International Journal of Innovative Computing, Information and Control
Volume **21**, Number **2**, April **2025**

ICIC International ©2025 ISSN 1349-4198
pp. **433–445**

CNN AND GAN BASED WEATHERING STEEL APPEARANCE EVALUATION IMPROVED BY IMAGE SIZE AND LOSS FUNCTION

Kengo Nagatani[1], Noritaka Shigei[1,*], Chihiro Morita[2], Yoichi Ishizuka[3] and Hiromi Miyajima[4]

[1]Graduate School of Science and Engineering
[4]Professor Emeritus of Kagoshima University
Kagoshima University
1-21-40 Korimoto, Kagoshima City, Kagoshima 890-0065, Japan
{ k3858973; k2356323 }@kadai.jp; *Corresponding author: shigei@ibe.kagoshima-u.ac.jp

[2]Department of Civil and Environmental Engineering
University of Miyazaki
1-1 Gakuen Kibanadai-nishi, Miyazaki City, Miyazaki 889-2192, Japan
cgmorita@cc.miyazaki-u.ac.jp

[3]Department of Electric and Electronic Engineering
Faculty of Engineering
Nagasaki University
1-14 Bunkyo, Nagasaki City, Nagasaki 852-8521, Japan
isy2@nagasaki-u.ac.jp

Received July 2024; revised October 2024

Abstract. *Weathering steel is a steel material that can maintain its performance outdoors for a long period of time by forming protective rust. Periodic inspection of the rust condition is necessary to maintain this steel in good condition, and CNN-based methods for evaluating the appearance of rust have been studied. One of the problems in improving the classification accuracy of the CNN is how to prepare the rust images for training. The objectives of this paper are 1) to consider rust patch image size and 2) to improve the diversity of the generated images in rust image generation using GAN. From 1), we show that the accuracy can be improved by using large cropping images and shrinking the image when training the model. From 2), we propose SSIM-target loss, which is an improvement of the conventional SSIM loss, and show that it improves the CNN Score. Using methods 1) and 2), we achieved an F1 score of 90.3% for the three-class classification of the rust images, compared to the baseline method's F1 score of 86.8%.*

Keywords: Rust images, Generative adversarial network, Convolutional neural network, Classification, Weathering steel

1. **Introduction.** Weathering steel is a steel material that produces a protective rust layer on the surface of the steel that suppresses the propagation of corrosion [1]. Many bridges have been constructed using this weather-resistant material. However, periodic inspections are required to evaluate the rusting condition because protective rusting may not be in good condition, depending on the environment. One of the used inspection methods is the cellophane tape test, in which rust on the surface of the steel is collected by adhering it to cellophane tapes and evaluated in terms of appearance [2]. Evaluation of this test is usually performed visually by an inspector. However, in this case, there is a problem that the evaluation depends on the subjectivity of the inspector. On the other hand, non-human inspection methods using Convolutional Neural Networks (CNNs) have

DOI: 10.24507/ijicic.21.02.433

been studied [3, 4, 5]. In the methods, CNN models trained with labeled images rate the rust condition from the target rust image.

In the previous studies [3, 4, 5], the rust image classification using CNN was trained by extracting patch images from tape images. 1) However, since the rust images were labeled by tape unit, there was a problem that the rust Scores were different within the same image. In [3], the effect of the size of the patch image to be cropped has been investigated, showing that the larger the patch image size, the better the accuracy. In [5], a decision method combining judgement of multiple patch images has been proposed and shown to be effective. On the other hand, to compensate for the lack of training data, generation of rust images using Generative Adversarial Networks (GANs) [6] has also been considered [7, 8]. 2) However, in order to use the generated rust images as training data for CNN, it is necessary to improve the quality and diversity of the generated images. In [7], a basic study on the generation of rust in GANs has been made. In [8], WGAN-GP and SSIM-L-WGAN-GP have been proposed, obtaining an F1 score of 88.0%, but further improvements are desired.

In this paper, we propose a method for 1) to suppress the variation of Scores of rusts in the same image by cropping a larger patch image. We also show that the accuracy of the CNN is improved by resizing the patch image to a smaller size when training the model. For 2), we add a loss function to the generator of the GAN to promote diversity in the generated images. We propose SSIM-target loss, which is an improvement of SSIM loss [8], and show that the accuracy of CNN classification is improved by improving the quality of the generated images.

This study has two objectives. First, to improve the accuracy of the CNN by training rust patch images by cropping them large and then resizing them small; second, to generate rust images that are effective for training the CNN by introducing SSIM-target loss into the GAN. The remainder of this paper is organized as follows. Section 2 describes the cellophane tape test and appearance evaluation using CNNs. Section 3 describes data augmentation using GANs and loss functions to improve the diversity of GANs. Section 4 proposes a method of cropping a patch image to a large size and then resizing it to a smaller size in order to suppress the variation of rust Scores in the same image. We also propose SSIM-target loss, which is an improvement of SSIM loss in the loss function of GAN. Section 5 shows the effectiveness of the proposed method, since the F1 score of the CNN classification accuracy reaches 90% by using the proposed method. Finally, in Section 6, we present our conclusions.

2. **Rust Appearance Evaluation and CNN.**

2.1. **Cellophane tape test.** The cellophane tape test is one of the methods to evaluate the rusting condition of weathering steel. Rust formed on the steel surface is collected by adhering it to a cellophane tape, and an expert evaluates the adhered rust's condition visually. Table 1 shows the evaluation categories and evaluation criteria for the current weathering steel products in Japan [9]. The higher the Score, the better the rust condition; the lower the Score, the worse the rust condition. The rust with a Score of 3 or higher does not require any action, while the rust with a Score of 2 or lower needs to be treated or observed and disposed of. It is especially important to distinguish between Scores 2 and 3. In this study, we examine three classifications of images: Scores 2 and 3, which are important to identify, and Score 4, which considers the presence or absence of damage.

2.2. **Convolutional Neural Network (CNN).** This paper uses CNN as a classifier for identifying the grade of rust conditions, as in [3, 4, 5, 8]. The input to the CNN model is a patch image shown in Figure 1, and the output of the model is the Score (2, 3, or 4) of the

TABLE 1. Rust evaluation categories

Score	Appearance	Sample patch
5	The amount of rust is small, and the color is relatively bright. No damage.	–
4	Rust is fine and uniform, with a size of less than 1 mm. No damage.	
3	Rust size is 1-5 mm and coarse.	
2	Rust is scaly, 5-25 mm in size.	
1	Rust has a layered exfoliation.	–

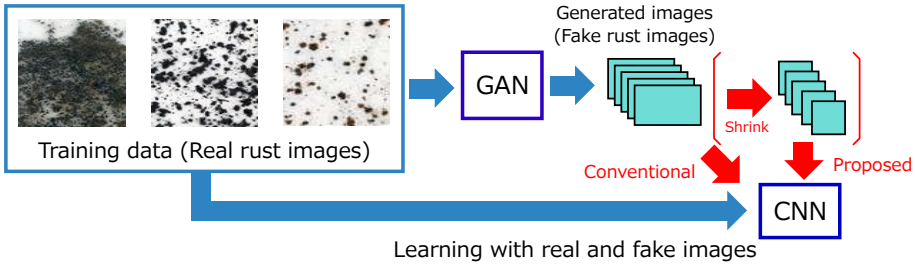


FIGURE 1. Data augmentation with GAN and its processing flow [8]

rust in the patch image. The CNN used is a general model consisting of convolutional, pooling, dropout, fully connected, and output layers. The Score corresponding to the maximum output unit in the output layer is selected as the decision result. The details will be presented in Section 5.2.

3. Data Augmentation with GANs.

3.1. Generative Adversarial Networks (GANs).

GANs are deep learning models consisting of two networks: a generator and a discriminator. The two networks learn by competing with each other. The generator $G(\boldsymbol{z})$ tries to generate images that resemble the training data, and the discriminator $D(\boldsymbol{x})$ tries to detect that the images generated by the generator are not real. There are many variations of GANs. [7] used Deep Convolutional Generative Adversarial Networks (DCGANs) and our previous study used Conditional Generative Adversarial Nets (CGANs) and Wasserstein GAN-Gradient Penalty (WGAN-GP). Since WGAN-GP performs better than DCGAN and CGAN, this study uses WGAN-GP.

WGAN-GP [11] is an improved GAN model from Wasserstein GAN (WGAN) [12]. Conventional GANs are difficult to learn and have problems of gradient vanishing and mode collapse. WGAN is based on the Wasserstein distance to design the loss function. WGAN has the advantage that the gradient does not disappear near the optimal point of the parameters, thus stabilizing the learning process. WGAN-GP is an improvement of WGAN by adding a gradient penalty to the WGAN loss function. This is a constraint for the discriminator $D(\boldsymbol{x})$ to be a Lipschitz function. The objective function of WGAN-GP, with the gradient penalty added, is expressed as follows:

$$L = \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[D(\hat{\boldsymbol{x}})] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})] + \lambda \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}\left[(\|\Delta_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2\right], \tag{1}$$

where $\mathbb{P}_r$ is the data distribution, $\mathbb{P}_g$ is the generator distribution, $\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}$ means random samples between sampled pairs $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$, the second term is the gradient penalty, and λ is the coefficient for the gradient penalty term.

For data augmentation using GAN, the process flow is shown in Figure 1. In order to apply the generated rust images as training data for the CNN, it is necessary to promote diversity of the generated images. Our previous study [8] proposed to use L_1, L_2, and LBP loss in order to promote diversity in the generated images. By applying these loss functions to the CGAN and the WGAN-GP, the diversity of the generated images can be improved.

The L1 loss L_1 and the L2 loss L_2 are defined as follows:

$$L_l = 1 - \frac{1}{{}_{bs}\mathrm{C}_2} \sum_{i\neq j} |G(\boldsymbol{z}_i) - G(\boldsymbol{z}_j)|^l, \tag{2}$$

where $l \in \{1, 2\}$, ${}_{bs}\mathrm{C}_2$ is the total combination of two generated images from within the batch with size bs, and $G(\boldsymbol{z}_i)$ is the i-th generated image by the generator. The loss function of the generator with L1 and L2 losses is as follows:

$$L_{\mathrm{L1\cdot L2-g}} = L_{\mathrm{g}} + \lambda_1 L_1 + \lambda_2 L_2, \tag{3}$$

where L_{g} is the generator loss, such as $L_{\mathrm{g}} = -\sum \log(D(\boldsymbol{x}))$, and λ_1 and λ_2 are the weights of the L1 and L2 loss terms.

3.2. **LBP loss function.** In [8], it has been proposed to use the texture feature Local Binary Pattern (LBP) [13] in loss functions in order to promote diversity in the generated images of GANs. The LBP is used to calculate the similarity of the textures of the rust images, and the diversity is improved by learning to reduce the similarity between the generated images. LBP is calculated as follows:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) 2^p, \tag{4}$$

where g_c is the value of the center pixel, g_p is the value of the p-th pixel in the neighborhood of g_c, $s(x) = 1$ for $x \geq 0$ otherwise $s(x) = 0$, P is the number of pixels in the neighborhood, and R is the radius of the neighborhood of the center pixel.

The LBP loss function is calculated as follows.

Step 1: LBP is calculated for all images in the batch using Equation (4).

Step 2: Make a histogram of the number of occurrences of each LBP value.

Step 3: For all pairs of images in a batch, the similarity of histograms is calculated using Euclidean distance.

Step 4: Normalize the similarity calculated in Step 3 and compute the average LBP_{hist}.

Step 5: Calculate the LBP loss $L_{\mathrm{LBP}} = 1 - LBP_{\mathrm{hist}}$. □

The loss function of the generator with the LBP loss is $L_{\mathrm{LBP_g}} = (1 - w_{\mathrm{LBP}})L_{\mathrm{g}} + w_{\mathrm{LBP}}L_{\mathrm{LBP}}$, and w_{LBP} is the weight of L_{LBP}.

4. Proposed Methods.

4.1. **Expansion of image cropping size.** In [5] and [8], the scanned images of cellophane tape are not used directly. Instead, cropped patch images are used as inputs for CNN. One reason for this is that increasing the size of the CNN input image tends to degrade the accuracy. On the other hand, since the rust condition is not uniform across the entire tape, there is a problem that the image does not always reflect the grade of the rust depending on the crop position of the patch image.

To solve the problem, we propose to expand the cropping image size and to use from 100×75 px to 150×75 px for 50 dpi. As shown in Figure 2, the height will be increased from 100 px to 150 px. Even when the cropping size is increased, it is desirable to crop as many images as possible. Therefore, the image is cropped by shifting the cropping position at regular intervals in the horizontal direction while overlapping. By increasing the size of the patch image, it can be expected that the correct label is applied to the patch image. In addition, by combining the proposed method with the image generation of GAN, the generated image size of GAN can be increased. The quality of the generated images is also expected to be improved by the GAN learning important features of rust.

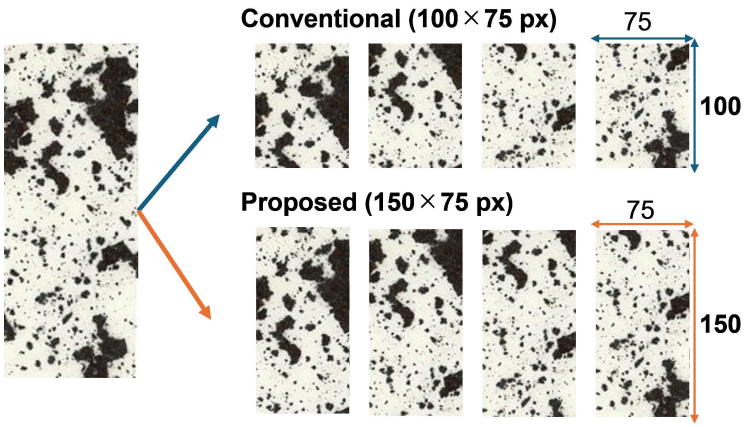


FIGURE 2. Expanded cropping from tape image to patch image

4.2. **Asymmetric reduction of CNN input image resolution.** In [5], the input image of CNN is 100×75 px in 50 dpi. On the other hand, in [8], the image of 100×75 px is reduced to 28×28 px and used as the input image. As a result, the resolution of the resulting reduced image is different in the horizontal and vertical directions. The horizontal and vertical DPIs are 18.7 dpi and 14.0 dpi, respectively, and their resolutions per pixel are 1.36 mm and 1.81 mm, respectively. The resolution of the reduced image is low resolution and asymmetric in the horizontal and vertical directions, but it has achieved a good classification accuracy in the upper 80% range.

We again explicitly propose this lower resolution of asymmetric input images as an effective method for improving CNN accuracy. Specifically, we propose to downscale a 150×75 patch image at 50 dpi to 28×28 and input it to the CNN. Our observations on why this approach works well are as follows. When labeling rusts, experts focus on the shape of large rusts in the tape and label them, so it is necessary for the CNN to capture the shape of large rusts. If the CNN is trained on large patch images, even small rust shapes will stand out and become noise, and the CNN may miss important rust features.

On the other hand, by downscaling the patch image it is expected that small rust features are suppressed and large rust features are emphasized.

4.3. **SSIM-target loss function.** In rust images generation, SSIM loss has been proposed in [8] to promote diversity in the generated images of GANs. SSIM loss calculates the similarity of images based on the mean and variance of pixel values, and trains GAN generators to reduce the similarity. SSIM loss is defined by the following equation:

$$L_{\mathrm{SSIM}} = \frac{1}{{}_{bs}\mathrm{C}_2} \sum_{i \neq j} \mathrm{SSIM}(G(\boldsymbol{z}_i), G(\boldsymbol{z}_j)) \tag{5}$$

$$L_{\mathrm{SSIM_g}} = (1 - w_{\mathrm{SSIM}}) L_{\mathrm{g}} + w_{\mathrm{SSIM}} L_{\mathrm{SSIM}}, \tag{6}$$

where the w_{SSIM} is the weight of L_{SSIM} and

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)}. \tag{7}$$

x and y are small regions centered on pixels at the same position in the two images to be compared. μ_x and μ_y are means of pixel values in the small region of x and y, σ_x and σ_y are standard deviations, and σ_{xy} is covariance. C_1 and C_2 are constants, $C_1 = (0.01 \times 255)^2$ and $C_2 = (0.03 \times 255)^2$ in the case of grayscale. This small area is calculated for all pixels, and the average value is SSIM. SSIM takes values between 0 and 1, and the closer the value is to 1, the higher the similarity.

SSIM loss is a learning method such that the value of L_{SSIM} in Equation (5) is close to 0. However, a fake image with too low SSIM may not be suitable as a rust image because it loses its realness.

To solve this problem, we propose a learning method such that the value of L_{SSIM} is close to the average value of SSIM of the real image in a batch. The difference between the average SSIM of the real image in the batch and the average SSIM of the generated image in the batch is defined as L_{SSIMT}, and the learning is performed such that L_{SSIMT} becomes 0. The average value of SSIM of the real image $SSIM_{\mathrm{real}}$ is calculated by the following equation:

$$SSIM_{\mathrm{real}} = \frac{1}{{}_{bs}\mathrm{C}_2} \sum_{i \neq j} \mathrm{SSIM}(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{8}$$

Therefore, using Equations (5) and (8), the SSIM-target loss is as follows:

$$L_{\mathrm{SSIMT}} = |SSIM_{\mathrm{real}} - L_{\mathrm{SSIM}}| \tag{9}$$

The loss function of the generator with the addition of SSIM-target loss is

$$L_{\mathrm{SSIMT_g}} = (1 - w_{\mathrm{SSIMT}}) L_{\mathrm{g}} + w_{\mathrm{SSIMT}} L_{\mathrm{SSIMT}}, \tag{10}$$

where w_{SSIMT} is the weight of L_{SSIMT}.

5. Evaluation Results.

5.1. **Experiment conditions.** The tape image is about 300 × 90 px at 50 dpi. The conventional patch image is 100 × 75 px, and the proposed patch image is 150 × 75 px. The 100 × 75 px dataset is named $w = 100$, and the 150 × 75 px dataset is named $w = 150$. When dividing the training data and the test data, for the comparison, the division procedure is the same for $w = 100$ and $w = 150$. The training dataset consists of 87 items for Score 2, 189 for Score 3, and 366 for Score 4, totaling 642. The test dataset consists of 24 items for Score 2, 51 for Score 3, and 97 for Score 4, totaling 172. The test data are used to evaluate the classification accuracy of the CNN.

The parameters for WGAN-GP are as follows: # of epochs is 5000, batch size is 16, image size is 150×75 or 28×28, image is monochrome, latent dimension is 128, generator learning rate is 2.0×10^{-4}, and the discriminator learning rate is 2.0×10^{-6} for Score 4 and 2.0×10^{-4} for others. For the structure for WGAN-GP, we refer to the official implementation of Keras [14]. Note that the parameters of the input layer are changed according to the size of the input images.

The following 3 models are considered. We generate rust images with each GAN and test which of the generated images contributes to improving the classification accuracy of CNN. The weights of the loss function are as follows: $w_{\mathrm{SSIM}} = 0.1$, $w_{\mathrm{SSIMT}} = 0.1$, $\lambda_1 = 0.001$ and $\lambda_2 = 0.001$.

1) WGAN-GP: Conventional WGAN-GP.
2) SSIM-L-WGAN-GP: WGAN-GP with SSIM loss and L1·L2 loss introduced.
3) SSIMT-L-WGAN-GP: WGAN-GP as proposed in Section 4.2.

5.2. **Evaluation with CNN.** We trained the WGAN-GP models described in Section 5.1 using the dataset $w = 150$ cropped by the proposed method, and generated rust images. We trained the CNN by data augmentation using the generated rust images. The following three types of the input image sizes are considered in the evaluation: 1) 150×75 px without resizing, 2) 56×28 px that is a reduction preserving the aspect ratio, and 3) 28×28 px that is a reduction ignoring the aspect ratio, which is the proposed method. The proposed image size 3) 28×28 px is compared with the conventional image sizes 1) 150×75 px and 2) 56×28 px. The structure of the CNN is tuned according to the size of each input image size. The structure of the CNN is shown in Table 2 for each input image size. The accuracy of the CNN classification is shown in Table 3. For comparison, Table 4 shows the Scores of the CNN when we trained the CNN on the conventional dataset $w = 100$ [8]. Note that when the CNN was trained for dataset $w = 100$, the CNN and GAN were trained at a size of 28×28 px. For comparison, we also show the Scores when the CNN is trained on the real images only and when the CNN is trained on the real images with the conventional online data augmentation using ImageDataGenerator in Keras. The processing contents of the online data augmentation are horizontal and vertical flipping and shifting ±0.3.

Table 2. CNN structure

	For 150×75 px		For 56×28 px		For 28×28 px	
Layer#	Layer type	Param	Layer type	Param	Layer type	Param
1	Input Layer	$75 \times 150 \times 1$	Input Layer	$56 \times 28 \times 1$	Input Layer	$28 \times 28 \times 1$
2	Conv2D	16, 3×3, ReLU	Conv2D	16, 3×3, ReLU	Conv2D	16, 3×3, ReLU
3	Maxpooling2D	2×2	Maxpooling2D	2×2	Maxpooling2D	2×2
4	Dropout	0.4	Dropout	0.4	Dropout	0.4
5	Conv2D	32, 3×3, ReLU	Conv2D	32, 3×3, ReLU	Conv2D	32, 3×3, ReLU
6	Maxpooling2D	2×2	Maxpooling2D	2×2	Maxpooling2D	2×2
7	Conv2D	64, 3×3, ReLU	Conv2D	64, 3×3, ReLU	Conv2D	64, 3×3, ReLU
8	Maxpooling2D	2×2	Maxpooling2D	2×2	Maxpooling2D	2×2
9	Conv2D	128, 3×3, ReLU	Dropout	0.4	Dropout	0.4
10	Maxpooling2D	2×2	Flatten	1344	Flatten	576
11	Conv2D	256, 3×3, ReLU	Dense	640, ReLU	Dense	512, ReLU
12	Maxpooling2D	2×2	Dense	512, ReLU	Dense	3, SoftMax
13	Dropout	0.4	Dense	3, SoftMax		
14	Flatten	2048				
15	Dense	640, ReLU				
16	Dense	512, ReLU				
17	Dense	3, SoftMax				

TABLE 3. Accuracy and F1 of CNN trained for dataset $w = 150$

Training data for CNN	Model	Score 2	Score 3	Score 4	Accuracy	MacroF1
Real images only*	–	0.770	0.752	0.927	0.857	0.816
Real images only**	–	0.892	0.822	0.935	0.897	0.883
Real images only***	–	0.899	0.836	0.933	0.901	0.889
Online Data Augmentation*	–	0.852	0.82	0.932	0.890	0.868
Online Data Augmentation**	–	0.908	0.808	0.924	0.890	0.880
Online Data Augmentation***	–	0.853	0.855	**0.946**	0.907	0.885
Real images+Generate $200 * 3$	WGAN-GP*	0.762	0.728	0.938	0.855	0.809
	WGAN-GP**	0.804	0.776	0.938	0.874	0.839
	WGAN-GP***	0.846	0.812	0.944	0.893	0.868
	SSIM-L-WGAN-GP*	0.782	0.804	0.954	0.886	0.847
	SSIM-L-WGAN-GP**	0.88	0.830	0.942	0.901	0.884
	SSIM-L-WGAN-GP***	0.896	0.828	0.931	0.897	0.885
	SSIMT-L-WGAN-GP*	0.798	0.769	0.936	0.870	0.834
	SSIMT-L-WGAN-GP**	0.86	0.799	0.936	0.888	0.865
	SSIMT-L-WGAN-GP***	**0.905**	**0.858**	**0.946**	**0.915**	**0.903**

CNN trained on 150×75 px for '*', 56×28 px for '**', 28×28 px for '***'

TABLE 4. Accuracy and F1 of CNN trained on dataset $w = 100$ [8]

Training data for CNN	Model	Score 2	Score 3	Score 4	Accuracy	MacroF1
Real images only	–	0.796	0.748	0.930	0.860	0.825
Online Data Augmentation	–	0.790	0.797	0.928	0.872	0.839
Real images+Generate $100 * 3$	WGAN-GP	0.840	0.766	0.923	0.868	0.843
	SSIM-L-WGAN-GP	**0.904**	0.811	0.925	0.890	**0.880**
Real images+Generate $200 * 3$	WGAN-GP	0.845	0.810	0.933	0.884	0.863
	SSIM-L-WGAN-GP	0.889	0.812	0.929	0.890	0.877
Real images+Generate $300 * 3$	WGAN-GP	0.876	**0.821**	**0.935**	**0.893**	0.877
	SSIM-L-WGAN-GP	0.850	0.796	0.932	0.882	0.859

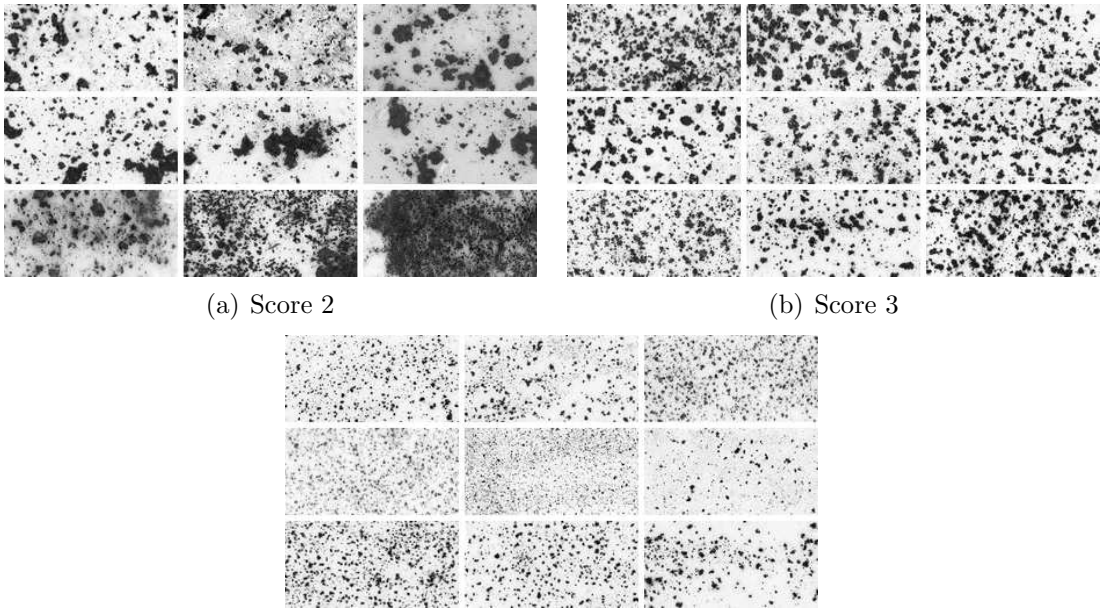
Table 3 shows that the classification accuracy of the rust images is higher when the input size of the patched images is smaller during the training of the CNN. In all cases, 3) 28×28 px, 2) 56×28 px, and 1) 150×75 px, in that order, show higher Scores. The result shows that the asymmetric reduced image of 28×28 px is a more effective data augmentation. By resizing the training image to a smaller size, the features of large rusts are emphasized, and the CNN Score is improved because the CNN has captured the features of large rusts. We consider that the effect of background and noise is relatively reduced by the resizing, and that the CNN was able to learn the rust images effectively. We also consider that the reason for the improved accuracy even when the aspect is ignored is that the shape and size of the rusts are diversified by the asymmetric downsizing, and the CNN is able to learn various features of the rusts. From these results, it can be said that resizing the training image to a smaller size has a significant effect on the improvement of accuracy in the classification problem of rust images.

On the other hand, SSIM-L-WGAN-GP is not as effective as WGAN-GP or SSIMT-L-WGAN-GP in 28×28 px. We guess that the reason for this is due to the nature of SSIM loss. SSIM loss has the property of focusing on the diversity of the generated image, and the generated image is expected to be less like the real rust images. SSIMT loss improves this nature.

Comparing the '***' of the dataset $w = 150$ with the results of the dataset $w = 100$ (Table 3 and Table 4), we find that the accuracy is generally improved by training on the dataset $w = 150$. We consider that this is because the larger size of the patches reduces

the variation of the Scores of rusts in the same image. Even in the case of training with only real images without data augmentation of GAN, the Macro F1 of 100×75 was 0.825, while that of 150×75 was 0.889, an improvement of 6.4% pt. It is shown that the larger size of the patch image when cropping the patch image from the tape image is better to effectively train the CNN, since the correct labels are reflected in the patch image.

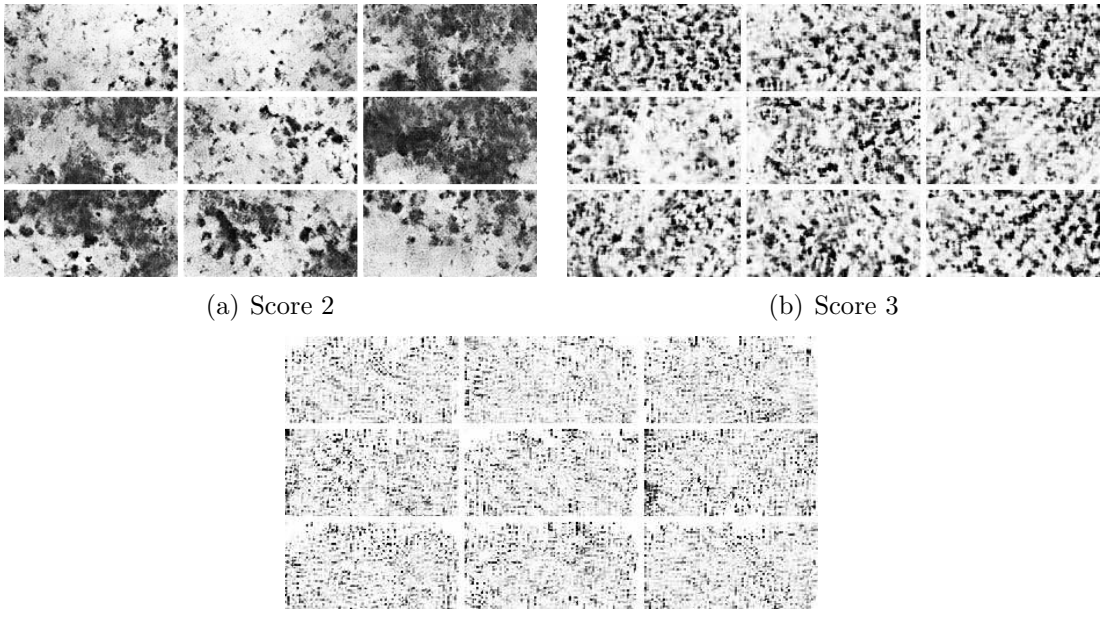
Compare the real rust images with the generated rust images. Figure 3 shows the real patch image with dataset $w = 150$, Figure 4 shows the generated SSIM-L-WGAN-GP



(a) Score 2 (b) Score 3

(c) Score 4

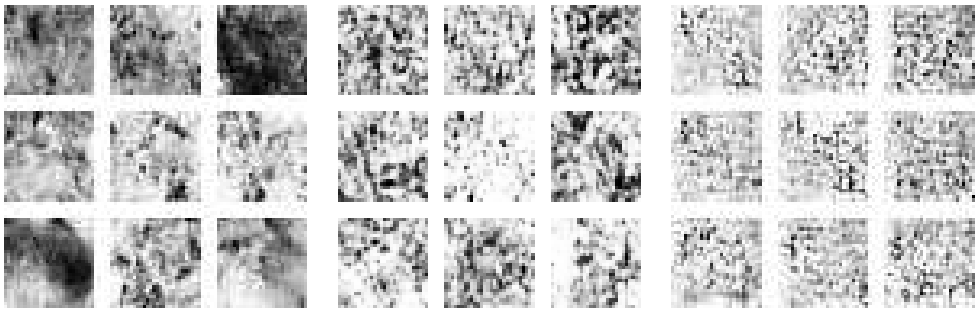
FIGURE 3. Real images with dataset $w = 150$ and image size 150×75 px



(a) Score 2 (b) Score 3

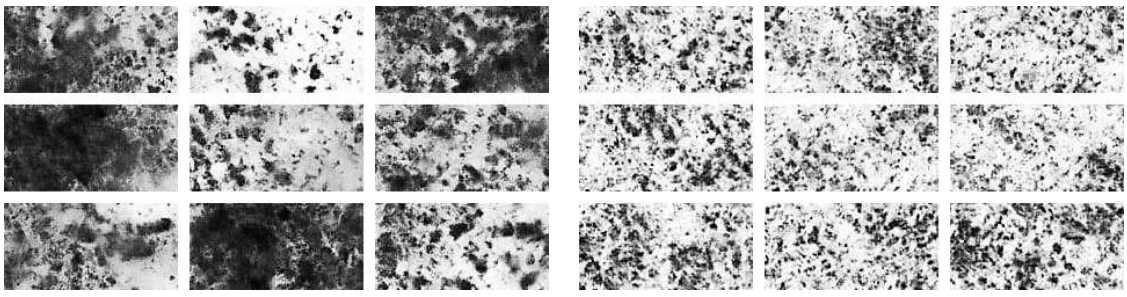
(c) Score 4

FIGURE 4. Generated images with dataset $w = 150$ and image size 150×75 px

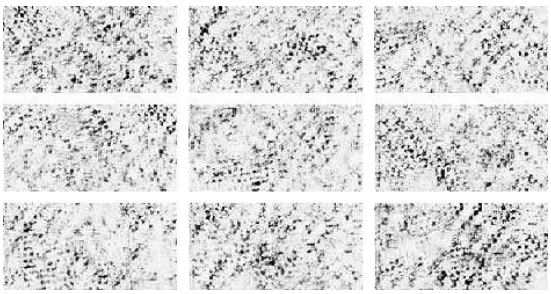


(a) Score 2 (b) Score 3 (c) Score 4

FIGURE 5. Generated images with dataset $w = 100$ and image size 28×28 px



(a) Score 2 (b) Score 3



(c) Score 4

FIGURE 6. Generated images of SSIMT-L-WGAN-GP with the best F1 score

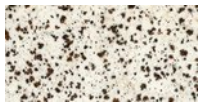
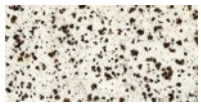
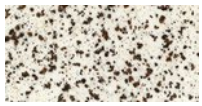
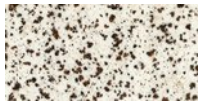
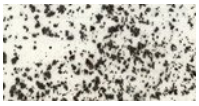
image trained with dataset $w = 150$, and Figure 5 shows the generated SSIM-L-WGAN-GP image trained with dataset $w = 100$. The image sizes of Figure 3 and Figure 4 are 150×75 px and Figure 5 is 28×28 px. From Figure 4, a high-quality rust image similar to the real image is successfully generated. This indicates that using the data with $w = 150$, the size of the input image to GAN and the size of the generated image are increased, and the rust images containing the important features of the rusts are generated. Comparing the generated images in Figure 4 and Figure 5, we can see that the generated image in Figure 4, in which the GAN is trained and generated with a larger size, has a better visual quality. Therefore, it is shown that the proposed method of training GANs with large size patch images is more effective as training data for CNNs, since it improves the quality of the generated images.

Figure 6 shows the generated images of SSIMT-L-WGAN-GP, which obtained the best F1 score of 0.903 in this paper. Visually, we can confirm the diversity of the generated images at each Score.

Regarding the effect of the proposed loss function, it can be seen that the generated images of SSIM-L-WGAN-GP and SSIMT-L-WGAN-GP are more improved in the accuracy of CNN than those of the conventional WGAN-GP. In addition, comparing SSIM-L-WGAN-GP and SSIMT-L-WGAN-GP, the latter tends to have higher Scores; thus, we consider that the quality of the generated images is improved by the proposed SSIM-target loss.

Finally, we show all 14 misclassified patch images for the best Score of 0.903 in Table 5. Compared to the actual images in Figure 3, the eight patch images (1)∼(3), (8) and (11)∼(14) appear to show that the evaluated Scores are not wrong[1]. Therefore, it is considered that the proposed method can achieve an F1 score higher than 0.903.

Table 5. Misclassfied patch images for the best F1 score of 0.903

Label	Evaluation	Patch images
Score 2	Score 3	(1) 
Score 3	Score 2	(2)  (3) 
Score 3	Score 4	(4)  (5)  (6)  (7)  (8)  (9) 
Score 4	Score 2	(10) 
Score 4	Score 3	(11)  (12)  (13)  (14) 

6. **Conclusion.** In this paper, we propose 1) a method to cropping a patch image to a large size in order to correctly reflect the labels of the patch image, 2) a method to train a CNN by resizing it to a small size in order for the CNN to capture important rust features, and 3) SSIM-target loss, an improved SSIM loss to improve the diversity of the generated images. The evaluation results showed that 1) increasing the size of the patch image improves the accuracy of the CNN by reducing the variability of the rust Scores in

[1]Note that the discrepancy between these labels and estimation is thought to depend on the position from which the patch image is cropped.

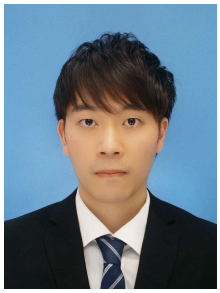
the same image, 2) increasing the size of the patch image enables the GAN to generate larger rust images and improves the quality of the GAN generated images, 3) training the CNN on resized and smaller patch images allowed the CNN to learn the important features of rust and greatly improved the accuracy of the CNN, and 4) the WGAN-GP generated images with SSIM-target loss introduced were effective in improving the accuracy of the CNN. Future work includes developing more effective GAN models and loss functions. One of the considerations is the application of Auxiliary Classifier Wasserstein GAN (ACWGAN) [15], which improves the quality of a small class of generated samples by introducing an auxiliary classifier.

Acknowledgments. This work was partially supported by JSPS KAKENHI Grant Number J24K07642.

REFERENCES

[1] *Application of Weathering Steel to Bridges*, Japan Iron and Steel Federation, Japan Bridge and Steel Construction Association, 2023 (in Japanese).
[2] C. Morita et al., Evaluation of rust appearance of weathering steel by image analysis of cellophane tape test, *J. of Structural Engineering*, vol.61A, pp.429-438, 2015 (in Japanese).
[3] M. Tai et al., Effect of CNN model and image size on the accuracy of identification of rust appearance grades for weathering steel plates, *AI Data Science*, vol.2, no.J2, pp.378-379, 2021 (in Japanese).
[4] R. Hasuike et al., An attempt to determine the degree of deterioration of corroded parts of weathering steel using convolutional neural network, *AI Data Science*, vol.2, no.J2, pp.813-820, 2021 (in Japanese).
[5] K. Arimura, N. Shigei, C. Morita et al., Rust appearance evaluation methods for weathering steel by using bagging CNN classifier and multiple patch images, *J. of Japan Society for Fuzzy Theory and Intelligent Informatics*, vol.34, no.2, pp.533-538, 2022 (in Japanese).
[6] I. J. Goodfellow et al., Generative adversarial networks, *Comm. of the ACM*, vol.63, no.11, pp.139-144, 2014.
[7] K. Tamura and T. Harada, Basic study on rust image generation of weathering steel by generative adversarial networks, *AI Data Science*, vol.2, no.J2, pp.792-800, 2021 (in Japanese).
[8] K. Nagatani, N. Shigei, C. Morita et al., Improvement of GAN-based training image generation for CNN-based rust evaluation of weathering steel, *The 12th International Conference on Computer and Communications Management*, 2024.
[9] *Rust Score Guideline*, Japan Bridge Construction Association, https://www.jasbc.or.jp/sabi/meyasu.html, Accessed on Feb. 24, 2023 (in Japanese).
[10] Z. Wang et al., Image quality assessment: From error visibility to structural similarity, *IEEE Transactions on Image Processing*, vol.13, no.4, pp.600-612, 2004.
[11] I. Gulrajani, F. Ahmed, M. Arjovsky et al., Improved training of Wasserstein GANs, *Proc. of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, vol.30, pp.5769-5779, 2017.
[12] M. Arjovsky, S. Chintala and L. Bottou, Wasserstein generative adversarial networks, *Proc. of the 34th International Conference on Machine Learning (ICML'17)*, vol.70, pp.214-223, 2017.
[13] T. Ojala, M. Pietikainen and D. Harwood, Performance evaluation of texture measures with classification based on Kullback discrimination of distributions, *Proc. of the 12th International Conference on Pattern Recognition*, Jerusalem, Israel, pp.582-585, 1994.
[14] A. K. Nain, *WGAN-GP overriding model.train_step*, https://keras.io/examples/generative/wgan_gp, Accessed on Oct. 30, 2023.
[15] C. Liao and M. Dong, ACWGAN: An auxiliary classifier Wasserstein GAN-based oversampling approach for multi-class imbalanced learning, *International Journal of Innovative Computing, Information and Control*, vol.18, no.3, pp.703-721, 2022.

Author Biography

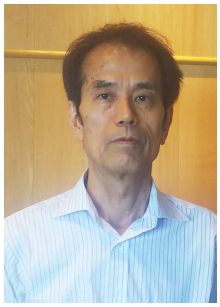


Kengo Nagatani received the bachelor's degree in Engineering from Kagoshima University, Kagoshima, Japan, in 2023. He is currently a master's student at the Graduate School of Science and Engineering, Kagoshima University. His research interests include deep learning and neural networks.

Noritaka Shigei received the Ph.D. degree from Kagoshima University, Japan, in 1997. He is currently a Professor at Kagoshima University, Japan. His research interests include machine learning, information networks, and digital communication systems.

Chihiro Morita received the Ph.D. degree from Kyushu University, Japan, in 1996. He is currently a Professor at University of Miyazaki, Japan. His research interests include structural analysis, maintenance, and soundness assessment of steel bridges.

Yoichi Ishizuka received a Ph.D. degree in Engineering from Kumamoto University in 1996. In 1996, he joined as an Assistant Professor with the Department of Electric and Electronic Engineering, Faculty of Engineering, Nagasaki University, Nagasaki, Japan, where he has been an Associate Professor since 2008 and Professor since 2021, respectively. He is involved in information and electronic circuits, including mixed-signal integrated circuits, power electronics, the Internet of Things, and artificial intelligence. He is a member of the Institute of Electrical and Electronics Engineers (IEEE) and the Institute of Electronics, Information and Communication Engineers (IEICE).



Hiromi Miyajima received the Ph.D. degree from Tohoku University, Japan, in 1979. He was a Professor at Kagoshima University from 1991 to 2016. He is currently an Emeritus Professor at Kagoshima University, Japan. His research interests include communication and network engineering and informatics/intelligent informatics.