

Metal Defect Detection Based on Convolutional Neural Networks and Data Augmentation

Runxin Zang

*School of Information Science and Engineering, Southeast University, Nanjing, China
213221841@seu.edu.cn*

Abstract: This paper explores the classification of metal surface defects, focusing on using deep learning techniques to improve automated detection and classification accuracy. Traditional detection methods have limitations such as insufficient accuracy and low efficiency, while convolutional neural network methods have gradually replaced them, making significant progress. However, issues like data scarcity, complex training processes, and overfitting remain. To address these, the paper uses the ResNet50 network combined with data augmentation techniques to expand the training dataset, enhancing the model's generalization ability and accuracy. Experimental results show that the model with data augmentation outperforms the one without it, improving validation accuracy from 95% to 96%. However, the improvement is not substantial, possibly due to image distortion during augmentation. In conclusion, the ResNet50-based metal surface defect classification model, enhanced with data augmentation, has made positive progress in improving accuracy but requires further research into specific enhancement methods to reduce distortion effects. Future work could focus on optimizing data augmentation strategies and improving the model's ability to identify rare defects.

Keywords: data augmentation, convolutional neural networks, ResNet50, metal surface defects

1. Introduction

The task of classifying metal surface defects primarily involves detecting and categorizing various defects that may occur on the metal surface during production. The types of metal surface defects are numerous, with common ones including scratches, cracks, holes, stains, and rolled scales. To ensure the quality of metal products, especially in fields such as aerospace, automotive, and construction, it is crucial to conduct timely and accurate detection and classification of these defects.

Traditional defect detection methods often rely on manual visual inspection, optical instruments, and ultrasonic testing. These methods have issues with accuracy, efficiency, cost, and scalability. With the rapid development of computer vision and deep learning technologies, convolutional neural networks (CNNs) have been widely applied to automated detection and classification tasks for metal surface defects, significantly enhancing the automation level and precision of defect detection.

The existing methods can be divided into two categories: traditional image processing methods and machine learning-based methods. Traditional image processing methods mainly include edge detection, morphological methods, color analysis, etc. Machine learning-based methods primarily

include SVM, random forest, K-means clustering, and generative adversarial networks (GANs). Additionally, CNN-based methods, such as transfer learning, commonly employ networks like AlexNet, VGGNet, etc., to directly train the original images and automatically extract image features through multiple convolutional layers, pooling layers, and fully connected layers for the classification of metal surface defects. To improve the network's performance, some improved CNN architectures, such as ResNet, DenseNet, and Inception, have been applied to metal defect classification, further enhancing classification accuracy and robustness.

Traditional image processing methods rely on image pre-processing and manual feature extraction, and they are sensitive to noise. The defects of different metal surfaces have varying features, and traditional methods are often unable to extract universal features, resulting in poor generalization ability. While machine learning methods also have many limitations. Traditional methods, such as SVM, rely on artificial features, and the feature selection and extraction process is complex and time-consuming. Furthermore, feature selection for different defects is difficult, and the ability to process large datasets is poor, which can make computational efficiency and memory consumption bottlenecks in large-scale datasets.

For GANs, the training process is complicated and prone to issues such as non-convergence or mode collapse. For convolutional neural networks, CNNs require a large amount of annotated data for effective training. In many practical scenarios, there is insufficient labeled defect data, especially for rare defects. CNNs also have long training times, especially with deep networks and large datasets, where the training process can be very time-consuming and require substantial computational resources. If the training set is small or the data distribution is unbalanced, CNNs are prone to overfitting, fitting the training data but exhibiting poor generalization ability on new data. Overall, although CNN-based defect classification methods have made remarkable progress in many areas, they still face challenges such as data scarcity, complex model training, and overfitting.

To address the problem of data scarcity and overfitting in CNNs, this paper uses ResNet50 to reduce the impact of overfitting on the model. In order to improve the accuracy of training parameters, this paper also adopts data augmentation to enhance the model's generalization ability and expand the dataset. The following seven methods are used:

- Rotation range: This parameter controls the angular range of image rotation. A value of 40 means the images will rotate randomly between -40 and +40 degrees.
- Width shift range: This parameter controls the range of horizontal translation of the image. A value of 0.2 means the image will be randomly translated within 20% of its width.
- Height shift range: This parameter controls the range of vertical translation of the image. A value of 0.2 means the image will be randomly translated within 20% of its height.
- Shear range: This parameter controls the shear transformation of the image, simulating perspective transformation. A value of 0.2 means a maximum shear transformation of 20% will be applied.
- Zoom range: This parameter controls the scale of the image. A value of 0.2 means the image will randomly zoom in or out by up to 20%.
- Horizontal flip: This parameter allows the image to flip horizontally. When set to True, the image will flip horizontally with a 50% probability.
- Fill mode: This parameter specifies how to fill the blank areas in the image during translation or rotation. "Nearest" represents filling the blank areas with the nearest pixel value.

The structure of the article is as follows: The second part presents related work on the NEU-Cluster open-source dataset and computer vision (CV). The third part introduces the concept and related formulas of data augmentation and neural network training (ResNet50). The fourth part presents the experimental results of the model with and without data augmentation. The fifth section highlights the conclusions and future work.

2. Related work

Song and Yan introduced a noise-robust CLBP descriptor for surface defect detection on hot-rolled steel [1]. Krizhevsky, Sutskever, and Hinton achieved state-of-the-art performance in image classification using deep convolutional neural networks [2]. He et al. improved image recognition with ResNet by solving deep network training challenges [3]. Shorten and Khoshgoftaar reviewed various image augmentation techniques in deep learning [4]. Li et al. developed a real-time YOLO network for steel strip defect detection [5]. Simonyan and Zisserman explored deep convolutional networks and proposed the VGG architecture [6]. Jarvis reviewed range-finding techniques for computer vision [7]. Zhang et al. improved YOLOv5s for hot-rolled steel defect detection with multi-scale feature fusion and attention mechanisms [8]. Zhang et al. introduced a YOLOv5s-based model for detecting small metal surface defects [9]. Lv et al. proposed an improved YOLOv7 model for high-precision strip steel defect detection [10]. Yang et al. provided a comprehensive survey on image augmentation techniques [11]. Akyon et al. proposed a new SAHI method for object detection [12]. While Yang et al. proposed a three-stage augmentation framework, Mask2Defect, which integrates prior knowledge and a generative pipeline to generate realistic steel defect images using semi-supervised mask generation and GAN models, thus improving the accuracy of steel defect detection [13]. Xu et al. introduced Scarcity-GAN, a GAN approach designed for data-scarce scenarios, leveraging similar features from auxiliary datasets and generating new defect samples via generative adversarial networks, thereby addressing the problem of limited defect images [14]. Tao proposed an erasing-inpainting augmentation strategy powered by a denoising diffusion probabilistic model (DDPM), which erases non-defective regions from images and inpaints them with synthetic defects, generating diverse defect images from small sample datasets [15]. Tai et al. developed StableSDG, which fine-tunes a pre-trained Stable Diffusion model and employs an "image-oriented" generation approach to create high-fidelity steel defect images, significantly expanding the training data [16]. He et al. conducted a comprehensive survey on the application of generative models in surface defect inspection in manufacturing, covering recent data augmentation techniques and discussing the effectiveness of different models in scenarios with limited data [17].

3. Our method

3.1. ResNet50

ResNet50 is a deep residual network architecture and a variant of ResNet, consisting of 50 layers. Unlike traditional CNNs, ResNet uses residual blocks to address common training issues in deep networks. By introducing skip connections—adding the input x to the output of the residual block—it effectively facilitates gradient flow and accelerates convergence. The core of ResNet is the residual block. The structure of a standard residual block is shown in equation (1).

$$y = F(x, \{W_i\}) + x \quad (1)$$

y is the output of the residual block, $F(x, \{W_i\})$ is the convolution operation and non-linear transformation in the residual block, and W_i represents the weight of the convolution kernel.

Specifically, the skip connection causes the input x to be added directly to the output $F(x, \{W_i\})$ of the convolution layer. The key advantage of this structure is that it allows the gradient to propagate backwards directly through these hopping connections, thus slowing down the gradient vanishing problem.

To understand the residual learning idea in ResNet, we can derive from a simple network block. Supposing we have a traditional neural network, the objective is to minimize the loss function $\mathcal{L}(y, \hat{y})$,

where y is the real label and $\hat{y}$ is the prediction output. For the traditional network, the loss function is shown in equation (2).

$$\mathcal{L}(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2 \quad (2)$$

For the residual block of ResNet, the output of the network is not a direct prediction result, but is obtained by adding the residual $F(x)$ and the jump connection x , as shown in equation (3).

$$\hat{y} = F(x, \{W_i\}) + x \quad (3)$$

The loss function after substituting into the $\hat{y}$ is shown in equation (4).

$$\mathcal{L}(y, \hat{y}) = \frac{1}{2}(y - (F(x, \{W_i\}) + x))^2 \quad (4)$$

In order to optimize network parameters through backpropagation, we need to calculate the gradient of the loss function. For ordinary neural networks, gradient calculation is based on the Chain Rule. In residual blocks, during gradient propagation, skip connections help the gradient pass directly through x .

For the residual network, the gradient of the loss function can be derived by equations (5) and (6).

$$\frac{\partial \mathcal{L}}{\partial F(x, \{W_i\})} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial F(x, \{W_i\})} \quad (5)$$

$$\frac{\partial \mathcal{L}}{\partial F(x, \{W_i\})} = (y - \hat{y}) \cdot \frac{\partial}{\partial F(x, \{W_i\})}(F(x, \{W_i\}) + x) \quad (6)$$

Since x is not affected by the training parameters and its gradient is 1, we have:

$$\frac{\partial \mathcal{L}}{\partial F(x, \{W_i\})} = (y - \hat{y}) \cdot 1 = (y - \hat{y}) \quad (7)$$

This gradient is unaffected by depth and is passed back directly, avoiding the vanishing or exploding gradient problem.

The network structure of ResNet50 includes the input layer, initial convolutional layer, residual blocks, pooling layer, and fully connected layer.

During the training of ResNet50, the following steps are mainly involved: forward propagation, loss calculation, backpropagation, and parameter updates.

3.2. Data augmentation

Data augmentation is a technique in deep learning used to generate additional training samples by applying various transformations to the original data. Its purpose is to expand the training dataset and increase the diversity of samples. By performing these transformations, the model can learn more variation patterns, enhance its ability to identify unseen data, avoid overfitting, and improve generalization. Data augmentation is a widely used technique to improve model robustness and generalization in tasks like image classification. In image classification, object detection, and speech recognition, data augmentation typically involves transforming the original data (e.g., through rotation, zoom, translation, cropping, etc.) to generate new training samples. These new samples not only retain the features of original data, but also incorporate transformation characteristics, allowing the model to better learn data variations and potential underlying patterns.

Furthermore, data augmentation is often performed online during training, meaning that the training data is transformed in real time at each step. This enables the model to encounter different variants of the data in each epoch, thereby improving its generalization ability. Although data

augmentation does not usually rely on specific mathematical formulas, some common augmentation techniques can be mathematically described.

3.2.1. Mathematical formulas for image data augmentation

- Rotation

The point (x, y) , which is in a two-dimensional image, yields a new coordinate (x', y') as shown in equation (8) after rotating an angle θ .

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad (8)$$

This means that the coordinates of each pixel are rotated at the θ angle.

- Translation

Image translation is a simple operation that can be achieved by adding a translation vector. For example, the transformation with a translation amount of (t_x, t_y) can be expressed in equation (9).

$$(x', y') = (x + t_x, y + t_y) \quad (9)$$

- Scaling

For the zoom transformation, if the scaling factor is α (the same scaling factor in the horizontal and vertical directions), the new coordinates are shown in equation (10).

$$(x', y') = (\alpha \cdot x, \alpha \cdot y) \quad (10)$$

- Affine Transformation

Affine transformation is a more general type of transformation, covering rotation, translation, scaling, etc., and the corresponding representation is shown in equation (11).

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \quad (11)$$

$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is the linear transformation matrix of 2×2 , and $\begin{pmatrix} t_x \\ t_y \end{pmatrix}$ is the translation vector.

3.2.2. The target formula for the data augmentation

The goal of data augmentation is to reduce overfitting by increasing the diversity of the training data and improving the model's generalization ability. Suppose we have an original training set $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where each sample x_i and label y_i form a training pair.

After data augmentation, we obtain an extended training set $D' = \{(x'_1, y_1), (x'_2, y_2), \dots, (x'_m, y_m)\}$, where the enhanced sample x'_i is included. The optimization objective of the enhanced training set D' is usually to minimize a specific loss function $\mathcal{L}$, such as cross-entropy loss or mean-squared error, as shown in equation (12).

$$\mathcal{L}(\theta) = \frac{1}{m} \sum_{i=1}^m \mathcal{L}(\hat{y}_i, y_i) \quad (12)$$

$\hat{y}_i = f(x'_i, \theta)$ is obtained by predicting the enhanced sample through the model f , and θ is the parameter of the model.

4. Experimental result

4.1. Experiment setup

4.1.1. Hyperparameter

The hyperparameters used in this experiment are shown in Table 1.

Table 1: Hyperparameters of the experimental results graph after data augmentation

Hyperparameter's name	Value
Model	ResNet50
Learning Rate	0.001
Optimizer	Adam
Batch Size	16
Epochs	30
Loss Function	CrossEntropyLoss
Image Size	224×224
Training Data Directory	"trainaug"
Validation Data Directory	"val"
Device	Cpu
Rotation Range	40°
Width Shift Range	0.2
Height Shift Range	0.2
Shear Range	0.2
Zoom Range	0.2
Horizontal Flip	True
Fill Mode	'nearest'
Number of Enhanced Samples Generated for Each Image	20
Saving Format of the Image	'jpeg'
Directory that Saves the Image	'./trainaug'

4.1.2. Training data

Currently, there are many open-source databases. We use the NEU-Cluster open-source dataset to train the multi-class neural network. The following figures illustrate the experimental use cases presented in this paper.

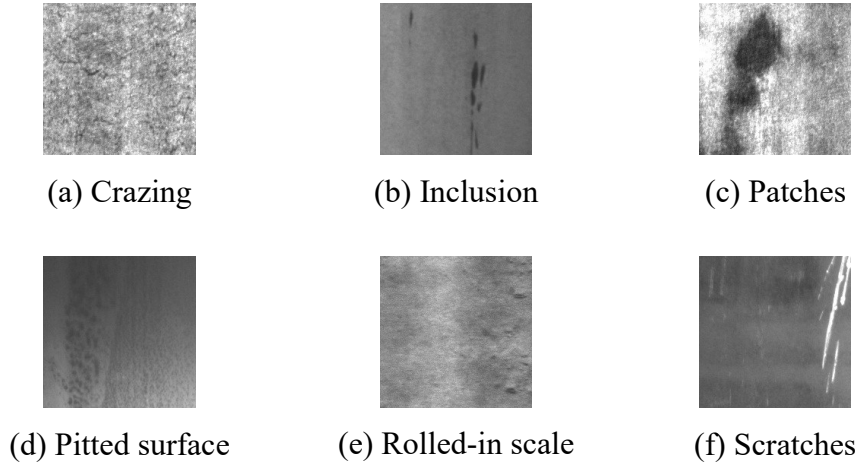


Figure 1: Metal surface maps with six types of defects before data augmentation

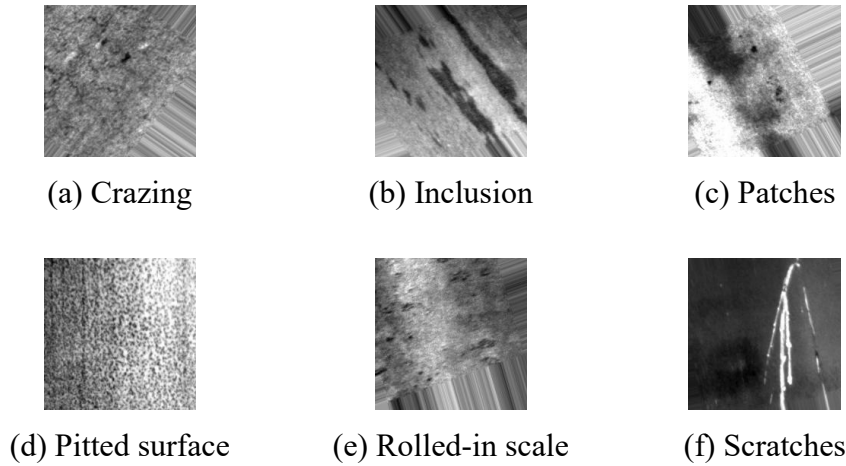


Figure 2: Metal surface maps with six types of defects after data augmentation

4.2. Diagrams of experimental results

In this paper, the training set for each round has been randomly shuffled, meaning that the dataset for each round is different. The experimental results before data augmentation are shown in Figures 3 and 4.

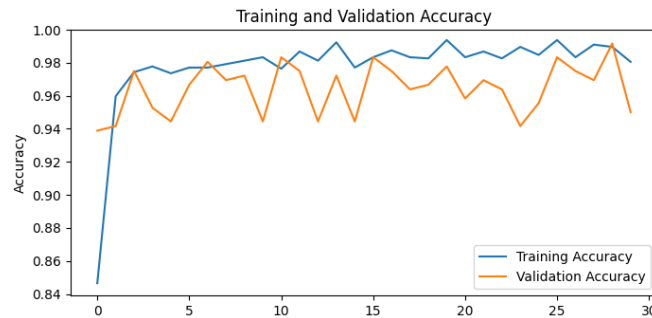


Figure 3: Changes in the training accuracy and validation accuracy before data augmentation with the number of training rounds



Figure 4: Training loss and validation loss versus training rounds before data augmentation

Figure 3 shows the trends in training accuracy (blue curve) and validation accuracy (orange curve). Training accuracy gradually improves as training progresses and eventually stabilizes around 0.98, indicating that the model's performance on the training set is steadily improving and converging. However, validation accuracy exhibits significant fluctuations. Although it improves during the first few epochs, its volatility increases with further training, eventually stabilizing around 0.94. This fluctuation may indicate overfitting. Overfitting typically occurs when the model performs very well on the training set but shows instability or decline on the validation set. This is because the model may have overfitted to the details and noise in the training data, leading to poor generalization on unseen data (i.e., the validation set). The fluctuations in validation accuracy suggest that the model has not adequately learned the general patterns of the data but instead relies too heavily on local features within the training data.

Figure 4 shows the changes in training loss (blue curve) and validation loss (orange curve). In the early stages of training, the training loss drops significantly. As training progresses, the training loss stabilizes, eventually settling around 0.1, indicating that the model's performance on the training set continues to improve, with errors gradually decreasing. However, the validation loss also decreases in the first few epochs but begins to rise after the 25th epoch. The increase in validation loss further confirms overfitting. Although the training loss continues to decrease, suggesting that the model learns more information from the training set, the fluctuations and increases in validation loss indicate a degradation in the model's performance on the validation set. Typically, when the training process overfits the training data, the model may overlook the broader and more general patterns of the data, leading to higher loss values on the validation set.

The experimental results of data augmentation are shown in Figures 5 and 6.

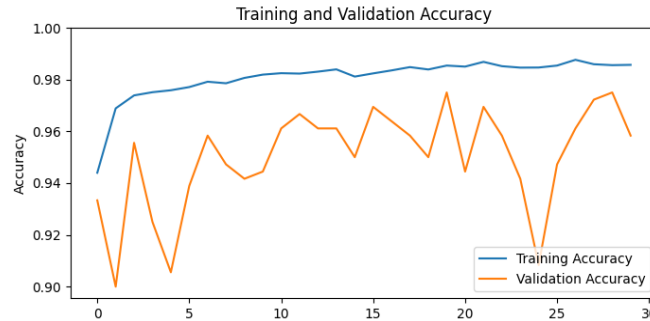


Figure 5: Changes of training accuracy and validation accuracy after data augmentation with the number of training rounds



Figure 6: Changes of training loss and validation loss with the number of training rounds

Figure 5 shows the changes in training accuracy and validation accuracy over the course of the training rounds. As training progressed, the training accuracy gradually increased and stabilized, approaching 0.98, indicating that the model's performance on the training data continuously improved. However, the validation accuracy exhibited significant fluctuations. Although it increased initially, it eventually leveled off at around 0.94. These fluctuations may reflect the onset of overfitting, where the model performs well on the training set but fails to maintain steady improvement on the validation set.

Figure 6 further illustrates the changes in training loss and validation loss. The training loss gradually decreased as the number of training rounds increased and then leveled off, suggesting that the model progressively optimized its performance on the training data, with the error gradually reducing. However, while the validation loss also decreased in the first few rounds, it began to increase in later stages of training, particularly after the 25th epoch. This phenomenon further confirms the presence of overfitting. Although the decreasing loss on the training set suggests that the model is learning effectively from the training data, its performance starts to deteriorate on the validation set. In conclusion, while data augmentation improves training accuracy, it cannot completely prevent overfitting. The instability in the validation set indicates that the model still faces challenges with generalization.

5. Conclusion

In this paper, the resnet50 model is used to classify the defects on the metal surface. After the data augmentation, the validation accuracy increases from 95% to 96%, but the effect is not significant during the data augmentation. The reason may be that there is no data augmentation method specifically for metal surfaces currently. However, according to the images produced after data augmentation in this paper, there are some methods without distortion: symmetrical reversal, multiple rotation of 90 degrees, image amplification. While random rotation and random translation will produce distortion areas as shown in image(a), image(c) and image(e) of Figure 2.

References

- [1] Song, K., & Yan, Y. (2013). A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. *Applied Surface Science*, 285, 858–864.
- [2] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 25, 1097–1105.
- [3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.
- [4] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1), 60.
- [5] Li, J., Su, Z., Geng, J., & Yin, Y. (2019). Real-time detection of steel strip surface defects based on improved YOLO detection network. *IFAC-PapersOnLine*, 52(30), 212–217.

- [6] Simonyan, K., & Zisserman, A. (2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. *International Conference on Learning Representations (ICLR)*.
- [7] Jarvis, R. A. (1983). *A perspective on range finding techniques for computer vision*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2), 122–139.
- [8] Zhang, H., Li, S., Miao, Q., Fang, R., Xue, S., Hu, Q., Hu, J., & Chan, S. (2024). *Surface defect detection of hot rolled steel based on multi-scale feature fusion and attention mechanism residual block*. *Scientific Reports*, 14, Article 7671.
- [9] Zhang, H., Li, S., Miao, Q., Fang, R., Xue, S., Hu, Q., & Hu, J. (2023). *Metal surface defect detection based on improved YOLOv5*. *Scientific Reports*, 13, Article 47716.
- [10] Lv, B., Duan, B., Zhang, Y., Li, S., Wei, F., Gong, S., Ma, Q., & Cai, M. (2024). *Research on surface defect detection of strip steel based on improved YOLOv7*. *Sensors*, 24(9), 2667.
- [11] Yang, S., Liu, H., Wang, Z., & Li, Y. (2023). *A comprehensive survey of image augmentation techniques for deep learning*. *Pattern Recognition*, 111, 107602.
- [12] Akyon, F. C., Altinuc, S. O., & Temizel, A. (2022). *Slicing aided hyper inference and a benchmark for small object detection*. *IEEE International Conference on Image Processing (ICIP)*, 2022, 3203–3207.
- [13] Yang, B., Liu, Z., Duan, G., & Tan, J. (2022). *Mask2Defect: A Prior Knowledge-Based Data Augmentation Method for Metal Surface Defect Inspection*. *IEEE Transactions on Industrial Informatics*, 18(11), 6743–6755.
- [14] Xu, C., Li, W., Cui, X., Wang, Z., Zheng, F., & Cui, X. (2024). *Scarcity-GAN: Scarce Data Augmentation for Defect Detection via Generative Adversarial Nets*. *Neurocomputing*, 566, 127061.
- [15] Tao, H. (2024). *Erasing-inpainting-based data augmentation using denoising diffusion probabilistic models with limited samples for generalized surface defect inspection*. *Mechanical Systems and Signal Processing*, 208, 111082.
- [16] Tai, Y., Yang, K., Peng, T., Huang, Z., & Zhang, Z. (2024). *Defect Image Sample Generation with Diffusion Prior for Steel Surface Defect Recognition*. *IEEE Transactions on Automation Science and Engineering*, early access (Jan 2024).
- [17] He, Y., Li, S., Wen, X., & Xu, J. (2024). *A Survey on Surface Defect Inspection Based on Generative Models in Manufacturing*. *Applied Sciences*, 14(15), 6774.