IEICE TRANSACTIONS

on Information and Systems

VOL. E103-D NO. 1
JANUARY 2020

A PUBLICATION OF THE INFORMATION AND SYSTEMS SOCIETY
The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER Special Section on Enriched Multimedia—Application of Multimedia Technology and Its Security—

Neural Watermarking Method Including an Attack Simulator against Rotation and Compression Attacks

Ippei HAMAMOTO[†], *Nonmember and* **Masaki KAWAMURA**[†a)], *Senior Member*

SUMMARY We have developed a digital watermarking method that use neural networks to learn embedding and extraction processes that are robust against rotation and JPEG compression. The proposed neural networks consist of a stego-image generator, a watermark extractor, a stego-image discriminator, and an attack simulator. The attack simulator consists of a rotation layer and an additive noise layer, which simulate the rotation attack and the JPEG compression attack, respectively. The stego-image generator can learn embedding that is robust against these attacks, and also, the watermark extractor can extract watermarks without rotation synchronization. The quality of the stego-images can be improved by using the stego-image discriminator, which is a type of adversarial network. We evaluated the robustness of the watermarks and image quality and found that, using the proposed method, high-quality stego-images could be generated and the neural networks could be trained to embed and extract watermarks that are robust against rotation and JPEG compression attacks. We also showed that the robustness and image quality can be adjusted by changing the noise strength in the noise layer.

key words: *digital watermarking, neural networks, CNN, rotation, JPEG compression*

1. Introduction

Digital watermarking is used to prevent individuals from illegally using digital content, e.g., images, movies, and audio data, and also to identify unauthorized users [1]. Watermarking works by embedding secret information into contents imperceptibly [2]. In the case of image watermarking, the host image is called an original image and the marked image is called a stego-image. Methods that require the original image in order to extract the watermark are called non-blind type, and ones that do not require it are called blind type. In commercial use, the blind-type watermarking methods are required, since the original contents are typically unavailable.

Digital images can be easily modified by compression, clipping, scaling, and rotation. Once such image processing is applied, the location of watermarks may be missing or a part of the watermarks may be lost. Therefore, image processing is regarded as an attack against the watermarks. Geometric attacks, which include clipping, scaling, and rotating images, modify the coordinates of pixels, while non-geometric attacks, which include compression and additive noise, modify pixel values. It is necessary for a watermark to be extracted from a stego-image, even if the image has been attacked illegally or modified legally.

When a stego-image is distorted by geometric attacks, it is necessary to synchronize the marked position in order to extract watermarks. In the case of the non-blind type, since the methods can use the original images, they can match the position [3], [4]. In the case of the blind type, the feature detector by Scale Invariant Feature Transform (SIFT) [5] is effective to find the position. The SIFT feature detector can detect feature points that are robust against geometric transform from an attacked image. By using the SIFT feature points, the marked position can be synchronized. Many watermarking methods using SIFT [6]–[9] have been proposed. In these methods, marked regions are normalized to be equal in size. Both the SIFT feature points and the normalization make it easy to extract watermarks. However, they cannot synchronize the rotation angle of the image. The Fourier-Mellin transform domain is effective for rotation, scaling, and translation (RST) [10]. Tone and Hamada's method [11] uses the Harris-Affine detector and log-polar mapping as the invariant feature detector. While it can extract scaling and rotation invariant features, the log-polar mapping causes distortion of watermarks.

Methods using a marker or synchronization code [6], [7] and ones using moment of image [8], [9] have also been proposed. In Kawamura and Uchida's method [7], the marked regions are selected around the SIFT feature points and then markers and watermarks are embedded into the regions. In the first process of extraction, a possible marker is extracted by rotating and then its similarity is calculated. The angle that gives the highest similarity is regarded as the estimated angle. In the method of Li *et al.* [9], the moment of region, which is invariant against rotation, is calculated. However, these methods are sensitive to estimation errors or inaccuracy. Since the stego-images are usually distorted by attacks, the angle estimation often fails. Therefore, methods that have robust angle estimation or methods that eliminate the need of angle estimation are required.

Neural networks are a promising approach because they can be used for adjusting the embedding strength [12], [13] and for calculating the correlation between an original image and a watermark [14], [15]. In these methods, the neural networks are a part of the watermarking process. The robustness against attacks is usually acquired while training by attacked images provided in advance [16]. Recently, the embedding and extracting processes have been totally modeled by the networks. The neural networks can learn

Manuscript received March 26, 2019.
Manuscript revised August 8, 2019.
Manuscript publicized October 23, 2019.

[†]The authors are with Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Yamaguchi-shi, 753–8512 Japan.
a) E-mail: m.kawamura@m.ieice.org

DOI: 10.1587/transinf.2019MUP0007

the embedding and extracting processes by end-to-end training [17], [18]. Zhu *et al.* [18] proposed a method using neural networks that consist of a stego-image generator, an attack simulator, and a watermark extractor. A Gaussian filter, JPEG compression, and clipping are modeled in the attack simulator. Their method is robust against these noises because the network is trained by the simulator. It is a distinct advantage that the neural network includes the noise simulator inside itself.

Information hiding criteria (IHC) are criteria for watermarking methods provided by the committee of information hiding and its criteria for evaluation, IEICE [19]. The IHC defines the type of attacks, the image quality, and the bit error rate of watermarks to be accomplished. Our ultimate objective in our study is to develop a watermarking method that can fulfill IHC. As described above, a good number of watermarking methods, e.g., the SIFT-based ones [6]–[9], are robust against all geometric attacks except the rotation attack and accomplish good PSNR and BER. Therefore, our objective in the present study is to develop a method that is robust against the rotation attack. We propose a method using neural networks that consist of a stego-image generator, another attack simulator, and a watermark extractor. Our attack simulator consists of a rotation layer and an additive noise layer. As mentioned above, there is no blind-type method that is robust against rotation. Since the proposed neural networks can simulate the rotation attack in the rotation layer, the watermark extractor can output robust watermarks by training. That is, the proposed method requires no angle estimation. The proposed neural networks acquire robustness against the JPEG compression and the additive noise by means of the additive noise layer. Therefore, the proposed networks can embed and extract watermarks robustly.

In Sect. 2 of this paper, we briefly discuss related work using a neural network proposed by Zhu *et al.* [18]. Section 3 presents our neural networks. Computer simulations in Sect. 4 demonstrate that our networks can extract watermarks robustly. We conclude in Sect. 5 with a brief summary.

2. Related Work

Zhu *et al.* [18] proposed a method using neural networks that can learn robust watermarks. The networks consist of a stego-image generator G_{ψ}, an attack simulator (a noise layer), a watermark extractor E_{φ}, and a stego-image discriminator D_{γ}, where ψ, φ, and γ represent the parameters, e.g., the synaptic connections between the neurons and thresholds in these modules. The attack simulator is configured to model Gaussian blur, JPEG compression, and clipping. They showed that their neural networks can learn good stego-images and can extract robust watermarks.

2.1 Watermarking Model

Figure 1 shows the structure of the watermarking model proposed by Zhu *et al.* [18]. The notation $W \times H \times K$ represents the image width W, height H, and number of channels K. An L-bit watermark is embedded into $W \times H \times K$-size original images, where $K = 1$ for gray scale images and $K = 3$ for color images. In the middle part of the neural networks, the number of channels K can be larger than three.

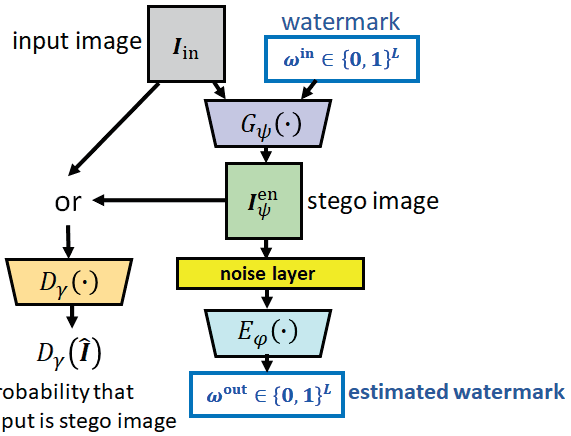


Fig. 1 Watermarking model by Zhu et al.

The stego-image generator G_{ψ} and the watermark extractor E_{φ} are convolutional neural networks (CNNs). In the stego-image generator, a $W \times H \times K$-size original image $\boldsymbol{I}^{\text{co}}$ and an L-bit watermark $\boldsymbol{\omega}^{\text{in}} = \left(\omega_1^{\text{in}}, \omega_2^{\text{in}}, \ldots, \omega_L^{\text{in}}\right)^{\top} \in \{0,1\}^L$ are fed into the network, and then, the network outputs a $W \times H \times K$-size stego-image,

$$\boldsymbol{I}_{\psi}^{\text{en}} = G_{\psi}\left(\boldsymbol{I}^{\text{co}}, \boldsymbol{\omega}^{\text{in}}; \boldsymbol{\psi}\right), \tag{1}$$

where $G_{\psi}(\cdot)$ stands for the stego-image generator as a function of the original image $\boldsymbol{I}^{\text{co}}$ and the watermark $\boldsymbol{\omega}^{\text{in}}$. $\boldsymbol{\psi}$ denotes all parameters of synaptic connections between neurons and thresholds in the generator. Next, the stego-image $\boldsymbol{I}_{\psi}^{\text{en}}$ is fed into the attack simulator. The image is transformed by the Gaussian blur, the JPEG compression, and clipping. The degraded image $\tilde{\boldsymbol{I}}_{\psi}^{\text{en}}$ is then fed into the watermark extractor E_{φ}. The extractor outputs an L-dimension vector,

$$\boldsymbol{E}^{\psi\varphi} = \left(E_1^{\psi\varphi}, E_2^{\psi\varphi}, \ldots, E_L^{\psi\varphi}\right)^{\top} \tag{2}$$

$$= E_{\varphi}\left(\tilde{\boldsymbol{I}}_{\psi}^{\text{en}}; \boldsymbol{\psi}, \boldsymbol{\varphi}\right), \tag{3}$$

where E_{φ} stands for the watermark extractor as a function of the image $\tilde{\boldsymbol{I}}_{\psi}^{\text{en}}$. $\boldsymbol{\varphi}$ denotes all parameters of synaptic connections between neurons and thresholds in the extractor. The L-bit estimated watermark $\boldsymbol{\omega}^{\text{out}} = \left(\omega_1^{\text{out}}, \omega_2^{\text{out}}, \ldots, \omega_L^{\text{out}}\right)^{\top}$ can be generated from the output $\boldsymbol{E}^{\psi\varphi}$.

The stego-image discriminator D_{γ} is a generative adversarial network (GAN). Either a stego-image $\boldsymbol{I}_{\psi}^{\text{en}}$ or an original image $\boldsymbol{I}^{\text{co}}$ is fed into the discriminator, which then outputs the probability $D_{\gamma}\left(\hat{\boldsymbol{I}}\right) \in (0,1)$ that the input $\hat{\boldsymbol{I}} \in \left\{\boldsymbol{I}^{\text{co}}, \boldsymbol{I}_{\psi}^{\text{en}}\right\}$ is the stego-image,

$$D_{\gamma}\left(\hat{\boldsymbol{I}}; \boldsymbol{\gamma}\right) = \begin{cases} 0, & \hat{\boldsymbol{I}} = \boldsymbol{I}^{\text{co}} \\ 1, & \hat{\boldsymbol{I}} = \boldsymbol{I}_{\psi}^{\text{en}} \end{cases}. \tag{4}$$

2.2 Training Embedding and Extraction Robust against Noise

In the training phase, several training images are generated from original images. The training images $\boldsymbol{I}^{\mathrm{co}}$ of the same size are clipped from the original image at random. The watermark $\boldsymbol{\omega}^{\mathrm{in}}$ is also generated in a random manner. The stego-image generator $G_{\boldsymbol{\psi}}$, the watermark extractor $E_{\boldsymbol{\varphi}}$, and the stego-image discriminator $D_{\boldsymbol{\gamma}}$ are trained by turns.

2.2.1 Training of the Generator and the Extractor

Zhu *et al.* define the error function $R^{\omega}_{\psi\varphi}$ for estimated watermarks as the mean square error (MSE) between the true watermark $\boldsymbol{\omega}^{\mathrm{in}}$ and the output $\boldsymbol{E}^{\psi\varphi}$ of the watermark extractor $E_{\boldsymbol{\varphi}}$, that is,

$$R^{\omega}_{\psi\varphi}\left(\boldsymbol{\omega}^{\mathrm{in}}, \boldsymbol{E}^{\psi\varphi}; \boldsymbol{\psi}, \boldsymbol{\varphi}\right) = \frac{1}{L}\|\boldsymbol{\omega}^{\mathrm{in}} - \boldsymbol{E}^{\psi\varphi}\|_2^2, \tag{5}$$

where $\|\cdot\|_2$ stands for 2-norm. It is better for the stego-image $\boldsymbol{I}^{\mathrm{en}}_{\psi}$ to be similar to the original image $\boldsymbol{I}^{\mathrm{co}}$. Therefore, they define the error function R^{I}_{ψ} as the MSE between the stego-image and the original image, that is,

$$R^{I}_{\psi}\left(\boldsymbol{I}^{\mathrm{co}}, \boldsymbol{I}^{\mathrm{en}}_{\psi}; \boldsymbol{\psi}\right) = \frac{1}{WHK}\|\boldsymbol{I}^{\mathrm{co}} - \boldsymbol{I}^{\mathrm{en}}_{\psi}\|_2^2. \tag{6}$$

Next, it is important that watermarks not be detected in stego-images. Both the watermark extractor $E_{\boldsymbol{\varphi}}$ and the stego-image discriminator $D_{\boldsymbol{\gamma}}$ are trained so as not to discriminate between the stego-images and the original images. Therefore, they define the error function $R^{G}_{\psi\gamma}$ as unnaturalness of stego-images, which is given by

$$R^{G}_{\psi\gamma}\left(\boldsymbol{I}^{\mathrm{en}}_{\psi}; \boldsymbol{\psi}, \boldsymbol{\gamma}\right) = -\log\left(1 - D_{\gamma}\left(\boldsymbol{I}^{\mathrm{en}}_{\psi}\right)\right), \tag{7}$$

where $D_{\gamma}\left(\boldsymbol{I}^{\mathrm{en}}_{\psi}\right)$ is the output of the discriminator $D_{\boldsymbol{\gamma}}$ when the stego-image $\boldsymbol{I}^{\mathrm{en}}_{\psi}$ is fed into it.

Finally, the generator $G_{\boldsymbol{\psi}}$ and the extractor $E_{\boldsymbol{\varphi}}$ are trained by minimizing the expectation of the weighted sum of the errors R^{I}_{ψ}, $R^{G}_{\psi\gamma}$, and $R^{\omega}_{\psi\varphi}$; that is, the parameters $\boldsymbol{\psi}$ and $\boldsymbol{\varphi}$, i.e., the values of synaptic connections and thresholds in the networks, are calculated by

$$\min_{\boldsymbol{\psi},\boldsymbol{\varphi}} \mathbb{E}_{\boldsymbol{I}^{\mathrm{co}},\boldsymbol{\omega}^{\mathrm{in}}}\left[R^{\omega}_{\psi\varphi} + \lambda^{I}R^{I}_{\psi} + \lambda^{G}R^{G}_{\psi\gamma}\right], \tag{8}$$

where λ^{I} and λ^{G} are weight parameters. $\mathbb{E}_{\boldsymbol{I}^{\mathrm{co}},\boldsymbol{\omega}^{\mathrm{in}}}$ represents the expectation of original images $\boldsymbol{I}^{\mathrm{co}}$ and watermarks $\boldsymbol{\omega}^{\mathrm{in}}$.

2.2.2 Training of the Discriminator

The stego-image discriminator $D_{\boldsymbol{\gamma}}$ attempts to distinguish the stego-images from the original images. It outputs the probability, $D_{\gamma}(\hat{\boldsymbol{I}}) \in (0, 1)$, that an input image $\hat{\boldsymbol{I}}$ is a stego-image. Therefore, they define the error function $R^{D}_{\psi\gamma}$ for decision errors as the cross-entropy given by

$$\begin{aligned} &R^{D}_{\psi\gamma}\left(\boldsymbol{I}^{\mathrm{co}}, \boldsymbol{I}^{\mathrm{en}}_{\psi}; \boldsymbol{\psi}, \boldsymbol{\gamma}\right) \\ &= -\log D_{\gamma}\left(\boldsymbol{I}^{\mathrm{en}}_{\psi}\right) - \log\left(1 - D_{\gamma}\left(\boldsymbol{I}^{\mathrm{co}}\right)\right). \end{aligned} \tag{9}$$

The discriminator is trained by minimizing the expectation of the cross-entropy. That is, the parameter $\boldsymbol{\gamma}$ is calculated by

$$\min_{\boldsymbol{\gamma}} \mathbb{E}_{\boldsymbol{I}^{\mathrm{co}},\boldsymbol{\omega}^{\mathrm{in}}}\left[\lambda^{D}R^{D}_{\psi\gamma}\left(\boldsymbol{I}^{\mathrm{co}}, \boldsymbol{I}^{\mathrm{en}}_{\psi}; \boldsymbol{\psi}, \boldsymbol{\gamma}\right)\right], \tag{10}$$

where λ^{D} is the weight parameter. Zhu *et al.* showed that the quality of stego-images could be improved by using the discriminator.

3. Proposed Method

We propose a blind-type watermarking method using neural networks that acquire the ability to embed and extract watermarks robust against rotation attack and JPEG compression. The same as the method of Zhu *et al.* [18], the proposed neural networks consist of a stego-image generator $G_{\boldsymbol{\psi}}$, an attack simulator, a watermark extractor $E_{\boldsymbol{\varphi}}$, and a stego-image discriminator $D_{\boldsymbol{\gamma}}$. However, our attack simulator differs in that we introduce a rotation layer and an additive noise layer instead of the noise layer. The rotation layer simulates the rotation of images, so the output of the layer is rotated images. In the additive noise layer, additive white Gaussian noise (AWGN) is added to the rotated images. Since the watermark extractor receives images that have been distorted and rotated, the network is able to output robust watermarks by training.

3.1 Watermarking Model

Figure 2 shows the proposed watermarking model. The regions of $W_0 \times H_0$-pixels are clipped from the original images. Since the regions are fed into the neural networks to train, we call these regions teacher images. However, $W \times H$-pixel input images are fed into the neural networks to train, where $W_0 \geq \sqrt{2}W$ and $H_0 \geq \sqrt{2}H$ (e.g. $W_0 = H_0 = 96$ and $W = H = 64$). When a $W \times H$-pixel input image is

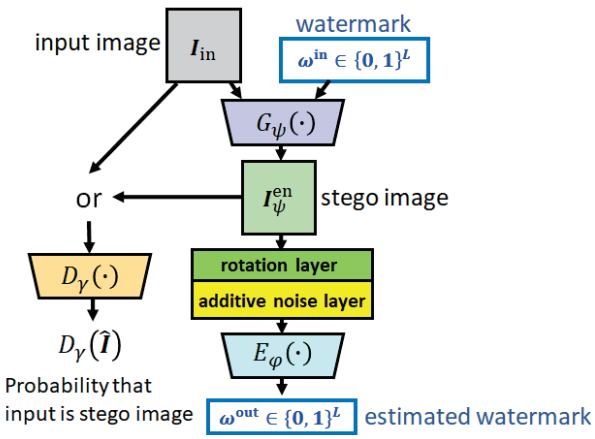


Fig. 2 Proposed watermarking model.

rotated, we consider a bounding rectangle of the rotated image. There are four triangular margins around the rotated image. Therefore, a teacher image is at most $\sqrt{2}$ times larger than the input images. The margins can be interpolated by the input image. The pixel value for each channel of an input image is normalized to the range of $[0, 1]$. An L-bit watermark is embedded into each region. Each region is the input image $\boldsymbol{I}^{\text{in}}$ for the stego-image generator $G_{\boldsymbol{\psi}}$. The generator outputs a stego-image $\boldsymbol{I}_{\psi}^{\text{en}}$. The size of the input image and the stego-image is $W \times H \times 1$, $(K = 1)$. That is, the watermark is embedded into the luminosity value of the image.

The rotation layer simulates the rotation attack. The stego-image $\boldsymbol{I}_{\psi}^{\text{en}}$ is rotated at θ radian around the center point of the stego-image. The angle θ is selected in a random manner. In the additive noise layer, AWGN is added to the rotated stego-image $\boldsymbol{I}_{\psi}^{\text{rot}}$ and a distorted stego-image $\tilde{\boldsymbol{I}}_{\psi}^{\text{rot}}$ is output. It is fed into the watermark extractor $E_{\boldsymbol{\varphi}}$. In our method, the output from the extractor, $E_i^{\psi\varphi}$, is regarded as the probability that the i-th watermark bit is 1. This is a different point from the method of Zhu *et al.* [18], whose output is the value of the watermark bit, 0 or 1.

3.2 Structure of Proposed Neural Networks

3.2.1 Convolution Layer

In the stego-image generator $G_{\boldsymbol{\psi}}$, the filter size of a convolution layer is 3×3 pixels, stride is $S = 1$ pixel, and padding is $P = 1$ pixel. In the watermark extractor E_{φ} and the stego-image discriminator D_{γ}, the filter size is 4×4 pixels, stride is $S = 2$ pixel, and padding is $P = 1$ pixel. We apply batch normalization (BN) [20] to the output of each layer, and use the leaky ReLU function as the activation function, unless otherwise stated.

3.2.2 Structure of the Stego-Image Generator G_{ψ}

Figure 3 shows the structure of the stego-image generator G_{ψ}. A $W \times H \times 1$-size input image $\boldsymbol{I}^{\text{in}}$ and an L-bit watermark $\boldsymbol{\omega}^{\text{in}}$ are fed into the generator. The $W \times H \times 64$-size feature map $\boldsymbol{I}^{\text{F}}$ can be obtained from the input image $\boldsymbol{I}^{\text{in}}$ by applying four convolution layers. The number of channels in each layer is 64. At the same time, the L-bit watermark $\boldsymbol{\omega}^{\text{in}}$ is fed into a layered network and is converted to a $W \times H \times L$-size feature map. These feature maps from the convolution layers (64 channels), the watermark (L channels), and the input image itself (1 channel) are joined together, thereby generating a $W \times H \times (64 + L + 1)$-size feature map. After that, the feature map is fed into two convolution layers. The first layer is the default convolution layer with the number of channels $K = 64$. The second layer is a different version with the number of channels $K = 1$, filter size 1×1, stride $S = 1$, and padding $P = 0$. The activation function is the sigmoid function and BN is not applied. Finally, a stego-image $\boldsymbol{I}_{\psi}^{\text{en}}$ is generated.

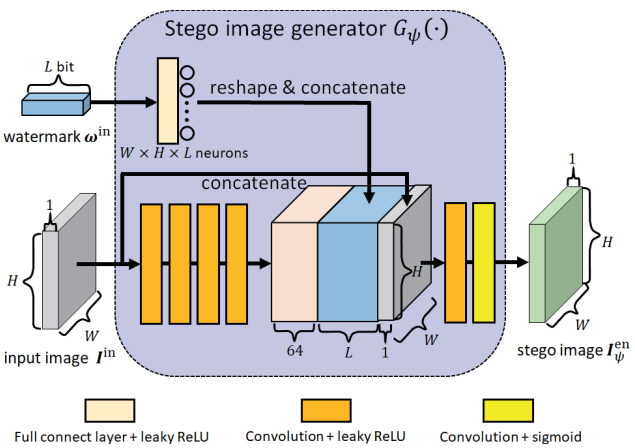


Fig. 3 Structure of the stego-image generator G_{ψ}.

3.2.3 Structure of the Watermark Extractor E_{φ}

Figure 4 shows the structure of the watermark extractor E_{φ}. The watermark extractor E_{φ} receives the distorted, rotated stego-image $\tilde{\boldsymbol{I}}_{\psi}^{\text{rot}}$ from the attack simulator. The feature map is generated by four convolution layers from the distorted image $\tilde{\boldsymbol{I}}_{\psi}^{\text{rot}}$. Each layer has 64 channels. The feature map is fed into two fully connected layers (FCL). The first layer has 128 output neurons, and their activation function is the leaky ReLU function. The second layer has L output neurons, and their activation function is the sigmoid function. Finally, the L-dimensional output $\boldsymbol{E}^{\psi\varphi}$ of the extractor is generated.

The output $E_i^{\psi\varphi}$ represents the probability that the i-th watermark bit is one. Therefore, the i-th estimated watermark bit ω_i^{out} is given by

$$\omega_i^{\text{out}} = \begin{cases} 0, & E_i^{\psi\varphi} \leq 0.5 \\ 1, & E_i^{\psi\varphi} > 0.5 \end{cases} . \tag{11}$$

3.2.4 Structure of the Stego-Image Discriminator D_{γ}

The stego-image discriminator $D_{\boldsymbol{\gamma}}$ is a generative adversarial network (GAN). The input image $\hat{\boldsymbol{I}}$ to the stego-image discriminator D_{γ} is either a stego-image $\boldsymbol{I}_{\psi}^{\text{en}}$ or an original image $\boldsymbol{I}^{\text{in}}$. The image $\hat{\boldsymbol{I}}$ is fed into four convolution layers

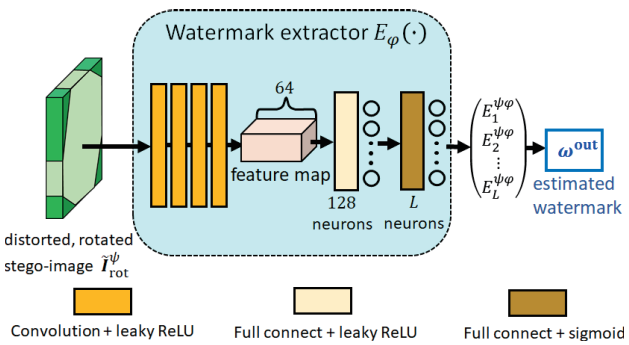


Fig. 4 Structure of the watermark extractor E_{φ}.

with $K = 8$, and then a feature map is generated. The map is fed into an FCL of one output neuron with the sigmoid function. The output $D_{\boldsymbol{\gamma}}(\hat{\boldsymbol{I}}) \in (0, 1)$ represents the probability that the input image $\hat{\boldsymbol{I}}$ is a stego-image.

3.3 Structure of the Attack Simulator

The attack simulator consists of a rotation layer and an additive noise layer. The rotation layer simulates the rotation of the image from the stego-image generator. The rotation angle is $0 \le \theta < 2\pi$ radian. Let $I^{\mathrm{en}}_{\boldsymbol{\psi}}(i, j)$, $i = 0, 1, \ldots, W-1$, $j = 0, 1, \ldots, H-1$ be a pixel value at a lattice point (i, j) of the stego-image $\boldsymbol{I}^{\mathrm{en}}_{\boldsymbol{\psi}}$, and let $I^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y)$, $r_x = 0, 1, \ldots, W-1$, $r_y = 0, 1, \ldots, H-1$ be a pixel value at a lattice point (r_x, r_y) of the rotated stego-image $\boldsymbol{I}^{\mathrm{rot}}_{\boldsymbol{\psi}}$. As shown in Fig. 5, the coordinate (Q_x, Q_y) is the point that is inversely rotated at θ radian from point $\boldsymbol{R}(r_x, r_y)$ around the center point $\boldsymbol{c}(c_x, c_y)$ of the stego-image. That is, the coordinate $\boldsymbol{Q}(Q_x, Q_y)$ is given by

$$\boldsymbol{Q} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}^{-1} (\boldsymbol{R} - \boldsymbol{c}) + \boldsymbol{c}. \tag{12}$$

Since the values of Q_x and Q_y are real numbers, the pixel value of $I^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y)$ is calculated from four neighboring lattice points around (Q_x, Q_y). Let the coordinate (i, j) of the upper-left point be

$$i = \lfloor Q_x \rfloor, \tag{13}$$
$$j = \lfloor Q_y \rfloor. \tag{14}$$

By using linear interpolation, the output $I^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y)$ is given by

$$\begin{aligned} &I^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y) \\ &= \{(i+1) - Q_x\}\left\{(j+1) - Q_y\right\} I^{\mathrm{en}}_{\boldsymbol{\psi}}(i, j) \\ &\quad + \{Q_x - i\}\left\{(j+1) - Q_y\right\} I^{\mathrm{en}}_{\boldsymbol{\psi}}(i+1, j) \\ &\quad + \{(i+1) - Q_x\}\left\{Q_y - j\right\} I^{\mathrm{en}}_{\boldsymbol{\psi}}(i, j+1) \\ &\quad + \{Q_x - i\}\left\{Q_y - j\right\} I^{\mathrm{en}}_{\boldsymbol{\psi}}(i+1, j+1). \end{aligned} \tag{15}$$

As mentioned in Sect. 3.1, there are four margins around the rotated image. These margins can be interpolated by the input image. In this way, the rotated stego-image $\boldsymbol{I}^{\mathrm{rot}}_{\boldsymbol{\psi}}$ is generated.

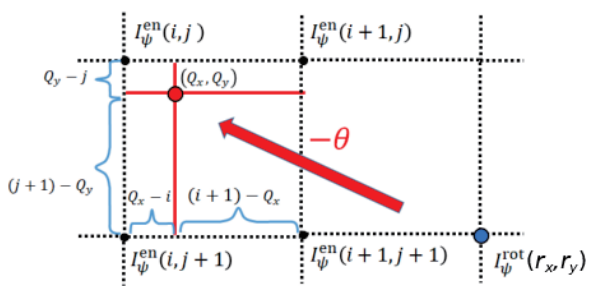


Fig. 5 Corresponding four neighboring lattice points.

In the additive noise layer, a noise is added to each pixel of the rotated stego-image $I^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y)$ independently. The noise ξ_{xy} is distributed according to the AWGN with average 0 and variance σ^2. Therefore, the output of the attack simulator, $\tilde{I}^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y)$, is given by

$$\tilde{I}^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y) = I^{\mathrm{rot}}_{\boldsymbol{\psi}}(r_x, r_y) + \xi_{xy}. \tag{16}$$

3.4 Training against Rotation and Additive Noise

The error function $R^{D}_{\boldsymbol{\psi\gamma}}$ for the stego-image discriminator $D_{\boldsymbol{\gamma}}$ is given by (9), the same as the method of Zhu *et al.* [18]. The parameter $\boldsymbol{\gamma}$, i.e., the synaptic connections between neurons and the thresholds in the discriminator, is given by minimizing (10). Note that here, we change the characteristic of the output in the watermark extractor $E_{\boldsymbol{\varphi}}$. Zhu *et al.* regard the output as the value of the watermark. In their case, it is reasonable to use the MSE. However, the output takes a real number in the range of $[0, 1]$ by the sigmoid function, so we regard the output as the probability that the watermark bit is 1. In this case, it is reasonable to use the cross-entropy. While the error function $R^{\omega}_{\boldsymbol{\psi\varphi}}$ in the method of Zhu *et al.* [18] is given by (5), the error function $\tilde{R}^{\omega}_{\boldsymbol{\psi\varphi}}$ in the proposed method is given by

$$\begin{aligned} &\tilde{R}^{\omega}_{\boldsymbol{\psi\varphi}}\left(\boldsymbol{\omega}^{\mathrm{in}}, E^{\boldsymbol{\psi\varphi}}; \boldsymbol{\psi}, \boldsymbol{\varphi}\right) \\ &= -\frac{1}{L}\sum_{i=1}^{L}\left\{\omega^{\mathrm{in}}_i \log E^{\boldsymbol{\psi\varphi}}_i + \left(1 - \omega^{\mathrm{in}}_i\right)\log\left(1 - E^{\boldsymbol{\psi\varphi}}_i\right)\right\}. \end{aligned} \tag{17}$$

Even though these concepts are slightly different, this difference does not affect the robustness of watermarks nor the quality of images.

The stego-image generator $G_{\boldsymbol{\psi}}$ and the watermark extractor $E_{\boldsymbol{\varphi}}$ are trained by minimizing the expectation of the weighted sum of the errors $R^{I}_{\boldsymbol{\psi}}$ of (6), $R^{G}_{\boldsymbol{\psi\gamma}}$ of (7), and $\tilde{R}^{\omega}_{\boldsymbol{\psi\varphi}}$. The parameters $\boldsymbol{\psi}$ and $\boldsymbol{\varphi}$ are calculated by

$$\min_{\boldsymbol{\psi},\boldsymbol{\varphi}} \mathbb{E}_{\boldsymbol{I}_{\mathrm{in}},\boldsymbol{\omega}_{\mathrm{in}}}\left[R^{I}_{\boldsymbol{\psi}} + \lambda^{\omega}\tilde{R}^{\omega}_{\boldsymbol{\psi\varphi}} + \lambda^{G}R^{G}_{\boldsymbol{\psi\gamma}}\right], \tag{18}$$

where λ^{ω} and λ^{G} are weight parameters.

4. Computer Simulations

In this section, we evaluate the effectiveness of the proposed attack simulator by comparing it with the method of Zhu *et al.* [18]. First, we compare the performances of the two methods by bit error rate (BER) and image quality in a case without rotation attack. Next, we calculate the suitable parameters of noise strength σ in the additive noise layer and the weight parameter λ^{ω} of the error function $\tilde{R}^{\theta\varphi}_{\omega}$ in a case where both the rotation and additive noise attacks are processed.

4.1 Experimental Conditions

Figure 6 shows the IHC standard images, which are provided by Information Hiding and its Criteria for evaluation,

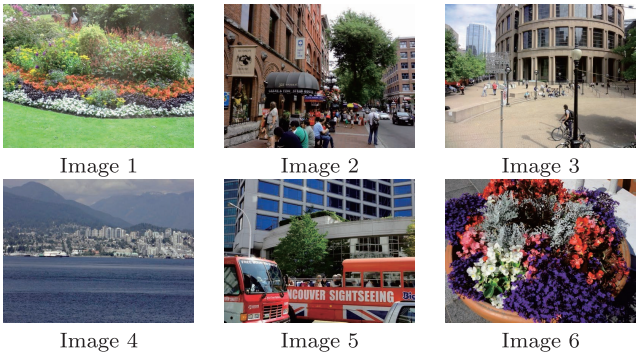


Fig. 6 IHC standard images.

IEICE [19]. The size of these images is 4608×3456 pixels. Of the six images, one is used for testing and the others are used for training of the neural networks as the teacher images. Specifically, the proposed neural networks are trained by the teacher images, which are clipped from the original images as described in Sect. 3.1, and then the networks are evaluated against the test image.

4.1.1 Training conditions

The size of input images $\boldsymbol{I}^{\text{in}}$ is $H = W = 64$, and the size of the training images is $H_0 = W_0 = 96$, as mentioned in Sect. 3.1. The 1024 training images (regions) are randomly clipped from a teacher image. Since we use five teacher images, there are 5120 images for training. The neural networks are trained to embed an 8-bit watermark ($L = 8$) into the central part of a teacher image.

The weight parameters λ^{ω}, λ^{G}, and λ^{D} for the error functions $\tilde{R}_{\omega}^{\theta\varphi}$, $R_{G}^{\theta\gamma}$, and $R_{D}^{\theta\gamma}$ are given by $\lambda^{\omega} \in \{0.001, 0.005, 0.01, 0.05, 0.1\}$ and $\lambda^{G} = \lambda^{D} = 0.0001$. The mini-batch size is 64 and the number of epochs is 300. The training algorithm is Adam [21], where the learning rate is $\alpha = 0.0004$ and the other parameters are Adam's default. The networks are implemented on TensorFlow [22]. Ten trials are performed by changing the initial condition of synaptic connections in the neural networks. The results show the average values.

4.1.2 Performance index

The image quality of a stego-image is evaluated by the peak signal-to-noise ratio (PSNR), which is given by

$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2}{\text{MSE}} \right) \text{ [dB]}, \tag{19}$$

$$\text{MSE} = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(I_{ij}^{\text{in}} - I_{ij}^{\text{en}} \right)^2, \tag{20}$$

where I^{in} and I^{en} are an input image and a stego-image, respectively. The stego-image is generated from the trained stego-image generator by using an 8-bit watermark and a 64×64-pixel test image. The embedding rate is $\frac{8}{64 \times 64} = 0.00195$ bits per pixel (bpp).

Table 1 PSNRs and BERs without rotation layer.

noise strength σ	0.0	0.02	0.04	0.06
PSNR [dB]	41.53	38.28	38.23	36.58
BER	0.43	0.33	0.20	0.08

Table 2 BERs and PSNRs for the proposed and Zhu et al.'s methods.

	method of [18]			Proposed method ($\sigma = 0.06$)
Channel	Y	U	V	Y
PSNR [dB]	30.09	35.33	36.27	36.58
BER ($Q = 50$)	0.15			0.08

The robustness of watermarks is evaluated by the bit error rate (BER), which is given by

$$\text{BER} = \frac{1}{L} \sum_{i=1}^{L} \omega_i^{\text{in}} \oplus \omega_i^{\text{out}}, \tag{21}$$

where $\oplus$ stands for the operation of exclusive OR. ω_i^{in} and $\boldsymbol{\omega}^{\text{out}}$ are the true watermark and a watermark extracted by the watermark extractor, respectively.

4.2 Comparison with the Method of Zhu et al.

We compare the proposed method and the method of Zhu *et al.* in terms of robustness against JPEG compression and image quality. The proposed networks are trained on 10,000 images from the COCO [23] training set, the same as [18]. A 1000-image test set is utilized for testing. Since there is no result for a rotation attack in [18], no training is performed in the rotation layer in this section. The weight parameter of the error function $\tilde{R}_{\omega}^{\theta\varphi}$ is $\lambda^{\omega} = 0.01$. Noise strength in the additive noise layer is $\sigma = 0.0, 0.02, 0.04, 0.06$ while training. That is, four different neural networks are generated by different noise strengths σ. After training the networks, stego-images are generated by the trained stego-image generator G_{ψ}. The second row in Table 1 shows the average PSNRs of the generated stego-images. These PSNRs were calculated inside a marked region of each image. We also investigated the robustness of watermarks against JPEG compression. The stego-images were compressed with the Q-value of $Q = 50$. The compressed stego-images were fed into the watermark extractors E_{φ}. The third row of Table 1 shows the average BERs. As shown, the robust embedding and extraction can learn by using the additive noise layer with $\sigma = 0.06$. When the noise strength σ is large, the stego-images are distorted, but the robustness against JPEG compression is improved.

Table 2 shows the results of Zhu *et al.'s* method and our own, where the noise strength is $\sigma = 0.06$. In the case of Zhu *et al.*, a 30-bit watermark is embedded into a 128×128-pixel YUV-image. The embedding rate is $\frac{30}{128 \times 128} = 0.00183$ bpp, which is smaller than our rate of 0.00195 bpp. These results indicate that the proposed method has good image quality comparable to that of Zhu *et al.'s* and also that it is more robust against JPEG compression than theirs.

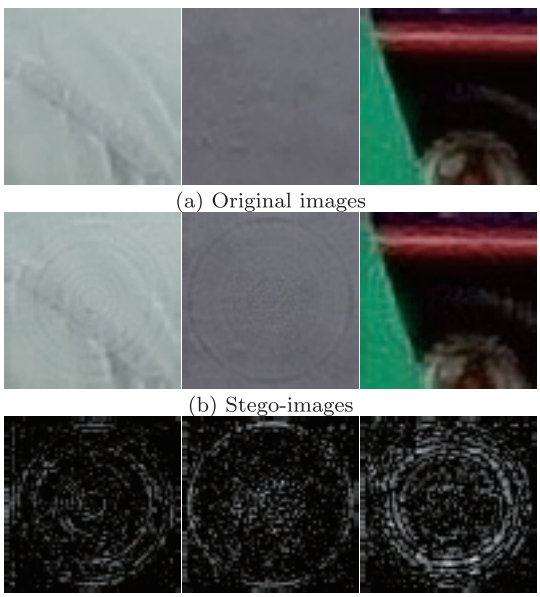
4.3 Effect of the Rotation Layer

We next examine the effect of the rotation layer in the attack simulator. That is, the rotation layer is also trained. The networks are trained by using the IHC standard images as described in Sect. 4.1. The weight parameter λ^ω for the error function $\tilde{R}_\omega^{\theta\varphi}$ is $\lambda^\omega = 0.01$. The rotation angle in the rotation layer is $0 \le \theta \le 2\pi$ radian, and the noise strength in the additive noise layer is $\sigma = 0.0, 0.02, 0.04$ while training. After training the networks, we evaluate the image quality and the robustness. Table 3 lists the average PSNRs of stego-images generated from different stego-image generators G_ψ trained with $\sigma = 0.0, 0.02, 0.04$. All PSNRs were over 35 dB. Even if the rotation layer simulates a rotation attack while training, the proposed method can learn high-quality images. Figure 7 shows (a) the original images, (b) stego-images, and (c) difference images, where the noise strength is $\sigma = 0.04$. Note that we examine a large noise case here in order to check the effect of the rotation layer. The difference images are generated from the difference between the original images and the stego-images. As shown, the brightness is ten times as large as the absolute value of the difference. Due to the rotation layer, a circular artifact appears in the stego-images.

Next, we evaluate the BERs for a rotation attack. The stego-images are rotated by an attacker and then the rotated images are fed into the trained watermark extractor E_φ. Figure 8 shows the average BERs in the cases where the attack angles θ are $0°, 10°, 20°, \ldots, 360°$. The abscissa and ordinate are the rotation angle θ and BER, respectively. The line with $\sigma = 0.0$ denotes the average BER by using only the rotation layer, i.e., no additive noise layer. The lines with $\sigma = 0.02, 0.04$ denote the average BERs by trained watermark extractors with $\sigma = 0.02, 0.04$. We can observe peaks of BER at the angles of $(45 + 90n)°, n = 0, 1, 2, 3$. At these angles, large interpolation occurred in attacked images due to rotation by the attacker. This caused image distortion. Moreover, we can see that the watermark extractor trained with the large noise strength $\sigma = 0.04$ has robustness against the rotation attack, since the average BERs are less than 0.001. In the following sections, we use the watermark extractor with $\sigma = 0.04$ to evaluate the proposed method.

Table 3 Image quality for noise strength.

noise strength σ	0.0	0.02	0.04
PSNR [dB]	39.0	37.6	35.9



(a) Original images

(b) Stego-images

(c) Difference images (10×)

Fig. 7 Examples of original images, stego-images and difference images ($\sigma = 0.04$).

4.4 Determination of Weight Parameter λ^ω

Let us determine the weight parameter λ^ω for the error function $R_\omega^{\theta\varphi}$ in the proposed method. We want to select a parameter value that best meets the requirement of both a small BER and a large PSNR. The noise strength is $\sigma = 0.04$ while training. Figure 9 shows the average BERs. The ab-

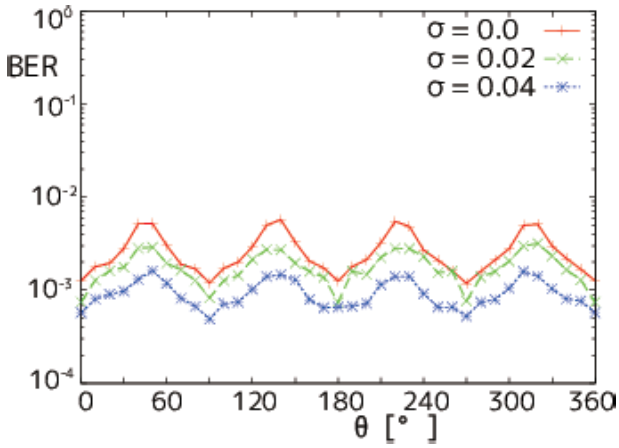


Fig. 8 Rotation angle θ vs. BER for different noise strengths.

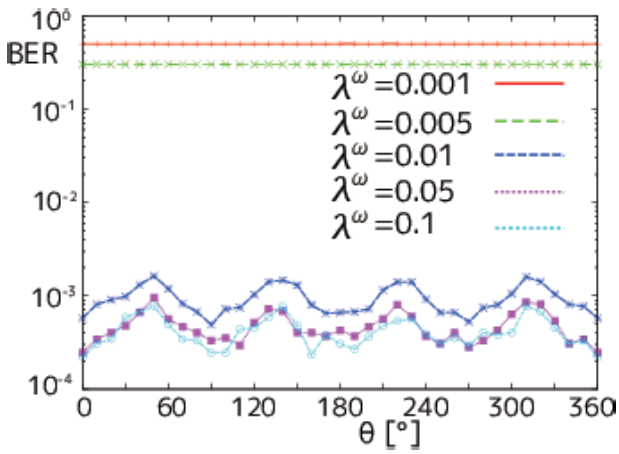


Fig. 9 Rotation angle θ vs. BER for weight parameter λ^ω.

Table 4 Weight parameter λ^{ω} and average of PSNR.

weight parameter λ^{ω}	0.001	0.005	0.01	0.05	0.1
PSNR [dB]	42.6	40.1	35.9	33.3	31.5

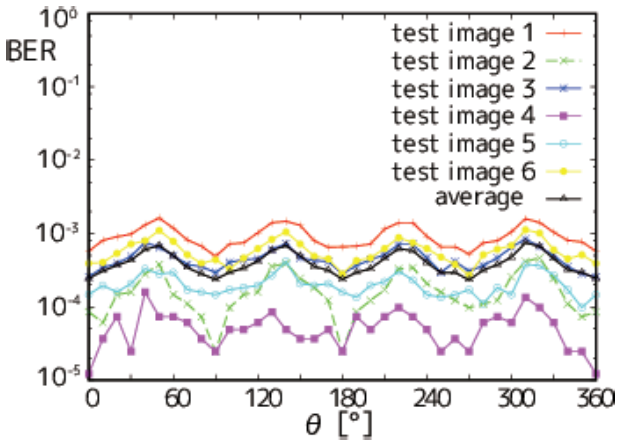


Fig. 10 BER vs. attack angle θ.

Table 5 PSNR for stego-images.

test image	1	2	3	4	5	6	Ave.
PSNR [dB]	35.9	35.7	36.5	36.8	37.1	35.6	36.3

scissa and ordinate are the attack angle θ and BER. The weight parameter is $\lambda^{\omega} = 0.001, 0.005, 0.01, 0.05, 0.1$. In the cases of small $\lambda^{\omega} = 0.001, 0.005$, the networks cannot embed and extract watermarks. In contrast, in the cases of $\lambda^{\omega} \geq 0.01$, we found that watermarks can be extracted with low BERs. Therefore, the proposed method has robustness against the rotation attack. The average PSNRs are listed in Table 4. The larger the parameter λ^{ω} is, the worse the image quality is. Therefore, we use the weight parameter $\lambda^{\omega} = 0.01$ in the following sections.

4.5 Performance Evaluation against Attack

We selected the noise strength $\sigma = 0.04$ and the weight parameter $\lambda^{\omega} = 0.01$ as the suitable parameters of the proposed method. In this section, we evaluate the robustness against rotation attack and JPEG compression. Figure 10 shows the average BERs vs. attack angle θ for test images 1 to 6 in Fig. 6. That is, one of the images is used for testing, and the other five are used for training. Since the average BER over six images is under 0.001, the proposed method has robustness against the rotation attack. Figure 11 shows the average BERs vs. Q-value of JPEG compression. When the Q-value is over 50, the average BER is under 0.01. Therefore, the proposed method has robustness against the JPEG compression. Table 5 lists the PSNRs for stego-images. We can see that all PSNRs are over 35 dB. As a result, the proposed method can produce high-quality stego-images with watermarks robust against rotation attack and JPEG compression, provided the suitable parameters are chosen.

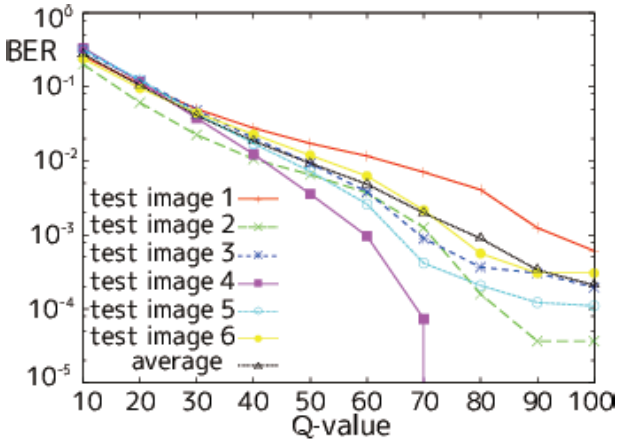


Fig. 11 BER vs. Q-value.

5. Conclusion

Among watermarking methods, there are many that can resist geometric attacks. However, there is no effective method that can resist a rotation attack while simultaneously fulfilling the IHC. Therefore, a method that is robust against rotation attack is required. We focused on neural networks that include an attack simulator to design attacks [18] and proposed adding a rotation layer and an additive noise layer to the attack simulator in order to resist the rotation attack and the JPEG compression. The networks also include a stego-image generator and a watermark extractor. Due to the attack simulator, both the generator and the extractor could learn robust embedding and extraction of watermarks. We demonstrated through simulations that the proposed method is robust against not only the JPEG compression but also the rotation attack. The robustness and image quality could be controlled by the noise strength in the additive noise layer and by the weight parameters. We determined the suitable parameters by computer simulations and showed that, by using these parameters, the proposed method could achieve low BERs and a high-quality image. We conclude that the proposed method can be utilized in prospective methods against any geometric attacks including the rotation attack by combining it with the SIFT-based watermarking methods [6]–[9] or the method of Zhu *et al.* [18]. This extension is a future work.

The neural watermarking scheme that includes the attack simulator is a promising approach. Both Zhu *et al.'s* [18] method and our own have demonstrated robust watermarking, so it may be possible to replace the attack simulator with another one that includes different attacks.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP16K00156. The computation was carried out using PC clusters at Yamaguchi University and the super computer facilities at Research Institute for Information Technology, Kyushu University.

References

[1] F.A.P. Petitcolas, R.J. Anderson, and M.G. Kuhn, "Information hiding—A survey," Proc. IEEE, vol.87, no.7, pp.1062–1078, 1999.
[2] K. Iwamura, M. Kawamura, M. Kuribayashi, M. Iwata, H. Kang, S. Gohshi, and A. Nishimura, "Information hiding and its criteria for evaluation," IEICE Trans. Inf. & Syst., vol.E100-D, no.1, pp.2–12, Jan. 2017.
[3] H. Luo, X. Sun, H. Yang, and Z. Xia, "A robust image watermarking based on image restoration using SIFT," Radio Engineering, vol.20, no.2, pp.525–532, 2011.
[4] X. Zhou H. Zhang, and C. Wang, "A robust image watermarking technique based on DWT, APDCBT, and SVD," Symmetry, vol.10, no.3, 77, 2018.
[5] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vis., vol.60, no.2, pp.91–110, 2004.
[6] H.-Y. Lee, H. Kim, and H.-K. Lee, "Robust image watermarking using local invariant features," Optical Engineering, vol.45, no.3, 037002, 2006.
[7] M. Kawamura and K. Uchida, "SIFT feature-based watermarking method aimed at achieving IHC ver.5," Advances in Intelligent Information Hiding and Multimedia Signal Processing, IIH-MSP 2017, Smart Innovation, Systems and Technologies, vol.81, pp.381–389, Springer, Cham, 2017.
[8] P. Dong, J.G. Brankov, N.P. Galatsanos, Y. Yang, and F. Davoine, "Digital watermarking robust to geometric distortions," IEEE Trans. Image Process., vol.14, no.12, pp.2140–2150, 2005.
[9] L. Li, X. Yuan, Z. Lu, and J.-S. Pan, "Rotation invariant watermark embedding based on scale-adapted characteristic regions," Information Sciences, vol.180, no.15, pp.2875–2888, 2010.
[10] J. O'Ruanaidh and T. Pun, "Rotation, scale and translation invariant digital image watermarking," Int. Conf. Image Processing, vol.1, p.536, IEEE Computer Society, 1997.
[11] M. Tone and N. Hamada, "Scale and rotation invariant digital image watermarking method," IEICE Trans. Inf. & Syst. (Japanese Edition), vol.J88-D1, no.12, pp.1750–1759, Dec. 2005.
[12] L. Mao, Y.-Y. Fan, H.-Q. Wang, and G.-Y. Lv, "Fractal and neural networks based watermark identification," Multimedia Tools and Applications, vol.52, no.1, pp.201–219, 2011.
[13] M. Vafaei, H. Mahdavi-Nasab, and H. Pourghassem, "A new robust blind watermarking method based on neural networks in wavelet transform domain," World Applied Sciences Journal, vol.22, no.11, pp.1572–1580, 2013.
[14] M.-S. Hwang, C.-C. Chang, and K.-F. Hwang, "Digital watermarking of images using neural networks," J. Electronic Imaging, vol.9, no.4, pp.548–555, 2000.
[15] L.-Y. Hsu and H.-T. Hu, "Blind image watermarking via exploitation of inter-block prediction and visibility threshold in DCT domain," J. Visual Communication and Image Representation, vol.32, pp.130–143, 2015.
[16] C.-T. Yen and Y.-J. Huan, "Frequency domain digital watermark recognition using image code sequences with a back-propagation neural network," Multimedia Tools and Applications, vol.75, no.16, pp.9745–9755, 2016.
[17] I. Hamamoto and M. Kawamura, "Image watermarking technique using embedder and extractor neural networks," IEICE Trans. Inf. & Syst., vol.E102-D, no.1, pp.19–30, Jan. 2019.
[18] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "HiDDeN: Hiding data with deep networks," European Conf. Computer Vision, Lecture Notes in Computer Science, vol.11219, pp.682–697, Springer, Cham, 2018.
[19] Information hiding and its criteria for evaluation, IEICE, http://www.ieice.org/iss/emm/ihc/en/ (accessed Jan. 27, 2019).
[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," Proc. Int. Conf. Machine Learning, pp.448–456, 2015.
[21] D.P. Kingma and J.L. Ba, "Adam: A method for stochastic optimization," Proc. 3rd International Conference on Learning Representations, 2015.
[22] TensorFlow, https://www.tensorflow.org/ (accessed Jan. 27, 2019).
[23] Microsoft COCO: Common Objects in Context, http://cocodataset.org/ (accessed Aug. 4, 2019).

Ippei Hamamoto received a B.S. from Yamaguchi University in 2017 and an M.S. from the Graduate School of Sciences and Technology for Innovation, Yamaguchi University in 2019. He received the EMM Best Poster award in March 2019. His research interests include neural networks and digital watermarking.

Masaki Kawamura received B.E., M.E., and Ph.D. degrees from the University of Tsukuba in 1994, 1996, and 1999. He joined Yamaguchi University as a research associate in 1999. Currently he is an associate professor there. His research interests include associative memory models and information hiding. He is a senior member of IEICE and a member of JNNS, JPS, and IEEE.