ChainMark: Integrating an Invertible Neural Network and Blockchain for Ensuring Ownership Rights in Image Watermarking

Haeun Jo and Jungwon Seo*

Department of Software Engineering, Jeonbuk National University, Jeonju-si, Jeonbuk-do, 54896 Republic of Korea
E-mail: 202411873@jbnu.ac.kr; jungwonrs@jbnu.ac.kr
**Corresponding Author*

Received 05 December 2025; Accepted 14 April 2026

Abstract

The widespread use of generative AI has intensified issues related to digital ownership, even as it enhances the efficiency of digital content creation. While watermarking is a primary method for protecting these rights, existing neural network-based approaches often prioritize robustness and imperceptibility, neglecting verifiable ownership. To address this limitation, this paper proposes ChainMark, a system that integrates an invertible neural network (INN) with blockchain technology. ChainMark employs an INN trained within the discrete wavelet transform domain to embed watermarks that are resilient to diverse signal processing attacks. Crucially, unlike traditional approaches that rely solely on watermark extraction, the proposed system secures the verification process through a blockchain smart contract. Experimental results validate the system's theoretical security and demonstrate that

Journal of Web Engineering, Vol. 25_5, 945–976.
doi: 10.13052/jwe1540-9589.2558

the joint LH-HL model configuration achieves an optimal trade-off between visual quality and extraction accuracy. Consequently, ChainMark effectively guarantees creator rights by ensuring both high-performance watermarking and trustworthy ownership verification.

Keywords: Watermarking, blockchain, digital contents, data ownership, invertible neural network.

1 Introduction

Generative artificial intelligence (AI) has exerted a profound influence on society, particularly within digital media content markets [1]. With the rapid advancement of generative AI, digital content that is created by humans is frequently mistaken for content that is generated by AI, and material that is generated by AI is often perceived as being made by humans. The development of generative AI has lowered the barrier to entry in the creative digital content industry, fundamentally transforming the creative environment and greatly improving the efficiency and productivity of content production [2].

Despite the considerable benefits associated with generative AI, its deployment raises substantial concerns, and the most prominent among them is the issue of content imitation and copyright infringement in the digital domain [3]. Recent studies indicate that the legal discourse surrounding generative AI has intensified, resulting in multiple lawsuits filed against developers of systems such as ChatGPT on the grounds of unauthorized use of protected content [4]. These developments demonstrate that the risks posed by generative AI are not merely hypothetical but are materializing as tangible legal and ethical conflicts. As generative models continue to advance in scale and capability, disputes related to authorship, data ownership, and copyright violations are expected to grow in both frequency and complexity. Therefore, establishing rigorous ethical frameworks and implementing advanced technological safeguards has become essential in order to protect the ownership and legitimate attribution of digital content within the rapidly developing artificial intelligence ecosystem [5].

Watermarking techniques have been widely used and developed as a method to protect ownership since their earliest forms in traditional digital contents [6]. In the context of digital content, watermarking enables the explicit identification of content origin and ownership by embedding unique identification information, referred to as a watermark, either within or outside the digital contents [7]. This approach has gained particular

importance in digital images, which are especially vulnerable to misuse because digital images can be easily replicated or modified without noticeable degradation [8].

However, safeguarding digital images presents unique challenges compared to other digital content. Images routinely undergo varying degrees of signal processing. Traditional image watermarking schemes, which rely on fixed handcrafted features, often fail to maintain robustness against these image-specific manipulations. Furthermore, balancing watermark durability with visual quality is particularly difficult in images; embedding a signal strong enough to survive compression inevitably introduces visible artifacts that degrade the aesthetic value of the content. These limitations underscore the insufficiency of static algorithms in handling the complex distributions of modern digital images.

With the advancement of neural networks grounded in deep learning, watermarking techniques have evolved considerably [9, 10]. Neural network based watermarking approaches integrate both the embedding and extraction processes into a unified learning framework and can be trained under diverse attack conditions. This allows them to achieve significantly stronger robustness compared with conventional watermarking methods [11].

Among recent deep learning approaches, the invertible neural network (INN) has attracted growing interest due to its reversible forward and backward transformation capability, which enables watermark embedding and extraction within a single model without information loss. This property provides structural advantages over other neural networks where irreversible feature compression occurs.

Recent studies have leveraged INN to improve the robustness and reversibility of digital watermarking. Lu introduced an invertible steganography network to address the vulnerability of traditional watermarking to content-preserving manipulation attacks [12]. Also, Xu and He enhanced INN-based watermarking through distortion-aware modules and Transformer-driven invertible architectures [13, 14]. Extensions to multi-user watermarking and deeper invertible models have further broadened the applicability of INN networks, enabling parallel watermark streams and more expressive invertible techniques [15–17].

Despite these advancements, INN-based watermarking techniques share a critical conceptual limitation: they implicitly equate successful watermark extraction with proof of ownership. This assumption is fundamentally flawed, as a sufficiently capable adversary may extract the same watermark without possessing any legitimate rights. Consequently, watermarking alone cannot

guarantee ownership. True ownership verification requires an identity binding mechanism that securely links an extracted watermark to the rightful creator. This gap persists because current research continues to place predominant emphasis on robustness and invisibility rather than verifiable ownership authentication [18–20].

To address this limitation, this study proposes the ChainMark system, which incorporates a blockchain-based trust verification mechanism to overcome the inherent constraints of watermarking techniques that rely exclusively on extractability. The proposed system achieves high robustness and imperceptibility through an INN while validating ownership through a smart contract, thereby providing a transparent and reliable ownership verification process. The contributions of this paper are as follows:

- The ChainMark system integrates an invertible neural network with a blockchain-based smart contract. This system simultaneously satisfies the requirement for watermark robustness, provided by the invertible neural network, and the requirement for reliable ownership verification, ensured by the decentralized trust framework of the blockchain.
- A comparative analysis of different DWT subband configurations for INN-based watermarking is provided. Experimental results demonstrate that the combined use of the LH and HL subbands yields the optimal balance between perceptual quality and extraction robustness compared with configurations that employ independent channels or full subbands.

The remainder of this paper is organized as follows. Section 2 presents the background knowledge necessary for understanding the proposed approach. Section 3 provides a review of related work. Section 4 describes the proposed approach in detail, and Section 5 presents the experimental evaluation. Finally, Section 6 concludes the paper.

2 Preliminaries

2.1 Watermarking and Discrete Wavelet Transform

Watermarking embeds identification information into digital content to prevent unauthorized copying and verify ownership [21, 22]. Among various embedding techniques, frequency domain methods utilizing the discrete wavelet transform (DWT) are preferred for their superior robustness against signal processing attacks. DWT decomposes an image into four multi-resolution subbands: low-low (LL), low-high (LH), high-low (HL), and high-high (HH) [23,24]. In this study, DWT is employed as the preprocessing

layer for the neural network, allowing the system to embed watermarks into specific frequency components that balance perceptual invisibility with extraction resilience.

2.2 Invertible Neural Network (INN)

Unlike conventional convolutional neural networks (CNNs) where information loss occurs during feature compression, an invertible neural network (INN) guarantees lossless reversibility between the input space x and the latent space z via a bijective mapping ($f : x \to z, f^{-1} : z \to x$) [25]. This structural property is critical for the proposed system, as it enables both the embedding of the watermark into the latent representation and its subsequent extraction within a single unified model. By leveraging INN, our approach minimizes information loss during the embedding process, thereby achieving higher imperceptibility compared to non-invertible architectures [26].

2.3 Blockchain and Smart Contract

Blockchain serves as a decentralized, immutable ledger that ensures data integrity and transparency, eliminating the need for a central authority. In the context of digital rights management, it provides a tamper-proof record of ownership. Specifically, this study utilizes programmable blockchain platforms (e.g., Ethereum) to deploy a smart contract that is self-executing code that automatically enforces predefined rules. By integrating smart contracts, the proposed system moves beyond passive record-keeping to active verification, where ownership proofs are cryptographically validated on-chain to prevent forgery or unauthorized alteration of copyright data.

3 Related Works

This section reviews prior studies on digital watermarking and blockchain enabled ownership protection, with emphasis on methodologies applied to digital images and related media domains. Existing research can be broadly categorized into deep learning based watermarking, INN based watermarking, and blockchain or non-fungible token (NFT) based provenance systems.

Deep learning watermarking approaches significantly improve robustness over traditional methods but remain fundamentally extraction driven. Padhi introduced a dual framework combining perceptual and cryptographic hashes to resist overwriting attacks [27], while Fu proposed a wavelet domain CNN

encoder and recovery decoder to address degradation arising from screen captured images [28]. These methods enhance resistance to distortion yet still rely solely on extracted watermarks for ownership claims.

INNs enable reversible embedding through bidirectional transformations. Lu proposed an INN based steganographic framework resilient to content preserving manipulations [12]. Xu improved robustness using a distortion aware message processor module [13], and He developed IWFormer, a Transformer enhanced INN architecture for stable extraction [14]. For multi user watermarking, Shi introduced a parallel INN to eliminate channel interference [15], and Ni expanded the expressive capacity of INN through frequency transform based feature separation [16]. StarINN further increased robustness through multi scale loss optimization [17]. Despite architectural advances, all INN based methods still treat extraction success as equivalent to ownership verification.

INN watermarking has been extended to neural scene and video representations. RWNeRF preserves watermark integrity across viewpoint changes in NeRF renderings [29], and MarkINeRV maintains watermark persistence in video oriented neural representations [30]. In audio, Zhu designed an INN based watermarking method tailored to perceptual constraints [31]. Although these works demonstrate versatility across media types, they remain bound to extraction only ownership assumptions.

Blockchain based watermarking focuses primarily on storing integrity metadata on the chain. Mannepalli applied blockchain to medical image watermarking to create permanent audit trails [32]. Broader NFT based systems record provenance for AI models [33], embed owner and buyer identifiers for leakage tracing [34], support metaverse copyright governance [35], authenticate deepfake and synthetic assets [36], and preserve 3D model authenticity using Fourier fingerprint signatures stored as NFTs [37]. However, in all cases, blockchain functions as a passive ledger and does not enforce identity binding between embedded watermarks and registered owners.

Table 1 summarizes the core techniques and provenance mechanisms across prior studies. As Table 1 indicates, deep learning and INN based watermarking systems provide robustness but lack verifiable identity binding, while blockchain and NFT systems provide provenance but are not coupled with signal level watermark verification. No existing method integrates robust INN watermarking with blockchain verified, key bound ownership authentication. The proposed ChainMark system fills this gap by combining DWT domain INN watermark embedding with a smart contract based verification

Table 1 Comparison of related works

Refs.	Core Technique	Ownership Technique
[27]	Dual DL watermark (perceptual + crypto hash)	Watermark extraction-based ownership
[28]	Wavelet CNN + recovery decoder	Watermark extraction-based ownership
[12]	INN high-capacity steganography	Watermark extraction-based ownership
[13]	INN + message processor	Watermark extraction-based ownership
[14]	Transformer-INN (IWFormer)	Watermark extraction-based ownership
[15]	Parallel INN (PINN)	Watermark extraction-based ownership
[16]	Frequency-transform multi-branch INN	Watermark extraction-based ownership
[17]	Efficient multi-scale INN	Watermark extraction-based ownership
[29]	INN for 3D rendering	Watermark extraction-based ownership
[30]	INN for neural video representation	Watermark extraction-based ownership
[31]	INN for perceptual constraint	Watermark extraction-based ownership
[32]	Edge/DWT watermarking	Blockchain = immutable integrity log
[33]	Non-fungible token	NFT provenance (no watermark)
[34]	ID-embedded watermark + NFT	Blockchain provenance + leakage tracing
[35]	NFT governance for digital works	Blockchain = immutable integrity log
[36]	Non-fungible token	Blockchain = immutable integrity log
[37]	Non-fungible token	Blockchain = immutable integrity log
Proposed	DWT-domain INN with split-channel watermark embedding	Blockchain-verified, key-bound ownership

framework, enabling both robust watermark extraction and cryptographically verifiable ownership.

4 Proposed Approach

4.1 ChainMark System Overview

The proposed ChainMark system integrates an INN with a blockchain smart contract to achieve both robust watermark embedding and verifiable ownership authentication. Figure 1 provides an overview of the watermark embedding, extraction, and ownership verification workflow. The complete procedure consists of two primary phases:

- **Phase 1: Encoding (watermark embedding)**
  1. The input image is decomposed into frequency bands (LL, LH, HL, HH) through the discrete wavelet transform (DWT).

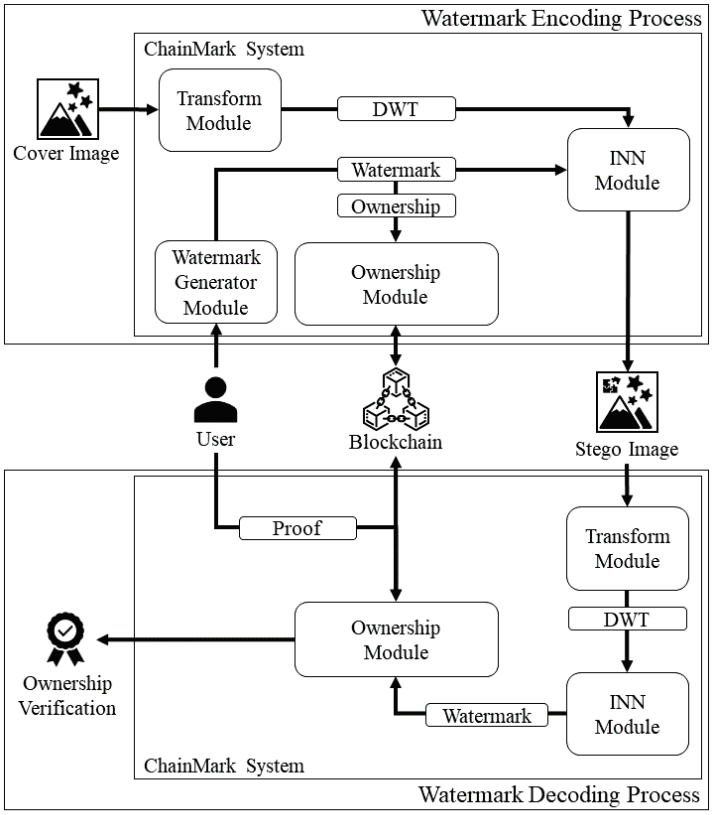

Figure 1 ChainMark system process overview.

2. A cryptographic watermark is generated from the user's secret key.
3. The watermark is mapped into a latent representation suitable for INN processing.
4. The INN encoder embeds the latent watermark into the LH and HL subbands using a split-channel embedding scheme.
5. The modified coefficients are recombined using the inverse DWT, producing the final watermarked image.

- **Phase 2: Decoding (ownership verification)**

1. The verification image undergoes DWT decomposition to extract the LH and HL subbands.
2. The INN decoder reconstructs the latent watermark from these subbands.

3. The latent watermark is decoded back into its original cryptographic form.
4. The claimant submits the extracted watermark along with their public key parameters for verification.
5. A smart contract validates the key–watermark pair against the on-chain record using a DLP-based proof.
6. Ownership is confirmed if the on-chain verification succeeds.

The ChainMark framework involves four main entities: the user, the ChainMark system, the blockchain network, and the smart contract responsible for enforcing verifiable ownership.

The user interacts with the ChainMark system to generate a unique watermark that is derived from a secret key (sk). This watermark is then embedded into an image or extracted from it when necessary. In the context of this system, the user is defined as the entity that possesses the capability to claim legitimate ownership of any image containing the generated watermark.

Upon a request from a user, the ChainMark system embeds a unique watermark into an image and subsequently restores or verifies it when needed. When ownership verification is initiated, the verification process is automatically executed through interaction with the smart contract on the blockchain. The ChainMark system comprises four distinct modules:

- *Transform module:* Applies the DWT to extract frequency components from the original image.
- *Watermark generator module:* Generates a unique watermark derived from the sk of the user.
- *INN module:* Utilizes an INN to perform both the embedding and extraction of watermarks.
- *Ownership module:* Verifies ownership through the smart contract deployed on the blockchain.

The ChainMark system is compatible with various types of blockchains, including public, private, and consortium networks. In this study, a private blockchain environment is assumed. This environment is selected for its support of programmable smart contract execution and its operation without gas fees. The smart contracts are implemented using the Solidity language. These contracts utilize structural keywords, such as *modifier* and *constructor*, to define conditional execution and enforce access control mechanisms.

To facilitate a clear understanding of the mathematical framework underlying the ChainMark system, the key notations and symbols used throughout this paper are summarized in Table 2. The nomenclature is organized into

Table 2 Main notation

Symbol	Description
$x,\ x_{\text{stego}}$	Original (cover) image and the corresponding watermarked (stego) image.
$f,\ f^{-1}$	Forward ($x \to z$) and inverse ($z \to x$) functions of the INN.
$z,\ z_{\text{ch}},\ z'$	Latent vector, channels for embedding, and extracted latent vector.
J	Jacobian matrix of the INN transformation w.r.t input image.
L_{base}	Base INN loss.
$L_{\text{dist}},\ L_{wm}$	Distortion loss (visual quality) and watermark reconstruction loss.
$\lambda_{z,j},\ I_g$	Hyperparameters for latent regularization and Jacobian determinant.
$\lambda_{\text{d,m}},\ \gamma$	Weights for distortion/magnitude loss and scaling factor for extraction.
$F_p^*,\ p,\ g$	Multiplicative group, large prime modulus, and generator.
$sk,\ se$	User's private key and auxiliary secret value.
$pv,\ fv$	Public value (g^{sk}) and verification value (g^{se}).
$w,\ w_{in}$	Original watermark derived from keys and extracted watermark.
$sp,\ lv$	Secret parameter and proof value for ownership verification.
$i,\ c$	Image identifier and on-chain commitment hash.
$\mathbb{A}$	Adversary modeled in the security analysis.
$ACC,\ NC$	Bit-wise accuracy and normalized correlation metrics.
$s_p,\ \tau_p$	Similarity score and threshold for ownership acceptance.

four primary categories: (1) cryptographic and blockchain parameters, which define the keys (sk, se), group structures (F_p^*), and on-chain commitments (c) required for secure ownership binding; (2) INN model variables, representing the image transformations ($x \to z$), latent channel allocations (z_{ch}), and Jacobian determinants; (3) loss function terms, including the specific weights (λ, β) used to optimize the trade-off between imperceptibility and robustness during the two-phase training; and (4) evaluation metrics, such as the similarity scores (s_p) and thresholds (τ_p) used for verification. Consistent use of these symbols ensures rigorous definitions in the subsequent methodology and security analysis sections.

4.2 Security Foundations

4.2.1 Hash function

A hash function converts input data of arbitrary length into an output value of a fixed length. Hash functions are generally differentiated into two categories: cryptographic and non-cryptographic. Non-cryptographic functions are optimized for high computational speed and uniform hash value distribution, and are commonly used in applications such as data retrieval. In contrast,

cryptographic hash functions are designed for security and possess three essential properties: collision resistance, pre-image resistance, and second pre-image resistance.

- *Collision resistance* means that it is computationally infeasible to find two distinct inputs $x \neq x'$ that produce the same output $H(x) = H(x')$.
- *Pre-image resistance* ensures that, for a given hash value y, it is computationally difficult to determine the corresponding input x.
- *Second pre-image resistance* implies that, for a given input x, it is difficult to find another input $x' \neq x$ that generates the same output.

Cryptographic hash functions are widely used in digital signatures, data integrity verification, and blockchain. The proposed system utilize the SHA-256 algorithm, which generates a 256-bit fixed length output. This ensures cryptographically strong collision resistance and compatibility with standard blockchain verification processes.

4.2.2 Discrete logarithm problem

The discrete logarithm problem (DLP) refers to the computational difficulty inherent in inverting an exponentiation operation within a finite field or group. For instance, in ordinary arithmetic, $2^x = 8$ easily yields $x = 3$. However, when modular arithmetic is introduced, the problem becomes significantly harder. Given a prime p, a generator $g \in F_p^*$, and an integer x such that $y = g^x \mod p$, it is computationally infeasible to efficiently recover x even if y, g and p are known. Because of this asymmetry, the DLP serves as a fundamental component in many cryptographic protocols. In the proposed approach, to guarantee security against modern computational attacks, the system utilizes a safe prime p with a length of at least 2048 bits. This parameter choice ensures that solving the DLP for the private key recovery remains computationally infeasible.

4.3 Security Requirements

The proposed ChainMark system is designed to satisfy the following security requirements. First, *ownership* verification must be supported, ensuring that the system can prove that a user is the legitimate owner of an image through the embedded watermark. This allows the rightful owner to be clearly identified even if a third party intentionally alters or damages the images. Second, the system must provide *unforgeability*, meaning that an attacker cannot impersonate a legitimate owner to generate a counterfeit watermark or modify an existing one.

4.4 Module Design

4.4.1 Transform module design

The Transform module applies a discrete wavelet transform (DWT) to decompose an input image into the LL, LH, HL, and HH bands of frequency through multi-resolution analysis. From these components, a predefined set of subbands is selected to serve as the input for the INN. The selected components of frequency are then forwarded to the INN module, which performs both watermark embedding and subsequent extraction. These operations can be implemented in Python by using the PyWavelets package [38]. To ensure reproducibility of the signal decomposition and reconstruction process, the precise operational steps are formalized in Algorithm 1.

Algorithm 1 Frequency transformation

Require: Cover Image x
Ensure: Stego Image x_{stego}
Procedure FORWARDTRANSFORM(x)
  $\{LL, LH, HL, HH\} \leftarrow \mathrm{DWT}(x)$
  return $\{LH, HL\}$ {Subbands for embedding}
Procedure INVERSETRANSFORM($LL, LH_{new}, HL_{new}, HH$)
  $x_{stego} \leftarrow \mathrm{IDWT}(LL, LH_{new}, HL_{new}, HH)$
  return x_{stego}

4.4.2 Watermark generator module design

The watermark generator module produces a unique watermark from a set of confidential system parameters $\{sk, F_p^*, p, g, se\}$. Here, $sk \in F_p^*$ denotes a private key, F_p^* represents the multiplicative group of integers modulo p, p is a large prime number, $g \in F_p^*$ denotes a generator, and $se \in F_p^*$ with $se \neq sk$ denotes an independent secret element. The public values are computed as $pv \equiv g^{sk} \pmod{p}$ and $fv \equiv g^{se} \pmod{p}$. A scalar watermark w is derived by hashing and mapping the two secrets to the exponent ring as $w \equiv \texttt{H2I}(\mathrm{hash}(sk \parallel se)) \pmod{(p-1)}$, where `H2I` converts the hash output to an integer form suitable for modular arithmetic. The watermark w is then delivered to the INN module for embedding in the latent space. Algorithm 2 provides a structured definition of this cryptographic derivation and parameter generation process.

4.4.3 INN module design

The INN module consists of multiple reversible blocks, each capable of performing forward and inverse transformations without information loss.

Algorithm 2 Watermark generation

Require: Private Key sk, Secret Element se, Public Parameters (p, g)
Ensure: Watermark w, Verification Values (pv, fv)
1: **Step 1: Compute Verification Values**
2: $pv \leftarrow g^{sk} \pmod{p}$
3: $fv \leftarrow g^{se} \pmod{p}$
4: **Step 2: Derive Watermark**
5: $w \leftarrow \texttt{H2I}(\text{hash}(sk \parallel se)) \pmod{(p-1)}$
6: **return** w, pv, fv

Each block includes a convolutional layer with a kernel of size 3×3 and 32 output channels, followed by a ReLU activation function that introduces nonlinearity, and another convolutional layer of the same configuration that refines the learned representation. This architecture enables the model to capture complex feature distributions while ensuring invertibility for reconstruction.

The training of the INN model is divided into two stages, referred to as *Phase A* and *Phase B*, in order to stabilize the representation of the latent space and optimize imperceptibility and robustness. The basic loss function of the INN is defined as:

$$L_{\text{base}} = \lambda_z \times I_g \times \text{mean}(z^2) - \lambda_j \times \text{mean}(\log|\det J|) \tag{1}$$

In Equation (1), λ_z represents the latent space regularization weight, and I_g denotes a coefficient that adjusts the scale of the loss term. The term $\text{mean}(z^2)$ is the mean squared value of the latent vector z, which encourages z to follow a stable normal distribution. λ_j is the weight assigned to the Jacobian determinant term, and $\text{mean}(\log|\det J|)$ represents the mean of the logarithm of the Jacobian determinant. This term preserves the information content of the mapping and ensures the stability of the inverse transformation.

Phase A: Latent space stabilization

The objective of *Phase A* is to secure the designated latent channels (z_{ch}) where watermark data will be embedded. The loss function, L_A, is defined as:

$$L_A = L_{\text{base}} + 0.05 \times \text{mean}(z_{ch}^2) \tag{2}$$

The additional term in Equation (2), $\text{mean}(z_{ch}^2)$, minimizes potential image distortion that may arise during watermark embedding by constraining energy in the watermark channels.

Phase B: Imperceptibility and robustness optimization

The goal of *Phase B* is to jointly optimize imperceptibility and robustness through scenarios of simulated attacks. During this phase, the model embeds the watermark into the secure latent region and learns to maintain watermark integrity under compression, noise, or blur attacks. The *Phase B* loss function is expressed as:

$$L_B = L_{\text{base}} + \lambda_{dist} \times L_{dist} + \beta \times L_{wm} \tag{3}$$

In Equation (3), $L_{dist} = \text{mean}((x_{stego} - x)^2)$ ensures that the stego image remains visually indistinguishable from the original, and λ_{dist} controls the weight for imperceptibility. The coefficient β in Equation (3) dynamically increases according to a cosine annealing schedule from 0 to β_{max}. This scheduling enables a gradual transition from optimization that is focused on imperceptibility to optimization that is focused on robustness as training proceeds. The loss for watermark reconstruction, L_{wm}, is defined as:

$$L_{wm} = (1 - \text{CosineSimilarity}(w, w_{in})) + \lambda_{mag} \times L_1 \tag{4}$$

The first term in Equation (4) evaluates the angular difference between the reconstructed and reference watermark vectors, enforcing structural consistency. The second term, which is weighted by λ_{mag}, stabilizes recovery by penalizing large absolute deviations.

Watermark embedding mechanism

The watermark is embedded within the latent space z by using an embedding strategy that splits channels. The bit sequence of the watermark is divided into two independent streams, each expanded into a map of two dimensions matching the spatial dimensions of the latent space. The resulting maps are embedded into the LH and HL latent channels that are related to frequency. This distributed embedding strategy leverages the distinct characteristics of the LH and HL frequency components, which can enhance robustness against attacks that disproportionately affect specific directional features.

Watermark extraction mechanism

During extraction, a target image is processed by the trained INN model to obtain its latent representation z'. Due to *Phase A* regularization, z' of a clean image approximates a distribution that is centered at zero, whereas that of an image that is watermarked contains the embedded watermark signal z'_{wm} that may be distorted by attacks. The signal z'_{wm} is normalized and then amplified

by a scaling factor γ to increase separation from the decision boundary. A sign function is applied to the amplified values to recover binary bits, and the bit streams from the LH and HL channels are concatenated to reconstruct the full watermark. The extracted watermark w_{in} is subsequently forwarded to the ownership module for verification.

To clarify the logical flow of these reversible operations, the embedding and extraction procedures are explicitly defined in Algorithm 3.

Algorithm 3 INN-based embedding and extraction

Require: Subbands $\{LH, HL\}$, Watermark w, INN Functions f, f^{-1}
Ensure: Modified Subbands $\{LH_{new}, HL_{new}\}$, Extracted Watermark w_{in}
Procedure EMBED($\{LH, HL\}, w$)
  $z \leftarrow f(LH, HL)$ {Map to latent space}
  $z_{ch} \leftarrow w$ {Embed w into reserved latent channels}
  $\{LH_{new}, HL_{new}\} \leftarrow f^{-1}(z)$ {Inverse transformation}
  return $\{LH_{new}, HL_{new}\}$
Procedure EXTRACT(x_{stego})
  $\{LL, LH, HL, HH\} \leftarrow \mathrm{DWT}(x_{stego})$
  $z' \leftarrow f(LH, HL)$ {Forward transformation}
  $w_{in} \leftarrow \mathrm{Sign}(z' \times \gamma)$ {Extract from watermark channels}
  return w_{in}

4.4.4 Ownership module design

The smart contract is implemented in Solidity with access control defined by *modifier* and initialization specified by *constructor*. After deployment, the contract records a commitment that binds the image identifier i to the verification value fv by computing $c = \mathrm{hash}(i \parallel fv)$. Data that is on the chain is immutable, and modification or deletion of stored values is not permitted. The parameters p and g are defined as global constants. The contract also defines weighting parameters δ and ε satisfying $\delta + \varepsilon = 1$, which are used in ownership verification to combine similarity metrics.

To verify the ownership of an image identified by i, the claimant transmits the watermark value w and verification parameters derived from private secrets: $sp \equiv se - w \times sk \pmod{(p-1)}$ and $lv \equiv g^{sp} \times (pv)^{w} \pmod{p}$.

The smart contract receives the extracted watermark w_{in} from the INN module and compares it with the submitted value w. First, it validates the cryptographic proof. If valid, it then checks whether $w_{in} = w$ (exact match). If the image has been degraded ($w_{in} \neq w$), the contract computes the bit accuracy and normalized correlation to measure similarity as defined in

Equations (5) and (6):

$$ACC = \frac{1}{N}\sum_{t=1}^{N}\zeta(w(t), w_{in}(t)) \tag{5}$$

where N is the total number of bits and $\zeta(a, b) = 1$ if $a = b$, and 0 otherwise. The normalized correlation is calculated as:

$$NC = \frac{\sum_{t=1}^{N} w(t)w_{in}(t)}{\sqrt{\sum_{t=1}^{N} w(t)^2}\sqrt{\sum_{t=1}^{N} w_{in}(t)^2}}. \tag{6}$$

A similarity score $s_p = \delta \times ACC + \varepsilon \times NC$ is computed, and the ownership claim is accepted if $s_p \geq \tau_p$, where τ_p denotes a predefined threshold. This verification logic is described in Algorithm 4.

Algorithm 4 Smart contract ownership verification

Require: Extracted w_{in}, Claimed w, Proof (sp, lv), On-chain fv, pv
Ensure: Verification Result (True/False)
Step 1: Cryptographic Validation (Zero-Knowledge Proof)
if $g^{sp} \times (pv)^{w} \pmod p \neq fv$ **then**
  return false{Invalid Proof}
end if
Step 2: Watermark Comparison
if $w_{in} == w$ **then**
  return true{Valid ownership (No attack)}
end if
Step 3: Check Robustness (Similarity Score)
$ACC \leftarrow \frac{1}{N}\sum \zeta(w, w_{in})$ {See Eq. (5)}
$NC \leftarrow \frac{\sum w \cdot w_{in}}{\sqrt{\sum w^2}\sqrt{\sum w_{in}^2}}$ {See Eq. (6)}
$s_p \leftarrow \delta \times ACC + \varepsilon \times NC$
if $s_p \geq \tau_p$ **then**
  return true{Valid ownership (Under attack)}
else
  return false
end if

5 Experiments

5.1 Security Analysis

This section presents security models for the defined security requirements, *ownership verification* and *unforgeability*, and proves the security of each model. The security of each model is defined as in Definition 1.

Definition 1: A security requirement is defined as secure if the probability that an adversary $\mathbb{A}$ wins in a defined security game is considered cryptographically negligible.

5.1.1 Ownership verification security analysis

The security game for the ownership verification requirement is defined as follows.

- **Setup:** The challenger (ch) generates the system parameters $\{g, p\}$, a user private key sk, and a secret value se. It computes the corresponding values $pv \equiv g^{sk} \mod p$ and $fv \equiv g^{se} \mod p$. The ch then registers the commitment $c = hash(i \parallel fv)$ on the smart contract and publishes the public parameters $\{g, p, pv\}$ to the $\mathbb{A}$.
- **Query:** The $\mathbb{A}$ can submit ownership verification queries to the ch. If $\mathbb{A}$ provides an arbitrary proof value, the ch responds with *True* or *False* depending on whether the submitted proof is valid.
- **Forge:** If $\mathbb{A}$ generates a new lv' such that it satisfies $g^{sp'} \cdot pv^{w'} \equiv fv \mod p$, then $\mathbb{A}$ is considered to have won the game.

Lemma 1: An adversary $\mathbb{A}$ cannot forge a valid ownership proof unless it can solve the discrete logarithm problem.

Proof 1: To generate a valid ownership proof, $\mathbb{A}$ must determine sp and w that satisfy $g^{sp} \times (pv)^{w} \equiv fv \mod p$. Since $sp = se - w \times sk \mod (p-1)$, obtaining a valid sp requires knowledge of either the secret key sk or the secret value se. However, deriving sk from the public value $pv = g^{sk} \mod p$ or se from $fv = g^{se} \mod p$ is equivalent to solving the discrete logarithm problem. Under the assumption that this problem is computationally infeasible, the probability that $\mathbb{A}$ can forge a valid ownership proof in polynomial time is negligible, as stated in Definition 1.

Lemma 2: Even if an image is corrupted, an $\mathbb{A}$ cannot forge a valid ownership proof unless it can solve the discrete logarithm problem.

Proof 2: If the image is corrupted, the system requires two conditions to be met for verification: the cryptographic proof (lv) must be valid, and the similarity score (s_p) between the claimed watermark (w) and the extracted watermark (w_{in}) must exceed the threshold (τ_p). An attacker $\mathbb{A}$ must therefore succeed at two tasks. First, $\mathbb{A}$ must forge a valid cryptographic proof (lv', w') that satisfies the group relation $g^{sp'} \cdot pv^{w'} \equiv fv \mod p$. As established

in Proof 1, this task is computationally infeasible as it requires solving the discrete logarithm problem. Second, $\mathbb{A}$ must also ensure that the claimed watermark w' is sufficiently similar to the watermark w_{in} that is extracted from the corrupted image. Because the attacker cannot pass the first check regardless of the similarity score, the joint probability of a successful forgery remains cryptographically negligible.

5.1.2 Unforgeability security analysis

The security game for the watermark unforgeability requirement is defined as follows.

- **Setup:** The ch generates the system parameters $\{g, p\}$, a user private key sk, and a secret value se. Based on these values, the watermark is computed as $w \equiv \texttt{H2I}(\text{hash}(sk \mid\mid se)) \mod (p-1)$. The ch provides the w to the $\mathbb{A}$.
- **Query:** The $\mathbb{A}$ may query the ch with arbitrary inputs and receive the corresponding hash outputs.
- **Forge:** The $\mathbb{A}$ wins the game if one of the following two conditions is satisfied:
  1. $\mathbb{A}$ finds the original input $(sk \mid\mid se)$ that corresponds to the given watermark w.
  2. $\mathbb{A}$ finds a new input $(sk' \mid\mid se') \neq (sk \mid\mid se)$ such that $w \equiv \texttt{H2I}(\text{hash}(sk \mid\mid se)) \mod (p-1)$.

Lemma 3: An $\mathbb{A}$ cannot forge a new watermark or alter an existing one unless it can violate the standard security properties of the cryptographic hash function.

Proof 3: In the ChainMark system, the watermark is defined as $w \equiv \texttt{H2I}(\text{hash}(sk \mid\mid se)) \mod (p-1)$. Therefore, forging or modifying a watermark is equivalent to breaking the core properties of a secure cryptographic hash function. The first property, *pre-image resistance*, ensures that even if the $\mathbb{A}$ obtains the w, it is computationally infeasible to reverse the H2I, modulo, and hash operations to retrieve the original input $(sk \mid\mid se)$. The second property, *second pre-image resistance* guarantees that it is also infeasible for the $\mathbb{A}$ to generate a different input $(sk' \mid\mid se')$ that results in the same output w after all transformations. Under the assumption that a secure hash function with these properties is employed, the probability that an $\mathbb{A}$ can forge or tamper with a valid watermark is cryptographically negligible.

5.2 Performance Evaluations

This section presents a comprehensive performance evaluation of the proposed ChainMark system. The experiments were designed to assess the effectiveness and reliability of the invertible neural network, which serves as the core component of the ChainMark system.

5.2.1 Experimental settings

The experiments were conducted under the following hardware environment. The system was equipped with an AMD RYZEN 9 9900x processor with 12 cores, 64 GB of memory and an NVIDIA GeForce RTX 5080 GPU. For image training, 10,000 images were randomly selected from the COCO Image Dataset, and 1000 images were randomly chosen from the Google Open Image Dataset for evaluation.

The INN network was trained with parameter settings designed to ensure robustness against diverse attack scenarios. For JPEG compression, quality factors of 50, 60, 70, 80, and 90 were used. For Gaussian blur, the standard deviation was set to 0.5, 1.0, 2.0, 3.0, 4.0 and 5.0. Additionally, Gaussian noise was applied with variance values ranging from 0.01 to 0.07, increasing in steps of 0.01.

5.2.2 Models for comparison

Three distinct INN models were trained to compare different embedding strategies. The first, the *ONLY* model (a joint LH-HL model), embeds watermark information simultaneously into the LH and HL subbands within a single network, improving embedding efficiency through shared feature representation. The second, the *BOTH* model (an independent LH-HL model), trains two separate networks for the LH and HL subbands and combines their outputs during embedding, thereby allowing examination of the effect of independent learning for each frequency. Finally, the *FULL* model (a full subband model), trains using all four DWT subbands (LL, LH, HL, HH), allowing comprehensive evaluation of the entire frequency spectrum.

5.2.3 Evaluation metrics

After generating and embedding the watermark, the perceptual quality of the images that were watermarked was evaluated using the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM). Subsequently, the watermark was extracted from both the original and attacked images, and its

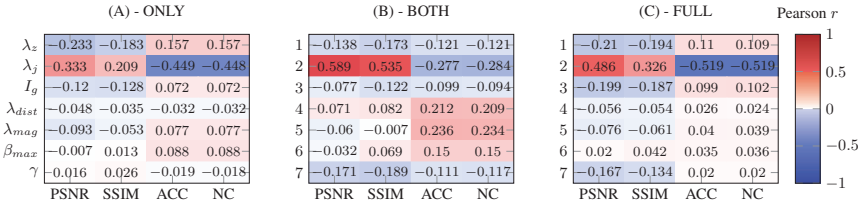

Figure 2 Correlation analysis between hyperparameters and performance metrics for the three INN models. Each heatmap visualizes the Pearson correlation coefficient between seven hyperparameters and four evaluation metrics. The legend indicates that red values represent a strong positive correlation and blue values represent a strong negative correlation. (A) results for the ONLY (joint LH-HL) model; (B) results for the BOTH (independent LH-HL) model; (C) results for the FULL (full subband) model.

similarity to the original watermark was measured using accuracy (ACC) and normalized correlation (NC).

5.2.4 Parameter selection

Figure 2 presents a heatmap visualization of the experimental results that were obtained by varying key hyperparameters across the three INN models (*ONLY, BOTH*, and *FULL*). Figure 2 illustrates how changes in weighting parameters affect the four primary metrics for evaluation, which are PSNR, SSIM, ACC, and NC, under both clean and attacked conditions (e.g., compression, noise, and blur).

Before analyzing the weighting parameters in Figure 2, preliminary experiments were conducted to determine the optimal structural parameters: the number of reversible blocks (BLOCKS), the number of training iterations (EPOCHS), and the batch size (BATCH).

- *BLOCKS*: Increasing the number of blocks strengthened robustness against blur attacks but resulted in degraded image quality and performance deviations against other attacks.
- *EPOCHS*: Increasing the number of training iterations resulted in a clear trade-off where the overall robustness to attacks (ACC) improved, but the perceptual quality (PSNR) decreased.
- *BATCH*: It was difficult to find a consistent trend when changing the batch size, as the advantages and disadvantages for each attack type fluctuated.

Based on these observations, a configuration that showed the most balanced outcome was chosen. When the number of blocks was set to 10, the training epochs to 200, and the batch size to 8, each performance indicator

achieved the highest average value. This outcome suggests that the balance between model complexity and training stability is optimized under this configuration. With these structural parameters fixed, a sensitivity analysis was performed on the seven hyperparameters shown in Figure 2: λ_z, λ_j, I_g, λ_{dist}, λ_{mag}, β_{max}, and γ.

In the *ONLY* model, λ_j showed the strongest overall correlation. It exhibited a moderate negative relationship with ACC and NC and a moderate positive relationship with PSNR and SSIM. This observation reveals a critical trade-off: increasing λ_j forces the model to be more perfectly invertible, which enhances the fidelity of the image reconstruction (PSNR/SSIM). However, this same pursuit of perfect invertibility also cleans the latent space more effectively, weakening the embedded watermark signal and thereby reducing the performance of recognition (ACC/NC).

For the *BOTH* model, λ_j demonstrated a strong positive relationship with PSNR and SSIM and a strong negative relationship with ACC and NC. This again highlights the same fundamental conflict between reconstruction fidelity and watermark robustness. In contrast, λ_{dist} and λ_{mag} exhibited mild positive relationships with metrics related to accuracy.

In the *FULL* model, λ_j dominated the correlation structure, exhibiting the same trade-off by positively influencing perceptual quality but negatively affecting measures related to recognition. Guided by this correlation analysis, which identified the critical influence of λ_j and the supplementary role of other parameters, an extensive search was conducted to determine the optimal configuration. After experimentation with multiple combinations, the best performance was obtained under the following settings: $\lambda_z = 0.07$, $\lambda_j = 0.07$, $I_g = 1.35$, $\lambda_{dist} = 250$, $\lambda_{mag} = 0.2$, $\beta_{max} = 150$, $\gamma = 110$. This configuration provided a balanced outcome across all four evaluation metrics (PSNR, SSIM, ACC and NC). Notably, simultaneous improvement in PSNR and SSIM and stable maintenance of ACC were observed under these settings.

5.2.5 Performance analysis

Table 3 presents the PSNR, SSIM, ACC and NC results for the three models. In terms of computational cost, the training times for the *ONLY*, *BOTH*, and *FULL* models were 128.54 minutes, 248.25 minutes, and 133.51 minutes, respectively. The *BOTH* model, which trains two independent networks, required significantly more time due to its architecture.

The models were trained against a wide range of attack scenarios, which included Gaussian blur with values from 0.5 to 5.0 and Gaussian noise with

Table 3 Performance by attack for each models

Model	Attack	PSNR	SSIM	ACC(std)	NC(std)
ONLY	Clean	34.07	0.9700	1.0000 (0.0000)	1.000 (0.000)
	JPEG 90	33.20	0.9560	0.9142 (0.0710)	0.828 (0.142)
	JPEG 80	32.34	0.9418	0.8523 (0.0858)	0.705 (0.172)
	JPEG 70	31.60	0.9294	0.8049 (0.0854)	0.610 (0.171)
	JPEG 60	30.96	0.9183	0.7598 (0.0829)	0.520 (0.166)
	JPEG 50	30.42	0.9081	0.7236 (0.0768)	0.447 (0.154)
	Blur 0.5	32.57	0.9557	0.9620 (0.0308)	0.924 (0.062)
	Noise 0.01	32.82	0.9370	0.9648 (0.0463)	0.930 (0.093)
	Noise 0.02	30.69	0.8628	0.9155 (0.0685)	0.831 (0.137)
	Noise 0.03	30.40	0.7800	0.8727 (0.0762)	0.745 (0.152)
BOTH	Clean	28.74	0.9002	0.6711 (0.0822)	0.342 (0.164)
	JPEG 90	28.47	0.8863	0.7276 (0.0315)	0.455 (0.063)
	JPEG 80	28.16	0.8742	0.7186 (0.0338)	0.437 (0.068)
	JPEG 70	27.79	0.8583	0.6932 (0.0359)	0.386 (0.072)
	JPEG 60	27.56	0.8563	0.6589 (0.0429)	0.318 (0.086)
	JPEG 50	27.37	0.8543	0.6412 (0.0396)	0.282 (0.079)
	Blur 0.5	29.74	0.9236	0.6773 (0.0769)	0.355 (0.154)
	Noise 0.01	28.40	0.8749	0.7302 (0.0308)	0.460 (0.062)
	Noise 0.02	27.53	0.8152	0.7338 (0.0311)	0.468 (0.062)
	Noise 0.03	26.42	0.7451	0.7371 (0.0310)	0.474 (0.062)
FULL	Clean	35.70	0.9802	1.0000 (0.0000)	1.000 (0.000)
	JPEG 90	34.18	0.9604	0.8632 (0.0543)	0.726 (0.109)
	JPEG 80	32.51	0.9401	0.7813 (0.0496)	0.563 (0.099)
	JPEG 70	31.33	0.9234	0.7437 (0.0432)	0.487 (0.086)
	JPEG 60	30.48	0.9097	0.7189 (0.0387)	0.438 (0.077)
	JPEG 50	29.85	0.8980	0.7005 (0.0379)	0.401 (0.076)
	Blur 0.5	32.25	0.9584	0.9165 (0.0534)	0.833 (0.107)
	Noise 0.01	34.28	0.9483	0.9187 (0.0536)	0.837 (0.107)
	Noise 0.02	31.75	0.8761	0.8209 (0.0796)	0.642 (0.159)
	Noise 0.03	29.40	0.7948	0.7550 (0.0781)	0.510 (0.156)

variance values from 0.01 to 0.07. The 'Attack' column in Table 3 specifies the representative subset of these attacks that was used to demonstrate key performance benchmarks. The 'JPEG' entries indicate JPEG compression with quality factors from 50 to 90. 'Blur 0.5' refers to Gaussian blur with a value of 0.5. The 'Noise' entries, labeled 'Noise 0.01', Noise 0.02', and 'Noise 0.03', correspond directly to those respective variance levels. To aid in visualization, cells that meet the predefined performance thresholds PSNR ≥ 30, SSIM ≥ 0.9, ACC ≥ 0.7, and NC ≥ 0.7 are highlighted with shading.

As shown in Table 3, the *ONLY* model shows strong overall performance. The *BOTH* model exhibits the lowest overall performance, even in the scenario that has no attack. The *FULL* model achieves higher PSNR values in several scenarios, such as 'Clean' and 'Noise 0.01', compared to the *ONLY* model. However, its NC results are consistently weaker than those of the *ONLY* model across all attack types.

The *ONLY* model is identified as the optimal model because it achieves the most effective balance between imperceptibility and robustness. While both the *ONLY* and *FULL* models achieve a perfect ACC and NC of 1.0 in the absence of an attack, which demonstrates flawless baseline embedding and extraction, their performance diverges significantly under attack.

The *ONLY* model consistently maintains superior robustness compared to the *FULL* model. For instance, under the 'Noise 0.03' attack, the *ONLY* model retains an ACC of 0.873 and an NC of 0.745, whereas the performance of the *FULL* model degrades significantly to an ACC of 0.755 and NC of 0.510. While the *FULL* model begins with slightly higher perceptual quality in the baseline scenario, the *ONLY* model represents the best overall compromise, maintaining high imperceptibility and stable, high-accuracy watermark recovery with high accuracy across the evaluated attacks.

The visual effects of the *ONLY* model are illustrated in Figure 3. For each sample, the stego image is shown under 'Clean' and various attack conditions. The corresponding difference maps, which have been magnified three times for clarity, are displayed below each image.

To benchmark the proposed system against contemporary watermarking techniques, a comparative evaluation was conducted using WAM [39] and a hybrid method that integrates Invisible Watermark [40] with VINE [41], hereafter referred to as WM_VINE. These methods were selected because they represent recent advances in learning-based watermarking and are among the few publicly available systems capable of processing payloads at the 256-bit scale, which corresponds to the default cryptographic configuration of the proposed approach.

Table 4 summarizes the performance of the two baseline models under representative attack scenarios, including clean images, JPEG compression with quality factors ranging from 90 to 50, Gaussian blur with standard deviation 0.5, and Gaussian noise with variance between 0.01 and 0.03.

The results reveal clear limitations in the ability of existing techniques to support long-form watermarking. WAM maintains an accuracy close to 0.50 across all evaluated conditions, which indicates failure in watermark extraction and is statistically indistinguishable from random guessing. This

Figure 3 Qualitative results of the optimal model (ONLY) for two sample images (Cover image 1 and Cover image 2). The first column displays the original cover images. The subsequent columns show the stego (watermarked) images under the 'Clean' (no attack) condition and various attack conditions (JPEG 90, JPEG 80, JPEG 70, JPEG 60, JPEG 50, Blur 0.5, Noise 0.01, Noise 0.02, and Noise 0.03). Below each stego image, a corresponding difference map, magnified three times for clarity, is displayed to visualize the perceptual impact of the embedded watermark.

outcome reflects the architectural design of WAM, which is tailored for short payloads and cannot be extended to cryptographic watermark lengths of 256 bits. Its reported PSNR values remain in the range of 30 to 32 decibels, offering no significant perceptual advantage over the proposed system.

WM_VINE achieves exceptionally high imperceptibility in the clean setting, with PSNR values exceeding 45 decibels. However, its robustness deteriorates sharply under minimal perturbations. JPEG compression with quality factor 90 reduces its decoding accuracy to approximately 0.56, and the model exhibits near-complete failure under noise attacks. Although WM_VINE produces visually appealing stego images, its frequency-localized embedding strategy lacks the resilience required for ownership verification in security-critical applications.

In contrast, the proposed ChainMark system, presented in Table 3, maintains stable performance with accuracy values exceeding 0.9 under most attack types while preserving acceptable visual quality with PSNR values around 34 decibels, even when encoding a 256-bit watermark. This comparative analysis demonstrates that ChainMark effectively unifies robust

Table 4 Performance comparison

Model	Attack	PSNR	SSIM	ACC (std)	NC (std)
WAM	Clean	30.31	0.8172	0.5002 (0.0094)	0.0004 (0.0189)
	JPEG 90	32.40	0.8690	0.5002 (0.0094)	0.0004 (0.0189)
	JPEG 80	32.74	0.8754	0.5002 (0.0094)	0.0004 (0.0189)
	JPEG 70	32.85	0.8766	0.5002 (0.0094)	0.0004 (0.0189)
	JPEG 60	32.84	0.8763	0.5002 (0.0094)	0.0004 (0.0189)
	JPEG 50	32.80	0.8759	0.5002 (0.0094)	0.0004 (0.0189)
	Blur 0.5	30.86	0.8267	0.5002 (0.0094)	0.0004 (0.0189)
	Noise 0.01	24.02	0.4681	0.5002 (0.0094)	0.0004 (0.0189)
	Noise 0.02	21.70	0.3699	0.5045 (0.0083)	0.0090 (0.0166)
	Noise 0.03	20.26	0.3156	0.5003 (0.0095)	0.0006 (0.0191)
WM_VINE	Clean	45.23	0.9890	0.8311 (0.0860)	0.6621 (0.1719)
	JPEG 90	45.96	0.9912	0.5623 (0.0347)	0.1246 (0.0693)
	JPEG 80	44.81	0.9887	0.5389 (0.0225)	0.0777 (0.0450)
	JPEG 70	43.54	0.9842	0.5340 (0.0170)	0.0680 (0.0340)
	JPEG 50	41.61	0.9797	0.5192 (0.0092)	0.0385 (0.0184)
	Blur 0.5	45.59	0.9902	0.8326 (0.0850)	0.6652 (0.1701)
	Noise 0.01	25.09	0.5064	0.4417 (0.0288)	−0.1166 (0.0577)
	Noise 0.02	22.27	0.3926	0.4541 (0.0115)	−0.0918 (0.0229)
	Noise 0.03	20.65	0.3322	0.4857 (0.0093)	−0.0285 (0.0187)

signal-processing behaviour with high-capacity cryptographic verification, a combination that existing baseline techniques do not achieve.

5.2.6 Threshold analysis

The robustness of the proposed ownership verification mechanisms was initially evaluated based on the *ONLY* model using the linear similarity score, $s_p = \delta \times ACC + \varepsilon \times NC$. To ensure statistical reliability, the experiment involved 100,000 imposter watermarks and 10,000 test images subjected to diverse and rigorous attack scenarios.

The analysis revealed a distinct separation between genuine and imposter sample distributions. A consistent equal error rate (EER) floor of 34.38% was identified across all linear weight combinations. Crucially, this value represents stress-test results derived from a dataset that includes extreme perturbations, such as severe Gaussain noise and high-ratio compression, rather than typical operating conditions. This indicates that the scoring function maintains structural validity even when the underlying image features are heavily degraded.

Subsequent investigations into nonlinear scoring functions did not yield performance improvements, confirming that the performance boundary is defined by the feature extraction capability under heavy distortion rather than the scoring method itself.

Consequently, the balanced linear configuration ($\delta = 0.5$ and $\varepsilon = 0.5$) was determined to be the most robust mechanism. This setting effectively balances ACC and NC, establishing an optimal verification threshold of $\tau_p = 0.262$. This results confirms the practical viability of the proposed verification framework, demonstrating resilience across a broad spectrum of signal processing attacks.

5.2.7 Ablation study

This section describes influence of λ_{mag} on performance. For this ablation study, the final *ONLY* model (which uses $\lambda_{mag} = 0.2$) was compared against an identical model trained with $\lambda_{mag} = 0$. The results are shown in Figure 4.

As shown in Figure 4, the results demonstrate a clear trade-off between perceptual quality and robustness. The model that omits the L_1 loss ($\lambda_{mag} = 0$) achieves slightly higher PSNR and SSIM values across all

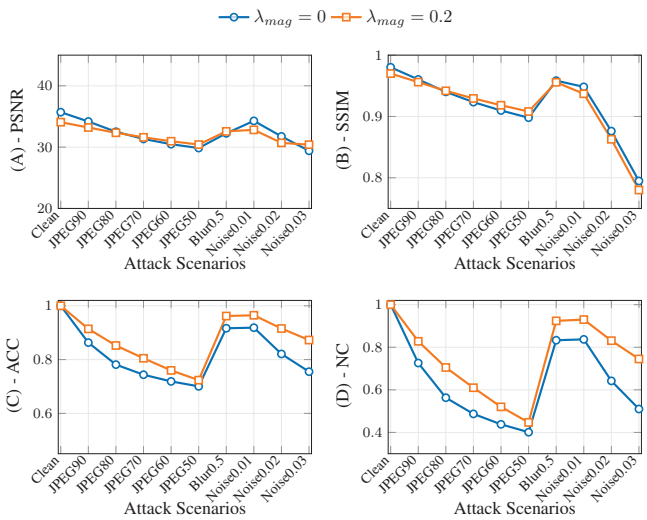

Figure 4 Ablation study results analyzing the influence of the L_1 loss term (λ_{mag}) in the ONLY model. The graphs compare the performance of the proposed model ($\lambda_{mag} = 0.2$) against the baseline model ($\lambda_{mag} = 0$). The x-axis denotes specific attack scenarios: JPEG (compression quality 90–50), blur (Gaussian blur $\sigma = 0.5$), and noise (Gaussian noise variance 0.01–0.03). (A) PSNR; (B) SSIM; (C) ACC; (D) NC.

attack conditions. However, this marginal gain in imperceptibility comes at a significant cost to robustness.

The model that includes the L_1 loss ($\lambda_{mag} = 0.2$) consistently and substantially outperforms the ablated model in both ACC and NC. This gap is especially evident under strong attacks. For instance, under the 'Noise 0.03' attack, the model with $\lambda_{mag} = 0.2$ retains an ACC of 0.873 and an NC of 0.745, whereas the performance of the model without this loss degrades significantly to an ACC of 0.755 and an NC of 0.510. This analysis confirms that the λ_{mag} component, which enforces the magnitude of the recovered signal, is critical for stabilizing the watermark recovery process and ensuring high extraction accuracy under adverse conditions.

6 Conclusion

To address the persistent limitation in conventional watermarking wherein successful extraction has been incorrectly regarded as evidence of ownership, this study presented ChainMark. This system integrates a discrete wavelet transform-based invertible neural network with a blockchain-driven verification mechanism. The joint LH and HL configuration demonstrated the most effective balance between perceptual fidelity and robustness. At the same time, ownership authentication was guaranteed through a smart contract whose correctness relies on the computational hardness of the discrete logarithm problem. This design ensures that watermark extraction alone cannot be misused to assert false authorship, thereby overcoming a structural weakness that has remained unresolved in prior neural watermarking research.

The proposed system exhibits several notable strengths. It provides strong resilience against a wide range of signal processing attacks through reversible embedding, and it establishes verifiable ownership by binding the watermark to cryptographic parameters stored on-chain. Consequently, the system prevents adversaries from exploiting watermark extraction as a mechanism for impersonation and enables reliable attribution even under substantial visual degradation.

Despite these advantages, the present work has inherent limitations. First, the invertible neural network architecture is sensitive to geometric transformations because it relies on strict spatial alignment between the cover image and the recovered latent representation. As a result, attacks such as cropping and rotation remain challenging. Additionally, while the blockchain ensures immutability, the current implementation on a public ledger may incur computational overhead and transaction latency, which can

affect scalability in high-frequency trading environments. Second, although authentication remains perfect in the absence of attacks, the equal error rate of 34.38% under severe perturbations reflects a fundamental limitation in the current model's ability to preserve the latent watermark structure. This value represents an aggregate metric computed across a comprehensive set of attack conditions, including Gaussian blur from 0.5 to 5.0, Gaussian noise with variance from 0.01 to 0.07, and JPEG compression from 50% to 90%. The reported rate, therefore, indicates the performance boundary under the most adverse scenarios. Future research will focus on mitigating these issues by incorporating ensemble invertible architectures, geometric invariant modules, and enhanced latent regularization techniques.

Looking forward, real-world deployment of ChainMark in digital content markets, licensing platforms, and creator-centric ecosystems will require further optimization. Integrating a lightweight consensus protocol or zero-knowledge proofs (zk-SNARKs) may provide practical improvements in throughput, confidentiality, and operational efficiency. These directions constitute promising avenues for refining the ChainMark framework into a comprehensive solution for secure digital ownership in modern creative environments.

Acknowledgment

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under ITRC (Information Technology Research Center) support program (IITP-2026-RS-2023-00259099) supervised by IITP (Institute for Information & Communications Technology Planning & Evaluation).

References

[1] Silva D, Kaynak O, El-Ayoubi M, et al. Opportunities and challenges of Generative Artificial Intelligence: Research, Education, Industry Engagement, and Social Impact. *IEEE Industrial Electronics Magazine*, 19(1):30–45, 2025. https://doi.org/10.1109/MIE.2024.3382962.

[2] Lemley MA, How Generative AI Turns Copyright Upside Down. *SSRN*, 4517702:1–24, 2023. http://dx.doi.org/10.2139/ssrn.4517702.

[3] Al-Busaidi AS, Raman R, Hughes L, et al. Redefining boundaries in innovation and knowledge domains: Investigating the impact of generative artificial intelligence on copyright and intellectual property

rights. *Science*, 9(4):100630, 2024. https://doi.org/10.1016/j.jik.2024.100630.

[4] Lucchi Nicola. ChatGPT: A Case Study on Copyright Challenges for Generative Artificial Intelligence Systems. *European Journal of Risk Regulation*, 15:602–604, 2024. https://doi.org/10.1017/err.2023.59.

[5] Samuelson P. Generative AI meets copyright. *Science*, 6654:158–161, 2023. https://doi.org/10.1126/science.adi0656.

[6] Cox IJ, Kilian J, Leighton FT, et al. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, 1997. https://doi.org/10.1109/83.650120.

[7] Evsutin O, Melman A, Meshcheryakov R. Digital Steganography and Watermarking for Digital Images: A Review of Current Research Directions. *IEEE Access*, 8:166589–166611, 2020. https://doi.org/10.1109/ACCESS.2020.3022779.

[8] Berghel H, O'Gorman L. Protecting ownership rights through digital watermarking. *Computer*, 29(7):101–103, 1996. https://doi.org/10.1109/2.511977.

[9] Bao Z, Xue R. Survey on deep learning applications in digital image security. *Optical Engineering*, 60(12):120901, 2021. https://doi.org/10.1117/1.OE.60.12.120901.

[10] Archana R, Jeevaraj PE. Deep learning models for digital image processing: a review. *Artificial Intelligence Review*, 57(11):11, 2024. https://doi.org/10.1007/s10462-023-10631-z.

[11] Hosny KM, Magdi A, Elkomy O, et al. Digital Digital image watermarking using deep learning: A survey. *Computer Science Review*, 53:100662, 2024. https://doi.org/10.1016/j.cosrev.2024.100662.

[12] Lu SP, Wang R, Zhong T, et al. Large-Capacity Image Steganography Based On Invertible Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[13] Xu Y, Mao Y. A Robustness Improved Watermarking Scheme based on Invertible Neural Network. In *2024 36th Chinese Control and Decision Conference*, 2024.

[14] He Z, Hu R, Wu J, et al. A Transformer-based invertible neural network for robust image watermarking. *Journal of Visual Communication and Image Representation*, 104:104317, 2024. https://doi.org/10.1016/j.jvcir.2024.104317.

[15] Shi J, Yang Z, Li L, et al. Multi-user watermarking based on paralleled invertible neural networks. *Signal, Image and Video Processing*, 19(503):1–12, 2025. https://doi.org/10.1007/11760-025-04015-9.

[16] Ni Z, Cheng H, Chen J, et al. NiNet: A new invertible neural network architecture more suitable for deep image hiding. *Information Processing & Management*, 62(6):104275, 2025. https://doi.org/10.1016/j.ipm.2025.104275.

[17] Shi W, Wang Z, Zhang X, et al. StarINN: An efficient invertible neural network for image steganography. *Displays*, 91:103175, 2026. https://doi.org/10.1016/j.displa.2025.103175.

[18] Craver S, Memon N, Yeo B-L, et al. Resolving rightful ownerships with invisible watermarking techniques: limitations, attacks, and implications. *IEEE Journal on Selected Areas in Communications*, 16(4):573–586, 1998. https://doi.org/10.1109/49.668979.

[19] Qiao L, Nahrstedt K. Watermarking Schemes and Protocols for Protecting Rightful Ownership and Customer's Rights. *Journal of Visual Communication and Image Representation*, 9(3):194–210, 1998. https://doi.org/10.1006/jvci.1998.0391.

[20] Sencar HT, Memon N. Watermarking and ownership problem: a revisit. In *DRM'05:Proceedings of the 5th ACM workshop on Digital rights management*, 2005.

[21] Hartung F, Kutter M. Multimedia watermarking techniques. *Proceedings of the IEEE*, 87(7):1079–1107, 1999. https://doi.org/10.1109/5.771066.

[22] Potdar VM, Han S, Chang E. A survey of digital image watermarking techniques. In *INDIN '05. 2005 3rd IEEE International Conference on Industrial Informatics*, 2005.

[23] Barni M, Bartolini F, Piva A. Improved wavelet-based watermarking through pixel-wise masking. *IEEE Transactions on Image Processing*, 10(5):783–791, 2001. https://doi.org/10.1109/83.918570.

[24] Ganic E, Eskicioglu AM. Robust DWT-SVD domain image watermarking: embedding data in all frequencies. In *MM&Sec'04:Proceedings of the 2004 workshop on Multimedia and security*, 2004.

[25] Hu K, Wang M, Ma X, et al. Learning-based image steganography and watermarking: A survey. *Expert Systems with Applications*, 249(C):123715, 2024. https://doi.org/10.1016/j.eswa.2024.123715.

[26] Guan Z, Jing J, Deng X, et al. DeepMIH: Deep Invertible Network for Multiple Image Hiding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):372–390, 2023. https://doi.org/10.1109/TPAMI.2022.3141725.

[27] Padhi SK, Tiwari A, Ali SS. Deep Learning-Based Dual Watermarking for Image Copyright Protection and Authentication. *IEEE Transactions*

on Artificial Intelligence, 5(12):6134–6145, 2024. https://doi.org/10.1109/TAI.2024.3485519.

[28] Fu L, Liao X, Guo J, et al. WaveRecovery: Screen-Shooting Watermarking Based on Wavelet and Recovery. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(4):3603-3618, 2025. https://doi.org/10.1109/TCSVT.2024.3510355.

[29] Sun W, Liu J, Dong W, et al. RWNeRF: Robust Watermarking Scheme for Neural Radiance Fields Based on Invertible Neural Networks. *Computers, Materials & Continua*, 80(3):4065–4083, 2024. https://doi.org/10.32604/cmc.2024.053115.

[30] Sun W, Liu J, Chen L, et al. MarkINeRV: A Robust Watermarking Scheme for Neural Representation for Videos Based on Invertible Neural Network. *Computers, Materials & Continua*, 80(3):4031-4046, 2024. https://doi.org/10.32604/cmc.2024.052745.

[31] Zhu J. Robust Audio Watermarking Based on Invertible Neural Network. *Academic Journal of Computing & Information Science*, 8(3):10–17, 2025. https://doi.org/10.25236/AJCIS.2025.080302.

[32] Mannepalli PK, Richhariya V, Gupta SK, et al. A robust blockchain-based watermarking using edge detection and wavelet transform. *Multimedia Tools and Applications*, 84:12739–12763, 2024. https://doi.org/10.1007/s11042-024-18907-4.

[33] Battah A, Madine M, Yaqoob I, et al. Blockchain and NFTs for Trusted Ownership, Trading, and Access of AI Models. *IEEE Access*, 10:2169–3536, 2022. https://doi.org/10.1109/ACCESS.2022.3215660.

[34] Alvar SR, Akbari M, Yue D, et al. NFT-Based Data Marketplace with Digital Watermarking. In *KDD '23: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.

[35] Dong Y, Wang C. Copyright protection on NFT digital works in the Metaverse. *Security and Safety*, 2:2023013, 2023. https://doi.org/10.1051/sands/2023013.

[36] Hasan HR, Salah K, Jayaraman R, et al. NFTs for combating deepfakes and fake metaverse digital contents. *Internet of Things*, 25:101133, 2024. https://doi.org/10.1016/j.iot.2024.101133.

[37] Mouris D, Tsoutsos NG. NFTs for 3D Models: Sustaining Ownership in Industry 4.0. *IEEE Consumer Electronics Magazine*, 13(5), 2024. https://doi.org/10.1109/MCE.2022.3164221.

[38] PyWavelets Documentation, PyWavelets - Wavelet Transforms in Python. Available Online: https://pywavelets.readthedocs.io/en/latest/.

[39] Hugging Face, Facebook/watermark-anything. Available Online: https://huggingface.co/facebook/watermark-anything.
[40] Github, invisible-watermark. Available Online: https://github.com/ShieldMnt/invisible-watermark?tab=readme-ov-file.
[41] Github, VINE. Available Online: https://github.com/Shilin-LU/VINE.

Biographies

Haeun Jo is an undergraduate student in the Department of Software Engineering at Jeonbuk National Univeristy. Her research interests include blockchain technology, consensus algorithm, and data security.

Jungwon Seo received his B.Sc. degree in management information systems from the State University of New York at Buffalo, USA, in May 2016, and his M.Sc. degree in computer science and engineering in March 2020. He obtained his Ph.D. degree in software engineering and blockchain from Sogang University, Seoul, Republic of Korea, in August 2024. He is currently an assistant professor with the Department of Software Engineering at Jeonbuk National University.