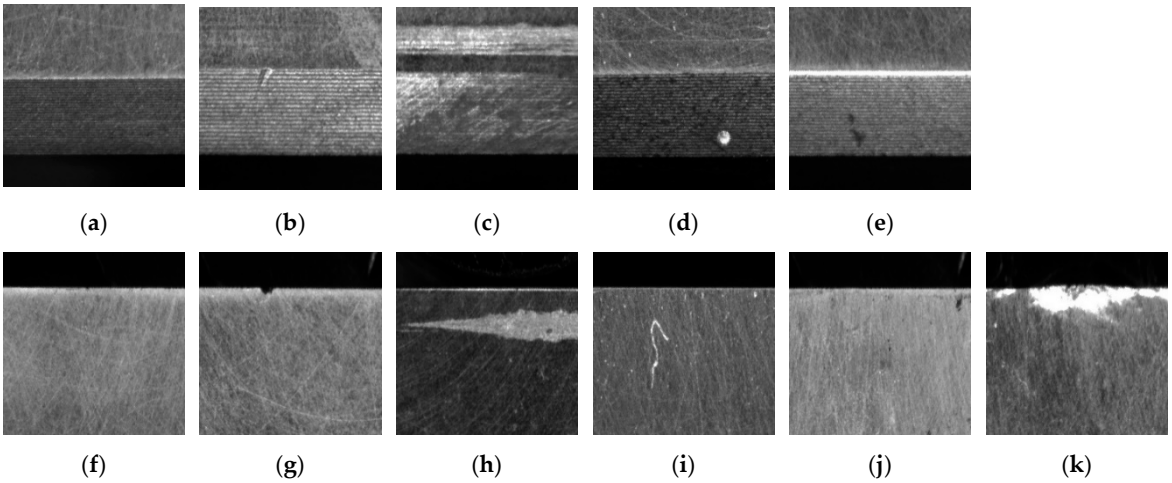


Figure 2. Examples of different roller samples: (**a**) CQ: chamfer qualified; (**b**) CC: chamfer cracks; (**c**) CI: chamfer indentations; (**d**) CSc: chamfer scratches; (**e**) CSt: chamfer stains; (**f**) EFQ: end-face qualified; (**g**) EFC: end-face cracks; (**h**) EFI: end-face indentations; (**i**) EFSc: end-face scratches; (**j**) EFSt: end-face stains; (**k**) EFSF: end-face serious fracture.

Table 1. Distribution of different categories of initial roller dataset.

Surface Type	EFQ	EFC	EFI	EFSc	EFSt	EFSF
Number of samples	300	94	14	32	18	44
Surface type	CQ	CC	CI	CSc	CSt	
Number of samples	200	70	31	6	21	

(4) Small data preprocessing. The initial roller dataset is expanded for the first time by the LD method for the sample imbalance problem. Then, the SSAD method is adopted to expand the dataset a second time. Finally, the dataset is divided into three parts: a training set, verification set, and test set, at a ratio of 3:1:1 [30];

(5) Sample training is conducted using four state-of-the-art CNN architectures, namely, SqueezeNet v1.1 [31], Inception v3 [32], VGG-16 [33], and ResNet-18 [34]. The trained model is then used to classify the roller image through a sliding window to obtain the final inspection result.

3.2. Small Data Preprocessing

3.2.1. Label Dilation

Figure 2 shows that the appearance characteristics of roller defects are very subtle, and it is difficult to catch defective rollers in actual production. This leads to an extreme imbalance in the number of positive and negative samples. Inspired by the work of L. Shen et al. [35], this paper adopts the LD method to expand the number of defect samples. The algorithm is described in Table 2.

Table 2. Algorithm steps of label dilation.

Input:
k categories of target samples: $\{C_1, C_2, \ldots, C_k\}$,
each of which has a number of $\{N_1, N_2, \ldots, N_k\}$;
Process:
For the most numerous category C_m, the sample size is N_m, and
the sample order is $P_{m0} = \{1, 2, \ldots, N_m\}$.
Randomly scramble the sample order to $P_{m\text{-}rand} = \{p_1, p_2, \ldots, p_{Nm}\}$.
Output:
For any other categories C_i ($i = 1, 2, \ldots, k, i \neq m$),
the original sample order is $P_{i0} = \{1, 2, \ldots, N_i\}$, and
the expanded sample order is
$P_{i\text{-}LD} = \{p_1 \bmod N_i, p_2 \bmod N_i, \ldots, p_{Nm} \bmod N_i\}$.

After the LD operation, the sample number of all categories is consistent with the sample number of the largest category before expansion. In this paper, the method is combined with the distribution of the initial roller dataset to expand the number of all end-face categories to 300 and the number of all chamfer categories to 200. In this way, the sample imbalance problem is solved, and the randomness of the dataset is maintained.

3.2.2. Semi-Supervised Data Augmentation

After the sample imbalance is resolved, the total sample size is still insufficient to support deep network training. Therefore, further data enhancement is required. In general, data enhancement methods include random cropping, scaling, rotation, flipping, and adding random noise. For the roller sample, the defects may appear at the edge of the picture because they are subtle and the position is not fixed. As a result, the range of random cropping is not likely to include the defect and its vicinity. Thus, the cropped sample no longer belongs to the original label category. Obviously, this goes against the label-preserving principle of data augmentation [4]. Inspired by the work of B. Zhou et al. [36], this paper proposes an SSAD method. The algorithm is as follows:

(1) As shown in Figure 3, a coarse classifier is obtained by training the GoogLeNet network with the label dilated dataset; (2) an occlusion experiment [37] is conducted to generate the characteristic response intensity map of all samples at the uppermost convolutional layer of the network; (3) a location is randomly selected around the region where the intensity $I < 0.4$. This location is mapped to the original image and uses it as a crop center; (4) each sample is cropped three times, and the cropped size is 244×244; (5) other steps are consistent with the method in GoogLeNet paper [38]. Using the SSAD method, the original label information is preserved and the diversity of the data is enhanced. This can effectively prevent overfitting and improve the inference ability of the model.

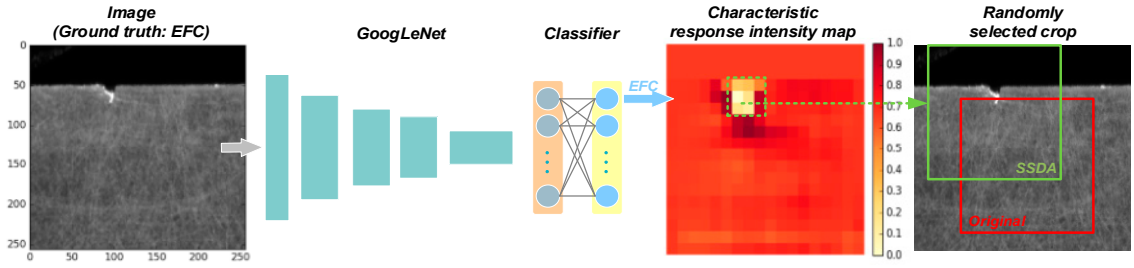


Figure 3. Flowchart of the proposed semi-supervised data augmentation (SSAD) method.

3.3. CNN Architectures

In this section, four state-of-the-art CNN architectures are briefly introduced: SqueezeNet v1.1, Inception v3, VGG-16, and ResNet-18. Among them, the VGG and ResNet architectures have the ability to extend to ultra-deep networks. However, considering the efficiency of actual deployment and the fairness of the comparison, this paper selects VGG-16 and ResNet-18 that have similar numbers of layers as those of the former two networks.

3.3.1. SqueezeNet v1.1

In recent years, much of the research on deep convolution networks has focused on improving the accuracy. At the same accuracy level, a smaller CNN architecture can provide more efficient distributed training, a smaller parameter model, and more suitability for deployment on memory-constrained devices such as FPGAs. SqueezeNet achieves the same accuracy as AlexNet [4] on ImageNet, but only uses 1/50 of the parameters of AlexNet.

The Fire module is the key to implementing SqueezeNet, which is composed of a squeeze layer and an expand layer. The squeeze layer only uses 1×1 convolution filters, which reduces the number of parameters by nine times compared with a 3×3 convolution filter. The expand layer has a mix of 1×1 and 3×3 filters, as shown in Figure 4. When the number of convolution filters meets

$$s_{1x1} < e_{1x1} + e_{3x3} \tag{2}$$

it is helpful to reduce the number of input channels of the 3×3 group. In addition, in SqueezeNet, the pooling layer is placed late in the network, providing a larger activation map for the convolutional layers. A larger activation map retains more information and provides a higher classification accuracy. In summary, these strategies allow SqueezeNet to dramatically reduce the number of parameters while maintaining the accuracy.

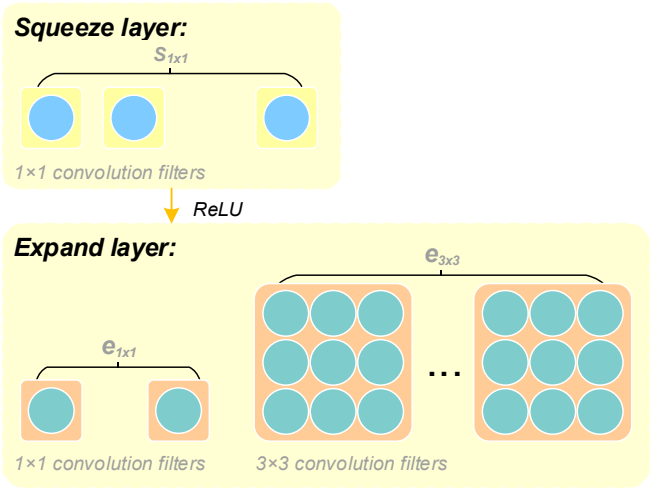


Figure 4. Organization of convolution filters in Fire module.

3.3.2. Inception v3

C. Szegedy et al. introduced new design principles and optimization ideas based on GoogLeNet and proposed Inception v3. The paper points out that the representational bottleneck caused by severe compression should be avoided in the early stage of a network. In addition, higher dimensional representations are more suitable for processing locally within a network. Accordingly, Inception v3 optimizes and improves Inception v1 from the following three aspects: (1) factorize convolutions with a large filter size, including the use of a multilayer perceptron to replace a 5×5 filter and the special factorization of a 3×3 filter into an asymmetric structure of 3×1 and 1×3; (2) introduce auxiliary classifiers with a regularization effect to accelerate the convergence of the network; (3) parallel a convolutional layer with *stride* = 2 and a pooling layer to achieve more efficient grid size reduction.

Noteworthily, Inception v3 proposed a mechanism to regularize the classifier layer by estimating the marginalized effect of label-dropout during training. For each training example x, the model computes the probability of each label $k \in \{1 \ldots K\}$:

$$p(k|x) = \frac{\exp(z_k)}{\sum_{i=1}^{K} \exp(z_i)} \tag{3}$$

where z_i represents the logits or unnormalized log-probabilities. Considering the ground-truth distribution over labels $q(k \mid x)$ for this training example, the loss can be defined as the cross entropy:

$$l = -\sum_{k=1}^{K} \log(p(k)) q(k) \tag{4}$$

Minimizing this is equivalent to maximizing the expected log-likelihood of a label. Additionally, the cross-entropy loss is differentiable with respect to z_k and the gradient has a simple form:

$$\frac{\partial l}{\partial z_k} = p(k) - q(k) \tag{5}$$

Consider the case of a single ground-truth label y, so that $q(y) = 1$ and $q(k) = 0$ for all $k = y$. For a particular example x with label y, the log-likelihood is maximized for $q(k) = \delta_{k,y}$, where $\delta_{k,y}$ is Dirac delta, which equals 1 for $k = y$ and 0 otherwise. This maximum is not achievable for finite z_k, but is approached if $z_y >> z_k$ for all $k \neq y$. This, however, can cause over-fitting and encourage the differences between the largest logit and all others to become large, which reduces the ability of the model to adapt. In order to encourage the model to be less confident about its predictions, a distribution over labels $u(k)$, independent of the training example x, and a smoothing parameter ϵ are introduced to replace the $q(k \mid x)$ with

$$q'(k \Big| x) = (1 - \epsilon)\delta_{k,y} + \epsilon u(k) \tag{6}$$

In this paper, considering that the labels of the roller dataset have been normalized by the LD method, the uniform distribution $u(k) = 1/K$ is used so that

$$q'(k) = (1 - \epsilon)\delta_{k,y} + \frac{\epsilon}{K} \tag{7}$$

where $K = 11$ classes and $\epsilon = 0.1$.

3.3.3. VGG-16

K. Simonyan et al. discussed the effect of network depth on image recognition accuracy based on AlexNet, and achieved a better performance by reducing the size of the filters and increasing the number of convolutional layers. For the first time, VGG pushed the CNN depth to 16–19 layers. The VGG-16 used in this paper has the following characteristics in the network design: (1) all the convolution layers use 3×3 or 1×1 filters to reduce the number of model parameters; (2) the pooling layer adopts maximum pooling, and is only placed behind the 2nd, 4th, 7th, 10th, and 13th convolution

layers to enhance the ability of feature expression. The configuration for the convolution layers is shown in Figure 5.

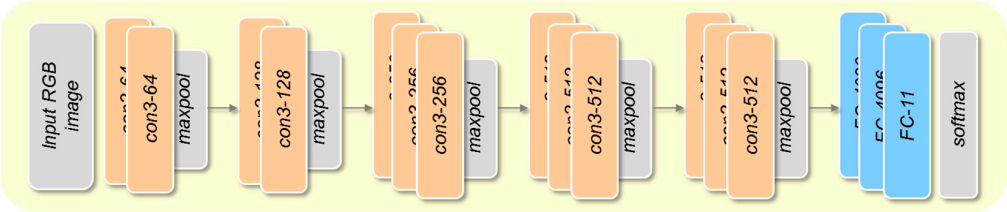


Figure 5. ConvNet configurations of VGG-16.

3.3.4. ResNet-18

After VGG was proposed, researchers found that with the gradual deepening of the network, the phenomenon of a decreased accuracy appeared, for which K.M. He et al. proposed a new network, namely, the deep residual network. In a traditional convolutional network or fully connected network, information loss occurs more or less in the process of information transmission. At the same time, gradient disappearance or gradient explosion may be caused, which makes the deep network unable to train.

As shown in Figure 6, ResNet solves this problem to some extent by directly detouring the input information to the output and protecting the integrity of the information. Therefore, the next layer only needs to learn the part of the difference between the input and output of the previous layer. This simplifies the learning objective and the difficulty of convergence. In a specific implementation, ResNet mainly has two kinds of residual modules: one connects two 3×3 filters in series, and the other uses 1×1, 3×3, and 1×1 filters in series.

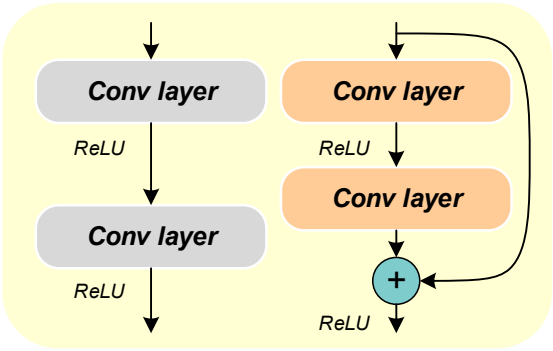


Figure 6. Comparison of a plain block and residual block.

4. Experiments and Results

4.1. Experimental Setup

A powerful machine is used for these experiments. Its specifications are as follows:

CPU: Intel E3-1230 V2*2 (3.30 GHz);
Memory: 16 GB DDR3;
GPU: NVIDIA GTX-1080Ti.
The software platform used is the following:
Ubuntu 16.04 LTS;
Visual Studio Code with Python 2.7.

All of the following experiments are conducted using our own fork of Caffe [39], which is a rapid, open-source framework for deep learning.

The roller dataset used in the experiment is augmented, with the specific number of samples in each category shown in Table 3. There are 13,440 samples in the four categories. Among these, 27,630 are used as a training set, 4480 as a verification set, and the remaining 4,480 as a test set.

Table 3. Distribution of 11 categories of roller datasets.

Type	EFQ/EFC/EFI/EFSc/EFSt/EFSF	CQ/CC/CI/CSc/CSt
Training set	1440	960
Validation set	480	320
Test set	480	320
Total number	2400	1600

4.2. Network Training and Performance Metrics

The four CNN architectures mentioned in Section 3.3 are trained for the roller surface classification task using three different strategies. Two of these strategies refer to training the CNN from scratch, which means starting from a random configuration of weights. The first from-scratch strategy uses a dataset augmented by a method similar to that in the GoogLeNet paper, while the second strategy uses the dataset augmented successively by the LD and SSAD methods. The third strategy is based on transfer learning from pretrained networks, which are respectively obtained from the corresponding papers cited above. Here, all of the network layers are fine-tuned, including convolution layers and fully connected layers. From this perspective, the approach is called a deep transfer strategy. The third strategy uses a dataset identical to that of the second one. We call the second and third networks an SDD-CNN because of the small data-driven augmentation method that they use. Note that these 12 training configurations (4 CNN architectures $\times$ 3 strategies) use the same hyperparameter values (*momentum* 0.9, *weight decay* 0.0005, *learning rate* 0.005, and *batch size* 32).

The number of network layers and parameters for each CNN model is displayed in Table 4. SqueezeNet benefits from its special Fire module, and the parameter size is the smallest of the four models. Inception v3 and ResNet18 have a similar number of parameters, and VGG16 is more than five times larger than Inception v3.

Table 4. Training strategies and basic scales for four CNN models.

Model Name.	Training Type	Number of Layers	Number of Parameters
SqueezeNet v1.1	From scratch	18	728,139
SDD-SqueezeNet v1.1	From scratch		
SDD-SqueezeNet v1.1	Deep transfer		
Inception v3	From scratch	18	24,734,048
SDD-Inception v3	From scratch		
SDD-Inception v3	Deep transfer		
VGG16	From scratch	16	134,305,611
SDD-VGG16	From scratch		
SDD-VGG16	Deep transfer		
ResNet18	From scratch	18	11,196,107
SDD-ResNet18	From scratch		
SDD-ResNet18	Deep transfer		

Figure 7 shows a comparison of the time and number of iterations required to achieve convergence in the training of different network models. Obviously, in both strategies that train from scratch, SDD-CNN requires less training time and iteration times than the original CNN. This is because, for the roller defect image, the dataset of SDD-CNN preserves the defect features more completely, which enables the model to extract the target features more precisely and to reach the convergence more quickly. Specifically, the convergence time of the SDD-CNN can be reduced by at least 20% over

the original CNN, and in the case of ResNet18, this number is up to 42%. With regard to the number of iterations, the SDD-CNN can also be reduced by at least 20%; for ResNet18, the figure is 36%. The SDD-CNN from deep transfer learning benefits from its pretraining model, and the convergence time and iteration times are less than those of the general SDD-CNN. Conversely, from the perspective of architecture, Inception v3 has the shortest convergence time no matter which training strategy is adopted. Considering that this is not a small number of parameters, this convergence rate is quite good and should be attributed to its efficient architecture design. VGG16 is the slowest of the four architectures in terms of convergence speed because it has a huge weight to optimize.

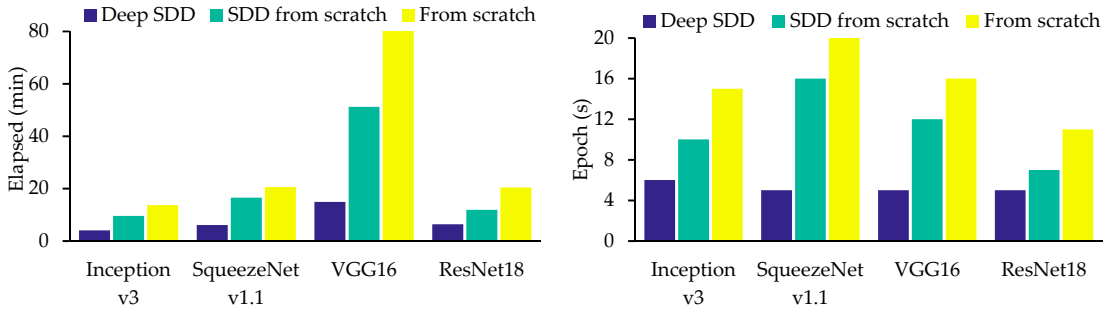


Figure 7. Comparison of elapse and epoch for convergence for different models.

In addition to surveying the convergence speed of the model, the accuracy of the verification set is an important indicator when considering the training performance. Figure 8 illustrates that, regardless of the architecture that is used, the accuracy of the SDD-CNN is higher than the original CNN for almost all types of defect. Specifically, for the CI, CSc, EFC, EFI, EFSc, and EFSt types of defect, both SDD-CNN and the original CNN exhibit an extremely high performance. In the remaining CQ, CC, CSt, EFQ, and EFSF types of defect, the accuracy of the SDD-Inception v3 increases by 4.46% over the Inception v3 on average, and this number is up to 8.13% for the CQ defect. The SDD-SqueezeNet v1.1 increases by 6.62% on average, which is up to 12.18% for the CQ defect. The SDD-VGG16 increases by 7.41% on average, which is up to 11.04% for the EFQ defect. The SDD-ResNet18 increases by 6.75% on average, which is up to 14.38% for the CQ defect. Furthermore, as shown in Figure 9, the SDD-CNN is always superior to the original CNN in terms of convergence speed and accuracy, regardless of the network architecture. This verifies the effectiveness and robustness of the method proposed in this paper in the classification and detection of roller surface defects.

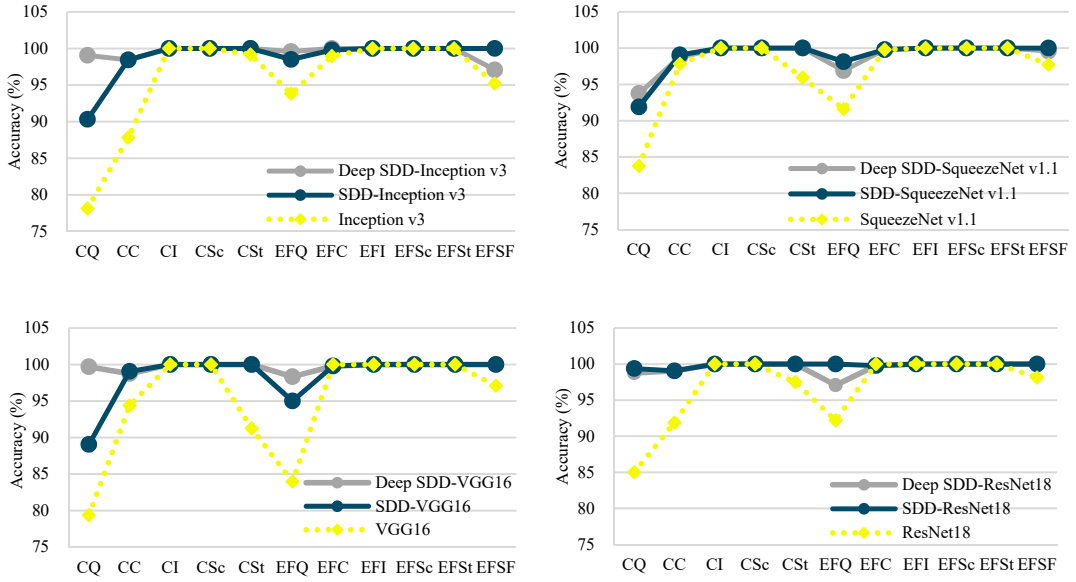


Figure 8. Comparison of validation accuracy for different models with different strategies.

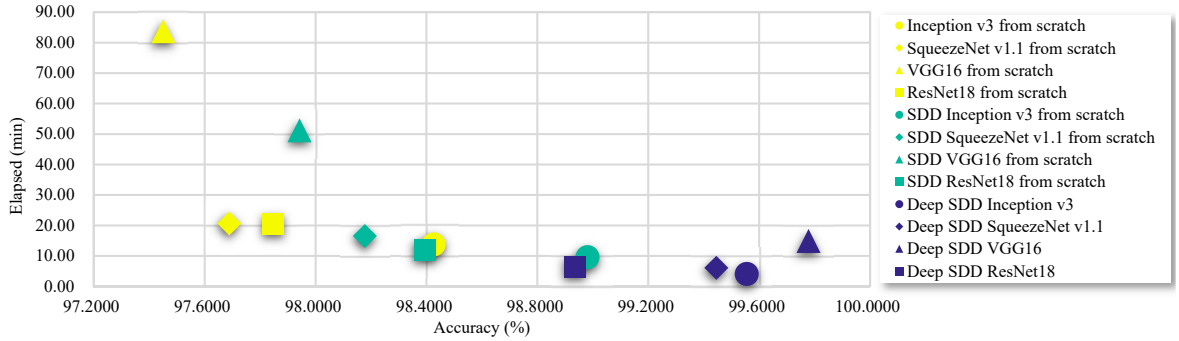
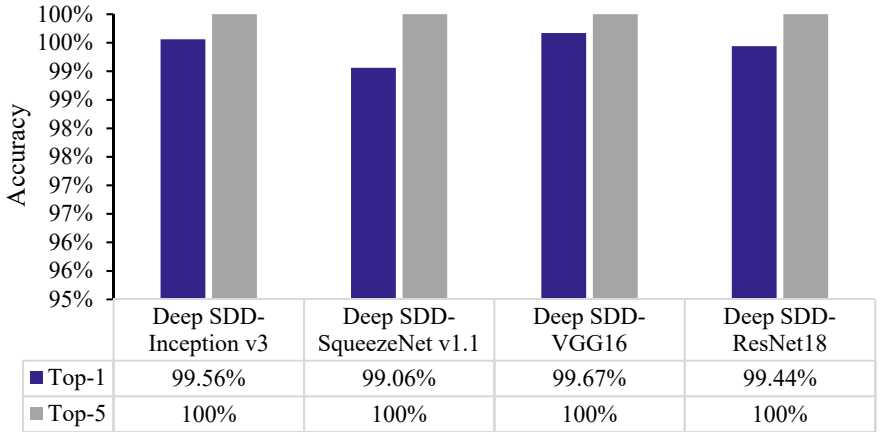


Figure 9. Diagram of convergence time and validation accuracy.

Furthermore, this paper compares the performance of four deep SDD-CNNs for the test set. Figure 10 shows that the Top-1 accuracy [4] of all four models exceeds 99%, and the Top -5 accuracy reaches 100%. Among them, SqueezeNet v1.1 has the lowest Top-1 accuracy, and Inception v3 and VGG16 both exceed 99.5%. Figure 11 shows the recall for each type of roller surface defect. It can be seen that the four models have a recall of 100% in the six defects of CI, CSc, CSt, EFI, EFSc, and EFSt, thus showing good stability. Because more than 60% of the region of the roller surface is of the end-face, and cracks are some of the most common defects on it [2], the recall of the two categories EFQ and EFC is a critical indicator for evaluating the performance of the classifier. Inception v3 has a recall of more than 99% in both categories. Considering its shortest training time and outstanding classification performance, it is the most suitable network model among the four architectures for the classification of roller surface defects.



	Deep SDD-Inception v3	Deep SDD-SqueezeNet v1.1	Deep SDD-VGG16	Deep SDD-ResNet18
Top-1	99.56%	99.06%	99.67%	99.44%
Top-5	100%	100%	100%	100%

Figure 10. Top-1 and Top-5 accuracy of test set across different models.

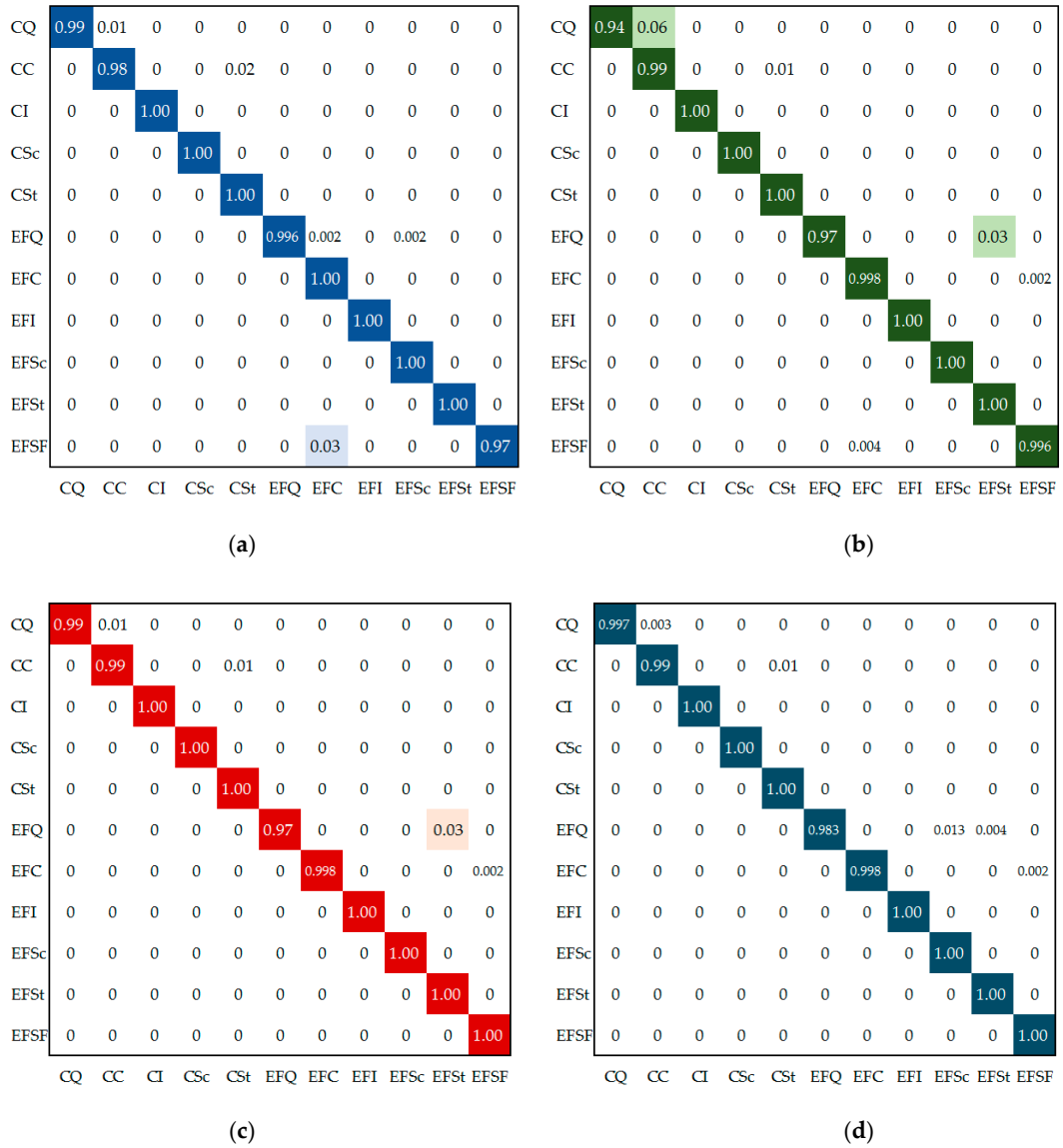


Figure 11. Confusion matrixes for different models: (**a**) deep SDD-Inception v3; (**b**) SDD-SqueezeNet v1.1; (**c**) deep SDD-ResNet18; (**d**) deep SDD-VGG16.

4.3. Model Visualizations

To evaluate a deep learning model more effectively, the most intuitive method shows the distribution of samples in the feature space directly. T-distributed stochastic neighbor embedding (t-SNE) [40,41] is currently the most popular dimension reduction algorithm for high-dimensional data and is commonly used for deep network visualization [11].

Figure 12 shows the t-SNE distribution of all training samples in the above four deep SDD-CNNs. Apparently, in the feature space of the deep SDD-Inception v3 model, all 11 types of roller samples have an excellent degree of discrimination. This result is also consistent with the conclusions of the previous section. The other three models (deep SDD-SqueezeNet v1.1, deep SDD-VGG16, and deep SDD-ResNet18) have some categories that do not achieve perfect segmentation. The discrimination of deep SDD-SqueezeNet v1.1 for the CC and CQ samples is particularly low, which is also consistent with the recall in Figure 11. In summary, no matter what the convergence speed, accuracy, or distribution of the feature space, the deep SDD-Inception v3 model has an excellent performance in the classification and detection of roller surface defects.

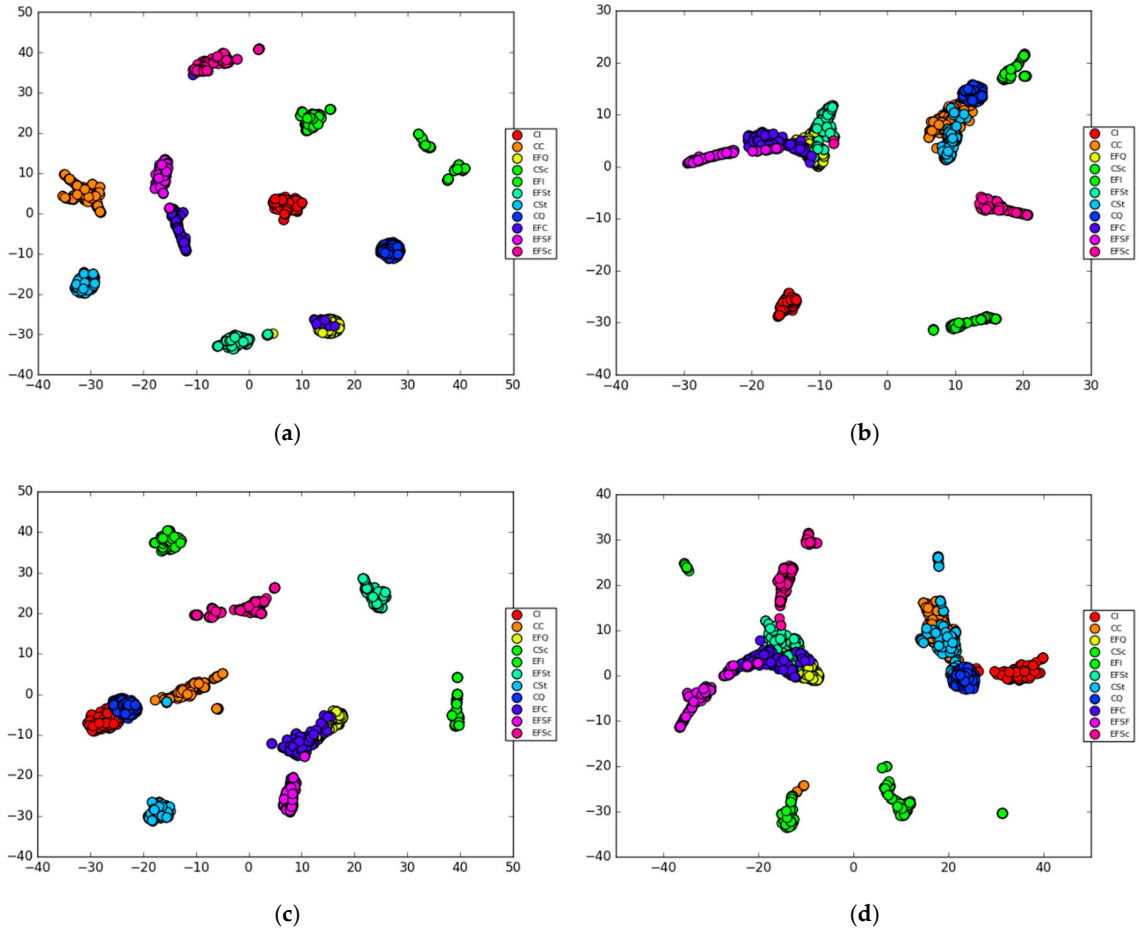


Figure 12. Comparison of t-SNE across the four networks at the end of training: (**a**) Deep SDD-Inception v3; (**b**) Deep SDD-SqueezeNet v1.1; (**c**) Deep SDD-VGG16; (**d**) Deep SDD-ResNet18.

To further explore the classification basis of the deep SDD-Inception v3 model for each category of samples, this paper conducts an occlusion experiment to analyze the corresponding intensities of different regions of the samples. In this experiment, a gray rectangle is used to block each part of the input image to test the characteristic response intensity of the uppermost convolution layer of the network. As shown in Figure 13, when an occlusion occurs in the background region of the sample, the intensity remains substantially unchanged; and when the rectangle blocks the critical position of the defect, it sharply decreases. The deep SDD-Inception v3 model can accurately locate the features of different types of roller surfaces and make precise classification judgments.

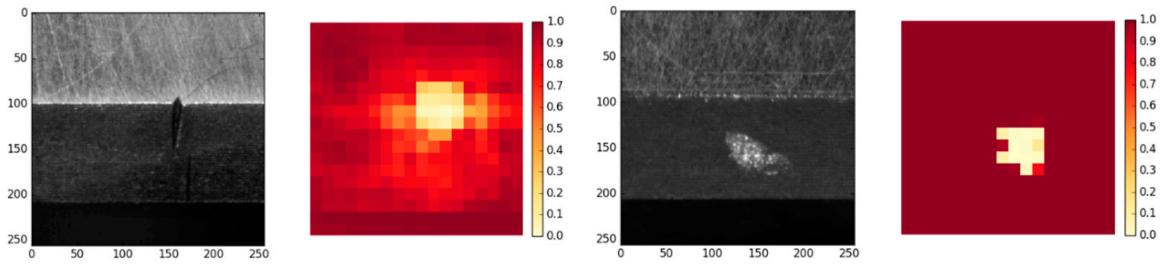


Figure 13. *Cont.*

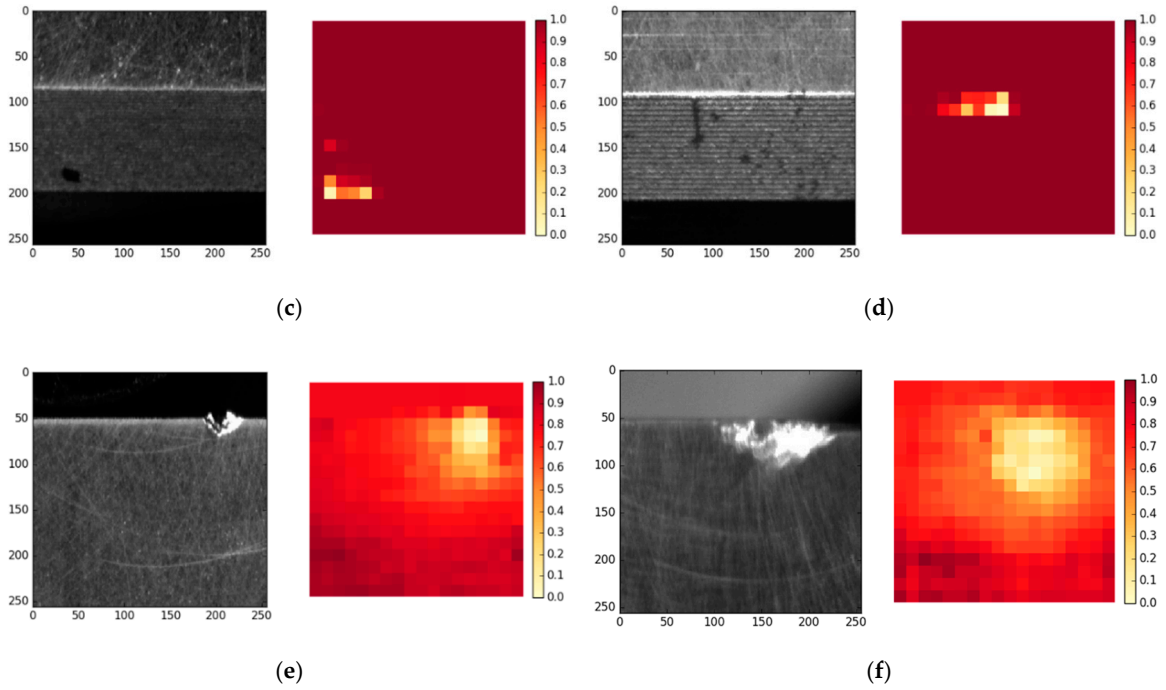


Figure 13. Occlusion experiments on a couple of roller surface defect images: (**a**) CC; (**b**) CSc; (**c**) CSt; (**d**) CSt; (**e**) EFC; (**f**) EFSF.

5. Conclusions

To solve the problem of classification and inspection of subtle defects on roller surfaces by deep learning, a small data-driven convolutional neural network is proposed in this paper. At present, the difficulty of this issue is that the shape of the defects is extremely unobservable and the occurrence probability is low, which makes the number of samples insufficient and the distribution extremely unbalanced. For this reason, this paper first adopted the label dilation method to solve the imbalance in the sample distribution. Then, a semi-supervised data augmentation method was proposed. The feature response map of each sample was generated through a pretrained coarse network to guide the pseudorandom cropping of samples and realize more accurate data expansion. Next, four state-of-the-art CNN architectures were introduced and trained for the roller defect classification task. Finally, a series of experiments were conducted to verify that the proposed SDD-CNN is significantly superior to the original CNN model in terms of convergence speed, training time, and classification accuracy. In particular, the SDD-inception v3 using a deep transfer learning strategy achieved a 99.56% Top-1 accuracy on the test set, and a visualization experiment also proved the reliability of the model. Overall, the deep-learning-based roller defect classifier presented in this paper provides a clear path toward higher reliability and stability in actual production.

Author Contributions: Conceptualization, X.X. and H.Z.; methodology and software, X.X.; validation and formal analysis, Z.G. and Z.Z.; writing, X.X. and Z.G.; supervision and project administration, H.Z.; funding acquisition, X.W.

Funding: This work was supported by the National Natural Science Foundation of China (Grant No. 61771352).

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

References

1. Zhang, B. Research on Surface Defect Detection of Bearing Roller Based on Machine Vision. Master's Thesis, Nanchang Hangkong University, Nanchang, China, June 2018.
2. Shen, H.; Li, S.; Gu, D.; Chang, H. Bearing defect inspection based on machine vision. *Measurement* **2012**, *45*, 719–733. [CrossRef]

3. Zheng, Z.; Ma, Y.; Zheng, H.; Ju, J.; Lin, M. UGC: Real-time, ultra-robust feature correspondence via unilateral grid-based clustering. *IEEE Access* **2018**, *6*, 55501–55508. [CrossRef]
4. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. In Proceedings of the 2012 International Conference on Neural Information Processing Systems, Lake Tahoe, ND, USA, 3–6 December 2012; pp. 1097–1105.
5. Chen, Z.; Deng, S.; Chen, X.; Li, C.; Sanchez, R.; Qin, H. Deep neural networks-based rolling bearing fault diagnosis. *Microelectron. Reliab.* **2017**, *75*, 327–333. [CrossRef]
6. Zhang, W.; Li, C.; Peng, G.; Chen, Y.; Zhang, Z. A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load. *Mech. Syst. Signal Process.* **2018**, *100*, 439–453. [CrossRef]
7. Shao, H.; Jiang, H.; Lin, Y.; Li, X. A novel method for intelligent fault diagnosis of rolling bearings using ensemble deep auto-encoders. *Mech. Syst. Signal Process.* **2018**, *102*, 278–297. [CrossRef]
8. Li, X.; Zhang, W.; Ding, Q. A robust intelligent fault diagnosis method for rolling element bearings based on deep distance metric learning. *Neurocomputing* **2018**, *310*, 77–95. [CrossRef]
9. Gan, M.; Wang, C.; Zhu, C. Construction of hierarchical diagnosis network based on deep learning and its application in the fault pattern recognition of rolling element bearings. *Mech. Syst. Signal Process.* **2016**, *72–73*, 92–104. [CrossRef]
10. Zhong, R.Y.; Xu, X.; Klotz, E.; Newman, S.T. Intelligent manufacturing in the context of industry 4.0: A review. *Engineering* **2017**, *3*, 616–630. [CrossRef]
11. Wang, J.; Ma, Y.; Zhang, L.; Gao, R.X.; Wu, D. Deep learning for smart manufacturing: Methods and applications. *J. Manuf. Syst.* **2018**, *48*, 144–156. [CrossRef]
12. Weimer, D.; Scholz-Reiter, B.; Shpitalni, M. Design of deep convolution neural network architectures for automated feature extraction in industrial inspection. *CIRP Ann. Manuf. Technol.* **2016**, *65*, 417–420. [CrossRef]
13. Ren, R.; Hung, T.; Tan, K.C. A generic deep-learning-based approach for automated surface inspection. *IEEE Trans. Cybern.* **2017**, *99*, 929–940. [CrossRef]
14. Cha, Y.J.; Choi, W.; Suh, G.; Mahmoudkhani, S.; Büyüköztürk, O. Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. *Comput. Aided Civ. Infrastruct. Eng.* **2017**, *33*, 731–747. [CrossRef]
15. Park, J.K.; Kwon, B.K.; Park, J.H.; Kang, D.J. Machine learning-based imaging system for surface defect inspection. *Int. J. Precis. Eng. Manuf. Green Technol.* **2016**, *3*, 303–310. [CrossRef]
16. Chen, Z.Q.; Li, C.; Sanchez, R.V. Gearbox fault identification and classification with convolution neural networks. *Shock Vib.* **2015**, *2015*, 390134.
17. Wang, P.; Ananya, Y.R.; Gao, R.X. Virtualization and deep recognition for system fault classification. *J. Manuf. Syst.* **2017**, *44*, 310–316. [CrossRef]
18. Dong, H.; Yang, L.; Li, H. Small fault diagnosis of front-end speed controlled wind generator based on deep learning. *WSEAS Trans. Circuits Syst.* **2016**, *15*, 64–72.
19. Janssens, O.; Slavkovikj, V.; Vervisch, B.; Stockman, K.; Loccufier, M.; Verstockt, S.; Walle, R.V.d.; Hoecke, S.V. Convolutional neural network based fault detection for rotating machinery. *J. Sound Vib.* **2016**, *377*, 331–345. [CrossRef]
20. Zhao, R.; Wang, D.; Yan, R.; Mao, K.; Shen, F.; Wang, J. Machine health monitoring using local feature-based gated recurrent unit network. *IEEE Trans. Ind. Electron.* **2018**, *65*, 1539–1548. [CrossRef]
21. Wu, Y.; Yuan, M.; Dong, S.; Lin, L.; Liu, Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing* **2017**, *226*, 853–860. [CrossRef]
22. Zhao, R.; Yan, R.; Wang, J.; Mao, K. Learning to monitor machine health with convolution bi-directional LSTM networks. *Sensors* **2017**, *17*, 273. [CrossRef]
23. Wang, P.; Gao, R.X.; Yan, R. A deep learning-based approach to material removal rate prediction in polishing. *CIRP Ann. Manuf. Technol.* **2017**, *66*, 429–432. [CrossRef]
24. Deutsch, J.; He, M.; He, D. Remaining useful life prediction of hybrid ceramic bearings using an integrated deep learning and particle filter approach. *Appl. Sci.* **2017**, *7*, 649. [CrossRef]
25. Zhang, W.; Duan, P.; Yang, L.T.; Xia, F.; Li, Z.; Lu, Q.; Gong, W.; Yang, S. Resource requests prediction in the cloud computing environment with a deep belief network. *Softw. Pract. Exp.* **2017**, *47*, 473–488. [CrossRef]

26. Sun, W.; Shao, S.; Zhao, R.; Yan, R.; Zhang, X.; Chen, X. A sparse auto-encoder-based deep neural network approach for induction motor faults classification. *Measurement* **2016**, *89*, 171–178. [CrossRef]
27. Yang, Z.; Wang, X.; Zhong, J. Representational learning for fault diagnosis of wind turbine equipment: A multi-layered extreme learning machines approach. *Energies* **2016**, *9*, 379. [CrossRef]
28. Wang, L.; Zhao, X.; Pei, J.; Tang, G. Transformer fault diagnosis using continuous sparse auto encoder. *SpingerPlus* **2016**, *5*, 448. [CrossRef] [PubMed]
29. Cha, Y.J.; Choi, W.; Büyüköztürk, O. Deep learning-based crack damage detection using convolutional neural networks. *Comput.-Aided Civ. Infrastruct. Eng.* **2017**, *32*, 361–378. [CrossRef]
30. Mohanty, S.P.; Hughes, D.P.; Marcel, S. Using deep learning for image-based plant disease detection. *Front. Plant Sci.* **2016**, *7*, 1419. [CrossRef] [PubMed]
31. Iandola, F.N.; Han, S.; Moskewicz, M.W. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. *arXiv*, 2016; arXiv:1602.07360.
32. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2818–2826.
33. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv*, **2014**, arXiv:1409.1556.
34. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
35. Shen, L.; Lin, Z.; Huang, Q. Relay backpropagation for effective learning of deep convolutional neural networks. In Proceedings of the 2016 European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 467–482.
36. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2921–2929.
37. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In Proceedings of the 2014 European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2014; pp. 1–16.
38. Szegedy, C. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 8–10 June 2015; pp. 1–9.
39. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 24–27 June 2014; pp. 1–4.
40. Maaten, L.V.d.; Hinton, G. Visualizing data using t-sne. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
41. Maaten, L.V.d. Accelerating t-sne using tree-based algorithms. *J. Mach. Learn. Res.* **2014**, *15*, 3221–3245.