

Article

A Robust Semi-Blind Watermarking Technology for Resisting JPEG Compression Based on Deep Convolutional Generative Adversarial Networks

Chin-Feng Lee ¹ , Zih-Cyuan Chao ², Jau-Ji Shen ²  and Anis Ur Rehman ^{1,*} 

¹ Department of Information Management, Chaoyang University of Technology, Taichung 41349, Taiwan; lcf@cyut.edu.tw

² Department of Management Information Systems, National Chung Hsing University, Taichung 40227, Taiwan; jj40709@gmail.com (Z.-C.C.); jjshen@nchu.edu.tw (J.-J.S.)

* Correspondence: rehmananis771@ustb.edu.pk

Abstract: In recent years, the internet has developed rapidly. With the popularity of social media, uploading and backing up digital images has become the norm. A huge number of digital images are circulating on the internet daily, and issues related to information security follow. To protect intellectual property rights, digital watermarking is an indispensable technology. However, the common lossy compression technology in the network transmission process is a big problem for watermarking. This paper describes an innovative semi-blind watermarking method with the use of deep convolutional generative adversarial networks (DCGANs) for hiding and extracting watermarks from JPEG-compressed images. The proposed method achieves an average peak signal-to-noise ratio (PSNR) of 49.99 dB, a structural similarity index (SSIM) of 0.95, and a bit error rate (BER) of 0.008 across varying JPEG quality factors. The process is based on an embedder, decoder, generator, and discriminator. It allows watermarking, decoding, or reconstruction to be symmetric such that there is less distortion and durability is improved. It constructs a specific generator for each image and watermark that is supposed to be protected. Experimental results show that, with the variety of JPEG quality factors, the restored watermark achieves a remarkably low corrupted rate, outstripping recent deep learning-based watermarking methods.



Academic Editor: Jie Yang

Received: 20 December 2024

Revised: 5 January 2025

Accepted: 7 January 2025

Published: 10 January 2025

Citation: Lee, C.-F.; Chao, Z.-C.; Shen, J.-J.; Rehman, A.U. A Robust Semi-Blind Watermarking Technology for Resisting JPEG Compression Based on Deep Convolutional Generative Adversarial Networks. *Symmetry* **2025**, *17*, 98. <https://doi.org/10.3390/sym17010098>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: invisible digit watermark; JPEG compression; deep learning; convolutional neural network; generative adversarial network

1. Introduction

Due to the rapid development of internet technology and social media in recent years, many business models have gradually changed. Digital images have become one of the main media for information transmission. To protect the intellectual property rights and copyrights of the owners, digital watermarking is an indispensable technology.

So-called “digital watermarking” refers to the hiding of secret information in digital media. Digital media can be images, videos, text, or audio, etc. The most common application is to embed trademarks in digital media to protect intellectual property. In 1994, Schyndel et al. [1] proposed the term digital watermark and published a paper on undetectable digital watermark technology. Information hiding emphasizes the hiding of secret information in the carrier, making it difficult for human eyes to detect. In 1996, in the First International Workshop on Information Hiding [2], relevant terms for information-hiding technology were defined. In 1999, Petitcolas et al. [3] classified the structure of

information-hiding technology. Information-hiding technology can be subdivided into steganography and watermarking. The goal of steganography is the same as cryptography, which is to protect secret information from being stolen by third parties. Watermarking is used in copyright protection, image authentication, medical, military, fingerprint, and other purposes. Steganography and watermarking are very similar in structure. They both hide secret information in digital media. However, the two technologies have different characteristics. Information-hiding technology has three basic attributes: capacity, security, and robustness [4]. The capacity indicates how much information a digital carrier can embed; security usually refers to its undetectability, and secret information will not be detected by statistical methods or steganalysis techniques; robustness indicates that the image can withstand various image-processing attacks, such as crop, scaling, rotation, noise, blur, compression, etc. In addition, for information-hiding technology, the ability to have imperceptibility is the top priority, and imperceptibility refers to the inability of human eyes to perceive secret information. Steganography focuses more on the imperceptibility, security, and capacity of secret information, while most watermarks prioritize their robustness.

With the vigorous development and popularization of social media, huge amounts of image data are transmitted on the internet every day. According to Google's statistics in 2020, about 4.3 million GB of data are uploaded to Gmail, Google Drive, Google Photos, and other services every day.

To meet the demand for storage space and maintain the efficiency of data transmission, image compression is a very important and indispensable technology. JPEG compression [5] is the most common image compression technology in the network transmission process, which can effectively reduce the capacity of digital images and maintain good image quality. However, this important technology, which is indispensable on the internet, is a big problem for digital watermarking.

The basic standard technology of JPEG is based on discrete cosine transform (DCT) to compress images. The method involves converting the image into a DCT matrix and determining the coefficients through a pre-designed quantization table. Therefore, traditional solutions mostly use DCT conversion, adopting a DCT watermarking approach based purely on psychovisual thresholds to minimize distortion and enhance robustness. By contrast, our GAN-based approach uses deep learning to increase adaptability and resistance to attacks [6]. DCT is used to transform and segment the scrambled watermark into small portions to embed [7]. DCT ensures watermark insertion in the mid-band of co-efficient robustness [8]. Using DCT, LWT sub-bands are transformed to prepare them for watermark embedding with better perceptibility [9]. DWT applies DWT to the RGB contributions of the cover image, wherein watermark coefficients are embedded in DWT bands for robust resilience [7]. DWT uses approximation coefficients of one block as a watermark by embedding them into the detail coefficients of another block, which improves recovery and enhancements during tampering detection [10,11]. DFT uses magnitudes that result in high imperceptibility of the watermarking [8] and other frequency domains to design algorithms to overcome the high-frequency information lost by lossy compression. Symmetry is essential in this approach, especially when using DCT in the embedding process. The symmetrical nature of DCT ensures that watermark information is evenly dispersed across the image, hence boosting resistance to lossy compression. Symmetry also enhances alignment between the embedder and decoder, which is necessary to keep the embedded watermark intact. The suggested watermarking technique is more imperceptible and robust to image distortion since it preserves symmetrical characteristics during embedding and decoding. In recent years, there have been many watermarking techniques using neural networks [12,13]. These methods use neural networks to replace traditional embedding and extraction algorithms to improve the imperceptibility of watermarks, but most of them

are non-blind watermarks, which are not suitable for practical applications. Therefore, since 2017, many scholars have proposed blind watermarking technology based on deep neural networks [14–18], which is more in line with reality, and uses various methods to train the resist image processing and steganalysis.

The watermarking method is based on a deep neural network designed to replace the traditional embedding and extracting algorithm to improve the imperceptibility of the watermark and improve the watermark's robustness by simulating the image-processing attacks. Although these methods show good performance, particularly in terms of robustness against attacks and imperceptibility of watermarks, their effectiveness is measured using deep learning-based evaluation metrics such as the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Bit Error Rate (BER). These metrics allow for a quantitative assessment of the robustness and accuracy of watermark recovery. On the other hand, these practices will make the overall network architecture larger and require a huge amount of training time.

The purpose of this research is to develop a digital watermarking mechanism based on deep learning technology to protect the copyright of JPEG-compressed images. This paper proposes a watermarking method based on deep convolutional generative adversarial networks (DCGANs) [19], which decodes the JPEG-compressed image and restores (retrieves) the watermark using the generator of DCGANs.

This study is aimed at defeating all the present watermarking techniques by designing a strong semi-blind watermarking method applicable for JPEG lossy compression. Specific objectives include:

- a. Designing embedding and decoding watermarking through a deep convolutional generative adversarial network (DCGAN).
- b. Introducing a generator into the framework for each image watermark pair to gain robustness and flexibility in the approach.
- c. Embedding a secret key mechanism, so that authorization is provisioned for a model not to be exploited without permission.
- d. Providing a framework that allows exclusive model training for specific applications and enhances manageability.
- e. Providing excellent robustness based on various quality factors, as validated by PSNR, SSIM, and BER metrics.

The rest of the paper is organized as follows: the proposed method is elaborated in Section 2. Section 3 demonstrates the evaluation results. The conclusions and future work are presented in Section 4.

1.1. Related Works

1.1.1. Generative Adversarial Networks

Deep learning has been widely used in fields such as classification, facial recognition, and medical imaging. With the development of deep learning, many scholars have proposed various algorithms based on CNN. Recently, Gao et al. (2024) [20] proposed a framework based on a generative adversarial network (RIW-GAN) for robust image watermarking. The encoder network comprises convolutional layers, and a residual block generates an encoded image that closely resembles the original with minimal distortion. A simulated noise layer, used as a differentiable network layer, enables end-to-end training before decoding, enhancing resilience against attacks. Experimental results demonstrate that RIW-GAN surpasses state-of-the-art models in terms of invisibility and robustness, effectively withstanding standard assaults like JPEG compression and geometric attacks such as resizing and cropping, thereby establishing its superiority in robust image watermarking. In 2020, Hao et al. [21] proposed a generative adversarial network-based robust

image watermark algorithm. The model comprises generator and adversary modules. The generator creates watermarked images and decodes noise-damaged images to retrieve the watermark, while the adversary identifies the presence of a watermark and introduces noise. A high-pass filter is applied before the discriminator, embedding the watermark in the mid-frequency image region based on the Hidden model. To calculate loss, a higher weight is assigned to the image center and a lower weight to the edges, aligning with the human visual system. Experimental results show superior visual performance and robustness against noise interference. Goodfellow et al. (2014) [22] proposed the architecture of generative adversarial networks (GAN). A GAN is an unsupervised learning method whose architecture consists of a generator and a discriminator. The generator network takes random noise as input, and its output generated pictures should be as similar as possible to the real samples provided by the training set. The input of the discriminator network is the real sample and the output of the generating network, and its purpose is to distinguish the output of the generator network from the real sample as much as possible. The two networks are trained against each other, and the ultimate goal is to prevent the discriminator network from correctly judging whether the generated image is true as much as possible.

In recent years, many GAN-based architectures have been proposed and applied in many fields. The original GAN only needs a random noise as input to start training. In other words, the appearance of the generated image is completely determined by the random noise. The entire training process is too free and uncontrollable. To solve this problem, Mirza et al. (2014) [23] proposed the conditional GAN, which uses random noise and a piece of additional information as input at the same time. The additional information can be categories, images, text, etc. This change makes the training process controllable. In other words, a conditional GAN transforms GAN from unsupervised learning to supervised learning, allowing researchers to better control the training process. In 2015, Radford et al. (2015) proposed deep convolutional generative adversarial networks (DCGANs) [19], which introduced convolutional neural networks (CNN) into the GAN architecture and improved the stability of training and the quality of the generated images. In the DCGAN architecture, the generator uses transposed convolution to replace the fully connected layer, and the discriminator uses the convolution layer to improve the image feature extraction. The discriminator uses Leaky ReLU instead of ReLU to avoid the gradient being too sparse (Dead ReLU Problem). Although DCGANs improve training stability and image quality, they still do not solve the two problems of GAN: the vanishing gradient problem and model collapse. Arjovsky et al. (2017) [24] proposed WGAN, which made four simple changes under the original GAN architecture: the last layer of the discriminator cancels the sigmoid function, uses Wasserstein divergence to replace JS divergence, and establishes a fixed constant to limit the update range of the parameters of the discriminator, and RMSProp is used by the optimizer. These changes successfully solve the problem of instability in the original GAN training and ensure the diversity of the generated image.

1.1.2. Watermarking Technology Based on Deep Learning

In 2023, Hu et al. [25] proposed a new blind digital watermarking method for color images: a downsized color image embedded into a host color image. The watermarking procedure is performed by dividing the host image into blocks of eight pixels non-overlappingly, performing discrete cosine transform (DCT) for each block of each channel, and manipulating the magnitudes of three designated DCT coefficients using the appropriate minimization constraint. Experimental results have shown that the watermarked image produced by this watermarking method outperformed four other methods based on zero-normalized cross-correlation (ZNCC), a metric that measures the similarity between

the original and retrieved watermarks, with values closer to one indicating higher accuracy. Furthermore, watermarks were still at a fair level of imperceptibility, as proved by the various measures of peak signal-to-noise ratio and mean structural similarity. This has accompanied the new paradigm in color image watermarking—the adoption of deep residual networks in the denoising and resolution enhancement of retrieved watermarks. Overall, more than 0.88 ZNCC levels are achieved by the residual network in super-resolving the watermark images by a factor of sixteen. Ahmadi et al. [17] proposed a watermarking method (ReDMark) based on a convolutional neural network. After training, the network can provide a blind watermark for grayscale images. The method includes three modules: Embedding, Attack Layer, and Extraction. The structure of Embedding and Extraction is fully convolutional networks (FCN), and the Attack Layer simulates various image-processing attacks to improve the model’s robustness. In addition, the method also introduces a strength factor to control the balance between imperceptibility and robustness. According to the experimental results, the method has good imperceptibility and robustness, but in the case of $\sigma = 25$ Gaussian blur, where σ represents the standard deviation, with higher values indicating stronger blurring effects, 10% salt-and-pepper noise, and 20% cropping, the robustness is poor.

In 2020, Zhong et al. [18] introduced a watermarking method designed to achieve both robustness and blindness. This approach leverages the fitting capabilities of deep neural networks to learn and encapsulate the watermarking algorithm, training the network in an unsupervised manner. The architecture features a one-way deep neural network comprising five key modules: Encoder, Embedder, Invariance, Extractor, and Decoder. Among them, Invariance is the key factor for the robustness of the method, and the fully connected layer is used to map the feature to another space to improve the tolerance for errors. According to the experimental results, this method has good imperceptibility and good robustness for most image-processing attacks. It is only inferior in certain severe distortion situations, such as 65% cropping, a compression ratio of quality factor 10, and 90% salt-and-pepper noise.

The comparative analysis of prior methods and proposed work is shown in Table 1.

Table 1. Comparative Analysis of Prior Methods and Proposed Work.

Method	Key Features	Limitations	How Our Work Addresses These Limitations
Gao et al. (2024) [20]	RIW-GAN framework; uses convolutional layers and residual blocks for encoding.	Focuses on robustness but may lack flexibility in handling diverse image distortions.	Our method uses DCGANs with a dedicated generator for each watermark, ensuring adaptability to diverse distortions.
Hao et al. (2020) [21]	GAN-based robust watermarking; uses a high-pass filter and weighted loss.	Limited robustness under severe noise interference.	Our method achieves better robustness under noise and JPEG compression, with low BER (≤ 0.008).
Goodfellow et al. (2014) [22]	Original GAN architecture; unsupervised learning with generator and discriminator.	Suffers from instability, vanishing gradients, and model collapse.	Our method uses DCGANs with Leaky ReLU and Wasserstein divergence, improving stability and robustness.
Mirza et al. (2014) [23]	Conditional GAN; uses additional information (e.g., categories, images) for control.	Requires additional input information, which may limit flexibility.	Our method does not require additional input information, making it more flexible and easier to implement.
Radford et al. (2015) [19]	DCGANs; introduces CNN into GAN architecture for better image quality.	Still faces issues with vanishing gradients and model collapse.	Our method uses DCGANs with transposed convolution and Leaky ReLU, improving training stability.

Table 1. Cont.

Method	Key Features	Limitations	How Our Work Addresses These Limitations
Arjovsky et al. (2017) [24]	WGAN; uses Wasserstein divergence and RMSProp for stable training.	Requires careful tuning of hyperparameters for optimal performance.	Our method simplifies training by focusing on JPEG compression robustness and using a secret key for security.
Hu et al. (2023) [25]	Blind watermarking for color images; uses DCT and deep residual networks.	Limited robustness under severe distortions (e.g., cropping, noise).	Our method achieves better imperceptibility (PSNR > 49 dB, SSIM $\approx$ 0.95) and robustness under distortions.
Ahmadi et al. (2020) [17]	ReDMark; uses FCN for embedding and extraction, with an attack layer.	Poor performance under severe distortions (e.g., Gaussian blur, salt-and-pepper noise, cropping).	Our method outperforms ReDMark in robustness to JPEG compression, with low BER (≤ 0.008).
Zhong et al. (2020) [18]	Deep neural network-based method; uses Encoder, Embedder, and Decoder modules.	Inferior performance under severe distortions (e.g., 65% cropping, 90% salt-and-pepper noise).	Our method maintains low BER (≤ 0.008) and high PSNR (>49 dB) even under severe distortions.
Proposed Method	DCGAN-based semi-blind watermarking; uses embedder, decoder, and generator.	Focused on JPEG compression; future work is needed for other attacks.	Our method outperforms prior works in robustness to JPEG compression, with low computational cost and flexible training.

2. Methods

Robust watermarks can protect digital multimedia copyrights to prevent or prevent unauthorized copying and copying of digital media. To improve the ability of watermarks to resist JPEG compression, this paper proposes an invisible watermarking method based on deep convolutional generative adversarial networks. The architecture of the method, as shown in Figure 1, includes two modules, the Embedder and Decoder, and two deep neural networks: Generator G and Discriminator D , and the proposed method is to train a unique generator for the multiple protected images, and one generator corresponds to one watermark.

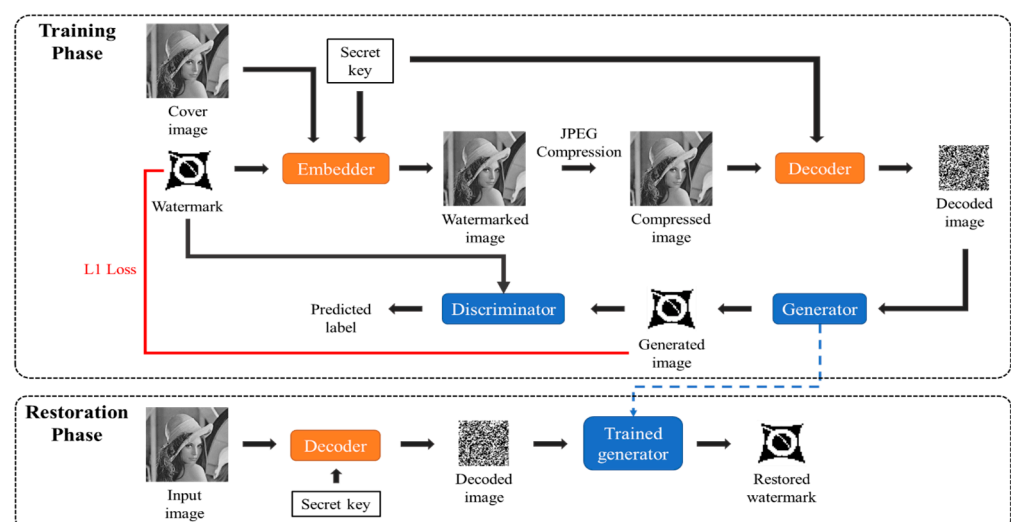


Figure 1. The framework diagram of the proposed model.

2.1. Training Phase

At this phase, a watermark is embedded into the original image through an embedder to produce a watermarked image. The watermarked image is compressed by JPEG to obtain a compressed image after JPEG compression. At this point, a decoder is used to extract the decoded image from the compressed image, that is, the corrupted watermark. The decoded image will be used as the input image of the deep convolutional generative adversarial networks. Generator G generates a generated image similar to the original watermark, and discriminator D inputs the generated image and the original watermark at the same time, and judges whether the generated image is similar to the original watermark. The ultimate goal of DCGANs is to enable generator G to restore the decoded image to an image similar to the original watermark.

2.2. Dataset Details

In the proposed study, 100 grayscale images (256×256) from the USC-SIPI image dataset were used as the authors' originals during the training phase. The watermark was a binary image of the logo of National Chung Hsing University (64×64). The test phase was carried out on 834 gray (256×256) images from the Microsoft COCO dataset and 10 binary images (64×64) from the CIFAR image dataset. All the images were resized to a uniform dimension. The pixel values were normalized to improve the stability of training.

Under the experiments, a concealed image of size 256 by 256 was cut into 16 blocks of 64 by 64 each to embed the same watermark in each block. This way, in case a protected image is cropped, some blocks of watermark can remain recoverable.

2.3. Experimental Setup

This research method was mainly written in Python 3.6, the deep learning framework used PyTorch 1.4.0, and the experiments were carried out on a Windows 10–64-bit system equipped with i7-9700 3.00 GHz CPU, 16 GB RAM, and RTX 2060 SUPER. The USC-SIPI [26] image data set and the open data set Microsoft COCO dataset [27] were mainly used as the carrier of the watermark. The six images in each of Figures 2 and 3 were taken from the USC-SIPI [26] image dataset and the open Microsoft COCO dataset [27].



Figure 2. Part of the image of the USC-SIPI image dataset. (a) Lena, (b) baboon, (c) horse, (d) F16, (e) peppers, (f) Sailboat.



Figure 3. Part of the image of the Microsoft COCO dataset. (a) Zebra, (b) boat, (c) plane, (d) cat, (e) motorcycle, (f) old man.

The watermark originated from the logo of National Chung Hsing University and the open dataset CIFAR-10 [28] had undergone binarization, which is shown in Figure 4. A key secret was used in both the embedder and decoder, which was generated by random numbers.

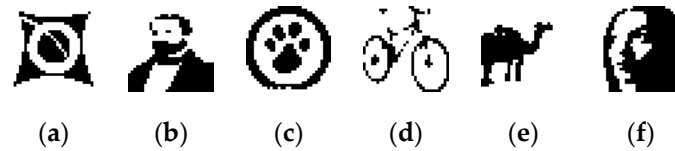


Figure 4. Part of the binary image of National Chung Hsing University LOGO and CIFAR. (a) NCHU logo, (b) Man, (c) paw, (d) bike, (e) camel, (f) women.

The learning rate was set to 0.0002, and the Adam optimizer was adopted as it provides a compromise between speed of convergence and stability in training. Empirically, a batch size of 32 was found to be a good compromise for optimal generation with efficient memory utilization while avoiding overfitting.

2.4. Evaluation Metrics

This experiment uses three image-quality evaluation indicators to evaluate the performance of the method—peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and the bit error rate (BER)—and uses the Confusion Matrix to evaluate the effect and accuracy of the model. The detailed definition of each indicator is as follows.

Peak signal-to-noise ratio (PSNR) is a measure of image quality. The larger the ratio calculated by PSNR, the closer the copied image is to the input image, and the image quality is good. The typical PSNR value is between 30 dB and 40 dB. The higher the value, the lower the distortion. The PSNR value is defined as Equation (1):

$$PSNR = 10 \times \log \frac{255^2}{MSE} \quad (1)$$

where $MSE = \frac{1}{H \times W} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [I(i, j) - K(i, j)]^2$

where H and W are the width and height of the image, respectively, $I(i, j)$ represents the pixel value of the original image at position (i, j) , and $K(i, j)$ represents the pixel value of the original image at (i, j) .

Structural similarity (SSIM) is an index that measures the similarity of two images. When the SSIM value is 1, it means that the two images are exactly the same. Generally speaking, the SSIM value must be at least 0.95 for the image quality to be acceptable. *SSIM* is defined as Equation (2):

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (2)$$

where x and y stand for the original image and the restored (retrieved) image, respectively, μ_x and μ_y are the average values of x and y , σ_{xy} is the covariance of the sum, σ_x^2 and σ_y^2 are the variation of x and y , respectively, and C_1 and C_2 are constants.

The bit error rate (Bit Error Rate) represents the error rate of the restored (retrieved) image. The lower the error rate, the more similar the restored image is to the original image. *BER* is defined as Equation (3):

$$BER = \frac{N_e}{H \times W} \quad (3)$$

where H and W are the length and width of the image, and N_e is the number of bits to restore the error.

The confusion matrix is an index to measure the accuracy of the classification model. The confusion matrix has two dimensions: Actual condition and Predicted condition, and each has two situations: Positive and Negative. Therefore, the following four indicators can be obtained:

True positive (*TP*): Represents that the real situation is true, and the model prediction is true.

False positive (*FP*): Represents that the real situation is not true, and the model prediction is true.

False negative (*FN*): Represents that the real situation is true, and the model prediction is not true.

True negative (*TN*): Represents that the real situation is not true, and the model predicts it is not true.

For the classification model, the more accurate the prediction performance, the better. Therefore, corresponding to the confusion matrix, the higher the number of *TP* and *TN*, the better, and the smaller the number of *FP* and *FN*, the better. With the above four indicators, four other advanced indicators can be calculated accordingly: Accuracy, Precision, Recall, and F1 Score. The definitions of the four indicators are as shown in Equations (4)–(7). Accuracy represents the proportion of correct judgments among the results predicted by all models. Precision represents the proportion of the true situation in the model judged to be true. The Recall rate represents the proportion of all the results that are judged to be true by the model. The F1 Score combines precision and recall. The closer the value of this index is to 1, the better the model performance, and the closer the value is to 0, the worse the model performance. In Equation (7), *P* stands for Precision and *R* stands for Recall. These four indicators can convert the results in the confusion matrix to a ratio between 0 and 1, to better measure the effectiveness of the model.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

$$P = \frac{TP}{TP + FP} \quad (5)$$

$$R = \frac{TP}{TP + FN} \quad (6)$$

$$\text{F1Score} = \frac{2PR}{P + R} \quad (7)$$

Metrics such as PSNR, SSIM, and BER are very important for assessing the watermark robustness and its imperceptibility. PSNR indicates how nearly identical the original image and watermarked image are; higher values thus indicate less distortion and better imperceptibility. SSIM quantifies the structural image similarity and indicates the perceptual quality of the watermarked image in comparison; values of SSIM close to 1 indicate high visual fidelity. BER is a measure of the error rate in rescuing the watermark, where lower values mean greater strength against attacks. The metrics are thus credibly comprehensive in evaluating the balance between invisibility and resilience.

2.5. Embedder and Decoder

The main purpose of the embedder is to embed a binary watermark with a size of $H \times W$ into a grayscale image. The embedding method uses the simplest and most well-known LSB substitution method and randomly generates a set of secret keys with a size of $H \times W$ pixels, and each value ranges from 1~4, where each of these values can be used to specify that each secret bit of information is hidden in the LSB position of the grayscale image pixel value. To explain the embedding algorithm in more detail, we define the following symbols. Original image $I = \{p_1, p_2, \dots, p_n | 0 \leq p_i \leq 255, p_i \in N\}$, Secret key $K = \{k_1, k_2, \dots, k_n | k_i \in \{1, 2, 3, 4\}\}$, Watermark bit $M = \{m_1, m_2, \dots, m_n | m_i \in \{0, 1\}\}$, where n is the number of pixels in the image, which is $H \times W$. This simplicity enables

effective watermarking without compromising pixel values, which is crucial to an image's quality. The detailed algorithm process is as follows:

- Step 1. Convert each pixel value p_i of the original image into a binary value b_i .
- Step 2. The secret key value k_i indicates that the watermark bit m_i is embedded in the position of the k_i th least significant bit (LSB) of the binary pixel value b_i by LSB substitution to obtain a new binary value b'_i .
- Step 3. Convert the binary value b'_i to the decimal value p'_i , which is the pixel value after embedding the watermark bits.
- Step 4. Repeat steps 1 to 3 until the entire image is processed, and the image I' embedded with the watermark is obtained.

An example is shown in Figure 5 to illustrate the detailed embedding method. Assuming an original grayscale image with a size of 4×4 pixels, the length of a secret message is 16 bits "1111 1111 0011 0011". Suppose there is a secret key generated by random numbers, its size is also 16 values, and the value range is {1, 2, 3, 4}. The first pixel value in the upper-left corner of the original grayscale image is 202, which can be converted into a binary 8-bit "11001010". The value of the first secret key in Figure 5 is 6, indicating that a secret bit of information can be embedded in the third LSB of the corresponding pixel value. Therefore, after embedding the secret bit 1 in the corresponding position, the binary representation of the new pixel value is "11001110", which is converted to 206 in decimals. The above steps are repeated until the entire grayscale image is processed, and the image with the embedded watermark can be obtained.

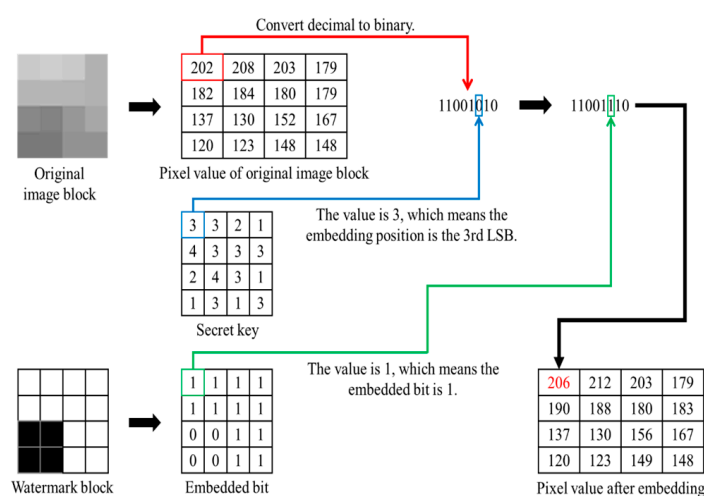


Figure 5. Example of embedder.

The main purpose of the decoder is to extract a binary image with a size of $H \times W$ from a JPEG-compressed grayscale image. Because the grayscale image is compressed by JPEG, the previously embedded watermark has been destroyed. The extraction method follows the LSB substitution method. The decoder and the embedder share the same secret key, which can be used as a basis for extracting the secret message. To explain the decoder algorithm in more detail, we have defined the following symbols. Given the input JPEG image $C = (c_1, c_2, \dots, c_n | 0 \leq c_i \leq 255, c_i \in N)$, and the secret key $K = \{k_1, k_2, \dots, k_n | k_i \in \{1, 2, 3, 4\}\}$, where $n = H \times W$, the decoding process first converts the pixel value c_i of the input image into a binary value b_i and then extracts the bit m_i which is the k_i th LSB of the value b_i . This is repeated to process the entire image until the decoded image is obtained. An example is shown in Figure 6 to illustrate the detailed extraction method.

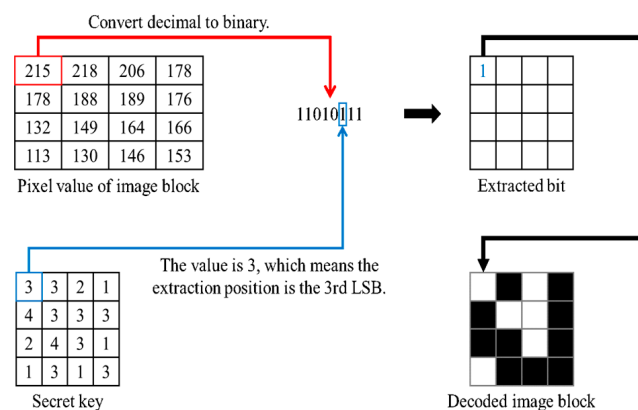


Figure 6. Example of decoder.

Suppose a compressed grayscale image with a size of 4×4 and the same secret key is used as the embedding process, the secret key size is 16 values, and the range of values is $\{1, 2, 3, 4\}$. The value of the first pixel in the upper-left corner of the grayscale image is 215, which can be converted into a binary 8-bit “11010111”. The value of the first secret key is 3, indicating that a secret bit of information should be extracted in the third LSB of the corresponding pixel value. Therefore, after the secret bit 1 is taken out, the pixel corresponding to the extracted image is white. The above steps are repeated until the entire image has been processed, and the decoded binary image can be obtained. This method ensures that the embedded watermark bits can be reliably recovered by the decoder after lossy JPEG compression, further evidencing the robustness of the proposed framework.

2.6. Generator and Discriminator Networks

The generator network architecture used in this paper is shown in Figure 7, which is different from the random noise used in the original DCGANs. The input image is a decoded binary image with a size of $H \times W \times 1$. After three layers of Conv after the block, an $8 \times 8 \times 128$ feature vector is generated, and then through three layers of ConvT Block, an image of $128 \times 128 \times 3$ is output. Conv Block contains a Batch Normalization (BN) layer, a convolutional layer with a kernel size of 4×4 , and a LeakyReLU activation function; ConvT Block includes a BN layer, a transposed convolutional layer with a kernel size of 4×4 , and a LeakyReLU activation function. Finally, the feature vector passes through a layer of the tanh activation function and outputs the generated image.

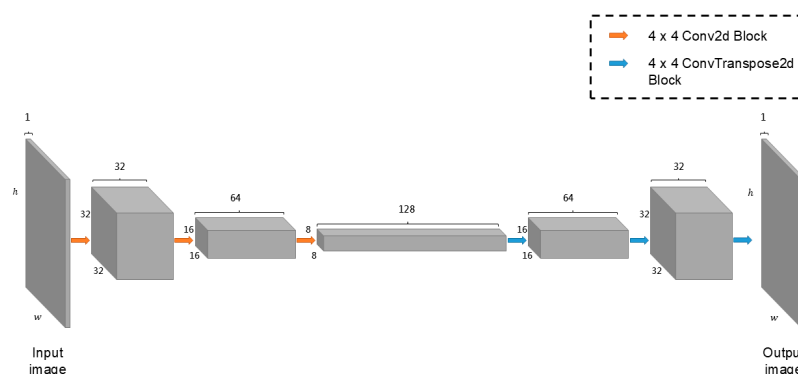


Figure 7. Generator architecture diagram.

The discriminator architecture of this study is similar to the original DCGAN discriminator. The discriminator architecture is shown in Figure 8. The input image size is $H \times W \times 1$, including three Conv Block 1 and one Conv Block 2. Conv Block 1 contains one Batch Normalization (BN) layer, a convolutional layer with a kernel size of 4×4 , and

the LeakyReLU activation function. Conv Block 2 contains a convolutional layer with a kernel size of 4×4 and the sigmoid activation function. The final output is the probability of whether the generated image is true.

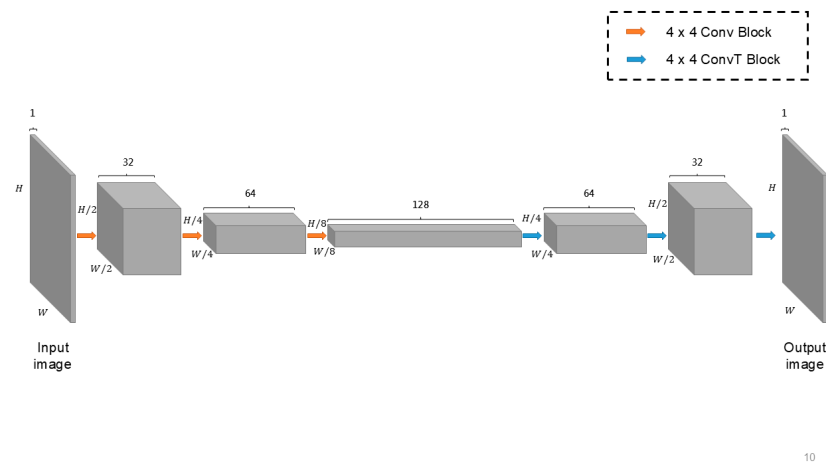


Figure 8. Discriminator architecture diagram.

In the training phase, the loss function used in this study is shown in Equations (8)–(10). Equation (8) represents the loss function L_D of the discriminator D , and its purpose is to maximize the value of the function. The closer the value of the real sample is to 1, the better, and the closer the value of the generated sample is to 0, the better, where x represents the real sample, $D(x)$ represents the value that the discriminator judges for the real sample, y represents the input sample, $G(y)$ represents the image generated by the generator with the input sample, and $D(G(y))$ represents the value that the discriminator judges for the generated image.

$$L_D = E_{x \sim P_{data}(x)} [-\log D(x)] + E_{y \sim P_{data}(y)} [-\log(1 - D(G(y)))] \quad (8)$$

Equation (9) represents the loss function L_G of generator G . Its purpose is to maximize the value of the function. We hope that generator G will eventually produce an image that can fool discriminator D . Therefore, the closer the value of the generated sample is to 1, the better; where y represents the input sample, $G(y)$ represents the image generated by the generator with the input sample, and $D(G(y))$ represents the value that the discriminator judges for the generated image.

$$L_G = E_{y \sim P_{data}(y)} [-\log D(G(y))] \quad (9)$$

In the method proposed in this paper, it is hoped that the generated image will be as similar to the original watermark as possible, so an additional loss function L_w is added, as shown in Equation (10). We hope that the generated image will be similar to the real sample at the pixelwise, so the loss function uses L1 Loss, where n is the number of pixels in the image, $G(y)_i$ represents the i -th pixel of the generated image, and x_i represents the i -th pixel of real sample.

$$L_w = \frac{1}{n} \sum_{i=1}^n |G(y)_i - x_i| \quad (10)$$

The overall loss function L_{total} is shown in Equation (11), which includes the loss function of the discriminator, the loss function of generator G , and the newly added loss function.

$$L_{total} = L_D + L_G + L_w \quad (11)$$

2.7. Restoration Phase

In the training phase, after the training of the deep convolutional generative adversarial networks is completed, a trained generator G will be obtained, which can restore the specific binary image to approximate the original watermark. To successfully restore the watermark, it is necessary to store the secret key used in the initial training phase and the model parameters completed during the training to facilitate the restoration phase.

In the restoration stage, the received image is first input to the decoder, a specific binary image is extracted through the decoder and secret key, then the trained model parameters are read, and the generator G is used to obtain the restored (retrieved) watermark.

3. Results

The proposed method of semi-blind watermarking using deep convolutional generative adversarial networks (DCGANs) showed very good performance at different JPEG quality factors with respect to imperceptibility and fidelity. It scored above 49 dB PSNR, SSIM around 0.95, and BER below 0.008, indicating that the watermark was very resistant to JPEG compression and achieved high fidelity in watermark restoration. Security was ensured with a secret key for unauthorized access or duplication. The proposed method maintained its performance under various quality factor tests (10 to 90), the number of epochs for training, and achieved convergence within 250 epochs. The method outperformed recent deep-learning-based techniques, especially at very low QFs (like 10) with a BER of 0.008. Furthermore, the DSC decreased the number of parameters of the model by 50 percent (from 1.27 MB to 0.69 MB) and was shown to be sufficiently robust with PSNR: ~48–49 dB, BER: 0.01–0.014, while a slightly reduced SSIM was noticed (0.95 to ~0.90). These results make it clear that it is a good technique in terms of robustness, imperceptibility, and security; it is promising for real-world applications aimed at digital image safeguarding.

3.1. Comparison of Different Quality Factors and Training Times

To explore whether the method designed by this study can restore the watermark embedded in a watermarked image after JPEG compression, the experiment in this section will use different quality factors to observe the image quality evaluation indicators under different training times, and discuss how to tune the model parameters based on the results. The results of the experiment are shown in Table 2, which shows the values of various evaluation indexes of the original watermark and the watermark generated by the model under different quality factors and training times.

In the case of a quality factor of 50, when the number of training epochs is 50 and 100, the PSNR values of the restored (retrieved) watermark generated by the model are approximately 40.87 dB and 44.06 dB, respectively. When the number of training epochs rises to more than 250, the PSNR value of the restored (retrieved) watermark generated by the model increases significantly and remains at about 47.5–51.57 dB. However, a closer observation reveals that when the Epoch is 200, the PSNR of the restored (retrieved) watermark generated by the model is only about 42.75 dB, which shows that the PSNR does not continue to increase as the number of training epochs increases. SSIM and BER have similar results. When we observe the compression of different quality factors (quality factor = 10, 20, 30, ..., 90), we will find that there are similar results regardless of the quality factor. In other words, the model has not yet converged when the Epoch is 50–100, but when the number of training epochs increases to 200 or more than 250, the generator and discriminator models have already converged, so the loss function of the model will not continue to decline. However, it will remain near a certain value; even if the Epoch is increased, the loss function value of the model will only make a small difference.

Table 2. PSNR, SSIM, and BER of the original watermark and the restored (extracted) watermark under different quality factors and training times.

Epoch	QF = 10			QF = 20			QF = 30		
	PSNR	SSIM	BER	PSNR	SSIM	BER	PSNR	SSIM	BER
50	41.890	0.851	0.028	43.776	0.909	0.015	43.108	0.897	0.017
100	41.269	0.800	0.041	47.032	0.951	0.007	44.480	0.907	0.016
150	44.561	0.904	0.016	48.359	0.963	0.005	50.405	0.979	0.003
200	44.528	0.878	0.022	47.667	0.957	0.006	49.001	0.967	0.004
250	45.618	0.903	0.017	46.887	0.941	0.009	49.421	0.966	0.005
300	46.063	0.902	0.017	45.761	0.924	0.011	47.543	0.938	0.009
350	48.839	0.938	0.010	46.333	0.929	0.011	45.581	0.906	0.016
400	49.158	0.931	0.012	47.853	0.949	0.008	51.324	0.970	0.003
450	49.093	0.922	0.014	49.013	0.960	0.006	50.477	0.965	0.004
500	51.756	0.947	0.008	49.037	0.958	0.006	49.071	0.954	0.007
Epoch	QF = 40			QF = 50			QF = 60		
	PSNR	SSIM	BER	PSNR	SSIM	BER	PSNR	SSIM	BER
50	43.807	0.914	0.013	40.872	0.816	0.035	41.910	0.873	0.022
100	47.251	0.954	0.007	44.068	0.921	0.012	45.728	0.943	0.009
150	47.426	0.956	0.006	47.052	0.960	0.006	46.634	0.955	0.006
200	48.406	0.957	0.006	42.758	0.821	0.033	47.104	0.952	0.007
250	49.685	0.969	0.004	47.458	0.956	0.007	47.432	0.950	0.007
300	47.377	0.938	0.009	50.390	0.965	0.005	48.765	0.958	0.005
350	49.733	0.968	0.005	51.071	0.980	0.003	49.493	0.967	0.004
400	48.341	0.947	0.009	49.079	0.956	0.007	49.987	0.968	0.005
450	57.996	0.993	0.001	47.769	0.944	0.009	44.868	0.884	0.022
500	49.348	0.959	0.006	51.572	0.976	0.003	49.764	0.965	0.005
Epoch	QF = 70			QF = 80			QF = 90		
	PSNR	SSIM	BER	PSNR	SSIM	BER	PSNR	SSIM	BER
50	41.306	0.815	0.034	42.374	0.849	0.028	42.151	0.843	0.035
100	44.462	0.909	0.014	43.210	0.879	0.021	46.328	0.876	0.027
150	46.934	0.938	0.011	43.846	0.880	0.021	49.362	0.887	0.023
200	47.048	0.947	0.008	46.301	0.924	0.012	51.921	0.900	0.020
250	50.341	0.970	0.004	47.243	0.938	0.010	55.811	0.929	0.014
300	47.558	0.948	0.008	47.407	0.939	0.010	57.215	0.925	0.015
350	47.795	0.937	0.010	46.904	0.925	0.012	59.367	0.930	0.014
400	46.051	0.898	0.017	49.876	0.955	0.006	60.614	0.937	0.012
450	46.538	0.912	0.014	50.767	0.966	0.005	62.908	0.940	0.012
500	50.038	0.955	0.006	50.101	0.948	0.007	64.150	0.948	0.010

On the whole, the method proposed in this paper is robust to any quality factor. Among them, the watermark generated by the trained model is compared with the original watermark, the average PSNR can reach 49.99 dB, the average SSIM reaches 0.95, and the average BER can be as low as 0.008. Therefore, the method proposed in this paper sets different quality factors in the training phase and optimizes the parameters for different training times. These optimal parameters will be stored as model parameters for restoring (retrieving) the watermark. That is, when a picture with a hidden watermark is compressed by JPEG, the DCGAN model in the method designed by us is capable of generating the watermark. Moreover, the generated watermark is quite similar to the original watermark.

3.2. The Model's Ability to Restore (Retrieve) Watermarks in Different Situations

To verify the safety of this research method, in the experimental design of this section we mainly explore whether the trained model can only restore (retrieve) the watermark for

the image with the watermark embedded at the beginning, and whether, in other situations, the original hidden watermark should not be restored (retrieved). We assume the following 6 situations:

- Situation 1. The same original image and the same watermark used in the training phase are also embedded with the same secret key.
- Situation 2. The same original image and the same watermark are used in the training phase, but using a different secret key to embed.
- Situation 3. The same original image and the same secret key are used in the training phase but with a different watermark embedded.
- Situation 4. The same original image is used in the training phase, but no watermark is embedded.
- Situation 5. The original image is different from the one used in the training phase, but the same secret key is used and the same watermark is embedded.
- Situation 6. The original image is different from the one used in the training phase, and no watermark is embedded.

The visualization results of the experimental settings in 6 cases are presented in Figure 9. Only in situation 1 will the trained model generate a restored watermark that is sufficiently similar to the original watermark. In the other cases (situations 2–6), the original watermark cannot be restored. Even if it is seen by human eyes, it has an outline similar to the original watermark, but the difference in the details is still very large.

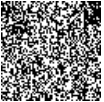
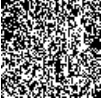
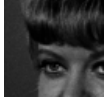
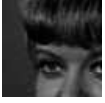
Test situation	Original image	Watermarked image	Compressed image (QF=50)	Decoded image	Restored watermark
Situation 1					
Situation 2					
Situation 3					
Situation 4		N/A			
Situation 5					
Situation 6		N/A			

Figure 9. Visualized results of restored (retrieved) watermarks in different situations.

Table 2 shows the PSNR, SSIM, or BER values of the restored (retrieved) watermark in different situations compared to the original watermark. The watermark restored (retrieved) by the model we designed in Situation 1 is indeed significantly different from the watermark restored (retrieved) by the model in the other situations (Situations 2–6). The PSNR value of Situation 1 in Table 3 is 49.832 dB, and the PSNR value of the other situations (Situations 2–6) is approximately 38.5 dB; the SSIM of Situation 1 in Table 3 is as high as 0.974, while the SSIM of the other situations (Situations 2–6) is only about 0.3 to 0.4. In Table 3, the BER of Situation 1 is 0.003, and the BER of the other situations (Situations 2–6) is about 0.2. In addition, Situation 2 used different secret keys to train the generator network, but the watermark could not be restored (retrieved) correctly. This experiment also proved the importance of using the correct secret key. Even if the network model and watermark are stolen by a third party, as long as the secret key is protected, the third party cannot copy a similar watermark. Therefore, the secret key can greatly enhance the security of this research method.

Table 3. The evaluation indicators of the original watermark and the restored (retrieved) watermark in different situations.

Test Situation	PSNR	SSIM	BER
Situation 1	49.556	0.962	0.005
Situation 2	38.586	0.391	0.188
Situation 3	38.464	0.364	0.199
Situation 4	38.453	0.355	0.202
Situation 5	38.602	0.394	0.187
Situation 6	38.447	0.345	0.206

3.3. Model Accuracy Indicators for Different Thresholds

According to the results in Table 3, the watermark restored (retrieved) in Situation 1 is indeed significantly different from the other situations (Situations 2 to 6). Among the three evaluation indicators, the value of SSIM is closer to the recognition of the human eye. Therefore, we calculated the SSIM value of the original watermark and the restored (retrieved) watermark, and used SSIM as the threshold value to determine whether the restored (retrieved) watermark was similar to the original watermark. If the SSIM value is greater than the threshold value, it means that the restored (retrieved) watermark is very similar to the original watermark. In other words, if an image can be restored (retrieved) by the generator to produce a generated image that is sufficiently similar to the original watermark, it means that the image is one of the original protected images. To verify the classification accuracy of different threshold values, in this section we use a confusion matrix and four other indicators to evaluate the effect of the model.

Table 4 shows the confusion matrix and evaluation indicators presented by experiments with different SSIMs as the threshold values. The TP in the confusion matrix indicates that the model must be restored (retrieved) to produce a watermark and is judged to be sufficiently similar to the original watermark; FN means that the model must restore (retrieve) the watermark, but the watermark is judged to be wrong, which is a missed judgment; FP means that the model should not restore (retrieve) the watermark but is judged to be similar to the original watermark, which is a false alarm; TN indicates that the model should not restore (retrieve) the watermark and is judged to be an erroneous watermark. It can be seen from Table 4 that under different thresholds, the performance of Accuracy, Precision, Recall, and F1 Score are all very good with little difference, so we can focus on how to reduce missed judgments and false alarms. From the perspective of the application of intellectual property rights protection, it is most important that a model can

completely protect the watermarked images. Therefore, in the two error situations, FN is more important than FP.

Table 4. Confusion matrix and accuracy indicators of model for different thresholds.

Threshold	TP	FN	FP	TN	Accuracy	Precision	Recall	F1 Score
0.9	1598	2	5	13,339	0.999	0.997	0.999	0.998
0.89	1598	2	7	13,337	0.999	0.996	0.999	0.997
0.88	1599	1	8	13,336	0.999	0.995	0.999	0.997
0.87	1599	1	9	13,335	0.999	0.994	0.999	0.997
0.86	1600	0	10	13,334	0.999	0.994	1.000	0.997
0.85	1600	0	10	13,334	0.999	0.994	1.000	0.997
0.84	1600	0	14	13,330	0.999	0.991	1.000	0.996

It can be seen from Table 4 that when the Threshold is 0.90, the TP is 1598 and the FN is 2, which means that the watermark should be restored (retrieved) correctly in the image; 1598 images are restored (retrieved) by the model we designed to produce a high-quality watermark, but 2 images are judged to be watermarks with lower image quality. The TN is 13,339 and the FP is 5, which means that in the image that should not be restored (retrieved) to produce the correct watermark, 13,339 images have not been restored (retrieved) with high-quality watermarks, but 5 images have been restored (retrieved) by our designed model to produce high-quality watermarks. It is important to note that the dataset was imbalanced, with 1600 watermarked images (positive cases) and 13,344 non-watermarked images (negative cases). On the whole, when the Threshold is smaller, the FN decreases, which means that the number of missed judgments is reduced, and the relative FP will increase accordingly, which means that the number of misjudgments increases. When the Threshold is 0.87–0.90, although the FP is less, the FN increases, indicating that the original protected image has missed judgment. When the Threshold is 0.84–0.86, the FN is all 0, which means that there is no missed judgment, but when the Threshold is 0.84, the FP number increases, which means that the number of misjudgments begins to increase. Therefore, from the perspective of missed judgment, Thresholds of 0.85 and 0.86 would be a better choice.

3.4. BER Comparison of Watermarking Methods Based on Deep Learning

The experimental results in Table 5 compare the BER values of retrieved watermarks under various quality factors for our method and recent deep-learning-based watermarking methods resistant to JPEG compression. The data are sourced from the experiments of Ahmadi et al. [17] and Zhong et al. [18]. Ahmadi et al. proposed the watermarking method of ReDMark, which uses three different types of network architecture models, including MT-Net, JT-Net, and GT-Net, during the training process. Ahmadi et al. confirmed in their experiments that, among the three different models, the two models of MT-Net and JT-Net have better robustness for JPEG compression. Therefore, we only compare the two models, MT-Net and JT-Net. In addition, Ahmadi et al. added a strength factor α to the watermarking method of ReDMark to control the balance between robustness and imperceptibility. According to the experimental results of the paper, when the strength factor $\alpha = 1.0$, the imperceptibility of the watermark is poor, but the robustness is relatively good; however, when the strength factor $\alpha = 0.6$, the imperceptibility of the watermark will increase, while the robustness will decrease.

Table 5. Comparison with other methods to the BER value of the restored watermark.

Method	QF = 10	QF = 50	QF = 70	QF = 90
ReDMark (MT-Net) [17] $\alpha = 1.0$	-	0.012	0.0	0.0
ReDMark (MT-Net) [17] $\alpha = 0.8$	-	0.035	0.004	0.0
ReDMark (MT-Net) [17] $\alpha = 0.6$	-	0.118	0.042	0.016
ReDMark (JT-Net) [17] $\alpha = 1.0$	-	0.013	0.002	0.001
ReDMark (JT-Net) [17] $\alpha = 0.8$	-	0.067	0.023	0.013
ReDMark (JT-Net) [17] $\alpha = 0.6$	-	0.173	0.107	0.079
Zhong et al. [18]	0.116	(0, 0.005)	(0, 0.003)	-
Ours	0.008	0.003	0.004	0.009

According to Table 5, our method still has the best BER when QF = 10. When QF = 50, in terms of BER, our method is superior to the method of Ahmadi et al. and is not much different from the method of Zhong et al.; when QF = 70 and QF = 90, our method still has a BER of less than 0.009. It is worth noting that Ahmadi et al. did not discuss a QF of less than 50, while Zhong et al. did not discuss a QF of greater than 70. On the whole, the BER value of the method proposed by Ahmadi et al. and Zhong et al. will increase as the QF decreases, which means that the more information is lost after compression, the higher the error of the restored (retrieved) watermark. The reason is that the above two methods both increase the dimension of features through the network layer, such that the model can learn more parameters and improve robustness.

Differently from the above two methods, our proposed architecture directly uses JPEG-compressed images for training and can train different models for different quality factors, and therefore has a low BER under any quality factor.

3.5. Comparison of Model Parameters of Different Model Architectures and Various Evaluation Indicators

The proposed method needs to store the secret key and the trained model parameters to generate watermarks, and how to reduce the amount of model parameters is very important. Therefore, we used depthwise separable convolution (DSC) [29] to ensure the feasibility of this idea. Compared with traditional convolution, depthwise separable convolution can reduce the number of parameters and calculations and has little effect on the model effect. To compare the effects of depthwise separable convolution on model training and results, we created two additional network models, DSC-1 and DSC-2. DSC-1 changes the convolutional neural network in the down-sampling part of the generator to the depthwise separable convolution, and the rest is the same as the original model; the generator part of DSC-2 is the same as that of DSC-1, and the convolutional neural network of the discriminator is changed to depthwise separable convolution and the rest is the same as the original model.

The comparison between the original model and the other two models is shown in Table 6. The visualization results of the restored (retrieved) watermark of the three models are shown in Figure 10.

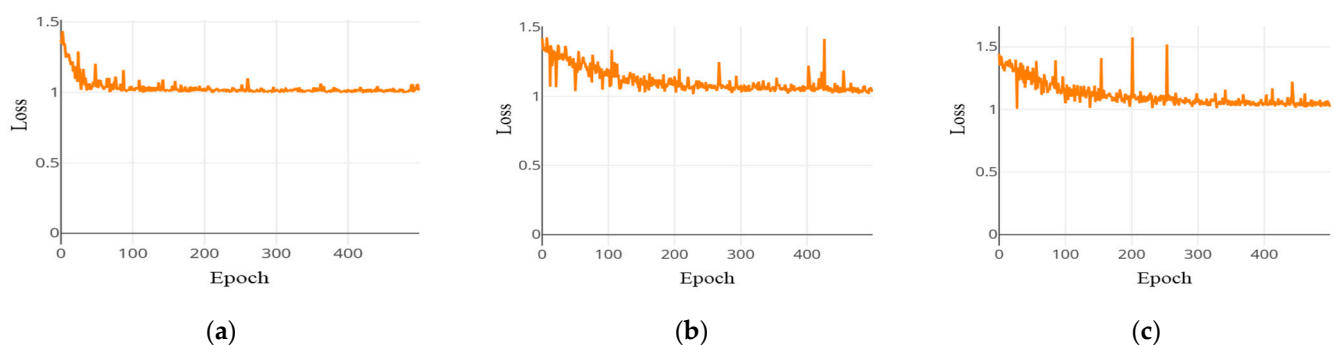
Table 6. Various evaluation indicators for restored (retrieved) watermarks with different models.

Model	Generator Parameters	Generator Size (MB)	PSNR	SSIM	BER
Original	331,715	1.27	49.348	0.959	0.006
DSC-1	178,358	0.69	49.320	0.911	0.014
DSC-2	178,358	0.69	48.497	0.900	0.016

Model	Restored watermark
Original	
DSC 1	
DSC 2	

Figure 10. Visualization results of restored (retrieved) watermark with different models.

In terms of generator parameters, the generator parameters are reduced by nearly half compared to the original model. The generator model size of the original model is about 1.27 MB, while DSC-1 and DSC-2 are reduced to 0.69 MB. In the image quality part of the restored (retrieved) watermark, DSC-1 and DSC-2 can still maintain good image quality for the restored (retrieved) watermark (the PSNR is 49.3 and 49.5 dB, respectively). However, the SSIM drops significantly after using DSC-1 and DSC-2, and it is already lower than 0.95. The human eyes can already see that the watermark restored (retrieved) by the model is different from the original watermark image. In the BER part, the BER of DSC-1 and DSC-2 has increased by about 0.01 but is still within an acceptable range. In addition, the depth of separable convolution will also affect the convergence speed of the model. The generator loss functions of the three models are shown in Figure 11, respectively. The original model (Figure 11a) has almost converged when Epoch is about 100, while DSC-1 (Figure 11b) only converges when Epoch is around 200, and DSC-2 (Figure 11c) converges when Epoch is around 300. The SSIM, BER, and convergence speed are slightly worse, but the benefit is that the number of parameters is greatly reduced, which is as much as half of the original model.

**Figure 11.** Generator loss with different models. (a) Generator loss of the original model. (b) Generator loss of the model DSC-1. (c) Generator loss of the model DSC-2.

This paper focuses on the importance of symmetry in watermark embedding, decoding, and reconstruction operations. It proposes a strategy for reducing distortions produced by lossy compression while also improving reconstruction stability. The experiment proves the method's dependability and capacity to handle a variety of JPEG quality parameters.

The proposed method has acquired the best capability of robustness and imperceptibility with PSNR values above 49 dB, SSIM values of up to about 0.95, and a BER up to 0.008, outperforming other methods like ReDMark and that of Zhong et al., especially at the lowest-quality factors. The difference from previous work is the capability to balance robustness with imperceptibility through negotiation with DCGAN and symmetry principles with security enhancement provided by secret keys. Furthermore, since depthwise separable convolutions halve the model size without decreasing performance, it is a secure, efficient, and advanced method for robust watermarking.

3.6. Computational Efficiency and Resource Requirements

The proposed methodology for semi-blind watermarking is robust and safe thanks to secret keys and the parameters of the model. However, the contribution of factors such as computational efficiency and resource requirements also plays a significant role in real-world application scenarios.

The training phase is carried out through deep convolutional generative adversarial networks (DCGANs), which would generally require heavy computational resources owing to the sustainable generator and discriminator networks employed for iterative training. All the above tests were carried out on a system that has an NVIDIA RTX 2060 SUPER GPU, 16 GB of RAM, and an Intel i7-9700 CPU.

The extracted hidden watermark is computed in an inference phase based under the trained model parameters along with secret keys. This computation is relatively less demanding compared to training, taking only a fraction of the hardware resources.

Depthwise separable convolution (DSC) has been used in some models, such as DSC-1 and DSC-2, to enhance computational efficiency. The model parameters accounted were reduced significantly, with a nearly 50% reduction from 1.27 MB to 0.69 MB, and yet still offered an acceptable level of robustness (PSNR: ~48–49 dB, BER: 0.01–0.014). However, there was a minor drop in SSIM (from 0.95 to approximately 0.90), which might adversely affect perceptibility for pictures with a higher level of detail.

4. Conclusions

This study proposes a robust semi-blind watermarking method for JPEG lossy compression, which uses a pre-designed embedder and decoder, combined with deep convolutional generative adversarial networks (DCGANs) for training, and finally, a well-trained generator model to restore (retrieve) the watermark. In addition, this study added a set of secret keys during the training process, which can ensure the security of the model and prevent third parties from copying a model that has a similar capability. Different from other watermarking methods based on deep learning, the training process of this method involves training a unique generator for the multiple protected images, and one generator corresponds to one watermark. Therefore, in the context of a third-party verification agency, the training process of this method is flexible and easy to manage.

According to the experimental results, the method proposed in this study can restore (retrieve) the watermark well under the influence of different quality factors of JPEG compression. The image quality and error bit rate (BER) of the restored (retrieved) watermark are both good. An example of the excellent robustness of the method is that it produces PSNR values consisting of more than 49 dB, SSIM values of around 0.95, and BER values of less than 0.008, depending entirely on the JPEG quality factors. These parameters are the

main character of an application that can outperform such distinguished conventional and new deep learning methods in retrieving watermark information under several conditions of compression.

Symmetry determines the effectiveness of the proposed methodology. It brings improved resistance to the compression of pictures as well as distortion by watermark symmetry principles focused on embedding and decoding. An under-explored area, though, is the practical implementation and comparative gains of symmetry. Further quantitative or ablation study might demonstrate the impact of symmetry directly on robustness against different image distortions, including scaling, cropping, noise, etc. It could further add to the contribution of the method toward robust watermarking technologies.

In this study, depthwise separable convolution (DSC) is used to reduce the amount of model parameters by about half. Although the image quality of the restored (retrieved) watermark is not ideal, the BER is still at a certain level. This confirms the feasibility of this direction. How to further improve the image quality while reducing the amount of model parameters is one of the important topics for future research. In addition, this method currently focuses on the robustness of JPEG compression; however, if it can resist other intentional image-processing attacks, the integrity of the method will be better.

The model holds promise with datasets like USC-SIPI and Microsoft COCO; however, its scaling capacity to more diverse datasets as well as scenarios such as very high resolutions and extremely compressed versions remains an avenue to be explored. Future studies will be directed toward experimentation on these specific scenarios to ascertain the adaptability of the model and its further nuances to be applicable in real-world scenarios.

Author Contributions: Conceptualization, C.-F.L. and Z.-C.C.; methodology, J.-J.S. and A.U.R.; software, C.-F.L.; and A.U.R.; validation, C.-F.L., Z.-C.C. and J.-J.S.; formal analysis, A.U.R.; investigation, A.U.R.; resources, C.-F.L.; data curation, C.-F.L. and A.U.R.; writing—original draft preparation, A.U.R.; writing—review and editing, C.-F.L. and J.-J.S.; visualization, A.U.R.; supervision, C.-F.L.; project administration, C.-F.L.; funding acquisition, C.-F.L. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the National Science and Technology Council, Taiwan, under contract number NSTC 113-2221-E-324-017-MY2.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity Index
BER	Bit Error Rate
JPEG	Joint Photographic Experts Group
DCGAN	Deep Convolutional Generative Adversarial Network
DSC	Depthwise Separable Convolution
BER	Bit Error Rate
JPEG	Joint Photographic Experts Group

References

1. Van Schyndel, R.G.; Tirkel, A.Z.; Osborne, C.F. A Digital Watermark. In Proceedings of the 1st International Conference on Image Processing, Austin, TX, USA, 13–16 November 1994; Volume 2, pp. 86–90.
2. Pfitzmann, B. Information Hiding Terminology. In Proceedings of the First International Workshop on Information Hiding, Cambridge, UK, 30 May–1 June 1996; Springer-Verlag: Berlin, Germany, 1996; Volume 1174, pp. 347–350.
3. Petitcolas, F.A.P.; Anderson, R.J.; Kuhn, M.G. Information Hiding—A Survey. In Proceedings of the IEEE, Special Issue on Protection of Multimedia Content, 6 August 1999; Volume 87, pp. 1062–1078.
4. Kadhim, I.J.; Premaratne, P.; Vial, P.J.; Halloran, B. Comprehensive Survey of Image Steganography: Techniques, Evaluations, and Trends in Future Research. *Neurocomputing* **2019**, *335*, 299–326. [\[CrossRef\]](#)
5. Wallace, G.K. The JPEG Still Picture Compression Standard. *IEEE Trans. Consum. Electron.* **1992**, *38*, xviii–xxxiv. [\[CrossRef\]](#)
6. Ernawan, F.; Kabir, M.N. A Robust Image Watermarking Technique with An Optimal DCT-Psychovisual Threshold. *IEEE Access* **2018**, *6*, 20464–20480. [\[CrossRef\]](#)
7. Abdulrahman, A.K.; Ozturk, S. A Novel Hybrid DCT and DWT Based Robust Watermarking Algorithm for Color Images. *Multimed. Tools Appl.* **2019**, *78*, 17027–17049. [\[CrossRef\]](#)
8. Hamidi, M.; El Haziti, M.; Cherifi, H.; El Hassouni, M. Hybrid Blind Robust Image Watermarking Technique Based on DFT-DCT and Arnold Transform. *Multimed. Tools Appl.* **2018**, *77*, 27181–27214. [\[CrossRef\]](#)
9. Singh, A.K. Robust and Distortion Control Dual Watermarking in LWT Domain Using DCT and Error Correction Code for Color Medical Image. *Multimed. Tools Appl.* **2019**, *78*, 30523–30533. [\[CrossRef\]](#)
10. Benrhouma, O.; Hermassi, H.; Belghith, S. Tamper Detection and Self-Recovery Scheme by DWT Watermarking. *Nonlinear Dyn.* **2015**, *79*, 1817–1833. [\[CrossRef\]](#)
11. Thakkar, F.N.; Srivastava, V.K. A Blind Medical Image Watermarking: DWT-SVD Based Robust and Secure Approach for Telemedicine Applications. *Multimed. Tools Appl.* **2017**, *76*, 3669–3697. [\[CrossRef\]](#)
12. Yen, C.T.; Huang, Y.J. Frequency Domain Digital Watermark Recognition Using Image Code Sequences with A Back-Propagation Neural Network. *Multimed. Tools Appl.* **2016**, *75*, 9745–9755. [\[CrossRef\]](#)
13. Sun, L.; Xu, J.; Liu, S.; Zhang, S.; Li, Y.; Shen, C. A Robust Image Watermarking Scheme Using Arnold Transform and Bp Neural Network. *Neural Comput. Appl.* **2018**, *30*, 2425–2440. [\[CrossRef\]](#)
14. Kandi, H.; Mishra, D.; Gorthi, S.R.S. Exploring the Learning Capabilities of Convolutional Neural Networks for Robust Image Watermarking. *Comput. Secur.* **2017**, *65*, 247–268. [\[CrossRef\]](#)
15. Vukotic, V.; Chappelier, V.; Furon, T. Are Deep Neural Networks Good for Blind Image Watermarking? In Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS), Hong Kong, China, 11–13 December 2018; pp. 1–7.
16. Mun, S.-M.; Nam, S.-H.; Jang, H.; Kim, D.; Lee, H.-K. Finding Robust Domain from Attacks: A Learning Framework for Blind Watermarking. *Neurocomputing* **2019**, *337*, 191–202. [\[CrossRef\]](#)
17. Ahmadi, M.; Norouzi, A.; Karimi, N.; Samavi, S.; Emami, A. ReDMark: Framework for Residual Diffusion Watermarking on Deep Networks. *Expert Syst. Appl.* **2020**, *146*, 113157. [\[CrossRef\]](#)
18. Zhong, X.; Huang, P.; Mastorakis, S.; Shih, F.Y. An Automated and Robust Image Watermarking Scheme Based on Deep Neural Networks. *arXiv* **2020**, arXiv:2007.02460. [\[CrossRef\]](#)
19. Radford, A.; Metz, L.; Chintala, S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv* **2015**, arXiv:1511.06434.
20. Gao, G.; Xu, T.; Hua, F. Robust Image Watermarking Based on Generative Adversarial Networks for Copyright Protection. *Res. Sq.* **2024**. [\[CrossRef\]](#)
21. Hao, K.; Feng, G.; Zhang, X. Robust image watermarking based on generative adversarial network. *China Commun.* **2020**, *17*, 131–140. [\[CrossRef\]](#)
22. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS’14), Montreal, QC, Canada, 8–13 December 2014.
23. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784.
24. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein Generative Adversarial Networks. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 214–223.
25. Hu, H.T.; Hsu, L.Y. Blind color image watermarking incorporating a residual network for watermark denoising and super-resolution reconstruction. *Soft Comput.* **2023**, *27*, 917–934. [\[CrossRef\]](#)
26. The USC-SIPI Image Database. Available online: <http://sipi.usc.edu/database> (accessed on 12 June 2017).
27. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft Coco: Common Objects in Context. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2014; pp. 740–755.

-
28. Krizhevsky, A.; Nair, V.; Hinton, G. The CIFAR-10 Dataset. Available online: <https://www.cs.toronto.edu/~kriz/cifar.html> (accessed on 31 December 2024).
 29. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.