

DWT-GAN Watermarking: Discrete Wavelet Transform domain-based Generative Adversarial Network for Digital Image Watermarking

Harish Sharma, Sandeep Chaurasia, Nitesh Pradhan, Ayush Singh

Department of Computer Science and Engineering, School of Computer Science and Engineering, Manipal University Jaipur, Rajasthan, India- 303007
e-mails: harish27j@gmail.com; sandeep.chaurasia@jaipur.manipal.edu; nitesh.pradhan@jaipur.manipal.edu; ayushsingh07x@gmail.com

Abstract

Digital image watermarking has proven its suitability for copyright protection and copy control of digital media. Digital Watermarking can be divided into three steps. Firstly, a feature vector is identified from digital media; then, it is modified to embed the watermark; after that, it is projected back into the media space while maintaining its visual properties. Transformation-based spread spectrum watermarking techniques are considered the most likely techniques for the same. Recently, advances in deep learning architectures have also proven their worth in the domains of steganography and watermarking. This article proposes a novel deep learning technique for watermarking digital images. It is a Generative Adversarial Network (GAN) based scheme that prepares the networks for both the host images and the watermark using a dataset of state-of-the-art embedding processes, i.e., transform domain techniques. Our model consists of three deep learning subnetworks: a watermark embedding (generator) network, a discriminator network for training, and a network of watermark extractors (decoders) to extract watermarks as required. This article shows that the quality of the watermarked images improves as the number of epochs increases from 10,000 to 25,000. These three networks in this scheme work well together. The experimental results show that this approach has achieved improvement as the mean squared error (MSE), peak signal-to-noise-ratio (PSNR), and structural similarity index (SSIM) quality metrics for watermarked images after different numbers of epochs have shown significant values. A lower MSE of less than 0.5, a higher PSNR above 40, and a higher SSIM towards 1 indicate better quality of the watermarked images.

Keywords: *digital watermarking, deep learning-based watermarking, generative adversarial network, DWT watermarking.*

1 Introduction

Watermarking is a data concealment technique to embed identification information into a digital media item that notifies users about the intellectual property of its owners. The identity information may be recovered, allowing the owners to be identified if a threat attempts to replicate or alter the original content [1]. As technology advances, unauthorized users copy, allow, and share infringing digital content. Ensuring the security of multimedia content remains one of the most challenging tasks. Watermarking has been widely adopted to protect the copyrights of multimedia content; the method for this can be split into two phases: embedding and extraction (or identification). In this process, the owner embeds watermarks into the multimedia in the embedding stage. Suppose someone at the identifying point has compromised and manipulated digital data. In that case, owners can remove the watermarks from encrypted multimedia as proof of intellectual property rights [2].

Digital image watermarking has traditionally been done with specialized algorithms categorized according to their work domain, i.e., spatial domain watermarking and frequency domain watermarking. By modifying bit streams or pixel values, spatial domain approaches infuse data directly into the cover media. In comparison to other approaches, their computational simplicity makes them more susceptible to adversary removal and distortion. Frequency domain methods work by manipulating the signal medium's frequency coefficients. These approaches are more resistant to attacks but are also more computationally demanding. In various methods, frequencies may be adjusted and partitioned into high or low regions; Then, the personal information is integrated into the frequency segments to improve the embedding quality [3]. For more robust methods, spatial and frequency domain techniques can be coupled [4] [5]. Traditional methods have the drawback of having finite applications, and their developers must be well-versed in the embedding process. Various methods are helpful for specific applications, but the feasibility of these algorithms may be compromised soon as watermark removal and deterioration methods become advanced. Adaptive watermarking methods [6-8] use a combination of spatial and frequency-based methodologies. Senders and receivers of data can change their plans to stop an attacker from damaging or changing a message if they know how the attacker finds out about it.

Deep learning methods in data concealment may provide adaptive, generic frameworks that may be used for watermarking and steganography applications. These machine-learning models may develop sophisticated embedding patterns that are considerably more successful than standard watermarking or steganography methods in resisting a broader spectrum of attacks. Recently, advanced computing and deep-learning developments have climbed as computers have advanced. Deep learning and neural networks [9] have come to be useful for image recognition [10], natural language processing [11], and speech recognition [12] issues.

In this work, we provide a unique digital image watermarking approach based on discrete wavelet transformation and generative adversarial network (GAN). Convolutional neural networks and other deep learning techniques are used in generative adversarial networks as a way of modeling. Generative modeling is an unsupervised machine learning technique that involves identifying and learning

the regularities or patterns in incoming data so that the model can be used to generate or output new instances that could have been derived from the original dataset.

In this article, we propose a new approach to digital image watermarking using a Discrete Wavelet Transform (DWT) domain's knowledge to train a Generative Adversarial Network (GAN). Once the network is trained, it will be computationally less complicated than the traditional approach. DWT is considered one of the state-of-the-art methods for image watermarking. We have used samples generated by various DWT methods as training samples for the generator in our GAN model. GAN is used for image watermarking because it can generate realistic images that are very difficult to distinguish from real images. This makes it difficult for someone to remove the watermark from an image without damaging it. We have trained our GAN model specifically for image watermarking. This model can be trained to generate images that are both realistic and contain the watermark.

In section 2, we have discussed related work like basic terminology, performance metrics, features of watermarking, and deep learning. We have discussed recent developments in watermarking using deep-learning approaches in the same section. In section 3, i.e., in method and materials, we have discussed pre-processing of datasets, training parameters (like optimization function, activation function, weight initialization, loss function), training procedure, proposed architecture, and algorithm. In section 4, we have demonstrated the experimental setup, dataset used, and hyperparameters. In section 5, we have discussed parameters used to evaluate our experiment, demonstrating our results and output of our proposed method. In the last section, we have given the conclusion of our work.

2 Related Work

When evaluating Digital Watermarking techniques, there are numerous elements to consider. The three most essential are capacity, imperceptibility, and robustness.

The capacity refers to how much data can be stored on a cover media. The imperceptibility refers to how easily the data is detectable, and the robustness describes the data's resistance to attacks; in this sense, attacks are any alterations made to the cover material to destroy or damage the watermark data. The three qualities all come with an inherent trade-off. For example, a large payload capacity makes the message more detectable, resulting in a lower level of imperceptibility. In the same way, making a system more resistant to attacks may reduce its payload capacity and make it harder to see because the encoded image will have more redundancy to prevent distortions.

Digital watermarking often focuses on robustness above secrecy because the capacity to survive intrusions and manipulation is more crucial than the watermark's invisibility. Since deep learning-based systems are adaptive, the user may directly control these metrics' trade-offs. The primary characteristics of robustness and imperceptibility serve the purpose of the deep learning system.

2.1 Deep Learning Approach in Watermarking:

Deep Neural Networks (DNN) can model complex non-linear relationships between problems and solution spaces. Compositional models can be produced using DNN designs, expressing the object as a layered architecture of a shallow prior layer. The extra layers allow input from lower layers to be combined, which in theory, means that complex data can be shown with fewer units than in a narrow network with the same performance.

Deep architectures offer several options for a few simple techniques. Each architecture has achieved success in critical sectors. If different designs have not been tested against the same datasets, comparing their performances is not necessarily promising. Several other DNN variations exist, including Residual Neural Networks (ResNet) and Convolution Neural Networks (CNN).

Objective functions are used during training to maximize the performance of deep learning models. This function calculates the difference between the actual and projected data points. The model learns to make more accurate predictions over time by enhancing the loss function. In addition to examining modern techniques that do not follow the encoder-decoder loss structure, this section will explore some of the most often-used objective functions for data hiding in deep learning models.

Definition 1: Mean square Error between the original work (I) and watermarked image (K) is given as:

$$MSE = \frac{1}{mn} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [I(i,j) - K(i,j)] \quad (1)$$

MSE is employed to calculate encoder loss in some works [13]– [15], while many others, like [16]–[26], Utilize it for both encoder and decoder loss computations.

Definition 2: Mean Absolute Error (MAE) uses the absolute value of the difference between the data points as an input to determine the difference between two data sets.

Therefore, it disregards mistakes outside the regression line, often known as outliers. It is utilized in [27] to determine the loss of the decoder.

Definition 3: Cross Entropy Loss compares two variables' probability distributions, in the case of watermarking, the initial and extracted binary watermark messages. This equation requires two parameters: $p(x)$, which represents the actual distribution, and $q(x)$, which describes the estimated distribution. This is like the original and extracted watermarks in digital watermarking, where $q(x)$ represents the probability of watermark bits as the output of the decoder network.

Cross entropy loss is proportional to the logarithm of the predicted probability; hence, the model is penalized for making accurate probability forecasts closer to the actual distribution than to the accurate distribution. The following equation represents the cross-entropy loss:

$$H(p, q) = - \sum p(x) \log(q(x)) \quad (2)$$

Definition 4: Adversarial Loss in deep learning models that contain a discriminator network for adversarial learning; a second loss function is added to

the overall optimization to consider the Discriminator's Capacity to detect encoded images.

As the encoder is a generative network, it is possible to link adversarial loss with encoder loss to enhance the imperceptibility of the encoded watermark. The Discriminator takes an image X , which may be changed or not, and predicts $A(I) \in [0, 1]$, which is the probability that X has a watermark. The Discriminator optimizes the function for adversarial loss using the following equation's predictions.

$$L_a(X, Y) = (1 - A(I_c)) + \log(A(Y)) \quad (3)$$

Digital watermarking models aim to enhance robustness and imperceptibility, achieving balancing the different qualities based on the application's requirements. The encoder-decoder architecture separates the model into two distinct networks for encoding and decoding and is the design most typically employed by deep data-hiding models. Using two distinct equations, one for the encoder and the other for the decoder and computing the system-wide loss as a weighted total is a popular method for evaluating loss. This amount also includes adversarial loss for architectures that contain an adversarial network. Minimizing loss between the original cover picture and the encoded image is essential while increasing imperceptibility. Optimal resilience requires minimizing the loss between the original secret message or watermark information and the final recovered message.

Given how expensive it is to train neural networks, especially DNNs, it is hard to find a good reason to train individual neural networks [28–30] to defend each image on a large scale. Traditional treatments present a considerable challenge and are inefficient for real-world applications. DNN, which has been popular in several applications, can be used for watermarking [31]. The general monotonic activation function shows the process that cannot be turned around to remove secret information from a watermarked image.

Zhong et al. [32] developed an adaptive watermarking framework based on deep neural networks to generalize image watermarking processes. Without knowing anything about possible attacks ahead of time, an unsupervised deep learning structure and a new way to calculate loss are made to get a system with high Capacity and resilience. After the network learned how to embed the watermark into original photos and recover the watermark from watermarked images, Ming et al. [33] trained the model on an image collection, and the model was then evaluated against several image sets to determine its generalizability. The recovered watermark often receives more attention throughout the de-watermarking procedure.

Kandi et al. [34] encode the watermark information using two images generated by the same neural network, assuming that the two images are visually indistinguishable but have different pixel values generated by the network. The watermarked image is converted using the outputs of two systems and the original carrier image by comparing the Distance metric between the watermark image blocks. Zhang et al. [35] used a traditional back-propagation process to update the network's weights, and the wavelet domain was subsequently used to conceal the secret information. CNN architecture, called ISGAN [15], conceals a secret grey

image into a color cover image on the sender side and extracts the secret image on the receiver side. Other works execute image steganography using deep learning, although these works still have concerns with Capacity, invisibility, and security [15][36].

3 Problem Formulations or Methodology

3.1. Pre-processing of Data

Wavelet techniques efficiently acquire sparse, compressible data representations or features that can be utilized in machine learning and deep learning workflows. Wavelet transforms have been widely used in signal processing and image compression, where they can provide sparse representations of signals and images. These sparse representations can be used as features in machine learning and deep learning workflows [37] [38]. Many digital image watermarking schemes have used discrete wavelet transform-based techniques over the last decade [39-42]. Since DWT-based watermarking techniques are considered the best for digital watermarking, we used this method to create our dataset for training the GAN model.

The Wavelet transform may use a variety of wavelets to decompose images [43]. Wavelet transform decomposes a signal into a set of resolution-related views [44]. The wavelet transform operates by convolving the target function with wavelet kernels to obtain wavelet coefficients representing the contributions in the function at different scales and orientations. The wavelet transform can be used to analyze signals at different resolutions, where the high-frequency components represent the details of the signal, and the low-frequency components represent the smooth parts of the signal [45]. The wavelet transform can be recursively applied to the low-frequency components to obtain a multiresolution signal analysis. This multiresolution analysis allows for extracting features at different scales, which can be useful in various applications, including image processing, signal processing, and data compression. Therefore, the wavelet transform's results depend on the type of wavelet used [46]. The state-of-the-art coding techniques use the wavelet transform as a fundamental and common step for their further technical advantages [44 - 46].

Many wavelets are being used nowadays for the decomposition of signals and images. In this paper, each test image was transformed using distinct wavelets. The types and families of wavelets are Daubechies, biorthogonal, coiflets, and eyelets [49]. Below are the few wavelets we used to create the training dataset.

The extremal phase wavelets of the Daubechies are called *dbN* wavelets. The number of vanishing moments is N. In the literature, these filters are also referred to by their 2N filter tap count. The types are *db1*, *db2*...*db45*. These are not symmetrical. The length of the filter is 2N. We have used '*db2*' and '*db10*' from this family. The Haar wavelet is a particular case of this family of wavelets and is known as '*db1*'.

With FIR filters, accurate reconstruction is achievable for the compactly supported Biorthogonal (Biro) Wavelets. The prefix *biorx* identifies each variant of the biorthogonal wavelet. *y*, where *x* denotes the analysis part's vanishing moments (decomposition), and *y* denotes the synthesis part's vanishing moments (reconstruction). The types are *bior1.1*, *bior1.3*, *bior1.5*, *bior2.2*, *bior2.4* ...*etc*.

Wavelets with the most vanishing moments are known as coiflets (coif). In coifN, N is the sum of the wavelet and scaling function's vanishing moments. The 3N number of filter coefficients is also used in the literature to identify these filters. We have used 'coif2' and 'coif5' wavelets from this family.

Wavelets used are "rbio4.4", "coif2", "coif5", "Haar", "db2", "db10", "bior1.1", "rbio1.1", "rbio2.4", "bior2.4".

3.2 Training Parameters

3.2.1 Optimizer

Optimizers calculate and update the gradients of loss for weights in the network. Adaptive gradient descent (AdaGrad), Root Mean Square Propagation (RMSProp), Stochastic Gradient Descent (SGD), and Adaptive moment estimation (Adam) are all frequently used optimization techniques in deep learning. In contrast, SGD applies the same learning rate to all the parameters and updates them simultaneously. However, this makes training on sparse datasets difficult. AdaGrad overcomes this problem by applying different learning rates to every parameter, thus making it plausible to train on sparse datasets. However, as the training progresses, the learning rate becomes negligible, thus making training at later stages difficult. RMSProp overcomes this problem by calculating exponential decaying average gradients. Adam optimizer uses the best of AdaGrad and RMSProp, i.e., momentum and exponentially decaying gradients, respectively.

The proposed network uses Adam as the preferred optimizer during experimentation. We found that it performed better than RMSProp. Like RMSProp, an exponentially declining average of previously observed gradients (m_t) is stored by Adam. Additionally, it preserves forgetting elements β_1 and β_2 . β_1 is a forgetting factor for the non-centered variance of gradients.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_1 \quad (4)$$

It also maintains past square gradients (g_t). As mentioned above, the value gives more importance to the previous gradients to avoid instability due to sudden changes.

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_1^2 \quad (5)$$

In our experimentation, the values of $\beta_1 = 0.5$ and $\beta_2 = 0.999$, along with a learning rate set to 0.0002, gave the best results.

3.2.2 Activation Function

Activation functions are employed to introduce non-linearity in Deep Neural Networks. The proposed network employs four activation functions: Sigmoid, ReLU, LeakyReLU, and Tanh.

A tanh activation function is used at the output layer of the generator. Since the input images are normalized between [0,1], tanh is used to get output values in the range [-1,1].

LeakyReLU activation is used in the Generator and Discriminator convolutional blocks to prevent the gradients from becoming zero. Unlike ReLU, it prevents the negative value from zeroing out, thus producing no update/gradient for the weights to get updated.

Since the task of the Discriminator is to classify images coming from real distribution and the generator as real or fake, the sigmoid activation function is used at the output layer. The sigmoid activation function is used for binary classification and outputs values between 0 and 1. It is also used in the output layer of the extractor as the watermark image is a grayscale image with values between 0 and 1. The ReLU activation function is used in hidden layers of the Extractor network.

3.2.3 Weight Initialization

Weight initialization plays a significant role in the optimization of the network. We found that initializing weights with Glorot/Xavier Initialization made the optimization process more manageable. It considers the total number of input units in the weight tensor and the total number of output units in the output tensor. Glorot initialization is defined as:

$$\sigma = \sqrt{\frac{2}{a+b}} \quad (6)$$

3.2.4 Loss Function

The loss functions for the three networks are as follows:

The Generator(G) loss function is the Binary-Cross-Entropy (BCE) Loss. The generator output is the watermarked image. The object of the generator is to minimize the BCE loss in-order to fool the discriminator. Data points from the real distribution (x_i) and data points from a fake distribution (z_i) are used to calculate the loss. The fake distribution (z_i) is fed as input to the generator (G).

$$G_{Loss} = Min \left[\sum (X \sim P(X)) [\log(D(X))] + \sum (Z \sim P(Z)) [\log(1 - D(G(Z)))] \right] \quad (7)$$

The loss function for the *Discriminator(D)* is the same as the generator. However, instead of trying to minimize it, the discriminator will try to maximize the $D(X)$ term and minimize the $D(G(Z))$ term to maximize the overall loss function. Data points from the real distribution (x_i) and the output of the Generator ($G(z)$) are utilized to calculate the loss.

$$D_{Loss} = Max \left[\sum (X \sim P(X)) [\log(D(X))] + \sum (Z \sim P(Z)) [\log(1 - D(G(Z)))] \right] \quad (8)$$

Therefore, the total loss function for training a generative adversarial network can be defined as:

$$Total\ Loss = Min_G Max_D \left[\sum (X \sim P(X)) [\log(D(X))] + \sum (Z \sim P(Z)) [\log(1 - G(Z)))] \right] \quad (9)$$

The loss function used in *the extractor* is *binary-cross entropy* loss.

$$BCE\ loss = -\frac{1}{N} \sum_{i=1}^N (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

We also experimented with several other loss functions as well, such as Mean-Squared Error, Mean Absolute Error, and L1 loss. However, we discovered that BCE loss best fit for our needs. The watermark image is ground truth (y_i), and the extractor's output (p_i) is used to calculate the loss for the extractor.

3.3 Training Procedure

3.3.1 Architecture of Generator and Discriminator

The proposed architecture uses two networks, i.e., Generator(G) and Discriminator(D). Each one of them is explained in detail:

The role of the *generator(G)* is to produce watermarked images that are successful in fooling the Discriminator. It consists of convolutional layers for extracting features from the input. The input to the network is size 128x128x2. During convolution operation, the shape of the input is not changed; only the filters of subsequent layers are increased. No pooling is used to keep the input size the same. Batch normalization layers are used after the convolution operation and before the activate function. It is done to ensure faster training and normalization of the output of previous layers. There are only partially connected layers. The Detailed Generator architecture is discussed in Table 1.

Discriminator(D) network is a simple convolutional neural network whose role is to classify images from real distribution and images from Generator as real or fake. Unlike the generator, it has fully connected layers at the end with sigmoid activation to classify images. It consists of convolutional layers with a stride of 2 to reduce the size of the input image. The detailed architecture is discussed in Table 2.

Table 1 Each Generator layer consists of a convolution filter of size 3x3 with stride '1' and "same" padding. After each convolution, batch normalization is applied before applying the activation function.

	No. of Filters	Activation Function	Size
1	-	-	128x128x2
2	16	Batch Normalization + LeakyReLU	128x128x16
3	32	Batch Normalization + LeakyReLU	128x128x32
4	64	Batch Normalization + LeakyReLU	128x128x64
5	128	Batch Normalization + LeakyReLU	128x128x128
6	256	Batch Normalization + LeakyReLU	128x128x256
7	128	Batch Normalization + LeakyReLU	128x128x128
8	64	Batch Normalization + LeakyReLU	128x128x64
9	32	Batch Normalization + LeakyReLU	128x128x32
10	16	Batch Normalization + LeakyReLU	128x128x16
11	1	Tanh	128x128x1

Table 2 Each Discriminator layer consists of a convolution filter of size 3x3 with stride ‘2’ and “same” padding. LeakyReLU activation function is applied after each convolution.

Layer	No. of Filters	Activation Function	Size
1	–	–	128x128x1
2	32	LeakyReLU	64x64x32
3	64	LeakyReLU	32x32x64
4	128	LeakyReLU	16x16x128
5	256	LeakyReLU	8x8x256
6	512	LeakyReLU	4x4x512
7	Dense Layer	Sigmoid	1

3.3.2 Architecture of Extractor

The role of the Extractor network is to extract the watermark from the output of the generator, i.e., watermarked image. We tried different Fully Convolutional Networks with different hyperparameters before settling down with an auto-encoder. The role of the auto-encoder is to map each output image of the generator to its respective watermark image. The detailed architecture of the extractor is shown in Table 3.

Table 3 Each Extractor network layer consists of convolution filters of size 3x3 with stride 2 and “same” padding. Convolutional Transpose layers with strides ‘4’, ‘5’, ‘5’ are applied to achieve the output of the same size as the input. The ReLU activation function is applied after each convolution.

Layer	No. of Filters/Layer Name	Activation Function	Size
1	–	–	128x128x1
2	32	ReLU	64x64x16
3	64	ReLU	32x32x32
4	128	ReLU	16x16x64
5	Dense1	ReLU	64
6	Dense2	ReLU	10
7	Dense3	ReLU	64
8	Dense4	ReLU	16384
9	ConvTranspose 128	ReLU	32x32x128
10	ConvTranspose 64	ReLU	64x64x64
11	ConvTranspose 1	Sigmoid	128x128x1

4 The Proposed Method

4.1 Proposed Algorithm

Algorithm: Training procedure of proposed DCGAN and Auto-encoder

Input(s): w , the concatenated image of Image and Watermark to G and w' , DWT watermarked image to D, watermark dataset(Q) to E for watermark extraction.

Output(s): Watermarked Image and Extracted Watermark from Watermarked Image.

1: $G \leftarrow$ concatenated image, W

2: $D \leftarrow$ output of generator, $G(W)$

3: $D \leftarrow$ DWT watermarked image, W'

4: Calculate Binary Cross Entropy Loss for the Discriminator

$$P(D(W', y_{real})) \sim 1 \text{ and } P(D(G(W), y_{fake})) \sim 0 \rightarrow \max[\log(D(W')) + \log(1 - D(G(W)))]$$

5: Backpropagate this loss calculated in step 4 to the Discriminator network.

6: Apply Adam optimizer to update the weights of the Discriminator network (D).

7: Calculate Binary Cross Entropy Loss for Generator

$$P(G(W), y_{fake}) \sim 1 \rightarrow \min[\log(D(W')) + \log(1 - D(G(W)))]$$

8: Backpropagate this loss calculated in step 5 to the Generator network.

9: Apply Adam optimizer to update the weights of the Generator network (G).

10: $E \leftarrow G(W)$

11: Calculate BCE loss between auto-encoder output and Q dataset.

12: Backpropagate this loss to train the E network.

13: Repeat steps 1 to 12 for each batch until the values of Binary Cross Entropy for both losses become negligible.

4.2 Architecture

The proposed architecture comprises a Generative Adversarial Network (GAN) and an Autoencoder. The inputs to the Generator are concatenation images of the cover image and the watermark. It is of the size 128x128x2. In contrast, the input to the Discriminator network is the DWT watermarked image of size 128x128x1. These images are watermarked using various wavelets. The Discriminator takes DWT watermarked images and outputs them from the generator to distinguish them as real or fake. The generator's output is also fed to the auto-encoder to regenerate the watermark. The whole network is shown in Fig.1.

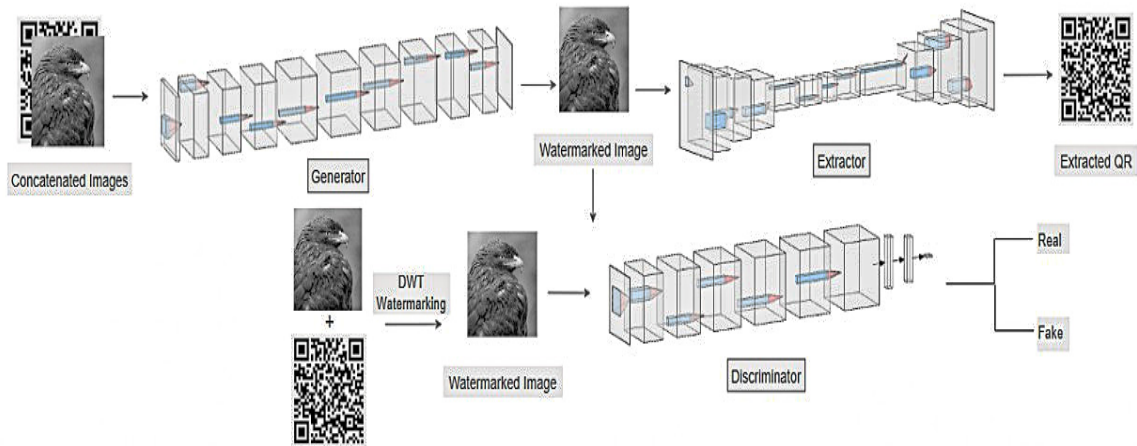


Fig. 1 Proposed DWT-GAN Architecture

4.3 Experimental Setup

The authors used Kaggle Kernels [45], [46] to train the proposed network. It is a freely available online training platform for machine learning and deep-learning framework. It provides an Nvidia Tesla P100 GPU with 16 GB of RAM, available for continuous use for 30 hours per week without any halt. The PASCAL-VOC 2012 dataset [47] is used to create training data. Every image is converted to grayscale and then resized to 128 x 128. Then these images are watermarked using different DWT wavelets with four different QR Code images as watermarks of size 128 x 128.

The idea is to train the system on various cover and watermark combinations. Therefore, these watermarked images are shuffled randomly and then fed to the Discriminator. The input images for the generator are grayscale PASCAL-VOC 2012 images concatenated with 4 different QR Code images as watermarks of size 128 x 128. So, the actual size of the input image to the generator network is 128 x 128 x 2.

Table 4 lists the hyper-parameters utilized in training and their experimentally determined values. A mini-batch comprises 64 watermarked pics and four unique 128 x 128 QR Code images. A watermark was applied to each mini batch. The training was carried out for 4000 epochs, or until the loss value became stable. We have used the Adam optimizer [18]. The strength factor was set to 1 during the training, and the weight decay rate was 0.01.

Table 4 Parameter Values

Hyperparameter	Value
Epoch	5000 to 30,000
Mini-Batch	64
Optimization	Adam
β_1	0.5
β_2	0.999
The learning rate of an	0.0002

embedded network(α)	
The learning rate of the extraction network(α)	0.0001

5 Results and Analysis

The following metrics can be utilized to determine the image's quality. It is connected to the imperceptibility property since embedded data should not result in perceptible modifications to digital media.

Definition 5: *Structural Similarity Index (SSIM) is a number between [-1.0,1.0] designated as the SSIM, with a value of "1" designating two identical images. The SSIM of two photos is shown as follows, where μ_x and μ_y represent the means of the two images, respectively., and the variances of the two images are denoted by σ_x and σ_y .*

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (11)$$

Definition 5: *Peak Signal to Noise Ratio (PSNR): It defines the relationship between a signal's maximal potential value (power) and the power of distorting noise, which determines its representation's efficiency.*

PSNR is generally expressed in the logarithmic decibel scale since specific signals have an extended dynamic range (the ratio between the maximum and smallest potential values of a changeable quantity). It is utilized in data concealment to differentiate between the cover picture and the encoded image, proving the efficacy of the embedding method. A PSNR score greater than 30dB indicates that the picture difference is not visible upon eye inspection. PSNR is measured on a logarithmic scale in decibels (dB).

$$PSNR = 20\log_{10} \left(\frac{MAX_f}{\sqrt{MSE}} \right) \quad (12)$$

The perceived quality of the watermarked picture can be evaluated using metrics like Mean Squared Error (MSE), PSNR, and SSIM, and the resilience of the recovered watermark can be evaluated using BPP and SSIM. We used the peak-signal-to-noise-ratio (PSNR), mean square error (MSE), and structural similarity index (SSIM) as proxies to examine the trade-offs between the quality of the watermarked images and the original cover image for the generator network.

Table 5 shows the quality of generated watermarked images compared to the original cover images. It is observed that the generator can produce high-quality watermarked images with an MSE value of less than 1, PSNR of around 50, and SSIM as high as 0.99. The watermarked image's quality is comparable to the cover image. Furthermore, a human eye examination of the watermarked image reveals that the watermark is likewise not identifiable.

Table 5 The quality metrics of generated watermarked images compared to the original cover images.

Epochs	MSE	PSNR	SSIM
5000	34.1481	32.7971	0.0012

10000	1.9931	45.1352	0.8998
15000	0.4804	51.3147	0.9751
20000	0.1544	56.2422	0.9901
25000	0.1126	57.6144	0.9922
30000	0.7561	49.3449	0.9532

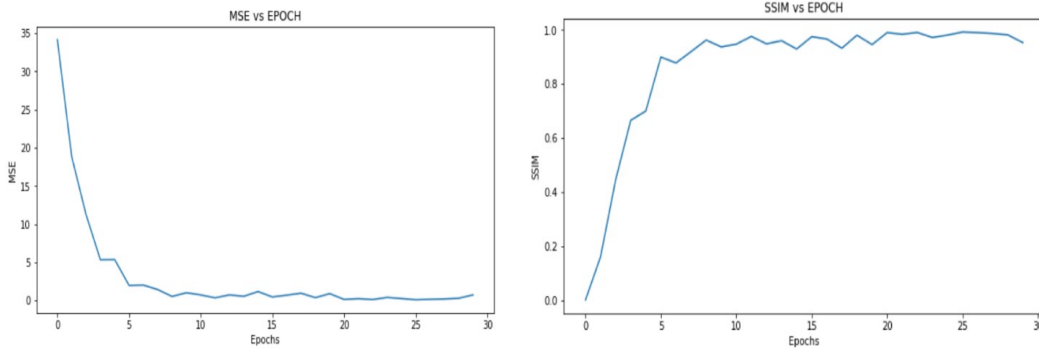


Fig. 2: (a) Mean Square Error (MSE) vs Epoch, (b) Structural Similarity Index Measure (SSIM) vs Epoch.

Fig. 2 (a-b) shows various parametric observations between the cover and watermarked image after various numbers of epochs. After 10K epochs, the generator becomes stable and generates watermarked images of acceptable quality.

Table 6 shows the resultant loss of the generator and Discriminator. As shown in the table, the authors have used BCE loss for the generator and the Discriminator as evaluation parameters to evaluate the performance of the proposed model. The table shows that Generator loss decreases as more epochs have been used. However, discriminator loss becomes stable and has minimal changes during most epochs.

Epochs	Generator Loss	Discriminator Loss
3000	0.9286	1.3736
6000	0.8013	1.3849
9000	0.6935	1.3712
12000	0.6898	1.4137
15000	0.4935	1.3884
18000	0.3859	1.3886
21000	0.3256	1.3973
24000	0.2723	1.3838
27000	0.1634	1.383
30000	0.1486	1.7065

Fig. 3 shows the result of some sample input images with different QR codes as watermarks. It is observed that the quality of watermarked images and extracted watermarks are very similar to the input cover images and embedded watermarks, respectively [20].



Fig. 3 the result of some sample input images with different QR codes as watermarks

Additionally, we scanned all the extracted QR codes using a QR code scanner, and their output was correct too. However, the QR code has an error correction mechanism that allows the perseverance of information even if there is some degradation in the quality of the code.

6 Discussion

In this research, we offer a unique approach for digital image watermarking based on the GAN model and DWT-based method examples. We have used various wavelets to create our sample dataset with three networks: embedder, extractor, and Discriminator. Once the embedded network is trained, the discriminator network will not be required.

In this work, the number of watermarks used is minimal (i.e., 4) because the network took a long time to understand the randomness of the watermark. Although, in the initial experiments, the network was more inclined towards one watermark, as most of the samples belonged to one in the beginning. Then we randomly distributed all four watermarks in work, which provides random samples to the network so that our system should be open to more than one type of watermark.

Earlier, researchers used simple watermarks to increase system parameters' performance accuracy (like MSE and SSIM). We have used quite a complex watermark (i.e., QR Code) for the experiments, which makes training quite challenging.

7 Conclusion

In this paper, the authors are proposing a novel image watermarking method that can embed highly informative watermarks, such as QR codes, into images using GAN. Our method can embed watermarks (like QR code) in images without compromising the quality of the host images. We have evaluated our method on a state-of-the-art dataset. The experimental results show that our method can achieve high performance in terms of watermark capacity, robustness, and imperceptibility. In this method, lower MSE, higher PSNR, and higher SSIM indicate better quality of the watermarked images.

Overall, GANs are a promising approach for image watermarking. They can generate realistic images that are very difficult to distinguish from real images. This makes them a good choice for protecting the copyright of images or for tracking the distribution of images.

The author of this paper suggests that for watermarking applications, one should use complicated and informative watermarks with enough randomness, such as QR code or random noise, to identify the real strength of the individual networks based on deep learning techniques.

ACKNOWLEDGEMENTS

The authors would like to thank the Manipal University Jaipur for providing the deep learning laboratory (lab 107, Academic Block 2) of computer science and engineering department infrastructure necessary to carry out this research. The availability of a well-equipped laboratory environment was an essential factor in allowing this research to be conducted in a timely and effective manner.

References

- [1]. Cox, I. J., Miller, M. L., Bloom, J. A., Fridrich, J., & Kalker, T. (2008). Robust Watermarking, *Digital Watermarking and Steganography*, 297–333.
- [2]. Shih, F. Y. (2017). Digital watermarking and steganography: Fundamentals and techniques. *CRC Press*.
- [3]. Al-Haj, A. (2007). Combined DWT-DCT digital image watermarking. *Journal of Computer Science*, 3(9), 740–746.
- [4]. Abed, S., Al-Roomi, S. A., & Al-Shayegi, M. (2019). Efficient cover image selection based on spatial block analysis and DCT embedding. (“Efficient cover image selection based on spatial block analysis and DCT ...”) *EURASIP Journal on Image and Video Processing*, 2019(1), 1–14.
- [5]. Tun, A., Thein, Y. (2013). Digital image watermarking scheme based on LWT and DCT. *International Journal of Engineering and Technology*, 272–277.
- [6]. Sheth, R. K., & Nath, V. V. (2016, Spring). Secured digital image watermarking with discrete cosine transform and discrete wavelet transform method. (“Secured digital image watermarking with discrete cosine transform and ...”) *In 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA)*. (“Social media: A new vector for cyber attack | IEEE Conference ...”)

- [7]. Kulkarni, T. S., & Dewan, J. H. (2016). Digital video watermarking using Hybrid wavelet transform with Cosine, Haar, Kekre, Walsh, Slant and Sine transforms. "In 2016 International Conference on Computing Communication Control and Automation (ICCCUBEA)." ("MoiréPose | Proceedings of the 28th Annual International Conference on ...")
- [8]. Qianli, Y., & Yanhong, C. (2012). A digital image watermarking algorithm based on discrete wavelet transform and Discrete Cosine Transform. ("A digital image watermarking algorithm based on discrete wavelet ...") In 2012 International Symposium on Information Technologies in Medicine and Education. ("2012 International Symposium on Information Technologies in Medicine ...")
- [9]. "Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., & Maida, A. (2019)." ("Sci-Hub | | 10.1016/j.neunet.2018.12.002") Deep learning in spiking neural networks. *Neural Networks*, 111, 47-63.
- [10]. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. *MIT Press*.
- [11]. Goldberg, Y. (2017). Neural network methods for natural language processing, *Synthesis Lectures on Human Language Technologies*, (Vol. 10, No. 1, pp. 1-309).
- [12]. Zhang, Y., et al. (2016, August). Towards end-to-end speech recognition with deep convolutional neural networks. *Interspeech 2016*.
- [13]. Qin, J., Wang, J., Tan, Y., Huang, H., Xiang, X., & He, Z. (2020). Coverless image steganography based on generative adversarial network. *Mathematics*, 8(9), 1394.
- [14]. Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, A., & Szczypiorski, K. (2016). ("Information Hiding: Challenges For Forensic Experts") Information hiding in communication networks: Fundamentals, mechanisms, applications, and countermeasures. ("Information Hiding in Communication Networks: Fundamentals, Mechanisms ...") *John Wiley & Sons*.
- [15]. Zhang, R., Dong, S., & Liu, J. (2019). Invisible steganography via generative adversarial networks. *Multimedia Tools and Applications*, 78(7), 8559-8575.
- [16]. Ahmadi, M., Norouzi, A., Karimi, N., Samavi, S., & Emami, A. (2020). ReDMark: Framework for residual diffusion watermarking based on deep networks. ("Convolutional neural network-based image watermarking using discrete ...") *Expert Systems with Applications*, 146, 113157.
- [17]. Zarrabi, H., Emami, A., Khadivi, P., Karimi, N., & Samavi, S. (2020). BlessMark: A blind diagnostically lossless watermarking framework for medical applications based on deep neural networks. *Multimedia Tools and Applications*, 79(31-32), 22473-22495.
- [18]. High-capacity robust image steganography via adversarial network. (2020). *KSIIT Transactions on Internet and Information Systems*, 14(1).
- [19]. Lee, J. E., Seo, Y. H., & Kim, D. W. (2020). "Convolutional neural network-based digital image watermarking adaptive to the resolution of image and watermark." ("Sci-Hub | Convolutional Neural Network-Based Digital Image Watermarking ...") *Applied Sciences*.



- [20]. Liu, Y., Guo, M., Zhang, J., Zhu, Y., & Xie, X. (2019). ("A robust document image watermarking scheme using deep ... - Springer") "A novel two-stage separable deep learning framework for practical blind watermarking." ("A Novel Two-stage Separable Deep Learning Framework for Practical Blind ...") In *Proceedings of the 27th ACM International Conference on Multimedia*. ("The VIA Annotation Software for Images, Audio and Video")
- [21]. Luo, X., Zhan, R., Chang, H., Yang, F., & Milanfar, P. (2020). Distortion agnostic deep watermarking. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. ("2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition ...")
- [22]. Sharma, K., Aggarwal, A., Singhania, T., Gupta, D., & Khanna, A. (2019). ("Neural style transfer for image steganography and ... - Springer") "Hiding data in images using cryptography and deep neural network." ("Hiding Data in Images Using Cryptography and Deep Neural Network ...") *Journal of Artificial Intelligence and Systems*, 1(1), 143–162.
- [23]. Wang, K., Li, L., Luo, T., & Chang, C.-C. (2020). Deep neural network watermarking based on texture analysis. *Communications in Computer and Information Science*, 558–569.
- [24]. Wu, P., Yang, Y., & Li, X. (2018). StegNet: Mega image steganography capacity with deep convolutional network. *Future Internet*, 10(6), 54.
- [25]. Wu, P., Yang, Y., & Li, X. (2018). Image-into-Image Steganography Using Deep Convolutional Network. In *Advances in Multimedia Information Processing – PCM 2018* (pp. 792–802).
- [26]. Zhu, J., Kaplan, R., Johnson, J., & Fei-Fei, L. (2018). HiDDeN: Hiding data with deep networks. In *Computer Vision – ECCV 2018* (pp. 682–697).
- [27]. Lee, J.-E., Seo, Y.-H., & Kim, D.-W. (2020). "Convolutional neural network-based digital image watermarking adaptive to the resolution of image and watermark." ("Sci-Hub | Convolutional Neural Network-Based Digital Image Watermarking ...") *Applied Sciences*, 10(19), 6854. doi:10.3390/app10196854
- [28]. Pradhan, N., Singh Dhaka, V., & Chaudhary, H. (2019). Classification of human bones using deep convolutional neural network. *IOP Conference Series: Materials Science and Engineering*, 594(1), 012024.
- [29]. Thomas, T., Pradhan, N., & Dhaka, V. S. (2020). Comparative analysis to predict breast cancer using machine learning algorithms: A survey. In *2020 International Conference on Inventive Computation Technologies (ICICT)* (pp. 1–5). Coimbatore, India.
- [30]. Pradhan, N., Dhaka, V. S., Rani, G., & Chaudhary, H. (2020). Transforming view of medical images using deep learning. *Neural Computation & Applications*, 32(18), 15043–15054.
- [31]. Quiring, E., Arp, D., & Rieck, K. (2018). Forgotten siblings: Unifying attacks on machine learning and digital watermarking. In *2018 IEEE European Symposium on Security and Privacy (EuroSP)* (pp. 488–502).
- [32]. Zhong, X., Huang, P.-C., Mastorakis, S., & Shih, F. Y. (2020). An automated and robust image watermarking scheme based on deep neural networks. ("An Automated and Robust Image Watermarking Scheme Based on Deep Neural Networks") arXiv [cs.MM]. Retrieved from <https://arxiv.org/abs/2005.03709>

- [33]. Ming, Y., Ding, W., Cao, Z., & Lin, C.-T. (n.d.). "A general approach for using deep neural network for digital watermarking." ("[PDF] A General Approach for Using Deep Neural Network for Digital ...")
- [34]. Kandi, H., Mishra, D., & Gorthi, S. R. K. S. (2017). Exploring the learning capabilities of convolutional neural networks for robust image watermarking. ("Sci-Hub | Exploring the learning capabilities of convolutional neural ...") *Computer Security*, 65, 247–268.
- [35]. Zhang, J., Wang, N., & Xiong, F. (2003). "Hiding a logo watermark into the multiwavelet domain using neural networks." ("Hiding a Logo Watermark into the Multiwavelet Domain Using Neural ...") "In 14th IEEE International Conference on Tools with Artificial Intelligence, 2002." ("14th IEEE International Conference on Tools with Artificial ...") (ICTAI 2002). Proceedings (pp. 507–514). Washington, DC, USA.
- [36]. Fu, Z., Wang, F., & Cheng, X. (2020). The secure steganography for hiding images via GAN. *EURASIP Journal on Image and Video Processing*, 2020(1), 1-18.
- [37]. AlZubi, S., Islam, N. and Abbod, M. (2011). 'Multiresolution analysis using wavelet, Ridgelet, and Curvelet transforms for medical image segmentation', *International Journal of Biomedical Imaging*, 2011, pp. 1–18. doi:10.1155/2011/136034.
- [38]. AlZubi, S., Sharif, M. S., Islam, N., & Abbod, M. (2011). Multi-resolution analysis using curvelet and wavelet transforms for medical imaging. *2011 IEEE International Symposium on Medical Measurements and Applications*. <https://doi.org/10.1109/memea.2011.5966687>
- [39]. Kashyap, N., & Sinha, G. R. (2012). Image watermarking using 3-level discrete wavelet transform (DWT). *International Journal of Modern Education and Computer Science*, 4(3), 50-56.
- [40]. Lee, Y. S., Seo, Y. H., & Kim, D. W. (2019). "Blind image watermarking based on adaptive data spreading in n-level DWT subbands." ("Blind Image Watermarking Based on Adaptive Data Spreading in n-Level ...") *Security and Communication Networks*, 1-14.
- [41]. Wang, J., Lian, S., & Shi, Y.-Q. (2017). Hybrid multiplicative multi-watermarking in DWT domain. *Multidimensional Systems and Signal Processing*, 28(2), 617-636.
- [42]. Gao, Y., Wang, J., & Shi, Y.-Q. (2019). Dynamic multi-watermarking and detecting in DWT domain. *Journal of Real-Time Image Processing*, 16(3), 565-576.
- [43]. Giri, K. J., Quadri, S. M. K., Bashir, R., & Bhat, J. I. (2020). DWT based color image watermarking: a review. *Multimedia Tools and Applications*, 79(43-44), 32881-32895.
- [44]. AlZu'bi, S., Jararweh, Y., Al-Zoubi, H., Elbes, M., Kanan, T., & Gupta, B. (2018, December 13). Multi-orientation geometric medical volumes segmentation using 3D multiresolution analysis. *Multimedia Tools and Applications*, 78(17), 24223–24248. <https://doi.org/10.1007/s11042-018-7003-4>

- [45]. Alzu'bi, S., Islam, N., & Abbod, M.F. (2010). 3D Multiresolution Analysis for reduced features segmentation of medical volumes using PCA. *2010 IEEE Asia Pacific Conference on Circuits and Systems*, 604-607.
- [46]. Choudhary, R., & Parmar, G. (2016, March). A robust image watermarking technique using 2-level discrete wavelet transform DWT. In *2016 2nd International Conference on Communication Control and Intelligent Systems (CCIS)* (pp. 120-124). ("A comprehensive survey on robust image watermarking") IEEE.
- [47]. Talebi, S. (2020, December 21). The wavelet transforms. Towards Data Science. <https://towardsdatascience.com/the-wavelet-transform-e9cfa85d7b34>
- [48]. Thakral, S., & Manhas, P. (2019). Image processing by using different types of discrete wavelet transform. ("Image processing by using different types of Discrete wavelet transform") In *Communications in Computer and Information Science* (pp. 499–507). Singapore: Springer Singapore.
- [49]. al-Qerem, A., Kharbat, F., Nashwan, S., Ashraf, S., & blaou, K. (2020, March). General model for best feature extraction of EEG using discrete wavelet transform wavelet family and differential evolution. *International Journal of Distributed Sensor Networks*, 16(3), 155014772091100. <https://doi.org/10.1177/1550147720911009>
- [50]. Yufeng, G. (2017, December 06). Introduction to Kaggle Kernels. Towards Data Science. <https://towardsdatascience.com/introduction-to-kaggle-kernels-2ad754ebf77>
- [51]. Run data science & machine learning code online. (n.d.). Kaggle. <https://www.kaggle.com/code>
- [52]. The PASCAL visual object classes homepage. (n.d.). <http://host.robots.ox.ac.uk/pascal/VOC/>

Notes on contributors



Harish Sharma   is a research scholar at the department of Computer Science and Engineering at Manipal University Jaipur. He have B.E. (CSE) and MTech (ICT) in Machine Intelligence specialization. His research areas are image processing, natural language processing, and information retrieval. He is the recipient of the young researcher award at the university level. He has been awarded one international patent. He can be contacted at harish27j@gmail.com.



Dr. Sandeep Chaurasia is a Professor and Director of the School of Computer Science & Engineering (SCSE) at Manipal University Jaipur, India. He did his B.E (CSE), MTech (CSE), Ph.D. (CSE) on the topic "Cancer Diagnosis & Prognosis using Machine Learning". He has authored more than 45 publications in indexed journals, conference proceedings, and book chapters. "He is an SM-IEEE, LM-CSI, M-ACM, and member of Machine Intelligence Research labs. He is currently working in the applied areas of machine & deep learning, including

natural language processing and medical diagnosis. He can be contacted at email: sandeep.chaurasia@jaipur.manipal.edu



Dr. Nitesh Pradhan is an assistant professor in the Department of Computer Science and Engineering at Manipal University Jaipur, India. His research interest includes Machine Learning, Data Mining, and Artificial intelligence. He has 5 years of teaching experience. He has 2 copyrights named ‘CropBid’ and ‘NavPark’. He has published good research papers in reputed journals and conferences.

Ayush Singh is a student of BTech (Computer Science & Engineering) at Manipal University Jaipur. His skill sets basically includes Artificial Intelligence and Image processing.