Hindawi
Journal of Sensors
Volume 2021, Article ID 1735386, 18 pages
https://doi.org/10.1155/2021/1735386

Research Article

Classification of Imbalanced Data Using Deep Learning with Adding Noise

Wan-Wei Fan and Ching-Hung Lee

Department of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Hsinchu, Taiwan

Correspondence should be addressed to Ching-Hung Lee; chleenctu@nctu.edu.tw

Received 24 July 2021; Revised 25 October 2021; Accepted 5 November 2021; Published 25 November 2021

Academic Editor: Binghua Cao

This paper proposes a method to treat the classification of imbalanced data by adding noise to the feature space of convolutional neural network (CNN) without changing a data set (ratio of majority and minority data). Besides, a hybrid loss function of crossentropy and KL divergence is proposed. The proposed approach can improve the accuracy of minority class in the testing data. In addition, a simple design method for selecting structure of CNN is first introduced and then, we add noise in feature space of CNN to obtain proper features by a training process and to improve the classification results. From comparison results, we can find that the proposed method can extract the suitable features to improve the accuracy of minority class. Finally, illustrated examples of multiclass classification problems and the corresponding discussion in balance ratio are presented. Our approach performs well with smaller network structure compared with other deep models. In addition, the performance is improved over 40% in defective accuracy by adding noise approach. Finally, the accuracy is higher than 96%; even the imbalanced ratio (IR) is one hundred.

1. Introduction

In industrial applications, the defect detection is very important since defects have an adverse effect on the quality and performance of products [1]. In this area, surface defect detection is one of the most applications and it is necessary for steel, wood, or solar wafers [2–4]. The more commonly used methods in recent years are machine vision due to high speed, cost savings, and high accuracy [5–10]. Traditional machine vision approach can be divided into four categories, namely, statistical approaches, structural approaches, filter-based methods, and model-based approaches [7]. However, the corresponding performance depends on application fields and the data set distribution affects the results [8–10]. On other applications such as cancer detection or environmental disaster prediction, a disproportionate amount of data available comes from negative cases [11, 12].

Recently, convolutional neural networks (CNNs) have been used for many fields [11–14]. It automatically learns to extract features without professional knowledge for feature extraction. In manufacturing, the defect detection is a typical imbalanced data problem and defect samples are usually less than nondefective ones [15]. The imbalance ratio (IR) is usually used to describe the ratio of minority to majority samples. The general used value of IR is greater than 1.5 that causes the learning result bias towards the majority class [16, 17]. To treat the imbalanced data problems, lots of researches have been presented, e.g., sampling methods, cost-sensitive methods, and kernel-based method [17–26]. The most used sampling method is random oversampling and undersampling [26]. Another variation of the oversampling method is synthetic minority oversampling technique (SMOTE), which generates new samples by synthesizing minority samples [19, 20]. SMOTE uses the modification of the data in the feature plane to solve the problem. Therefore, we have to use feature extraction to get the features first. On the other hand, data augmentation method modifies image data to add samples [17, 21–25]. Generative Adversarial Network (GAN) is used to generate realistic data from minority sets [17, 22, 23]; however, the corresponding results are not good enough on diverse data sets. Cost-sensitive method is changing the threshold or weight of the

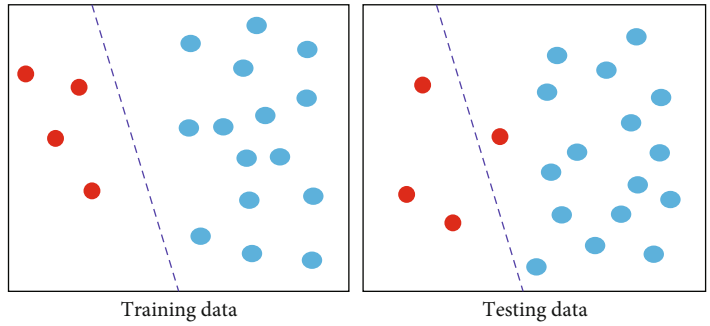


Figure 1: Problem of classification caused by imbalanced data in a learning model.

network to bias the network to minority classes [22, 23]. The kernel-based method works on the classification boundary of the feature space [24, 25]. Literature [27] presents the MetaBalance, an algorithm that uses meta-earning for deep neural network on class-imbalanced data.

Since the performance of CNN is closely related to the data, the number or quality of data affects the classification results. Therefore, some methods have been introduced to deal with this problem [28–30]. However, CNN misclassifies defective samples. Therefore, a CNN with adding feature space noise method is proposed to improve the accuracy.

In this paper, we propose a classification method using CNN with adding noise in feature space for imbalanced data classification. Our target is to improve the accuracy of minority samples in the classification that also preserves the total accuracy. Herein, a hybrid loss function of crossentropy and KL divergence is adopted for training. In addition, a simple design method for selecting structure of CNN is introduced and then, we add noise in feature space of CNN to obtain proper features by a training process and to improve the classification results. As our experience, the proposed approach has the ability to find the information that does not exist in the original minority samples, thereby improving the accuracy of minority class in the testing data. To demonstrate the generalization, this method is applied in three imbalanced data sets with different sizes. Experiments are introduced to show that the method effectively improves the accuracy of a minority samples. In addition, we also apply this method to multiclass classification or different imbalance ratios.

The rest of this paper is as follows. Section 2 introduces the imbalanced data problem and data sets. The major contributions are introduced in Section 3, which include CNN with adding feature space noise and network structure selection. Section 4 presents the experimental and validation results. Finally, conclusion is given.

2. Problem Formulation and Data Sets

This section introduces the imbalanced data classification problem and the experimental data sets. Three open data sets (DAGM 2007 [31], NEU surface defect [32], and MNIST [33]) are utilized to demonstrate the performance and effectiveness of our method.

Table 1: Sample number of DAGM 2007 in each subdata set.

Subdata set	Training data		Testing data	
	Defective	Nondefective	Defective	Nondefective
1	79	496	71	504
2	66	509	84	491
3	66	509	84	491
4	82	493	68	507
5	70	505	80	495
6	83	492	67	508
7	150	1000	150	1000
8	150	1000	150	1000
9	150	1000	150	1000
10	150	1000	150	1000

2.1. Imbalanced Data Classification. Any data set with an unequal data distribution is an imbalanced data, and the minority samples usually present significant concepts for classification [18, 34–36]. Figure 1 shows the problem of classification caused by imbalanced data in the training process; the red and blue points are the minority and majority samples, respectively; and the dashed line denotes the decision boundary. The model is trained and judges by the decision boundary. Since the minority samples in the training data do not have enough concepts to present the minority class, the model may classify some minority samples as wrong class in testing data. If these minority samples are defective ones, it will have a great impact on the quality of products. Therefore, our goal is to solve the misclassification of minority samples in imbalanced data.

For the binary classification problems, the corresponding confusion matrix is used to show the classification results. If p and n denote the true positive and negative, Y and N mean the predicted positive and negative and TP, FP, FN, and TN represent true positives, false positives (or type 2 errors), false negatives (or type 1 errors), and true negatives. In general, the positive samples are defective or minority samples, and the negative samples are nondefective or majority samples. Thus, *accuracy* is defined as

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}. \quad (1)$$

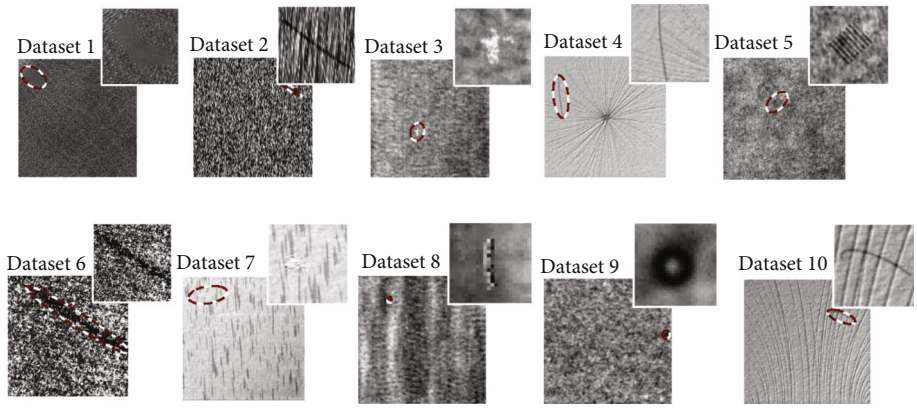


Figure 2: Samples of the DAGM 2007 data set.

Table 2: Number of the NEU surface data set.

Class	1	2	3	4	5	6	7	8	9
Training data	605	148	388	574	399	100	795	387	219
Testing data	605	148	387	574	398	100	794	386	219

We use *recall* to evaluate our model as

$$\mathrm{REC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{2}$$

Herein, we mainly compare the problems of binary classification with *recall*. To make the experimental results clearer, we present *recall* by accuracy of defective samples or minority samples instead. In addition, we also use *precision* to evaluate

$$\mathrm{PRE} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{3}$$

2.2. Data Sets

2.2.1. DAGM 2007 Data Set. The DAGM 2007 data set is a synthetic data set for defect detection on textured surfaces. It was originally created for a competition at the 2007 symposium of the DAGM (Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., the German chapter of the International Association for Pattern Recognition) [31]. Table 1 presents the number of samples for each subdata set with imbalance ratio -20/3. The sizes are unified to 512×512. Each subdata set has different kinds of textures and defects; samples are shown in Figure 2.

2.2.2. NEU Surface Defect Data Set. The data set is provided by Northeastern University, and the dimensions of the version are 64×64 pixels [33]. Nine classes of typical surface defects of the hot-rolled steel strip are collected. The NEU surface defect data set includes two difficult challenges, which are large differences in the same class and similarities between different defects. Table 2 shows the number of images for each class, and Figure 3 shows the samples of the data set.

2.2.3. MNIST Data Set. MNIST is an abbreviation of Modified National Institute of Standards and Technology from American Census Bureau employees [33]. Each picture is normalized to 28×28 pixels, shown in Figure 4, and Table 3 shows the number of images for each digit. MNIST is not an imbalanced data set; therefore, the sample number of some classes is modified in later experiments.

3. Convolutional Neural Network with Adding Noise in Feature Space

CNN is one of the representatives of deep learning and artificial intelligence [14]. Figure 5 shows the basic architecture of CNN with input image size 6×6 [37], in which the convolution computing with eight 3×3 filters results in eight feature maps with 4×4 resolution and processed by 2×2 maximum pooling to reduce dimensions. After passing the flatten layer, the feature maps are rearranged in one dimension; subsequently, the full connection with four hidden neurons and six outputs is connected. In general, the corresponding structure affects the performance of CNN; thus, we here introduce an architecture design method in this section.

3.1. CNN with Adding Noise in Feature Space. In order to solve the imbalanced data classification using CNN, the CNN is modified by adding noise to features extracted. The purpose of adding noise in feature space is to change the distribution of features by the training process [38–40]. When the noise is added to the feature space, we have the chance to extract suitable features for classification, especially for minority samples. Therefore, to get better results on the training data, it is necessary to get better features that can identify the random noise samples. Thus, it will be able to identify testing samples. An illustration of adding noise to the feature space is introduced in Figure 6; the red and blue points denote the majority and minority samples,

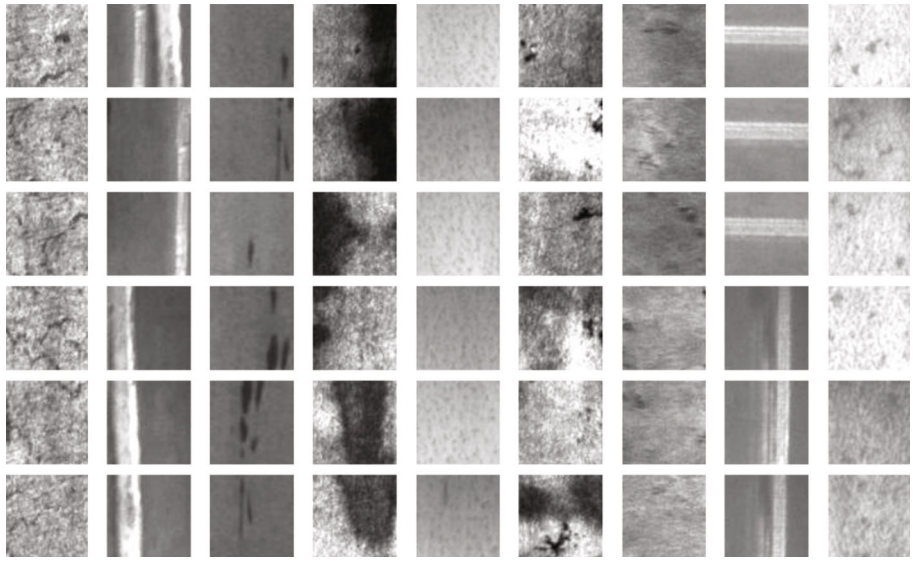


Figure 3: Samples of the NEU surface data set.

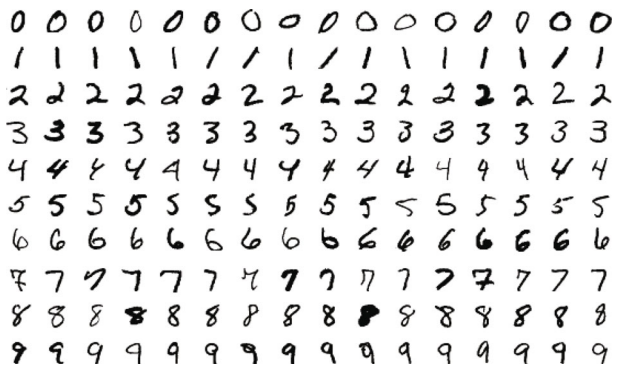


Figure 4: Samples of the MNIST data set.

Table 3: Number of the MNIST data set.

Digit	Training data	Testing data
0	5932	980
1	6742	1135
2	5958	1032
3	6131	1010
4	5842	982
5	5421	892
6	5918	958
7	6265	1028
8	5851	974
9	5949	1009

respectively. The dashed line denotes the decision boundary by training. Note that the trained neural network cannot distinguish the minority samples due to the fact that they are closer to the majority samples in the testing process. When we disturb the point, illustrated as the dashed circle in Figure 6, it has the chance to be similar to the point in the testing data that is misjudged. For being able to classify this point in the training data correctly, the network will be able to classify these points by the extracted feature and classification fully connected layer. Therefore, the minority samples that are close to the majority samples in the testing data may also be correctly classified.

Herein, we implement the concept to propose a CNN with adding noise in the feature space to obtain proper features by the training process and to improve the classification results. Figure 7 shows the proposed CNN with adding noise architecture. The noise is added in the last extracted feature layer, called $\mathrm{CNN}_{\mathrm{noise}}$. In addition, the structure selection will be introduced in the next subsection. The adding noise with standard normal distribution is multiplied with e^{σ} to ensure the value is positive. Note that nodes m are features extracted from CNN. For the testing process, we only take m and remove the noise. After the adding noise part, we get the new feature c as

$$c = \mathrm{ND} \times e^{\sigma} + m, \tag{4}$$

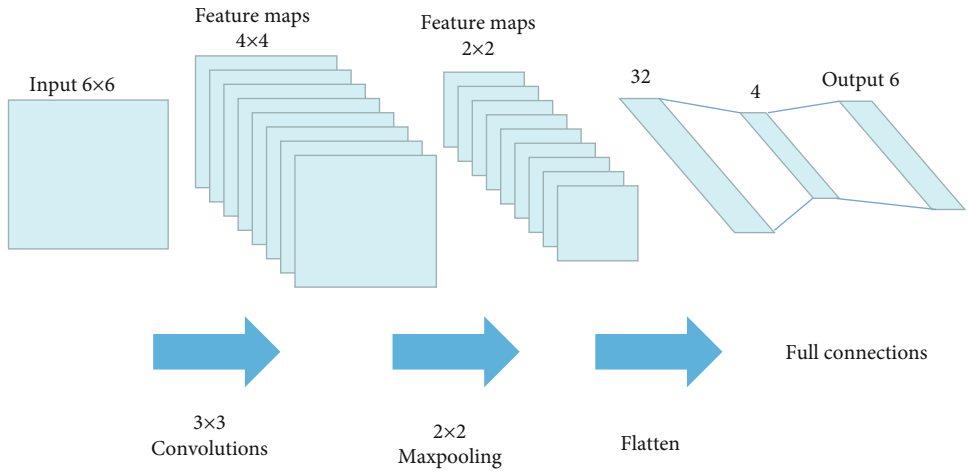


Figure 5: CNN architecture with convolutions, max pooling, flatten, and full connection.

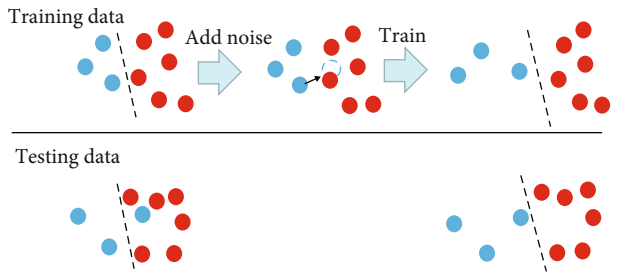


Figure 6: Illustration of effect of adding noise to features.

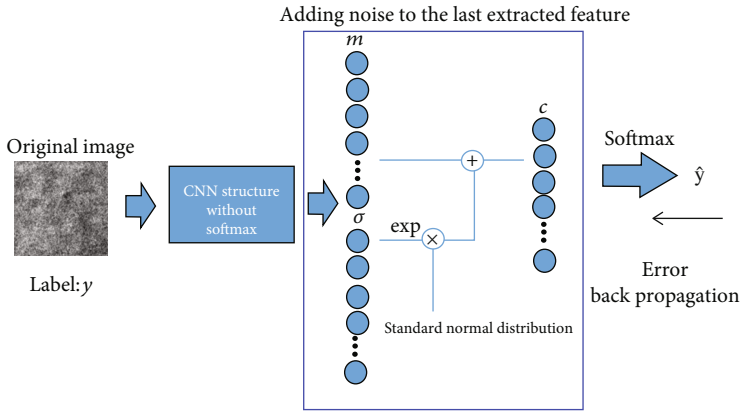


Figure 7: Architecture of the proposed method.

where ND means the standard normal distribution. Finally, c passes softmax and outputs $\hat{y}$ to make predictions on y and the network is adjusted by error back propagation. However, the noise added after training will approach zero without any constraints in loss function. Since a small value of noise results in a small loss function value, therefore, we adopt the KL divergence to constrain the CNN to ensure the existence of noise. KL divergence is used to calculate the difference in probability distribution. By using the KL divergence as one of our loss functions, m and σ will be close to the mean and standard deviation of standard normal distribution. Since the standard deviation of the standard normal distribution is one, we can ensure that noise is not zero. Therefore, our loss function is

$$L = \alpha L_{\mathrm{KL}} + L_{\mathrm{ce}}, \tag{5}$$

where

$$L_{\mathrm{KL}} = -\frac{1}{N}\sum_{n=1}^{N}\frac{1}{2}\sum_{i=1}^{k}\left(\sigma_{ni} + m_{ni}^{2} - \ln\left(\sigma_{ni}\right) - 1\right), \tag{6}$$

Table 4: Confusion matrix of transfer learning on the DAGM 2007 data set.

	Labels/predictions	Nondefective	Defective
VGG16	Nondefective	86.70%	13.29%
	Defective	71.83%	28.17%
InceptionV3	Nondefective	99.40%	0.60%
	Defective	100%	0%
ResNet50	Nondefective	84.92%	15.08%
	Defective	77.96%	22.54%

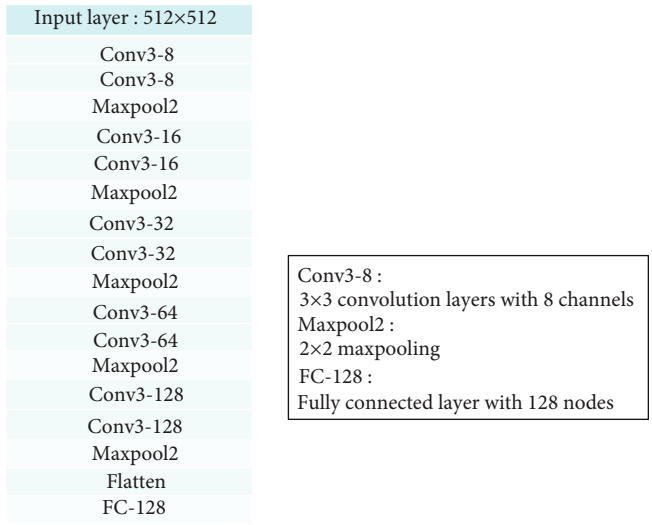


Figure 8: Architecture of $\mathrm{CNN}_{\mathrm{designed}}$.

$$L_{\mathrm{ce}} = -\frac{1}{N}\sum_{n=1}^{N}\left(y_n \log \hat{y}_n + (1-y_n)\log\left(1-\hat{y}_n\right)\right). \tag{7}$$

L_{KL} and L_{ce} denote the loss functions of KL divergence and crossentropy; parameter α is used to suppress the KL divergence. In addition, N denotes number of samples, k denotes dimension of feature, and y and $\hat{y}$ denote the label of data and prediction, respectively. An illustrated result of DAGM 2007 for selecting value of α, in equation (5), will be introduced in Section 4. By our experience, α is set to be 0.00025 and it can be fine-tuned by experimental results.

3.2. Architecture Design of CNN. The structure of CNN affects the classification results and computation complexity. Herein, a simple method to select the CNN structure is introduced. Recently, some literature uses transfer learning method for defect detection [41–44]. At first, we adopt the transfer learning for the DAGM 2007 subdata set 1 as test by using VGG16, InceptionV3, and ResNet50 [45–47]. The classic models are pretrained by ImageNet data set, and results of transfer learning are presented in Table 4. The algorithm used in training is Adam. The initial learning rates used during training are all 0.0001, and decay is 0.00001. The number of iterations used for learning is 100 epochs. The average results here are obtained by 10 independent runs. From Table 4, the accuracy of defective is not good enough although the accuracy of nondefective is acceptable. In addition, the total adjustable parameters of them are very huge, more than 18 million; the corresponding computation effort for training and defect is large. More details and comparisons will be introduced in Section 4. Thus, we introduce a designed method of CNN architecture to improve the classification results. The method of designing for specific data set was also used in other articles [48, 49]. The advantage of this method compared to transfer learning is that the calculation time is short.

Table 5: Defect detection results of $\mathrm{CNN}_{\mathrm{designed}}$ by our method on the DAGM 2007 data set.

Labels/predictions	Nondefective	Defective
Nondefective	100%	0%
Defective	5.63%	94.37%

Here are the steps to design the CNN architecture.

Step 1. We first decide the number of pooling layers.

Step 2. Use a smaller amount of convolution layers, and gradually increase it to increase the correct rate.

Step 3. Stop when the correct rate no longer increases.

Step 4. The number of hidden layers of the fully connected layer is selected to be one.

Table 6: Comparison results of previous models with $CNN_{designed}$.

	VGG16	InceptionV3	ResNet50	$CNN_{designed}$
Trainable parameters	4,194,690	12,845,442	16,777,602	2,654,586
Fixed parameters	14,714,688	21,802,784	23,587,712	0
Total parameters	18,909,378	34,648,226	40,365,314	2,654,586
Training time	20 seconds/epoch	22 seconds/epoch	22 seconds/epoch	5 seconds/epoch
Testing time	3.48 seconds/100 pictures	11.25 seconds/100 pictures	15.11 seconds/100 pictures	0.87 seconds/100 pictures

4. Experimental Results

The following experiments are run on the platform of Taiwan Computer Cloud (TWCC), a cloud computing platform provided by the National Center for High-performing Computing (NCHC) of the National Applied Research Labs (NARLabs). The computing speed of Taiwania 2 and Taiwania is ranked 23rd and 314th in the TOP500 international rankings in 2019, respectively. Its GPU hardware specifications are NVIDIA Tesla V100 32 GB, and we use eight of them at the same time.

To demonstrate the proposed approach, several comparison and illustration experiments are introduced. To illustrate the versatility, we also present the results of the three data sets and the multiclassification problem to show the performance of higher imbalance ratio case. In the following experiments, the data are randomly selected as 80% training, 10% testing, and 10% validation (each experiment is fine-tuned). In addition, the random selection for each epoch is done by "shuffle."

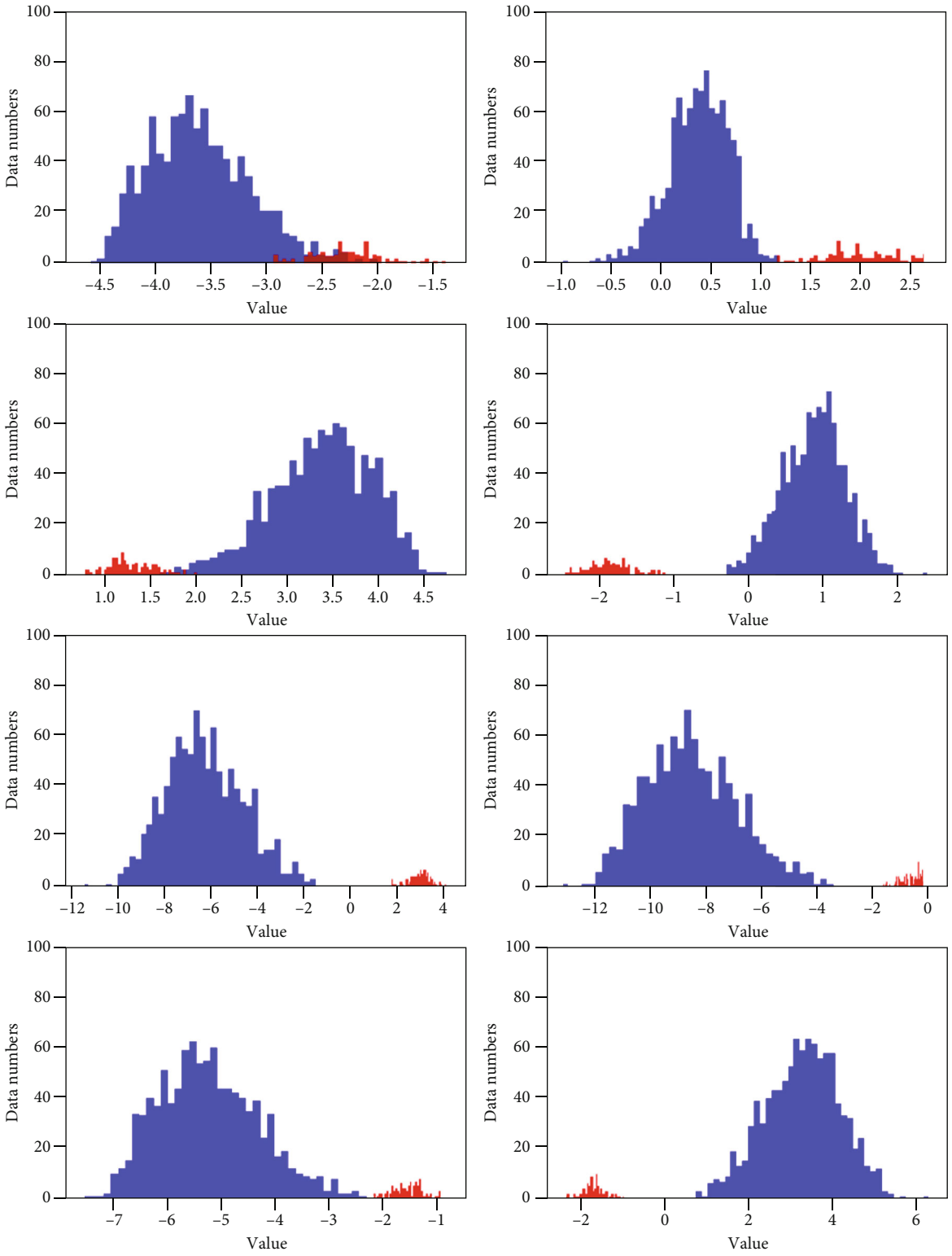
4.1. Architecture Design of CNN. Herein, an illustrated example for CNN architecture design by the DAGM 2007 data set for performance evaluation is introduced. At first, the number of pooling layers is determined by the range whose features are extracted according to the results of [50]. The number of convolutional layers determines the features; i.e., more complex features require more convolutional layers. We here gradually increase it and stop when the accuracy no longer increases. Since the DAGM 2007 data set is a binary classification problem, the number of hidden layers of the fully connected layer is selected to be one. In addition, the numbers of kernels of the convolutional layers and the fully connected layer are just set to appropriate values. Herein, our objective is to introduce usable results with fewer parameters and shorter computation time to obtain well accuracy. If we need to optimize the accuracy by architecture, uniform design method and optimization can be adopted [51, 52]. Figure 8 shows the CNN architecture, and Table 5 shows the corresponding results on the DAGM 2007 data set. Obviously, the accuracy of minority data is 94.37%; it is better than the results of VGG16, InceptionV3, and ResNet50 by transfer learning, shown in Figure 4. We use it as our CNN architecture (called $CNN_{designed}$) in the following discussions. Besides, Table 6 shows the comparison results of previous models with our method in parameter number and computation efforts. We can see that the parameters of the previous models are much more than our method. Note that the trainable parameters affect training time and the fixed parameters affect training and testing time; the total parameters affect the computing requirement. From Table 6, the parameter number of our method is smaller than that of others and the corresponding training and testing time of our method is relatively small. These may cause difficulties in the implementation of transfer learning approach. Considering the low accuracy of transfer learning in defective samples and implementation difficulties, we adopt the $CNN_{designed}$ in the following experiments.

4.2. Performance of Adding Noise in Feature Space. As shown in Figure 1, decision boundary in feature space is easily obtained when the distribution of features is separable. Thus, the purpose of adding noise is to get more separable features. Herein, we present the distribution of the trained features to demonstrate the proposed method. Figure 9 shows the distribution of eight features for training and testing data, respectively; blue and red are majority and minority samples. Figures 9(a) and 9(b) show the distribution of feature extracted by training samples and Figures 9(c) and 9(d) are the results by testing samples. From Figure 9(a), we can find that the extracted features by CNN without adding noise are very close and some samples are overlapping. Figure 9(b) shows the feature distribution extracted by the CNN_{noise}. Obviously, the features extracted by CNN_{noise} can increase the distance between majority samples and minority samples. In order to be able to adapt to these noise-added features, the network must be able to extract more separated features between classes. Figures 9(c) and 9(d) show the comparison results in testing data. We can see that there are many indistinguishable samples in the extracted feature by CNN without noise. In contrast, in Figure 9(d), we find that CNN_{noise} can also effectively separate features in the testing data. It can be found that the features confused in Figure 9(d) are very few, which proves that the features obtained by CNN_{noise} can effectively classify the samples. In addition, an evaluation of confusion feature is introduced as follows.

To evaluate the confusion of extracted features, we here define confusion number as

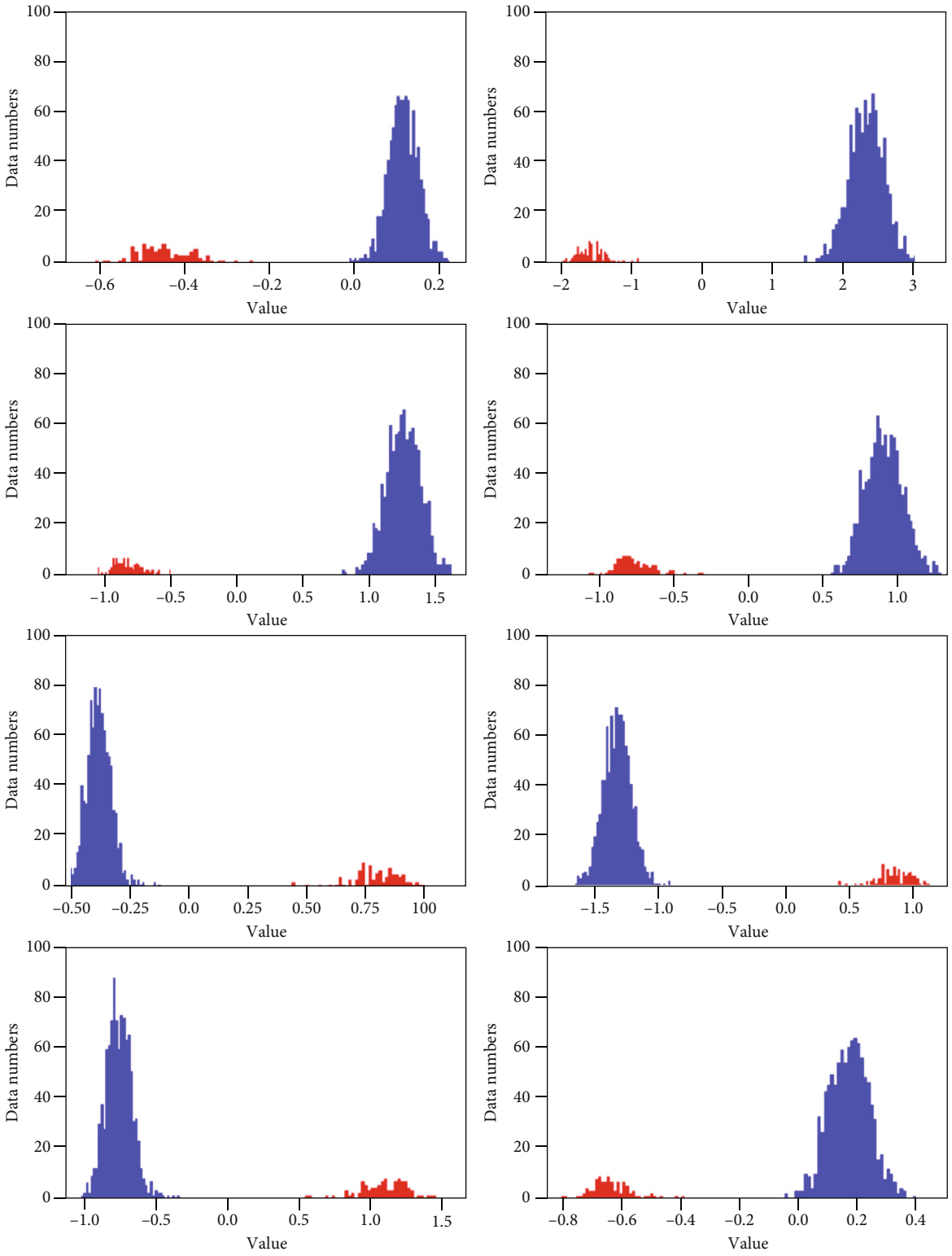
$$\text{Confusion number} = \min\left(\text{overlap } A, \text{overlap } B\right), \tag{8}$$

where overlap A denote the sample number of class A overlap class B, and overlap B means the sample number of class B overlap class A. Since we usually consider the fewer overlapping samples as being mixed into another class, we choose the small one to be the confusion number. For example, if $A = \{0, 1, 2, 5\}$ and $B = \{3, 4, 7, 8\}$, the sample of class



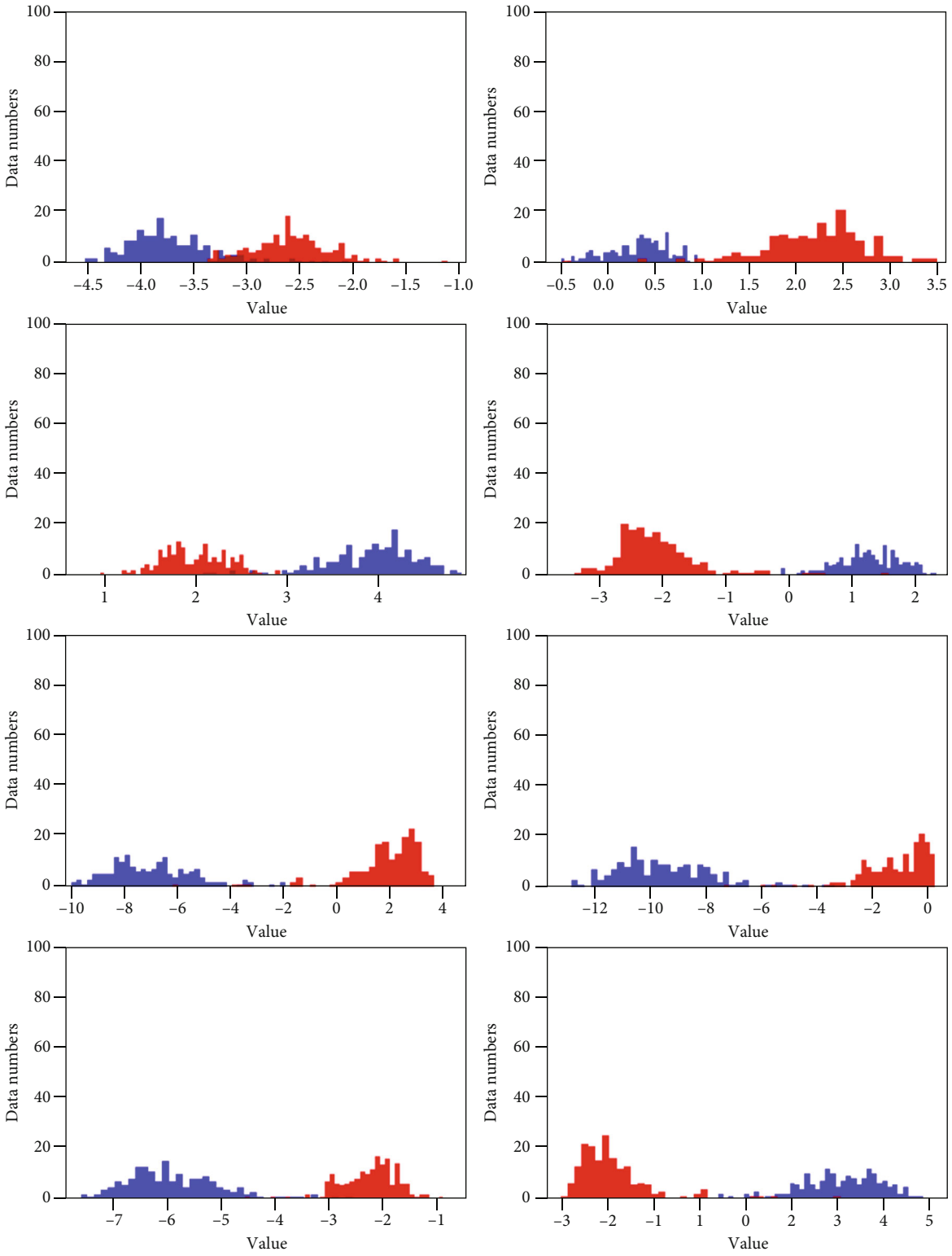
(a)

Figure 9: Continued.



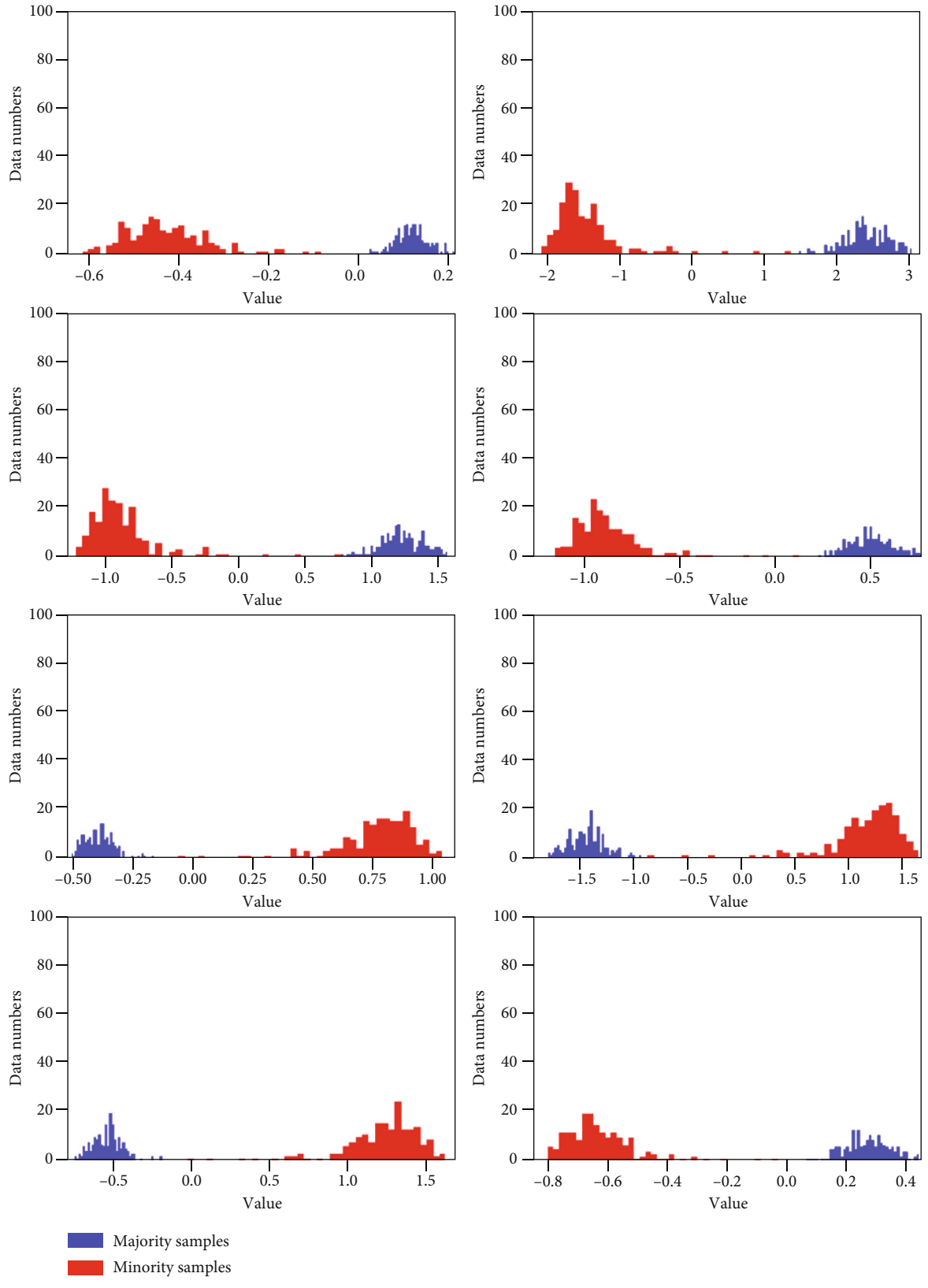
(b)

Figure 9: Continued.



(c)

Figure 9: Continued.



(d)

Figure 9: Distribution of the feature value: (a) extracted in training data by CNN without adding noise; (b) extracted in training data by CNN_{noise}; (c) extracted in testing data by CNN without adding noise; (d) extracted in testing data by CNN_{noise}.

A overlap class B is {5} and class B overlap class A is {3, 4}. Therefore, we have confusion number = min $(1, 2) = 1$. Table 7 presents the confusion number in different features of CNN without ($CNN_{designed}$) and with noise (CNN_{noise}). We can see that CNN_{noise} results in no confusing samples whether in training or testing data; this means that the features are separable. However, the CNN without adding noise results a number of minority samples in the testing data, even though feature extracted by CNN can be classified in the training data. Our method can obtain new features during training by adding noise to features of minority samples. These new features have the opportunity to approach those features that are not included in minority samples. In this way, our network also performs well in testing data. In training data, although $CNN_{designed}$ has several features that are not perfect, other features can still make the network get a high accuracy in training data. This also causes the network to no longer continue to converge, and the significant drop in the quality of the features in testing data highlights the lack of the imbalanced sample.

4.3. Comparison Results in Accuracy. This section presents the comparison results of classification accuracy between $CNN_{designed}$ and CNN_{noise} for the DAGM 2007 data set. Herein, the same learning parameters are utilized; the initial learning rate is 0.0001, and decay is 0.00001; the learning algorithm is Adam, and epoch is 200. Table 8 shows the comparison results by the accuracy of all test samples and defective samples in each subdata set. From Table 8, the accuracy rate can be improved by CNN_{noise}; it obtains much better results in these subdata sets, especially the accuracy of minority samples (accuracy improvement to 90.33%). This verifies the effect of our proposed method to detect imbalanced defect data; we can greatly improve the accuracy on minority samples. We could also conclude that the accuracy can be increased both in the high-accuracy subdata set and the low-accuracy subdata set. Especially, the subdata sets with lower accuracy of the defect samples, such as subdata set 4, subdata set 5, subdata set 7, and subdata set 9, can also be improved well. These results verify that if we use this method to detect imbalanced defect data, we can greatly improve the accuracy of $CNN_{designed}$ on minority samples.

4.4. Comparison Results of Other Data Sets. In order to verify the proposed method, other two commonly used data sets (NEU and MNIST) are adopted for demonstration. The NEU surface data set is also a problem with the defective data, and it also has different numbers of samples in different classes, and the MNIST is a commonly used data set for image recognition. The corresponding CNN structure is introduced in Table 9. Table 10 introduces the sample number of selected classes. They are divided into majority and minority classes for binary classification. Table 11 shows the accuracy results of NEU surface data set by $CNN_{designed}$ and CNN_{noise}. We can find that CNN_{noise} still gets better results for the NEU surface data set. Herein, we use an initial learning rate of 0.0005 and a decay of 0.00001, and Adam is also used as the training algorithm.

Table 7: Confusion number in different features of CNN-designed and CNN_{noise}.

	Training data		Testing data	
Feature	$CNN_{designed}$	CNN_{noise}	$CNN_{designed}$	CNN_{noise}
Feature 1	42	0	31	0
Feature 2	0	0	13	0
Feature 3	10	0	11	0
Feature 4	0	0	9	0
Feature 5	0	0	7	0
Feature 6	0	0	8	0
Feature 7	0	0	18	0
Feature 8	0	0	7	0

Table 8: Comparison in accuracy between CNN and CNN_{noise} (DAGM 2007).

	Accuracy			
Subdata set	$CNN_{designed}$		CNN_{noise}	
	All test samples	Defective (recall)	All test samples	Defective (recall)
1	99.30%	94.37%	100.00%	100.00%
2	99.48%	96.43%	100.00%	100.00%
3	95.30%	67.86%	98.09%	86.90%
4	88.70%	35.29%	97.39%	79.41%
5	82.09%	7.50%	94.26%	58.75%
6	96.87%	76.12%	98.78%	89.55%
7	80.70%	7.33%	92.52%	98.00%
8	94.09%	70.67%	97.65%	98.00%
9	82.61%	8.00%	99.13%	92.67%
10	99.65%	97.33%	100.00%	100.00%
Average	91.88%	50.09%	97.78%	90.33%

Table 9: CNN architecture for the MNIST and NEU surface data set.

Input layer: 64×64 or 28×28
conv3-8
conv3-8
maxpool2
Flatten
FC-128

Table 10: Sample number of selected classes.

	Majority classes		Minority classes	
Class	1	7	2	6
Training data	605	795	148	100
Testing data	605	794	148	100

Herein, we also apply CNN_{noise} on multiclass due to the fact that multiclass classification is common in real applications. Note that the NEU data set categorizes to multiple-class data; therefore, it is adopted for demonstrating

Table 11: Accuracy results of the NEU surface data set.

Group	Accuracy			
	$CNN_{designed}$		CNN_{noise}	
	All test samples	Minority (recall)	All test samples	Minority (recall)
Classes 1, 2	99.40%	96.96%	99.68%	98.38%
Classes 1, 6	91.35%	45.04%	94.61%	70.50%
Classes 2, 7	96.87%	80.07%	98.66%	91.44%
Classes 6, 7	99.13%	96.75%	99.31%	98.80%

Table 12: Confusion matrix results in multiclass classification.

(a) $CNN_{designed}$ (without noise)

Labels	Predictions			
	Class 1	Class 2	Class 6	Class 7
Class 1	98.96%	0.00%	0.87%	0.16%
Class 2	0.23%	79.05%	0.00%	20.72%
Class 6	63.31%	1.67%	34.35%	0.67%
Class 7	1.81%	0.00%	0.00%	98.19%

(b) CNN_{noise}

Labels	Predictions			
	Class 1	Class 2	Class 6	Class 7
Class 1	98.91%	0.00%	1.09%	0%
Class 2	0.68%	87.16%	0.45%	11.71%
Class 6	40.00%	3.32%	56.68%	0.00%
Class 7	0.04%	0.00%	0.00%	99.96%

Table 13: Number of samples in the modified MNIST.

Digit	Training data	Testing data
0	5932	980
1	6742	1135
2	5958	1032
3	6131	1010
4	5842	982
5	5421	892
6	5918	958
7	60	1028
8	60	974
9	60	1009

CNN_{noise}. Table 12 presents the confusion matrix results of $CNN_{designed}$ and CNN_{noise}. We can observe that class 2 and class 6 are minority classes here and they also get improvement by CNN_{noise}. It can be observed that the classification effect in multiple classes decreases since the addition of other classes, but CNN_{noise} can still obtain a higher accuracy rate than CNN without adding noise.

Table 14: Confusion matrix results on the modified imbalanced MNIST.

(a) $CNN_{designed}$ (without noise)

Labels	Predictions									
	0	1	2	3	4	5	6	7	8	9
0	1.00									
1		1.00								
2			1.00							
3				1.00						
4					1.00					
5				0.10		0.99				
6	0.01						0.99			
7	0.02	0.03	0.08	0.05	0.03			0.77		0.02
8	0.05	0.01	0.08	0.10	0.03	0.06	0.02		0.64	0.01
9	0.03	0.01	0.01	0.03	0.15	0.08		0.02		0.67

(a) CNN_{noise}

Labels	Predictions									
	0	1	2	3	4	5	6	7	8	9
0	1.00									
1		1.00								
2			1.00							
3				1.00						
4					1.00					
5				0.10		0.99				
6	0.01						0.99			
7	0.01	0.01	0.05	0.01	0.02			0.90		
8	0.03		0.04	0.01	0.02		0.02		0.84	0.04
9	0.02			0.01	0.10	0.04		0.03		0.80

Table 15: Comparison results in different imbalance ratios.

Imbalance ratio	Accuracy			
	$CNN_{designed}$		CNN_{noise}	
	Class 1 (recall)	Class 7	Class 1 (recall)	Class 7
2	100.00%	100.00%	100.00%	100.00%
4	99.52%	100.00%	100.00%	100.00%
10	97.14%	100.00%	100.00%	100.00%
20	92.38%	100.00%	99.52%	100.00%
50	70.48%	100.00%	97.62%	100.00%
100	11.90%	100.00%	96.43%	100.00%

The second validation is the MNIST data set; note that MNIST is first manually modified to make it an imbalanced data set. MNIST is a data set with ten classes of handwritten digits from 0 to 9; we here choose the digits 7, 8, and 9 as minority classes. There are 6000 samples per class in the original training data. The imbalance ratio 100 by randomly selecting the minority classes is created; the number of

Table 16: Comparison results with other methods in function analysis.

	For image	Simplicity	Versatility	Possibility
Random oversampling [54]	○	Δ	○	×
SMOTE [19–21]	Δ	Δ	Δ	Δ
Data augmentation [17]	○	Δ	Δ	○
Cost-sensitive [22, 23]	○	○	Δ	×
CNN_{noise}	○	○	○	○

samples in modified MNIST is introduced in Table 13. Table 14 shows the confusion matrix results on the modified imbalanced MNIST. Herein, we use an initial learning rate of 0.0001 and a decay of 0.00001, and the training algorithm is also Adam. From Table 14, it can be found that the minority classes of 7, 8, and 9 can only get about 77%, 64%, and 67% of the accuracy through CNN without adding noise. In contrast, CNN_{noise} can improve the accuracy of the minority classes 7, 8, and 9 with about 13%, 20%, and 13% without reducing the accuracy of the majority classes.

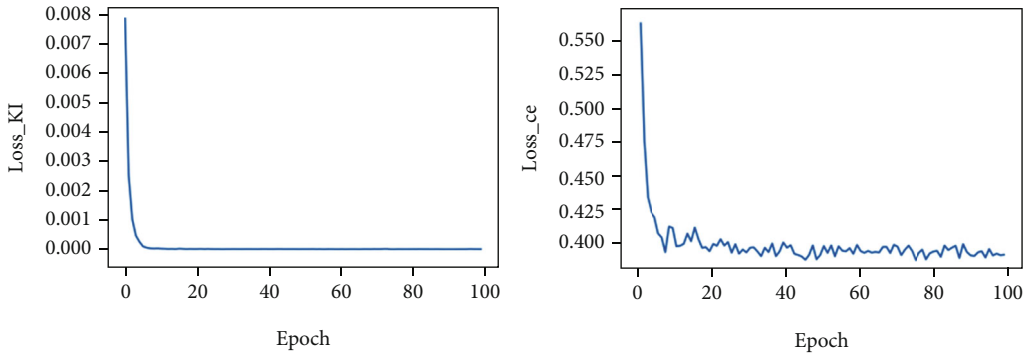
4.5. Discussion of Imbalanced Ratio. In practical problems, we may need to deal with a higher imbalanced ratio [53]. Therefore, we increase the imbalanced ratio artificially to verify CNN_{noise} by binary classification. It is also confirmed whether the results are affected by the imbalanced ratio. Herein, we use the class 1 (majority) and class 7 (minority) of the NEU surface defect for verification. Different imbalance ratios 2, 4, 10, 20, 50, and 100 are chosen, and random selection is utilized to create the data set. Table 15 shows the comparison results between CNN_{noise} and CNN in different imbalance ratios. We can see that when the imbalance ratio increases, the minority sample accuracy of CNN decreases rapidly. In contrast, CNN_{noise} decreases slowly and can maintain the accuracy of minority samples above 96%. This shows that CNN_{noise} has a higher tolerance for higher imbalance ratios.

4.6. Comparison Results with Others. Herein, comparison results with CNN_{noise} are introduced, shown in Tables 16 and 17. The corresponding function analysis between CNN_{noise} and other methods is introduced in Table 16, where ○ is able; Δ is partial able; and × is unable. The second column means whether the method can apply to image data directly. SMOTE must extract the features of the picture before we use it. The third column means whether the method requires complicated processing. Random oversampling needs to select the sampled images before classifying them. SMOTE and cost-sensitive method also require to generate samples first. The fourth column means whether the method is general for different data. SMOTE and data augmentation need different generation methods to deal with different nature of data. The cost-sensitive method may encounter the problem that the accuracy of minority class must be sacrificed. The last column means whether the method has the possibility of obtaining information beyond the existing information. Both SMOTE and data augmentation are generation methods. However, since SMOTE synthesizes in the original data, it is more limited than other methods. CNN_{noise} is able to add noise to the feature space and obtain information that does not exist in data. Since CNN_{noise} only needs to train a classification model and can be applied to different data sets directly, the versatility and simplicity of CNN_{noise} are also well. In these methods, random oversampling is mainly used to solve the problem of imbalanced sample distribution in the learning algorithm. On the other hand, cost-sensitive methods can work well if we do not care about the accuracy of the majority class. In addition, SMOTE can be used to find and synthesize information in data. CNN_{noise} and data augmentation can find information outside the data. Table 17 shows the comparison results of performance for the NEU surface data set with an imbalance ratio of 100. These comparison results are obtained by averaging the same ten experiments. We can see that all these methods can improve the accuracy of the minority class using only CNN. Although random oversampling can improve the accuracy, the improvement is lower than other methods. Cost-sensitive methods can increase the accuracy of minority class to a high level, but it sacrifices the accuracy of the majority. SMOTE and data augmentation require some preprocessing after using them. In contrast, CNN_{noise} can get well accuracy without complicated processing.

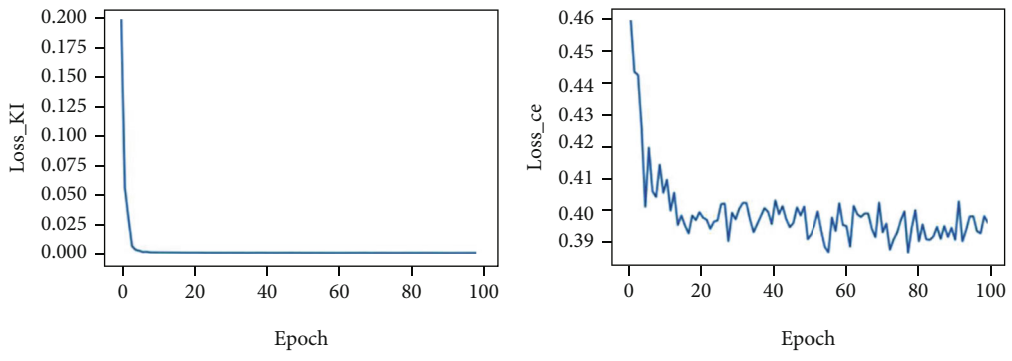
Table 17: Comparison results in the NEU surface data set (IR = 100).

	Class 1 samples (majority)	Class 7 samples (minority)
Only CNN	11.90%	100.00%
Random oversampling [54]	58.76%	100.00%
SMOTE [19–21]	95.19%	100.00%
Data augmentation [17]	87.81%	100.00%
Cost-sensitive [22, 23]	94.95%	74.70%
CNN_{noise}	96.43%	100.00%

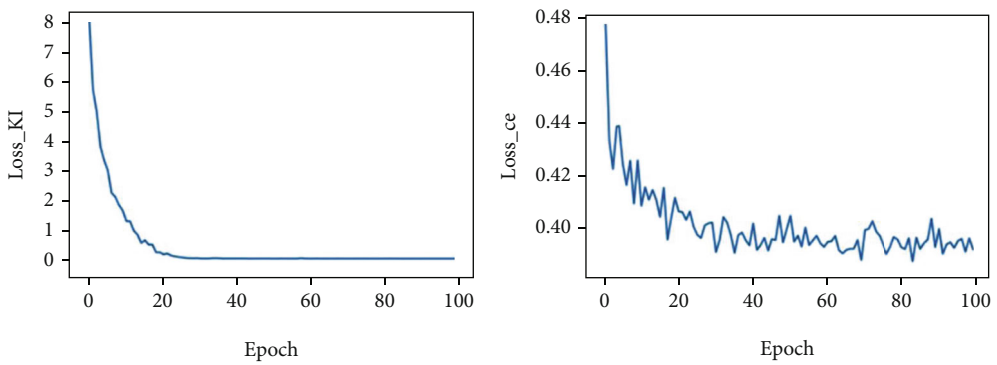
4.7. Discussion for Selection Parameter α of Hybrid Loss Function. Herein, a comparison result for selecting parameter α of hybrid loss function (5) is introduced by experimental testing. Figure 10 shows the training history with different α in the DAGM 2007 subdata set 1. Left column figures are L_{KL} and the right ones are L_{ce}. Herein, we attempt to balance L_{KL} and L_{ce} by gradually reducing α. From



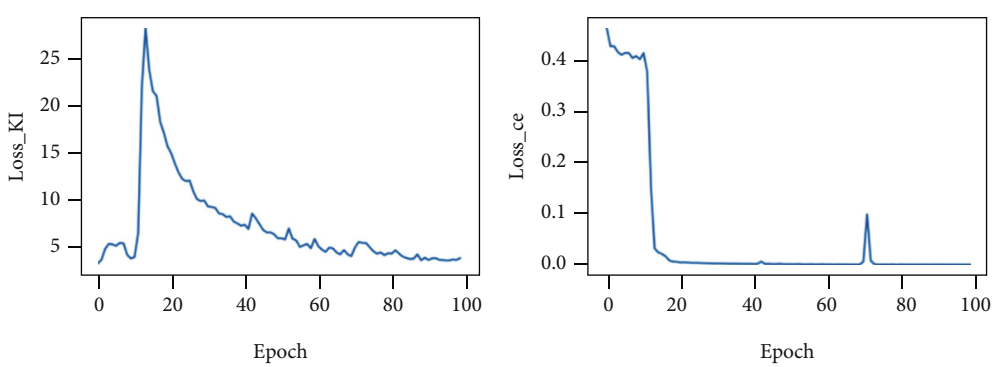
(a)



(b)

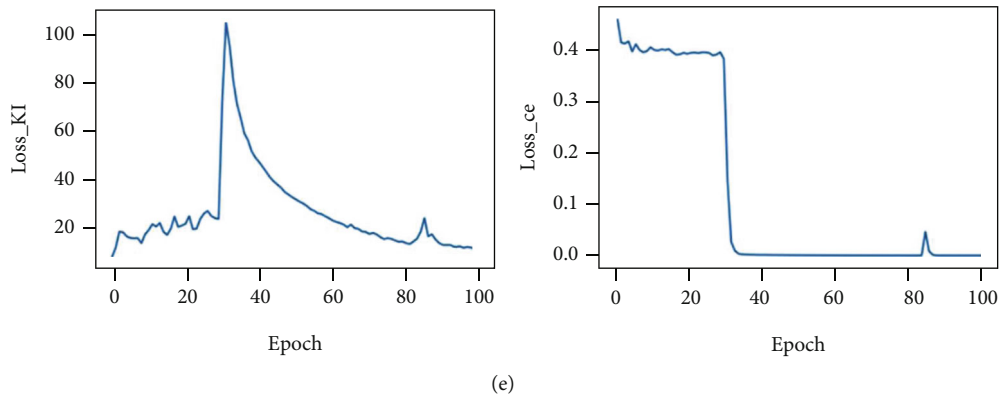


(c)



(d)

Figure 10: Continued.



(e)

Figure 10: Training history with different α in the DAGM 2007 subdata set 1: (a) $\alpha = 1$, (b) $\alpha = 0.01$, (c) $\alpha = 0.01$, (d) $\alpha = 0.00025$, and (e) $\alpha = 0.0001$.

Figures 10(a)–10(c), we can find that L_{ce} is difficult to converge when α is too small. Besides, L_{KL} in Figure 10(e) is larger than that in Figure 10(d). As our experience, we hope to get smaller L_{KL}; α is set to be 0.00025 for all experiments.

5. Conclusions

In this paper, we have proposed a method to improve the detection problem of CNN in the imbalanced defect data set by adding noise in the feature space. A simple design method for selecting structure of CNN was first introduced, and then, we add noise in feature space of CNN to obtain proper features by the training process and to improve the classification results. In addition, a hybrid loss function of crossentropy and KL divergence was adopted for training. For general uses in training data, CNN can distinguish the defective samples from nondefective samples. However, the results on the testing data are not as good as those on the training data; therefore, we added noise to the feature space of CNN to use it to prevent the network from being limited by the minority samples in training. We prevented noise from being removed by the KL-divergence limit during training. Finally, several comparison results of three data sets are introduced to demonstrate the performance and effectiveness of $\text{CNN}_{\text{noise}}$. Through different data sets, we can also verify that our method is a general method in imbalanced data. Especially, DAMG 2007 and NEU are synthetic data sets for defect detection on textured surfaces; our approach performs well with smaller network structure compared with other deep models. In addition, the performance is improved over 40% in defective accuracy by the adding noise approach. Finally, the accuracy is higher than 96%; even the imbalanced ratio (IR) is one hundred. As shown above, the proposed method can be applied for defect detection problems and other imbalanced data sets after fine-tuning.

Data Availability

Three open data sets (DAGM 2007, NEU surface defect, and MNIST) are utilized to demonstrate the performance and effectiveness of our method.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Ministry of Science and Technology, Taiwan, under contracts MOST 110-2634-F-009-024, 109-2634-F-009-031, and 109-2218-E-005-015.

References

[1] X. Tao, D. Zhang, W. Ma, X. Liu, and D. Xu, "Automatic metallic surface defect detection and recognition with convolutional neural networks," *Applied Sciences*, vol. 8, no. 9, p. 1575, 2018.

[2] D. Soukup and R. Huber-Mörk, "Convolutional neural networks for steel surface defect detection from photometric stereo images," *International Symposium on Visual Computing*, pp. 668–677, 2014.

[3] T. He, Y. Liu, C. Xu, X. Zhou, Z. Hu, and J. Fan, "A fully convolutional neural network for wood defect location and identification," *IEEE Access*, vol. 7, pp. 123453–123462, 2019.

[4] D. M. Tsai and J. Y. Luo, "Mean shift-based defect detection in multicrystalline solar wafer surfaces," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 1, pp. 125–135, 2011.

[5] H. Y. Ngan, G. K. Pang, and N. H. Yung, "Automated fabric defect detection–a review," *Image Vision Computing*, vol. 29, no. 7, pp. 442–458, 2011.

[6] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015.

[7] X. Xie, "A review of recent advances in surface defect detection using texture analysis techniques," *Electronic Letters on Computer Vision Image Analysis*, vol. 7, no. 3, pp. 1–22, 2008.

[8] C. Jian, J. Gao, and Y. Ao, "Automatic surface defect detection for mobile phone screen glass based on machine vision," *Applied Soft Computing*, vol. 52, pp. 348–358, 2017.

[9] Y. J. Jeon, D. C. Choi, S. J. Lee, J. P. Yun, and S. W. Kim, "Steel-surface defect detection using a switching-lighting scheme," *Applied Optics*, vol. 55, no. 1, pp. 47–57, 2016.

[10] Z. He, Y. Wang, F. Yin, and J. Liu, "Surface defect detection for high-speed rails using an inverse P-M diffusion model," *Sensor Review*, vol. 36, no. 1, pp. 86–97, 2016.

[11] R. Kail, A. Zaytsev, and E. Burnaev, "Recurrent convolutional neural networks help to predict location eqrthquakes," 2020, arXiv prprint arXiv: 2004.09140.

[12] S. Fotouhi, S. Asadi, and M. W. Kattan, "A comprehensive data level analysis for cancer diagnosis on imbalanced data," *Journal of Biomedical Information*, vol. 90, article 103089, 2019.

[13] Y. J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, 2017.

[14] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Into Imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[15] J. Yin, C. Gan, K. Zhao, X. Lin, Z. Quan, and Z.-J. Wang, "A novel model for imbalanced data classification," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 6680–6687, 2020.

[16] N. Noorhalim, A. Ali, and S. M. Shamsuddin, "Handling imbalanced ratio for class imbalance problem using SMOTE," *Proceedings of the Third International Conference on Computing, Mathematics and Statistics*, , pp. 19–30, Springer, Singapore, 2019.

[17] A. G. Adama and E. Eyad, "MFC-GAN: class-imbalanced dataset classification using multiple fake class generative adversarial network," *Neurocomputing*, vol. 361, pp. 212–221, 2019.

[18] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from Imbalanced Data Sets*, Springer, 2018.

[19] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup-beyond empirical risk minimization," 2017, ArXiv preprint X: 1710.09412.

[20] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "Cutmix: regularization strategy to train strong classifier with locatizable features," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6023–6032, Seoul, Korea (South), 2019.

[21] P. Shamsolmoali, M. Zareapoor, L. Shen, A. H. Sadka, and J. Yang, "Imbalanced data learning by minority class augmentation using capsule adversarial networks," *Neurocomputing*, vol. 459, no. 12, pp. 481–493, 2020.

[22] K. Deepshikha and A. Naman, "Removing class imbalance using polarity-GAN: an uncertainty sampling approach," 2020, arXiv preprint arXiv:2012.04937.

[23] E. Sharifnia and R. Boostani, "Instance-based cost-sensitive boosting," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 34, no. 3, article 2050002, 2020.

[24] X. Hong, S. Chen, and C. J. Harris, "A kernel-based two-class classifier for imbalanced data sets," *IEEE Transactions on Neural Networks Learning Systems*, vol. 18, no. 1, pp. 28–41, 2007.

[25] D. Wu, Z. Wang, Y. Chen, and H. Zhao, "Mixed-kernel based weighted extreme learning machine for inertial sensor based human activity recognition with imbalanced dataset," *Neurocomputing*, vol. 190, pp. 35–49, 2016.

[26] B. W. Yap, K. A. Rani, H. A. A. Rahman, S. Fong, Z. Khairudin, and N. N. Abdullah, "An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets," *Proceedings of the First International Conference on Advanced Data and Information Engineering*, vol. 285, pp. 13–22, 2014.

[27] A. Bansal, "Meta balance: high-performance neural networks for class-imbalanced data," 2021, arXiv preprint arXiv:2106.09643.

[28] Y. Yan, M. Chen, M. L. Shyu, and S. C. Chen, "Deep learning for imbalanced multimedia data classification," in *2015 IEEE International Symposium on Multimedia (ISM)*, pp. 483–488, Miami, FL, USA, 2015.

[29] M. M. Lopez and J. Ventura, "Dilated convolutions for brain tumor segmentation in MRI scans," in *International MICCAI Brainlesion Workshop*, pp. 253–262, Springer, Cham, 2018.

[30] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1–55, 2019.

[31] M. Wieler and T. Hahn, *Weakly Supervised Learning for Industrial Optical Inspection*, 2019, https://hci.iwr.uni-heidelberg.de/node/3616.

[32] S. Kechen and Y. Yunhui, *NEU Surface Defect Database*, 2019, http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html.

[33] L. Deng, "The MNIST database of handwritten digit images for machine learning research [Best of the Web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141-142, 2012.

[34] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics," *Information Sciences*, vol. 250, pp. 113–141, 2013.

[35] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.

[36] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, "Training deep neural networks on imbalanced data sets," in *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 4368–4374, Vancouver, BC, Canada, 2016.

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 1, pp. 1097–1105, 2012.

[38] G. Wu and E. Y. Chang, "KBA: kernel boundary alignment considering imbalanced data distribution," *IEEE Transactions on Knowledge Data Engineering*, vol. 17, no. 6, pp. 786–795, 2005.

[39] J. Sum and C. S. Leung, "Learning algorithm for Boltzmann machines with additive weight and bias noise," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 10, pp. 3200–3204, 2019.

[40] L. Lin, G. Ravitz, M. L. Shyu, and S. C. Chen, "Effective feature space reduction with imbalanced data for semantic concept detection," in *2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (sutc 2008)*, pp. 262–269, Taichung, Taiwan, 2008.

[41] J. Wang, Y. Ma, L. Zhang, R. X. Gao, and D. Wu, "Deep learning for smart manufacturing: methods and applications," *Journal of Manufacturing Systems*, vol. 48, pp. 144–156, 2018.

[42] L. Shang, Q. Yang, J. Wang, S. Li, and W. Lei, "Detection of rail surface defects based on CNN image recognition and classification," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pp. 45–51, Chuncheon, Korea (South), 2018.

[43] X. Xia, C. Xu, and B. Nan, "Inception-v3 for flower classification," in *International Conference on Image, Vision and Computing*, pp. 783–787, 2017.

[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[45] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2014, arXiv preprint arXiv:1409.1556.

[46] G. Panchal, A. Ganatra, Y. Kosta, and D. Panchal, "Behaviour analysis of multilayer Perceptronswith multiple hidden neurons and hidden layers," *International Journal of Computer Theory Engineering*, vol. 3, no. 2, pp. 332–337, 2011.

[47] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra, *Reducing overfitting in deep networks by decorrelating representations*, International Conference on Learning Representations, 2016.

[48] M. W. Akram, G. Li, Y. Jin et al., "CNN based automatic detection of photovoltaic cell defects in electroluminescence images," *Energy*, vol. 189, no. 116319, p. 116319, 2019.

[49] C. Souad, B. P. Jenny, and B. A. Chokri, "ChaboNet : design of a deep CNN for prediction of visual saliency in natural video," *Journal of Visual Communication and Image Representation*, vol. 60, pp. 79–93, 2019.

[50] G. Xu, H. Z. Wu, and Y. Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, 2016.

[51] H. Y. Chen and C. H. Lee, "Vibration signals analysis by explainable artificial intelligence (XAI) approach: application on bearing faults diagnosis," *IEEE Access*, vol. 8, pp. 134246–134256, 2020.

[52] C. H. Hung, S. X. Zeng, C. H. Lee, and W. T. Li, "End-to-end deep learning by MCU implementation: an intelligent gripper for shape identification," *Sensors*, vol. 21, no. 3, p. 891.

[53] K. Napierała, J. Stefanowski, and S. Wilk, "Learning from imbalanced data in presence of noisy and borderline examples," *International Conference on Rough Sets and Current Trends in Computing*, vol. 6086, pp. 158–167, 2010.

[54] J. V. Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental perspectives on learning from imbalanced data," in *International Conference on Machine Learning*, pp. 935–942, 2007.