Hindawi
Security and Communication Networks
Volume 2022, Article ID 7259469, 11 pages
https://doi.org/10.1155/2022/7259469

Research Article

Embedding Guided End-to-End Framework for Robust Image Watermarking

Beibei Zhang,[1] **Yunqing Wu**,[1] **and Beijing Chen**[1,2,3]

[1] *School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China*
[2] *Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu University of Information Technology, Chengdu 610225, China*
[3] *Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, Nanjing 210044, China*

Correspondence should be addressed to Beijing Chen; nbutimage@126.com

Received 7 April 2022; Accepted 21 June 2022; Published 9 July 2022

Academic Editor: Andrea Michienzi

In recent years, deep learning-based watermarking algorithms have received extensive attention. However, the existing algorithms mainly use the autoencoder to insert watermark automatically and ignore using the prior knowledge to guide the watermark embedding. In this paper, an end-to-end framework based on embedding guidance is proposed for robust image watermarking. It contains four modules, i.e., prior knowledge extractor, encoder, attacking simulator, and decoder. To guide the watermark embedding, the prior knowledge extractor providing chrominance and edge information of cover images is used to modify cover images before inserting the watermark by the encoder. To enhance the robustness of watermark extraction, the attacking simulator applying various differentiable attacks on the encoded images is introduced before extracting the watermark by the decoder. Experimental results show that the proposed algorithm achieves a good balance between invisibility and robustness and is superior to state-of-the-art algorithms.

1. Introduction

The unauthorized distribution of copies has become a threat to sharing of multimedia products. Hence, how to declare the ownership of the products is an urgent problem to be solved [1]. Digital watermarking technologies are widely used in copyright protection by embedding copyright information into digital products [2], such as digital literature, music, film, photography, and face portrait. Robustness against different attacks is significant for the practical application of digital watermarking. Traditionally, watermarking algorithms mainly rely on hand-crafted features to improve the robustness, such as applying various transforms [3–5] or using perceptual masking [6, 7]. The drawback to these hand-crafted algorithms is that they are not simultaneously robust to some types of distortions because different types of distortions often require different techniques [8]. Consequently, some deep learning-based algorithms have been presented [9–23]. They usually utilize convolutional neural network (CNN) to design end-to-end architecture with an encoder and a decoder. In order to further improve robustness, some improvement measures are proposed. These improvements can be categorized into two classes, i.e., attacking simulation and model architecture design [10]. The summary of different watermarking algorithms is listed in Table 1.

1.1. Attacking Simulation. Zhu et al. [11] were the first to propose a robust watermarking network HiDDeN with an attacking simulator. The attacking simulator was inserted into the network to satisfy the end-to-end training. However, HiDDeN can only be robust to a single attack, such as JPEG, Gaussian blur, crop, and dropout. Then, Mellimi et al. [12] and Ahmadi et al. [13] improved the attacking simulator to resist combined attacks. Since JPEG compression attack is

Table 1: Review of different deep learning-based watermarking algorithms.

Improvements	References	Techniques	Attacks	Capacity
Attacking simulation	Zhu [11]	The first work to simulate attacks by inserting the noise layers	Crop, Gaussian, dropout, and JPEG	90 bits
	Mellimi [12]	Simulation of noise layers against agnostic attacks	JPEG, noise, and noise	1024 bits
	Ahmadi [13]	Simulation of noise layers to resist mixture attacks	Crop, Gaussian, resize, and JPEG	1024 bits
	Chen [14]	Simulation of differentiable JPEG quantization	JPEG	1024 bits
	Jia [15]	Combination of simulated and real JPEG in noise layer	JPEG, crop, and Gaussian	1024 bits
	Ying [16]	Training a network to simulate JPEG compression	JPEG, scaling, and Gaussian	A whole image
Model architecture design	Dhaya [17]	Lightweight CNN scheme	JPEG, Gaussian, and median	512 bits
	Fang [18]	U-net architecture	Transparency, JPEG, and crop	128 bits
	Cun [19]	Combination of SplitNet and RefineNet	Crop and color	A whole image
	Mun [20]	Attention mechanism	JPEG, crop, filtering, and noise	512 bits
	Yu [21]	Generative adversarial network with attention mask	Noise, crop, and shift	A whole image
	Hao [22]	Generative adversarial network with a high-pass filter	Crop, Gaussian, and flip	30 bits
	Li [23]	Generative adversarial network with perceptual losses	Noise, filtering, and sharpen	1024 bits

nondifferentiable, some works [14–16] focused on JPEG compression simulation and improved the JPEG simulator by various differentiable methods to enhance the robustness against JPEG compression.

1.2. Model Architecture Design. Dhaya et al. [17] proposed a lightweight convolution neural network (LW-CNN) for the digital watermarking scheme, which had more resilience than other standard approaches. Fang et al. [18] exploited a template-based approach combined with U-Net to achieve better robustness. Cun et al. [19] used SplitNet and RefineNet to smooth watermarked regions for a better quality of watermarked images. Mun et al. [20] introduced attention mechanism into the watermarking field to achieve good performance in robustness against attacks. In addition, some notable algorithms with adversarial training [21–23] have greatly improved the perceptual quality of the watermarked images.

However, these existing CNN-based robust watermarking algorithms focus on attacking simulation and model architecture design before the watermark extraction. They do not consider prior knowledge to guide the watermark embedding. To further balance between invisibility and robustness, motivated by the traditional algorithms, some prior knowledge, such as the chrominance and edge saliency of cover images, is considered before the watermark embedding. The major contributions are as follows.

(1) We propose a prior knowledge extractor to obtain the chrominance and edge saliency of cover images for guiding watermark embedding.

(2) We propose an embedding guided end-to-end framework for robust watermarking based on the proposed prior knowledge extractor and attacking simulator.

(3) We conduct a lot of empirical experiments to evaluate the performance of the proposed algorithm in terms of invisibility and robustness. Experimental results demonstrate that our algorithm achieves a good balance between invisibility and robustness and performs better than state-of-the-art algorithms.

2. Methods

In this section, the proposed framework is described in detail. The overall architecture and loss functions are presented in subsection 2.1. Then, each module is explained in subsections 2.2–2.6 one by one, i.e., prior knowledge extractor, encoder, attacking simulator, decoder, and discriminator.

2.1. Model Architecture. The main framework is presented in Figure 1. As shown in Figure 1, the proposed model is based on autoencoder structure, which consists of four modules: a prior knowledge extractor, an encoder, an attacking simulator, and a decoder. The prior knowledge extractor obtains prior knowledge to modify cover images for guiding watermark insertion. After that, the encoder hides the watermark into the modified cover image. Then, the attacking simulator performs various simulated attacks on encoded images as a network layer. Finally, the decoder extracts the watermark from attacked (or unattacked) encoded images. These modules achieve their objectives through the following loss functions.

The encoder aims to insert the watermark into the cover image invisibly. So, the distortion loss is used to limit the distortion of the encoded image by

$$L_D(I_{\mathrm{co}}, I_{\mathrm{en}}) = \left\| I_{\mathrm{co}} - I_{\mathrm{en}} \right\|_2^2, \tag{1}$$

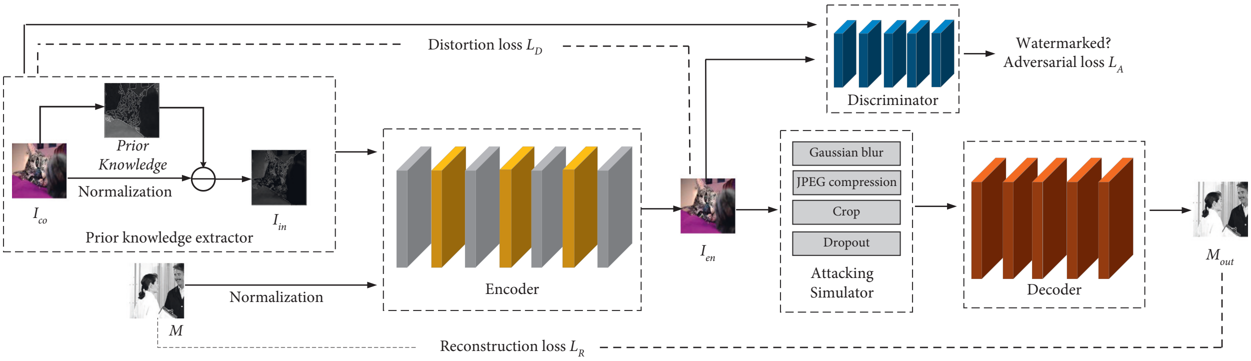


Figure 1: Overall architecture of the proposed model.

where I_{co} and I_{en} represent the cover image and encoded image, respectively.

The decoder wants to extract the watermark from the encoded images as much as possible. So, the reconstruction loss is adopted to improve the quality of the extracted watermark by

$$L_R(M, M_{\text{out}}) = \|M - M_{\text{out}}\|_2^2, \tag{2}$$

where M and M_{out} are the original watermark and extracted watermark, respectively.

The discriminator is used to judge whether the generated images are similar enough to the cover images. The discriminator and encoder compete with each other. So, the adversarial loss is considered to optimize the visual quality of the encoded image by

$$L_A = (I_{\text{co}}, I_{\text{en}}) = \log(1 - D(I_{\text{co}})) + \log(D(I_{\text{en}})), \tag{3}$$

where D represents the discriminator.

Therefore, the total loss for the proposed framework is

$$L_{\text{total}} = \alpha L_D(I_{\text{co}}, I_{\text{en}}) + \beta L_R(M, M_{\text{out}}) + \gamma L_A(I_{\text{co}}, I_{\text{en}}), \tag{4}$$

where α, β, and γ are three hyper-parameters.

2.2. Prior Knowledge Extractor Module. Most existing deep learning-based algorithms mainly use the autoencoder to insert watermark automatically and ignore using the prior knowledge to guide the watermark embedding. According to the human visual system (HVS), people are less sensitive to modification in regions with rich chrominance and edge information [24–29]. So, the chrominance and edge saliency proposed in [30] are considered prior knowledge in this paper. The cover image is modified before watermark insertion to make the watermarking robust. Figure 2 depicts the flow diagram of our proposed prior knowledge extractor.

In order to obtain the chrominance information of the cover images, first, the cover image is converted into $\mathrm{YC_bC_r}$ color space by

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B, \\ C_b &= 0.564(B - Y), \\ C_r &= 0.713(R - Y), \end{aligned} \tag{5}$$

where Y represents the luminance component and C_b and C_r represent chrominance components.

Then, the chrominance saliency $S_C(x)$ of a point x is obtained by

$$S_C(x) = 1 - \exp\left(-\frac{f_b^2(x) + f_c^2(x)}{\delta^2}\right), \tag{6}$$

where $f_b(x)$ and $f_c(x)$ are the normalization mappings of the C_b and C_r components, respectively, δ is a parameter set as 0.25 in this paper.

In order to obtain the edge information of cover images, the canny operator [31] is used to extract edge information. The edge saliency $S_E(x)$ of a point x is computed by

$$S_E(x) = \exp\left(-\frac{\text{Canny}(x) + 1}{\tau}\right), \tag{7}$$

where Canny(x) represents the result calculated by the canny operator for a given point x and τ is a threshold set as 2 in this paper.

Finally, as is known to all, the stronger the chrominance and edge saliency are, the less sensitive the human eye is. So, the cover image is modified by

$$I_{in} = I_{co} - \left(1 - \frac{S_C(x) + S_E(x)}{2}\right), \tag{8}$$

where I_{co} is the original cover image after normalization and I_{in} is its modified one. According to (8), the greater the chrominance and edge saliency is, the smaller the modification of the cover pixel is, consequently, the relatively greater the change of cover pixel is in the watermark insertion.

2.3. Encoder Module. The architecture of the encoder network is illustrated in Figure 3. As shown in Figure 3, the encoder network has two parallel branches corresponding to the cover image and watermark image, respectively. One branch uses some convolutional layers to extract shallow detail features and deep semantic features of input normalized watermark images. The other branch uses a sequence of convolutional layers to extract features of the input cover image for merging with the features extracted from the watermark image. Specifically, in order to embed

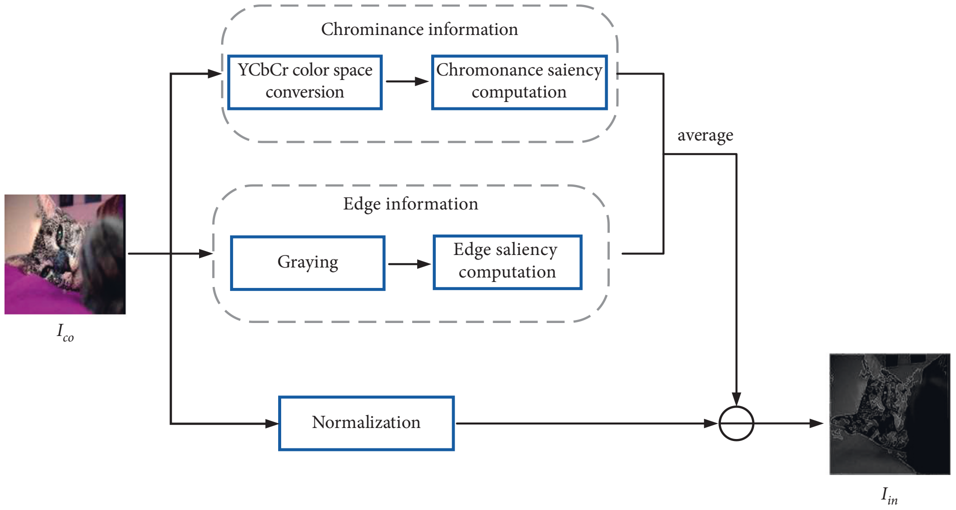

Chrominance information
YCbCr color space conversion
Chromonance saiency computation
average
Edge information
Graying
Edge saliency computation
Normalization
I_{co}
I_{in}

Figure 2: Flow diagram of the proposed prior knowledge extractor.

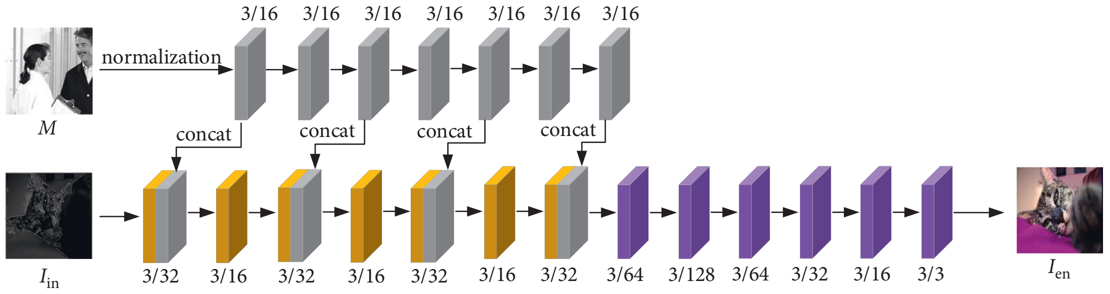


Figure 3: Architecture of the encoder. The numbers in the form *m/n* represent the kernel size (m) and the number of kernels (n) in each convolution layer.

watermark images into cover images, the encoder concatenates the features map extracted from each alternate layer of the watermark branch to the corresponding output features of the cover branch. Like [32], this concatenating process is repeated four times. Finally, the cover image and watermark are entirely fused as encoded images.

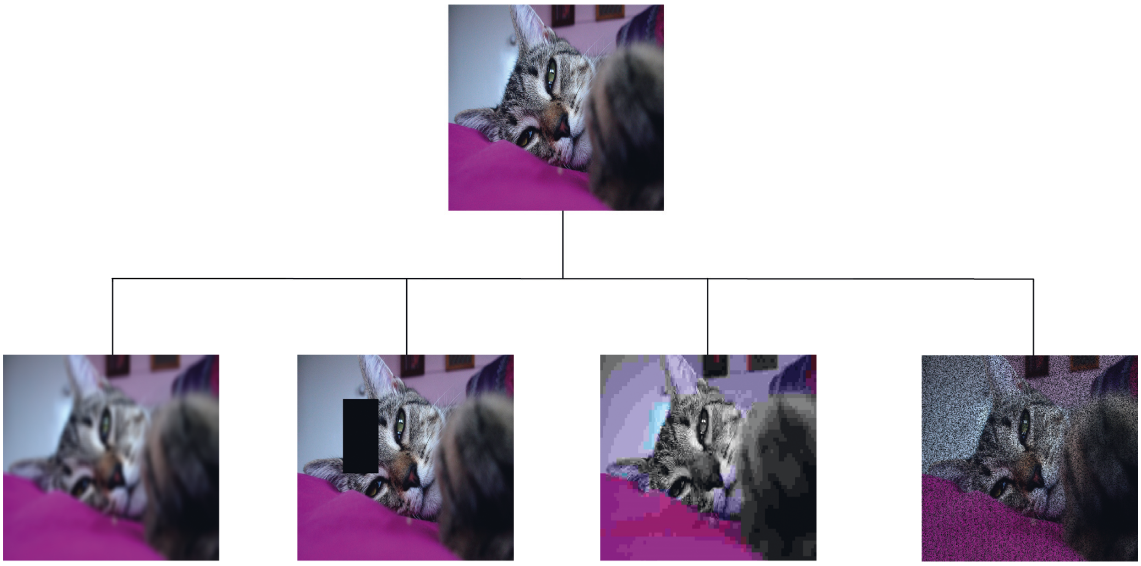
2.4. Attacking Simulator Module. In order to be robust against a variety of image distortions, as shown in Figure 1, an attacking simulator is inserted between the encoder and decoder to simulate various attacks by differentiable methods. Its parameters do not require to be updated during the entire network training process. Note that each iteration randomly selects one type of attack with equal probability. Specifically,[33], as shown in Figure 4, our attacking simulator includes four types of attacks: Gaussian blur, crop, JPEG compression, and dropout.

2.4.1. Gaussian Blur. Gaussian blur is also called Gaussian smoothing. It blurs the encoded images by performing a convolution operation with a Gaussian kernel. The larger the size of the convolution kernel, the stronger the blur attack.

2.4.2. Crop. Crop operation is simulated by randomly cropping out a small rectangle from the encoded images, namely, by replacing all the pixel values in this rectangle with zero. Specifically, the attack is simulated by multiplying with a 0–1 mask of the same size as the encoded image. In this mask, the region with pixel value 0 represents the cropped region, while the region with pixel value 1 represents the remaining region.

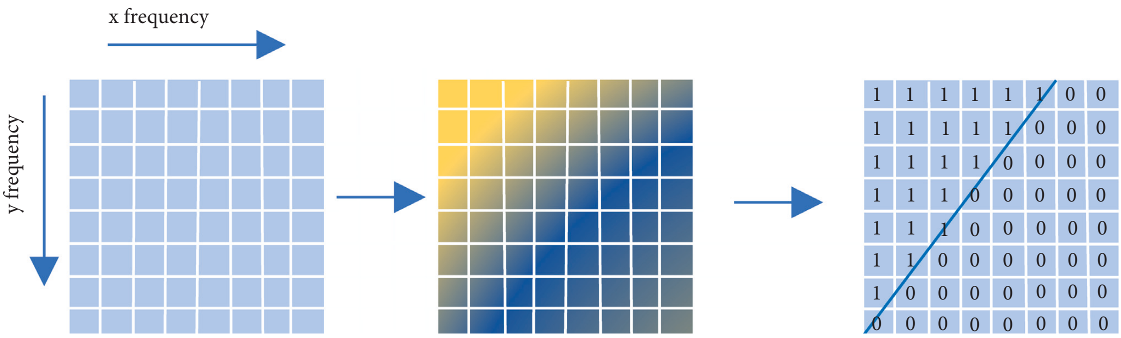
2.4.3. JPEG Compression. The steps of JPEG compression are composed of color space transformation, discrete cosine transform, quantization, and entropy coding. The sampling and discrete cosine transform steps are modeled by the max-pooling layer and convolution layer, respectively. Especially, as shown in Figure 5, the nondifferentiable quantization step is approximately simulated by performing JPEG-mask on the feature maps [11].

2.4.4. Dropout. Dropout attack is a common noise in image processing. It is implemented by arbitrarily replacing a certain ratio of pixels with zero. The detailed processing is similar to crop attack by multiplying with a 0–1 mask. The



(a) Gaussian blur (b) Crop (c) JPEG compression (d) Dropout

Figure 4: Samples of various attacks: (a)Gaussian blur; (b)crop; (c)JPEG compression; (d)dropout.



1	1	1	1	1	1	0	0
1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(a) DCT block (b) Real quantization table (c) Simulated JPEG-Mask

Figure 5: Simulation of nondifferentiable quantization step by JPEG-mask; (a)DCT block; (b)real quantization table; (c)simulated JPEG-mask.

difference is that the pixel values 0 and 1 are randomly distributed in the mask.

2.5. Decoder Module. In the end-to-end training, the decoder carries out the decoding procedure after encoding or attacking. The structure of the decoder is shown in Figure 6. The decoder takes the encoded or attacked image as input and extracts the watermark image. It uses seven Conv-BN-ReLU blocks to extract the watermark image from the input image. In this process, the function of convolutional operation is to extract features, and batch normalization (BN) speeds up the calculation while ReLU activation plays the filtering role. The final convolutional layer with a 3×3 kernel outputs watermark images.

2.6. Discriminator Module. The primary role of the discriminator is to improve the visual similarity between the encoded and cover images by adversarial training. The architecture of the discriminator is presented in Figure 7. It is similar to that of the decoder. The difference is that the discriminator outputs binary classification results to judge whether the image contains the watermark or not. Therefore, the discriminator is built with five Conv-BN-ReLU blocks, an adaptive average pooling layer, a linear layer with a single output unit, and a Sigmoid activation layer.

3. Experimental Results and Analysis

In this section, experiments are carried out to prove the effectiveness and robustness of the proposed algorithm. The training datasets and experimental details are described in subsection 3.1. Then, the ablation experiments in subsection 3.2 are performed to demonstrate the improvements in the proposed algorithm. Finally, the robustness of the model for different types of attacks is tested in subsection 3.3.

3.1. Experimental Datasets, Implementation Details, and Evaluation Metrics

3.1.1. Experimental Datasets. 5,000 images randomly selected from the COCO dataset [34] are used as cover images.

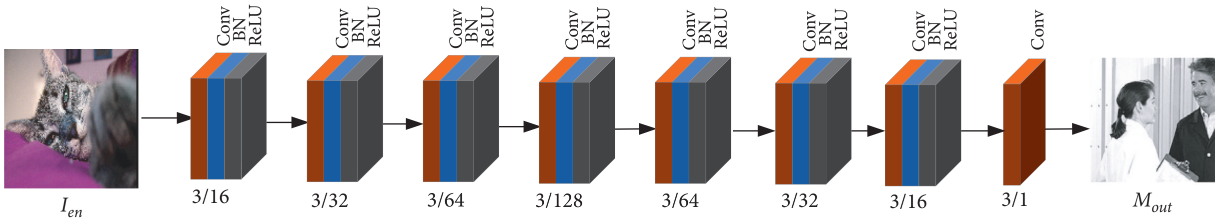


Figure 6: Architecture of the decoder. The numbers in the form m/n represent the kernel size (m) and the number of kernels (n) in each convolution layer.

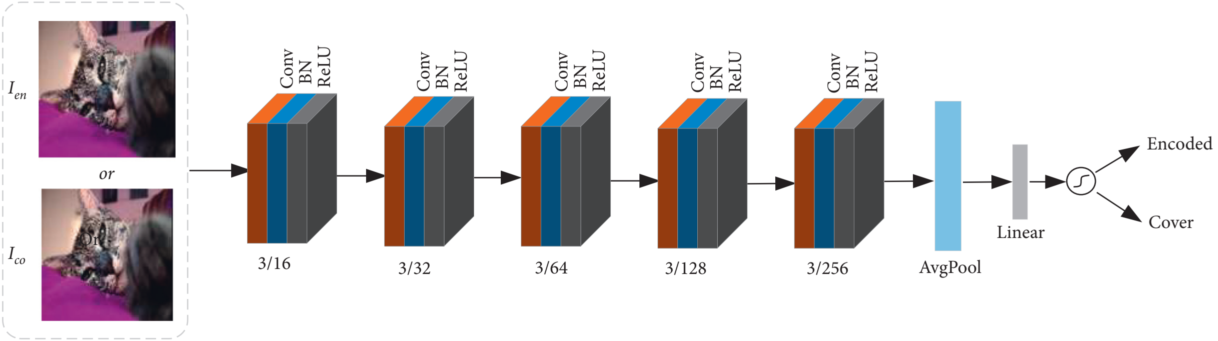


Figure 7: Architecture of the discriminator. The numbers in the form m/n represent the kernel size (m) and the number of kernels (n) in each convolution layer.

Three types of images are taken as watermark images for experiments. They are 5,000 logo images randomly selected from logo-2k + [35], 5,000 digital number images from MNIST [36], and 5,000 general images from ImageNet [37]. These watermarks are converted into grayscale images before embedding. 5,000 cover images and each 5,000 watermark images are regarded as 5,000 pairs for the following experiments. Then, the cover images and watermark images are, respectively, divided into training/validation/testing sets according to the ratio of 8 : 1:1 and resized to 128 × 128.

3.1.2. Implementation Details. The proposed watermarking model is trained iteratively using the ADAM optimizer [38] with an initial learning rate of 1.0e-3. The batch size is set as 16. The weights in the loss function shown in (4) are set as $\alpha = 0.3$, $\beta = 0.7$, and $\gamma = 0.001$. In addition, all simulated attacks have a hyperparameter governing the strengths: the kernel width ω of Gaussian blur is 3; quality factor QF of JPEG compression is 90; and ratios p of crop and dropout are 0.1 and 0.15, respectively.

3.1.3. Evaluation Metrics. The image visual quality is commonly evaluated by peak signal-to-noise ratio (PSNR) and structural similarity index metric (SSIM) metrics. Their definitions are given in the following.

Given two images U and V, the PSNR can be defined as

$$\mathrm{PSNR}(U, V) = 10\log_{10}\left(\frac{L^2}{\mathrm{MSE}(U, V)}\right), \tag{9}$$

where L is the maximum pixel value, which is usually set as 255, *MSE* is mean squared error defined as

$$\mathrm{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(U_i - V_i\right)^2}, \tag{10}$$

where n is the number of pixels.

The SSIM between two images U and V is defined as

$$\mathrm{SSIM}(U, V) = \frac{\left(2\mu_U\mu_V + C_1\right)\left(2\sigma_{UV} + C_2\right)}{\left(\mu_U^2\mu_V^2 + C_1\right)\left(\sigma_U^2\sigma_V^2 + C_2\right)}, \tag{11}$$

where μ_U and μ_V are the means, σ_U and σ_V are the standard deviations, σ_{UV} is the cross-covariance of U and V, and C_1 and C_2 are two constants used to avoid a null denominator.

3.2. Ablation Experiments. Here, some ablation experiments are conducted to validate the proposed model. All the experiments are performed under the combined attacks with all four different types of attacks.

Firstly, we begin by analyzing what the prior knowledge extractor can do. Table 2 shows the average PSNR and SSIM values of 5,000 encoded images and 5,000 extracted watermarks with/without the extractor. As the results are shown, the visual quality of both the encoded images and extracted watermarks is improved after introducing the prior knowledge extractor. This is because the extractor obtains prior knowledge to find more suitable locations for watermark embedding.

Then, we verify the effectiveness of the attacking simulator. So, we compared the proposed models without and with the attacking simulator in the training stage. The results are shown in Table 2. As shown in Table 3, when the attacking simulator is considered, although the quality of the

Table 2: Performance comparison between the proposed model without and with the prior knowledge extractor.

Prior knowledge extractor	Encoded image		Extracted watermark	
	PSNR	SSIM	PSNR	SSIM
Without	37.56	0.9626	35.29	0.9595
With	40.69	0.9904	38.19	0.9722

Table 3: Performance comparison between the proposed model without and with an attacking simulator.

Attacking simulator	Encoded image		Extracted watermark	
	PSNR	SSIM	PSNR	SSIM
Without	41.02	0.9913	24.12	0.7824
With	40.69	0.9904	38.19	0.9722

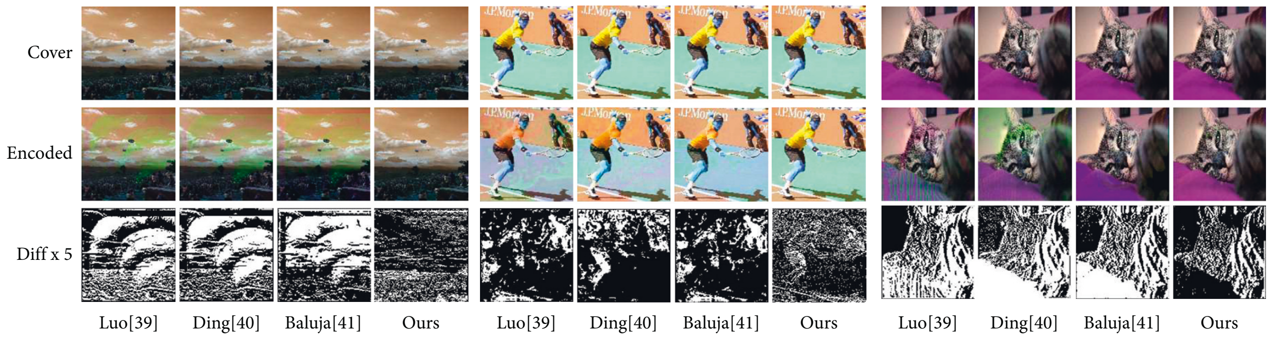


Figure 8: Some examples of the cover images (landscape, sportsman, and cat) and their corresponding encoded images, as well as their five times magnified differential images.

encoded images is sacrificed a little bit and the quality of extracted watermarks improves significantly. The PSNR values of the extracted watermarks increase from 24.12 dB to 38.19 dB and SSIM from 0.7824 to 0.9722.

The experimental results in Tables 2 and 3 show that either the prior knowledge extractor or attacking simulator is significant to the robust watermarking.

3.3. Comparison Experiments with Other Algorithms. In order to further evaluate the performance of the proposed algorithm, our algorithm is compared with some existing deep learning-based algorithms [39–41] in terms of both invisibility and robustness.

3.3.1. Invisibility. The challenge for digital watermarking is to improve robustness while keeping invisibility. Figure 8 shows the visual comparison of different watermarking algorithms. In addition, Table 4 presents their corresponding numerical results by PSNR and SSIM. It can be observed from Figure 8 and Table 4 that the watermarks are invisible in the encoded images for the proposed algorithm with high PSNR and SSIM values, while it is not the case for the other three algorithms who suffer from a little color bias. This is due to the use of prior knowledge for guiding watermark insertion in our algorithm.

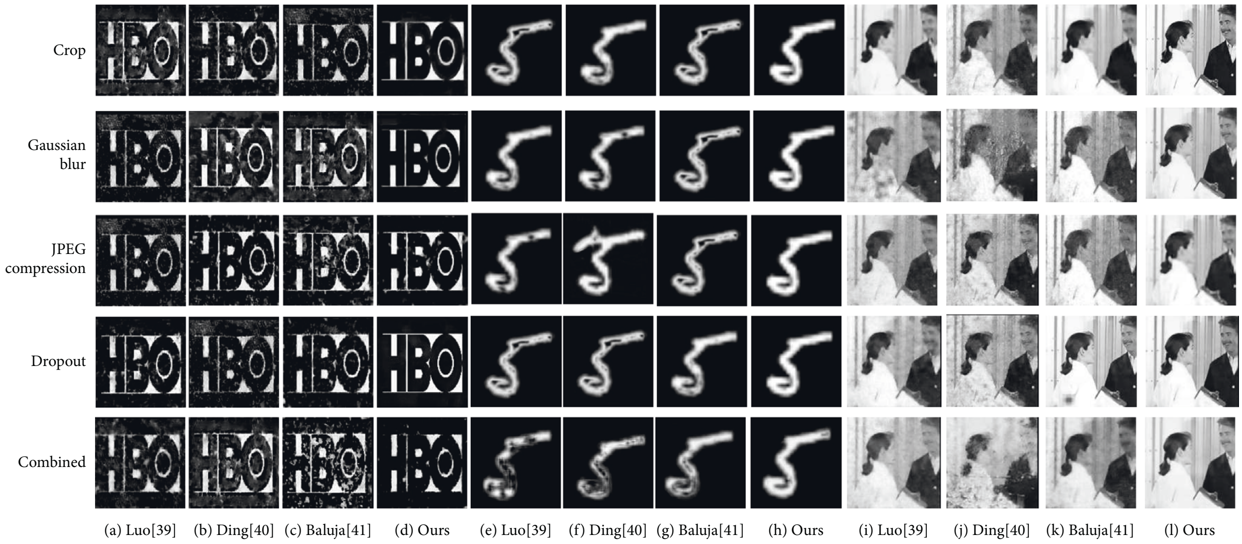
Table 4: PSNR and SSIM values of three encoded images in Figure 8 for different algorithms.

Algorithms	Landscape		Sportsman		Cat	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Luo [39]	30.19	0.883	31.34	0.896	30.46	0.886
Ding [40]	30.53	0.892	32.78	0.912	31.89	0.905
Baluja [41]	31.42	0.9012	33.97	0.924	32.75	0.917
Ours	36.71	0.957	38.12	0.963	36.92	0.959

3.3.2. Robustness. In order to test the robustness, the encoded images are carried out in five different types of attacks. Table 5 presents the average PSNR and SSIM values of 5,000 encoded images and 5,000 watermark images for four compared algorithms. In addition, Figure 9 shows some visual samples of the extracted watermarks. It can be observed from Table 5 and Figure 9 that the proposed algorithm achieves the best performance for all five types of attacks in both numerical and visual aspects, especially for the combined attack. Although the encoded images are distorted under various attacks, our algorithm can preserve watermark fidelity to a great extent with few errors. However, it is not the case for the other three algorithms, whose extracted watermarks suffer from some errors with some noise in vision. This is attributable to the watermarking guidance of prior knowledge and the consideration of

Table 5: Comparison of robustness against different types of attacks.

Attacks	Algorithms	Encoded image		Extracted watermark					
				Logo		MNIST		ImageNet	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Gaussian blur ($\omega = 9$)	Luo [39]	33.79	0.9213	31.49	0.9109	32.76	0.9267	30.38	0.9015
	Ding [40]	32.56	0.9146	30.24	0.8937	31.95	0.9086	28.89	0.8812
	Baluja [41]	34.24	0.9322	32.62	0.9148	33.97	0.9275	31.04	0.9068
	Ours	41.11	0.9751	36.24	0.9641	37.87	0.9753	34.92	0.9549
JPEG compression ($QF = 60$)	Luo [39]	33.08	0.9225	30.37	0.9021	31.43	0.9078	29.12	0.8994
	Ding [40]	32.67	0.9114	29.73	0.8864	30.82	0.9026	28.23	0.8801
	Baluja [41]	34.49	0.9511	31.39	0.9115	32.74	0.9248	30.59	0.9082
	Ours	38.35	0.9657	34.17	0.9512	35.86	0.9573	33.11	0.9448
Crop ($p = 0.4$)	Luo [39]	32.93	0.9325	31.37	0.9121	32.66	0.9264	30.25	0.9027
	Ding [40]	32.20	0.9007	30.88	0.8964	32.05	0.9128	30.08	0.8901
	Baluja [41]	34.36	0.9461	32.59	0.9323	33.74	0.9456	31.23	0.9119
	Ours	39.45	0.9727	37.06	0.9605	38.85	0.9773	36.39	0.9597
Dropout ($p = 0.45$)	Luo [39]	34.16	0.9321	32.45	0.9409	33.76	0.9487	31.07	0.9339
	Ding [40]	32.64	0.9145	30.36	0.9047	31.83	0.9102	29.99	0.8995
	Baluja [41]	33.97	0.9343	32.82	0.9222	33.75	0.9298	32.01	0.9186
	Ours	39.18	0.9710	35.64	0.9586	36.83	0.9642	34.75	0.9503
Combined	Luo [39]	32.17	0.9189	30.01	0.8832	30.95	0.9045	28.67	0.8974
	Ding [40]	31.01	0.9013	28.47	0.8813	31.08	0.8942	27.84	0.8757
	Baluja [41]	33.12	0.9387	30.96	0.9102	31.89	0.9211	30.22	0.9032
	Ours	37.64	0.9724	34.11	0.9505	35.79	0.9568	34.68	0.9501



(a) Luo[39] (b) Ding[40] (c) Baluja[41] (d) Ours (e) Luo[39] (f) Ding[40] (g) Baluja[41] (h) Ours (i) Luo[39] (j) Ding[40] (k) Baluja[41] (l) Ours

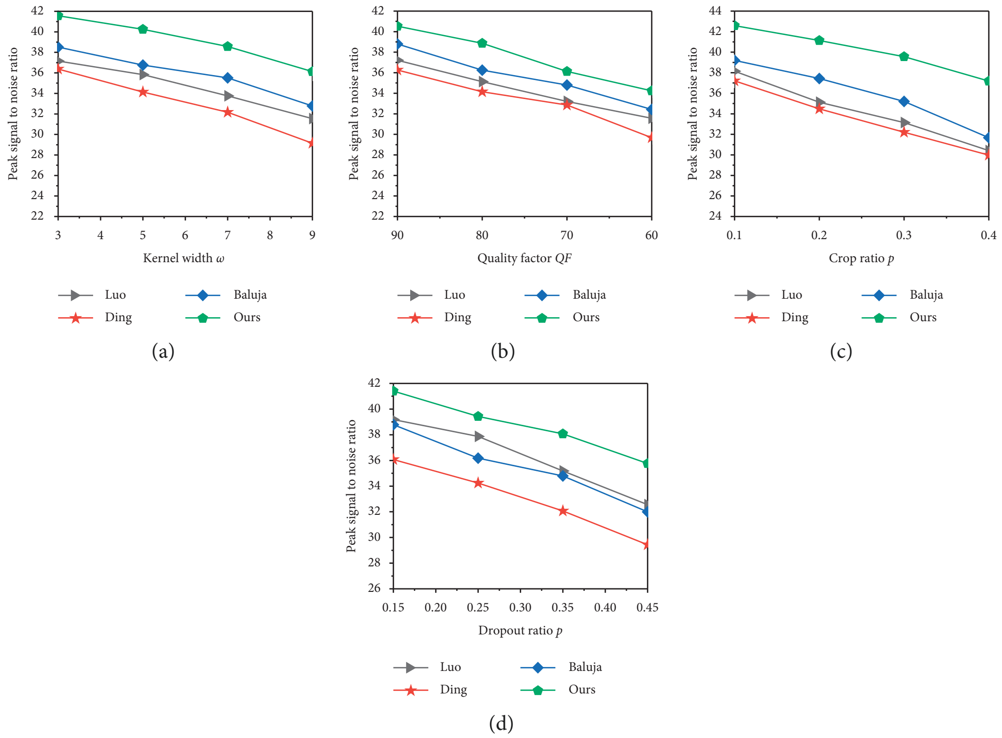
Figure 9: Visual comparison of robustness against different kinds of attacks. Samples (a)–(d) are from Logo-2k, (e)–(h) from MNIST, and (i)–(l) from ImageNet.

attacking simulator in our algorithm. Regarding three different types of watermark images, all the compared algorithms perform best on the MNIST watermarks. The main reason is that the other two types of watermark images are more complex and contain more semantic information, which results in more difficulty in the watermark extraction.

In addition, we evaluate the generalization performance of different watermarking algorithms against attacks different from those in the training stage in two aspects, i.e., different attack levels and different attack types.

3.3.3. Different Attack Levels. Figure 10 shows the average PSNR values of the extracted watermark images under different attack levels. As shown in Figure 10, our algorithm still performs better than the other three algorithms when being attacked by different levels of various attacks. In addition, the performance of all four compared algorithms decreases with the increase in attack levels.

Different attack types. To evaluate the performance in resisting the attacks that were not considered during the training stage, we select four kinds of black-box image



(a)

(b)

(c)

(d)

Figure 10: Comparison with other algorithms under different attacks with different levels: (a)Gaussian blur; (b)JPEG compression; (c)crop; (d)dropout.

Table 6: Comparison with other algorithms against the attack types different from those in the training stage.

Algorithms	Resizing ($T = 2$)	Medium blur ($\omega = 3$)	Salt and pepper noise ($p = 0.2$)	Gaussian noise ($\sigma = 1.0$)
Luo [37]	27.97	29.48	30.15	29.75
Ding [38]	26.64	28.57	29.31	28.43
Baluja [39]	28.89	30.32	31.67	30.46
Ours	32.87	33.91	34.02	33.52

attacks (resizing, medium blur, salt and pepper noise, and Gaussian noise) to test the model. The levels of these attacks are as follows: the scaling factor T of resizing is 2; kernel width ω of medium blur is 3; ratio p of salt and pepper noise is 0.2; and standard deviation σ of Gaussian noise is 1.0. Table 6 shows the average PSNR values of the extracted watermark images of different algorithms. As can be seen from Tables 5 and 6, the proposed algorithm still maintains higher PSNR values than the other three algorithms, though its performance decreases when facing attacks different from the training stage.

4. Conclusion

In this paper, we propose an embedding guided end-to-end framework for robust image watermarking. In this algorithm, a prior knowledge extractor and attacking simulator are introduced to guide watermarking embedding and enhance the robustness of watermark extraction, respectively. The experiment results demonstrate that, compared to the existing algorithms, the proposed algorithm performs better in both invisibility and robustness. However, the proposed algorithm does not consider other common attacks in practical application, such as printing, screen photography, and geometric transformation. Therefore, in the future, we will focus on the simulation of these attacks and study the deep learning-based watermarking algorithms against these attacks.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Open Fund of Advanced Cryptography and the System Security Key Laboratory of Sichuan Province (Grant no. SKLACSS-202113) and the

National Natural Science Foundation of China (Grant no. 62072251), the PAPD fund.

References

[1] H. Wu, G. Liu, Y. Yao, and X. Zhang, "Watermarking neural networks with watermarked images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 7, pp. 2591–2601, 2020.

[2] U. H. Panchal and R. Srivastava, "A comprehensive survey on digital image watermarking techniques," in *Proceedings of the 2015 Fifth International Conference on Communication Systems and Network Technologies*, pp. 591–595, Gwalior, India, April 2015.

[3] H. Nazir, I. S. Bajwa, M. Samiullah, A. Waheed, and M. Muhammad, "Robust secure color image watermarking using 4D hyperchaotic system, DWT, HbD, and SVD based on improved FOA algorithm," *Security and Communication Networks*, vol. 2021, Article ID 6617944, 17 pages, 2021.

[4] H. C. Fu, Z. F. Zhao, and X. L. He, "Improving Anti-compression Robustness of JPEG Adaptive Steganography Based on Robustness Measurement and DCT Block Selection," *Security and Communication Networks*, vol. 2021, Article ID 9153468, 15 pages, 2021.

[5] C. S. Yang, J. B. Li, U. A. Bhatti, J. Liu, J. Ma, and M. Huang, "Robust zero watermarking algorithm for medical images based on zernike-DCT," *Security and Communication Networks*, vol. 2021, Article ID 4944797, 8 pages, 2021.

[6] W. F. Qi, Y. X. Liu, S. R. Guo, X. Wang, and Z. Guo, "An adaptive visible watermark embedding method based on region selection," *Security and Communication Networks*, vol. 2021, Article ID 6693343, 11 pages, 2021.

[7] K. Zebbiche, F. Khelifi, and K. Loukhaoukha, "Robust additive watermarking in the DTCWT domain based on perceptual masking," *Multimedia Tools and Applications*, vol. 77, no. 16, Article ID 21304, 2018.

[8] M. Asikuzzaman and M. R. Pickering, "An overview of digital watermarking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2131–2153, 2017.

[9] Y. Quan, H. Teng, Y. Chen, and H. Ji, "Watermarking deep neural networks in image processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 1852–1865, 2021.

[10] H. Kandi, D. Mishra, and S. R. S. Gorthi, "Exploring the learning capabilities of convolutional neural networks for robust image watermarking," *Computers & Security*, vol. 65, pp. 247–268, 2017.

[11] J. Zhu, R. Kaplan, J. Johnson, and L. Fei, "Hidden: hiding data with deep networks," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 657–672, Munich, Germany, September 2018.

[12] S. Mellimi, V. Rajput, I A Ansari, and C W Ahn, "A fast and efficient image watermarking scheme based on deep neural network," *Pattern Recognition Letters*, vol. 151, pp. 222–228, 2021.

[13] M. Ahmadi, A. Norouzi, N. Karimi, E. Ali, and S. Samavi, "ReDMark: framework for residual diffusion watermarking based on deep networks," *Expert Systems with Applications*, vol. 146, Article ID 113157, 2020.

[14] B. J. Chen, Y. Q. Wu, G. Coatrieux, X. Chen, and Y. Zheng, "JSNet: a simulation network of JPEG lossy compression and restoration for robust image watermarking against JPEG attack," *Computer Vision and Image Understanding*, vol. 197-198, Article ID 103015, 2020.

[15] Z. Y. Jia, H. Fang, and W. M. Zhang, "MBRS: enhancing robustness of DNN-based watermarking by mini-batch of real and simulated JPEG compression," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 41–49, Chengdu, China, October 2021.

[16] Q. Ying, H. Zhou, X. Zeng, H. Xu, Z. Qian, and X. Zhang, "Hiding Images into Images with Real-World Robustness," 2021, https://arxiv.org/abs/2110.05689.

[17] R. R, "Light weight CNN based robust image watermarking scheme for security," *Journal of Information Technology and Digital World*, vol. 3, no. 2, pp. 118–132, 2021.

[18] H. Fang, D. Chen, Q. Huang et al., "Deep template-based watermarking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1436–1451, 2020.

[19] X. Cun and C. M. Pun, "Split Then Refine: Stacked Attention-Guided ResUNets for Blind Single Image Visible Watermark Removal," 2020, https://arxiv.org/abs/2012.07007.

[20] S M Mun, S H Nam, H Jang, D Kim, and H-K Lee, "Finding robust domain from attacks: a learning framework for blind watermarking," *Neurocomputing*, vol. 337, pp. 191–202, 2019.

[21] C. Yu, "Attention based data hiding with generative adversarial networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1120–1128, NY, USA, February 2020.

[22] K. Hao, G. Feng, and X. P. Zhang, "Robust image watermarking based on generative adversarial network," *China Communications*, vol. 17, no. 11, pp. 131–140, 2020.

[23] Q. Li, X Y. Wang, B. Ma et al., "Concealed attack for robust watermarking based on generative model and perceptual loss," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[24] B. J. Chen, X. Liu, Y. H. Zheng, G. Y. Zhao, and Y. -Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved xception," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 3527–3538, 2022.

[25] H. Y. Qi, D. Zhao, and J. Y. Zhao, "Human visual system based adaptive digital image watermarking," *Signal Processing*, vol. 88, no. 1, pp. 174–188, 2008.

[26] T. Sridevi and S. Fathima, "Watermarking algorithm based using genetic algorithm and HVS," *International Journal of Computer Application*, vol. 74, no. 13, 2013.

[27] F. Kabir and M. N. Kabir, "A Block-based RDWT-SVD Image Watermarking Method Using Human Visual System Characteristics," *The Visual Computer*, vol. 36, no. 1, pp. 19–37, 2020.

[28] B. J. Chen, W. J. Tan, C. Coatrieux, Y. H. Zheng, and Y. -Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.

[29] Y. L. Bei, S. Qiao, M. X. Liu, Q. Zhang, and X. R. Zhu, "A color image watermarking scheme Against geometric rotation attacks based on HVS and DCT-DWT," in *Proceedings of the 2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pp. 343–347, Jinan, China, December 2018.

[30] L. Zhang, Z. Gu, and H. Li, "SDSP: a novel saliency detection method by combining simple priors," in *Proceedings of the 2013 IEEE International Conference on Image Processing*, pp. 171–175, Melbourne, VIC, Australia, September 2013.

[31] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[32] B. J. Chen, J. X. Wang, Y. Y. Chen, Z. Jin, H. J. Shim, and Y. Q. Shi, "High-capacity robust image steganography via adversarial network," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 14, no. 1, pp. 366–381, 2020.

[33] M. Jamali, N. Karim, P. Khadivi, Z. Jin, H. J. Shim, and Y. Q. Shi, "Robust Watermarking Using Diffusion of Logo into Autoencoder Feature Maps," 2021, https://arxiv.org/abs/2105.11095.

[34] T. Y. Lin, M. Maire, S. Belongie et al., "Microsoft coco: common objects in context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 740–755, Zurich, Switzerland, September 2014.

[35] J. Wang, W. Min, S. Hou et al., "Logo-2K+: a large-scale logo dataset for scalable logo classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 6194–6201, NY, USA, February 2020.

[36] Y. L. Cun, "The MNIST Database of Handwritten Digits," 1998.

[37] J. Deng, W. Dong, R. Socher, F. F. Li, L. J. Li, and K. Li, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Miami, FL, USA, June 2009.

[38] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2014, https://arxiv.org/abs/1412.6980.

[39] X. Luo, R. Zhan, H. Chang, F. Yang, and M. Peyman, "Distortion agnostic deep watermarking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Article ID 13557, Seattle, WA, USA, June 2020.

[40] W. P. Ding, Y. R. Ming, Z. H. Cao, and C. T. Lin, "A generalized deep neural network approach for digital watermarking analysis," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 3, pp. 1–15, 2021.

[41] S. Baluja, "Hiding images within images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 7, pp. 1685–1697, 2020.