Study and Development of Reversible Watermarking Methods for Securing Deep Neural Networks

A THESIS SUBMITTED

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
ARTIFICIAL INTELLIGENCE

Submitted by

VAIBHAV KUMAR SINGH

(23/AFI/15)

Under the supervision of

DR. RAJEEV KUMAR

Assistant Professor

Department of Computer Science Engineering

Department Of Computer Science Engineering
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi 110042

MAY, 2025

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, **VAIBHAV KUMAR SINGH**, Roll No – **23/AFI/15**, hereby certify that the work which is being presented in the thesis entitled **"Study and Development of Reversible Watermarking Methods for Securing Deep Neural Networks"** in partial fulfillment of the requirements for the award of the Degree of Master of Technology (ARTIFICIAL INTELLIGENCE), Submitted in the **Department of Computer Science and Engineering**, Delhi Technological University is an authentic record of my own work carried out under the supervision of **DR. RAJEEV KUMAR.**

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Place: Delhi — Vaibhav Kumar Singh

Date: 30/05/2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

<u>CERTIFICATE</u>

Certified that **VAIBHAV KUMAR SINGH (23/AFI/15)** has carried out their search work presented in this thesis entitled **"Study and Development of Reversible Watermarking Methods for Securing Deep Neural Networks"** for the award of Master of Technology from Department of Computer Science and Engineering, Delhi Technological University, Delhi under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University.

Place: Delhi

signature

DR. RAJEEV KUMAR

Assistant Professor

Date: 30/05/2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I would like to express our heartfelt gratitude to all the individuals who have supported and assisted me throughout my MTech. thesis, First and foremost, I would like to thank my supervisor, **"Dr. Rajeev Kumar"**, Assistant Professor, Department of Computer Engineering, Delhi Technological University for his constant guidance, support, and encouragement throughout the project. I am indebted to him for sharing his knowledge, expertise, and valuable feedback that helped me in shaping the thesis..

I would like to extend my sincere thanks to the Vice Chancellor of Delhi Technological University and the faculty members of the Department of Computer Engineering for their support and encouragement throughout our academic journey

Place: Delhi Vaibhav Kumar Singh

Date: 30/05/2025

Abstract

The widespread sharing and reuse of pre-trained deep convolutional neural networks (CNNs) have raised serious concerns about model integrity and security. Malicious modifications such as backdoor and model-reuse attacks can compromise the trustworthiness of these models without visibly affecting their performance, making integrity authentication a critical requirement in sensitive domains. While watermarking has emerged as a prominent method for model verification, most existing techniques are irreversible and alter the model's internal structure permanently, making them unsuitable for integrity validation. This research proposes a reversible watermarking scheme that enables the embedding of authentication information into CNNs without any permanent modification. The method leverages model pruning theory to identify less critical parameters with low entropy, constructing an optimal host sequence for watermark insertion. Using a histogram shifting technique adapted from image watermarking, the watermark is embedded in a manner that ensures full recovery of the original model parameters upon extraction. Experimental validation is conducted across several popular CNN architectures, including AlexNet, VGG19, ResNet152, DenseNet121, and MobileNet. The results show that the reversible watermarking process has a negligible effect on model accuracy (within $\pm 0.5\%$) and demonstrates complete reversibility with zero reconstruction error. Lastly, if cryptographic hash values are embedded, identity verification is possible: any modification to the model will make the hash values differ between the original hash and the hash generated after each step. The researchers have made a new approach by reversing the process. using watermarking to check that the structure of CNNs is not tampered with, presented a way to protect sensitive models from deep learning by using secure parts For example, healthcare, finance and defense. This methodology can still be broadened and improved as we go. seen if LSTM is useful for other kinds of neural network structures. ability to help restore models that have had changes made to them unauthorized alterations

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Overview

Working with CNNs has allowed artificial intelligence to achieve much better results in image recognition. Using voice recognition, speech interpretation and operating independently through systems. They have become famous mostly due to how successful they are. Because ImageNet offered vast amounts of training images, ResNet and VGG were designed and using pretrained networks became popular, deep learning progressed a lot. Models that make it easier to implement change in multiple fields. But, having frequent The worry is that reuse of these models might introduce tampering with the parameters of the network. In healthcare and financial areas, this issue is an especially big concern. Autonomous technologies are also at risk, since incorrect AI models can be extremely dangerous. Because of this, confirming the legitimacy and integrity of distributed models is now essential for securely using AI.

1.2 Security Concerns

Using pretrained CNNs is being adopted more and more. notable issues of security and how they connect to maintaining the integrity of shared models. Improvements or alterations may be made to these models while they are being distributed. possible changes that can go unnoticed if you are unaware. The threat known as a backdoor attack is managed by including malicious triggers. that can make the model produce wrong or threatening results under some situations, all acting like normal even when being validated. There is also the threat of some other vectors. is the model-reuse attack which is when compromised models are introduced unnoticed. transferring knowledge learned from one system to another often leads to vulnerabilities appearing in other systems. Unlike ordinary software, deep learning models are built in complicated ways that are hard to see inside. Because of this, it is not easy to see or follow any unauthorized changes that may occur. The absence of transparency makes it clear that we need strong methods to confirm if a model is truthful. Authenticity and integrity must be tested and evaluated before the system is used outside of test environments. Standard methods for detecting problems are commonly used after something has happened. By comparison, this work supports using reversible watermarking can help prevent tampering and catch changes to documents. restoring an old model so that performance is not reduced.

1.3 Need for Integrity Verification

As CNNs and other deep learning models are applied to areas such like healthcare, autonomous vehicles, the military and financial risk assessment, their validity becomes more important. Neural networks are not like traditional software systems because their trainable features (weights and biases) which often remain mysterious and difficult interpretable. This makes detecting unauthorized modifications especially difficult and highlights the need for robust integrity verification mechanisms.

1.3.1 Risks of Tampered Models

Pre-trained models are commonly shared across research communities and deployed in commercial applications. However, during distribution or deployment, models can be illegally modified by third parties to embed malicious behavior, such as:

- **Backdoors**: Trigger-specific manipulations that cause the model to behave correctly under normal conditions but output incorrect or malicious predictions when specific inputs are provided.
- **Model-Reuse Attacks**: Malicious models serve as parent models in transfer learning, causing inherited vulnerabilities to propagate into new applications.

Such tampering not only compromises performance but can also jeopardize user safety and privacy, especially when these models are deployed in safety-critical environments.

1.3.2 Limitations of Traditional Detection Methods

Existing detection methods focus on analyzing model outputs, statistical patterns, or internal activations to infer the presence of tampering. However, these methods are:

- **Passive**: They attempt to detect anomalies without confirming model integrity.
- **Heuristic**: Their effectiveness heavily depends on assumptions about attack behavior.
- **Vulnerable to Evasion**: Sophisticated attackers may design tampering methods that evade these detection mechanisms.

Therefore, relying solely on detection is inadequate.

1.3.3 The Role of Integrity Authentication

Integrity authentication offers a proactive and formal mechanism to confirm whether a model has been modified. This involves:

- Embedding a secure, hidden watermark into the model at deployment.
- Extracting and verifying this watermark later to ensure it matches the original.

If the watermark differs or cannot be extracted, it serves as evidence that the model has been altered.

1.3.4 The Case for Reversible Watermarking

Most watermarking methods used in models today are irreversible, meaning they permanently alter the model's parameters, which compromises its original state. This is not acceptable in high-stakes applications where:

- **Regulatory compliance** mandates the preservation of original model behavior.
- **Security-sensitive applications** demand that no permanent changes be made.
- **Auditability** requires exact reproduction of the model for validation and verification.

Thus, a reversible watermarking approach is crucial. It enables:

- The embedding of watermark data without sacrificing model integrity.
- Full restoration of the model to its original form after verification.

1.4 Problem Statement

1.4.1 Core Challenge

Most existing model authentication and watermarking approaches are irreversible in nature. That is, once a watermark is embedded into a model's parameters, the original weights cannot be recovered. While this might be sufficient for ownership verification or intellectual property protection, it is not appropriate for integrity authentication, where the ability to restore the model to its pristine state is paramount.

The fundamental problem lies in the trade-off between embedding information and preserving the model's integrity. When information is embedded (e.g., through modifying weights), it introduces a deviation from the original parameter values. Even slight, irreversible changes conflict with the requirements of systems where exact reproducibility and structural purity are essential.

1.4.2 Formal Problem Definition

The goal is to develop a mechanism that allows a reversible watermark to be embedded into a CNN model in such a way that:

- A predefined watermark (such as a digital signature or cryptographic hash) can be embedded into the model without compromising its performance.
- After embedding, the modified model (called the marked model) should behave identically or nearly identically to the original model in terms of inference accuracy.
- The embedded watermark can be extracted **without requiring access to the original model**.
- Once the watermark is extracted, **the original model must be fully restored**, ensuring that no permanent changes remain.
- The watermark should be sensitive to tampering—any unauthorized modification to the model should result in a mismatch or corruption of the embedded watermark.

Mathematically, the problem can be represented as follows:
Let **M** be the original trained CNN model, and B be the watermark to be embedded. The embedding and extraction functions must satisfy:

- $\mathbf{M} = \mathbf{Emb(M,\ B)}$ $\rightarrow$ Embeds watermark B into model M, producing a watermarked model M.
- $\mathbf{(M, B)} = \mathbf{Ext(M)}$ $\rightarrow$ Extracts both the watermark B and reconstructs the original model M from M.

Where $\mathbf{Emb(\cdot)}$ and $\mathbf{Ext(\cdot)}$ are reversible operations, meaning the original model M can be perfectly recovered from M, and the integrity of B can be verified.

1.5 Research Objective

To address these limitations, the core objective of this research is to design a reversible watermarking framework tailored specifically for CNNs that enables secure and efficient integrity verification. This solution should be:

- Model-aware, leveraging the internal structure of CNNs.
- Fully reversible, ensuring complete recovery of original parameters.
- Tamper-evident, providing a reliable mechanism to detect modifications.
- Lightweight and non-invasive, with minimal computational overhead and performance impact.

Chapter 2

LITERATURE REVIEW

This chapter reviews existing research and methodologies that form the foundation of this work on reversible watermarking in CNNs for model integrity authentication. The focus is placed on prior developments in model protection, watermarking strategies, and reversible data embedding. The review highlights the strengths and limitations of various approaches and identifies the research gaps this thesis aims to address.

2.1 Early Foundations of Reversible Watermarking (2003–2006)

Watermarking that can be reversed started with the developments in digital image processing. In 2003, Tian (2003) published an important method which includes data in an image and still makes it possible to restore the original image perfectly. information was gathered [1]. It was proved with this method that one could remove added information from the files without altering the original files. On this basis, Ni et al. (2006) introduced a histogram shifting method that earned much attention because of its easy use and low distortion [2]. This technique makes space for hiding data by modifying pixel values which supports both high volume and high quality. They formed the basis for the later progress in reversible data hiding which spread into medical imaging, military messages and machine learning studies.

2.2 Advances in Reversible Techniques and Image Embedding (2009–2012)

After the first progress was made, researchers started using different techniques to improve reversible watermarking. The method suggested by Sachnev et al. (2009) uses both sorting and prediction to cut down on the distortions usually seen in watermarking [3]. Choosing to predict image pixels and add data to prediction errors, their technique improved how well the images hide data and how useful the technique is. After that, Zhang et al. (2012) proposed optimized binary cover codes, letting the data use the space in the host medium better while keeping reversibility [4]. By making these advances, data hiding methods could store more information without losing the ability to decode the image back to its original state, so their usefulness was greatly expanded.

2.3 Rise of Deep Neural Networks (2012–2016)

The arrival of AlexNet by Krizhevsky et al. (2012) opened new doors for computer vision since it was the first to use deep CNNs and achieved the best results in big image classification contests such as ImageNet [5]. As a result of this discovery, founded use of deep learning being adopted by many different fields. Following this success, He and others (2016) offered the ResNet architecture which employed residual connections to solve the vanishing gradient issue and successfully trained much deeper networks [6]. At about the same time as these advances in model architecture, researchers continued to improve reversible data hiding as well. Shi et al. (2016) have discussed changes in techniques over the past two decades. Research covers how these watermarking methods can be used with both images and more types of digital media such as audio and video as well [7].

2.4 Model Vulnerabilities and Security Threats (2017–2018)

As more deep learning tools became available and easy to use, people began to worry about how secure and reliable they are. Gu et al. (2017) pointed out that backdoor attacks are possible, where changing certain inputs causes a neural network to make errors, but standard tests will show normal results [8]. Another survey by Song et al. (2017) revealed that using least significant bit replacement (LSBR) can embed watermarks so that the model performs differently, causing worries about information being revealed and model credibility [9]. According to Ji et al. It was also found in (2018) that using the same model in transfer learning can result in unnoticed vulnerabilities migrating from one field to another which is known in the field as model-reuse attacks [10]. Various techniques for defense have been suggested because of these growing risks. By using Fine-Pruning, Liu et al. (2018) achieved stronger models, since it removes neural connections that appear unnecessary, therefore improving performance [11]. Similarly, Chen et al. (2018) introduced Activation Clustering to spot irregular activation in the neural networks that can be a sign of infected training samples [12].

2.5 Watermarking in Neural Networks (2017–2018)

During these days, more studies begun focusing on using watermarking techniques for neural networks. Uchida and colleagues (2017) were the first to propose watermarking methods by automatically putting signals into the weights during model training [13]. Built on these ideas, Adi and his team (2018) introduced a new way to use backdoors as "signatures" to confirm if a copyrighted model was used by a different person [14]. Also at this time, Rouhani, Abe and others (2018) introduced DeepSigns, inserting watermarks into the internal workings of neural networks to avoid their misuse and unauthorized copying [15]. At the same time, Zhang et al. (2018) looked into various approaches for producing key images for black-box watermarking. They help to check model accuracy by testing incidents, without accessing the inner workings of the model [16]. All this led to other improvements that resulted in the development of secure and tamper-resistant models in the quickly emerging AI field.

2.6 Extension to Modern Architectures and Applications (2017–2020)

At this point, several new architectures in convolutional neural networks (CNNs) were created and played a big role in deep learning development. DenseNet, created by Huang et al. (2017), ensures every layer shares features directly with the others which improves training and the quality of gradients [17]. Also, Sandler et al. (2018) introduced MobileNetV2, a model designed mainly for mobile and embedded systems by making operations less complex and using inverted residuals [18]. Such diverse architectures resulted in both difficulties and new chances for watermarking. In order to stay robust and invisible to the eye while still having top performance, watermarking had to conform to the particular layer and compression structures of transformers.

Luo and Wu (2017) applied entropy-based channel pruning for model compression, a technique later leveraged in the current study to construct host sequences for watermark embedding [19]. Willems and Kalker (2003) earlier established theoretical capacity bounds for reversible embedding under distortion constraints, providing a framework still referenced today [20].

2.7 Recent Watermarking Frameworks and Integrity Approaches (2020)

Zhang et al. (2020) offered a black-box framework for image processing networks, addressing use cases where only API access is available [21]. In the same year, the authors of the current paper built upon these foundations to propose a reversible watermarking framework tailored for CNNs, focusing on integrity authentication by exploiting model pruning theory and histogram shift techniques.

2.8 Summary of Literature Review

Over the past two decades, reversible watermarking has evolved from basic image embedding techniques to sophisticated applications in deep learning models. Early methods like difference expansion and histogram shift were crucial in establishing the foundational concepts of lossless data embedding in digital images. These techniques enabled exact recovery of the original content, making them highly suitable for sensitive applications.

With the rise of deep learning, particularly convolutional neural networks (CNNs), new challenges and opportunities emerged. Researchers began to explore vulnerabilities in shared models, such as backdoor and model-reuse attacks, which posed significant threats to model integrity.

Therefore, neural networks started using watermarking to improve security. At first, approaches within networks were made irreversible and concentrated on proving ownership. robustness is preferred to how realistic the model seems. In many cases, using traditional watermarking means the model's parameters are permanently altered which makes it hard to use these techniques for integrity assessment. For overcoming this limitation, researchers came up with reversible watermarking that is designed for CNNs. The technique uses practices like removing data using entropy and histogram shifting to make authentication possible modify things in a way that does not change the model's core structure. This way, the original model parameters can be completely recovered which is why it is used in sensitive industries like healthcare, defense and finance to prove model integrity and accuracy.

Using image-based watermarking was common in earlier years, but this field has now progressed to methods made for working with neural networks. It indicates the ongoing growth of collaboration between machine learning security and digital watermarking because unique solutions are required. The issues that arise when trying to protect AI models.

Chapter 3

METHODOLOGY

3.1 Introduction

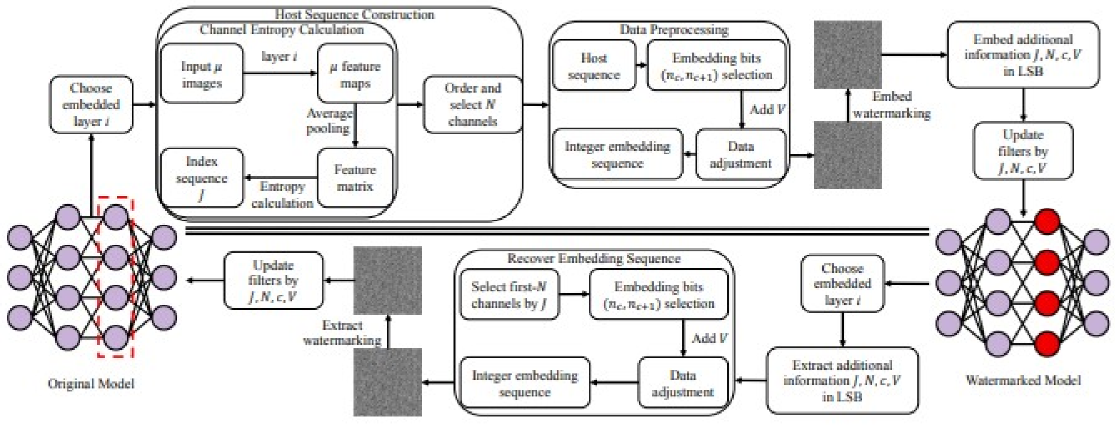
The chapter presents the detailed framework for reversible watermarking to develop convolutional neural networks (CNNs) for checking the integrity of our models. The approach is made so that all the original model settings can be completely restored after removing the embedded watermark. It also lets you find any unapproved changes that can take place during the distribution or deployment process. Principles from entropy-guided pruning, reversible data embedding and histogram shift are used together in this approach and all are chosen to fit the particular structure and characteristics of deep networks.

3.2 Overview of the Framework

The reversible watermarking procedure is divided into two fundamental stages:

- **Embedding Stage:** A binary watermark is inserted into specific parameters of a CNN in a way that preserves the model's original performance.
- **Extraction Stage:** The embedded watermark is recovered, and the network is reverted precisely to its initial, unaltered state.

These stages operate through a structured methodology composed of five key components, each tailored to maintain reversibility and integrity throughout the process.



Host Sequence Construction
Channel Entropy Calculation
Input μ images
layer i
μ feature maps
Average pooling
Index sequence J
Entropy calculation
Feature matrix
Order and select N channels
Choose embedded layer i
Data Preprocessing
Host sequence
Embedding bits (n_c, n_{c+1}) selection
Add V
Integer embedding sequence
Data adjustment
Embed watermarking
Embed additional information J, N, c, V in LSB
Update filters by J, N, c, V
Original Model
Watermarked Model
Update filters by J, N, c, V
Extract watermarking
Recover Embedding Sequence
Select first-N channels by J
Embedding bits (n_c, n_{c+1}) selection
Add V
Integer embedding sequence
Data adjustment
Choose embedded layer i
Extract additional information J, N, c, V in LSB

Figure 3.1: Framework of Reversible Watermarking

1. Host sequence construction
2. Data preprocessing and digit extraction
3. Reversible embedding using histogram shifting
4. Embedding of auxiliary metadata
5. Watermark extraction and model restoration

3.3 Workflow

In this chapter, the way to insert and confirm an image reversible watermark on a picture is shown. The function of a convolutional neural network (CNN) is covered. We want to make sure that we find any changes to the model and we can still restore it to its original form.

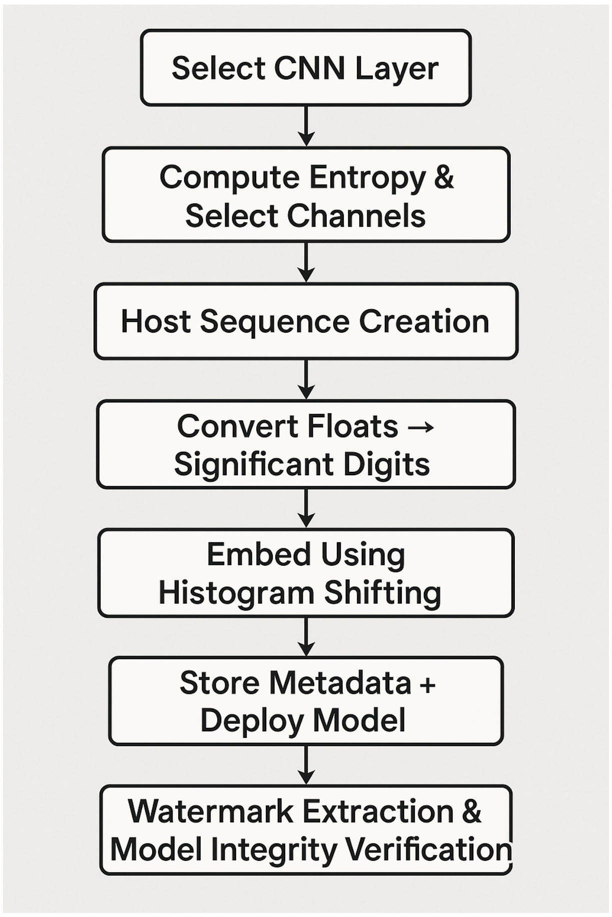

Select CNN Layer
Compute Entropy & Select Channels
Host Sequence Creation
Convert Floats → Significant Digits
Embed Using Histogram Shifting
Store Metadata + Deploy Model
Watermark Extraction & Model Integrity Verification

Figure 3.2: Flow Chart For Reversible Watermarking

3.3.1 Step 1: Select CNN Layer

To initiate the watermark embedding process, a specific convolutional layer of the neural network is selected. This layer acts as the embedding target. Typically, deeper layers are preferred because they are less sensitive to minor changes in weights, thus reducing the risk of performance degradation.

3.3.2 Step 2: Compute Entropy and Select Channels

Using a validation dataset, the entropy of each channel within the selected layer is calculated. Entropy reflects the information content or variability of a channel's activation maps.

- Low-entropy channels are less significant to model accuracy and are thus ideal for embedding.
- Channels are ranked, and the least informative ones are selected for watermarking.

This approach is inspired by entropy-based pruning techniques, ensuring minimal impact on model behavior.

3.3.3 Step 3: Host Sequence Creation

From the selected low-entropy channels, the corresponding kernel weights are extracted and organized into a flat or two-dimensional structure known as the **host sequence**. This sequence serves as the embedding medium, analogous to pixels in image watermarking.

The size and shape of the host sequence depend on the number of selected filters and the dimensionality of the convolution kernels.

3.3.4 Step 4: Convert Floats to Significant Digits

CNN weights are stored as floating-point numbers. To facilitate reversible embedding, each value in the host sequence is:

- Converted into its significant decimal digits.
- A fixed number of significant digits (typically two) are extracted..
- An offset is added to bring the values into a uniform, positive integer range

This step ensures that the data is discrete and bounded, making it suitable for histogram-based embedding methods.

3.3.5 Step 5: Embed Using Histogram Shifting

The prepared integer sequence is subjected to **histogram analysis**, where:

- A peak value (most frequent) and a zero or least-used point are identified.
- The watermark bits (e.g., a cryptographic hash) are embedded by slightly shifting values around the peak.

Only a portion of the weights is altered, preserving the majority of the host data. The changes are structured such that they can be fully reversed using the same histogram.

3.3.6 Step 6: Store Metadata and Deploy Model

In addition to the watermark, certain metadata is needed to extract and reverse the embedding, including:

- Index of selected channels
- Offset value used for digit shifting
- Position of embedded bits

This metadata is either embedded into unused parts of the model (e.g., LSBs) or stored securely elsewhere. Once completed, the watermarked model is deployed for usage or sharing.

3.3.7 Step 7: Watermark Extraction and Model Integrity Verification

When a model is retrieved or suspected of tampering, the embedded watermark is:

- Extracted using the stored or retrieved metadata.
- Compared against the expected value (e.g., a SHA-256 hash).

If the extracted watermark is missing, altered, or mismatched, it indicates **unauthorized modification**. In cases where the watermark is intact, **the model's original parameters can be completely restored**, confirming integrity and allowing continued usage.

3.4 Summary

This methodology ensures:

- Minimal change to the model's accuracy
- Full reversibility of watermark embedding
- Reliable detection of model tampering

By combining entropy analysis, digit-based data transformation, and histogram shifting, this framework provides a **secure and non-destructive way to authenticate CNN models.**

Chapter 4

EXPERIMENTAL SETUP AND RESULTS

4.1 Experimental Setup

4.1.1 Models Used

The framework was tested on several widely adopted CNN architectures to demonstrate generalizability:

- AlexNet
- VGG19
- ResNet152
- DenseNet121

4.1.2 Dataset

All models were evaluated using the ImageNet validation dataset, which consists of 50,000 color images across 1,000 classes. A subset of these images was used for entropy calculations during host sequence construction.

4.1.3 Parameters and Configurations

- Number of selected channels N for watermarking varied by model/layer.
- Offset value V=128 was used to shift significant digits into a usable range.
- Digit selection for embedding: second and third significant digits.
- Watermark: Random binary sequence or SHA-256 hash (256 bits).

4.2 Results

4.2.1 Classification Accuracy Comparison

To evaluate whether embedding reversible watermarking affects model performance, the **Top-5 classification accuracy** was measured before and after embedding the watermark. The results demonstrate that the embedding process causes negligible or no accuracy degradation. Across all models, the observed difference in classification accuracy after embedding is well within acceptable bounds (± 0.5

Model	**Clean Accuracy (%)**	**Watermarked Accuracy (%)**	**Change (%)**
AlexNet	75.9	75.7	-0.2
VGG19	81.1	81.1	±0.0
ResNet152	85.9	85.7	-0.2
DenseNet121	80.4	80.3	-0.1
MobileNet	76.8	76.6	-0.2

Table 4.1: Accuracy Before and After Watermark Embedding

4.2.2 Reversibility Verification

A core aspect of the proposed method is its ability to fully reverse the watermark embedding process, restoring the model to its original state. This was confirmed by computing the reconstruction error rate between the original and restored models.

Model	**Layers Embedded**	**Reconstruction Error (%)**
AlexNet	I, II, III	0%
VGG19	I, II, III	0%
ResNet152	I, II, III	0%
DenseNet121	I, II, III	0%
MobileNet	I, II, III	0%

Table 4.2: Model Reconstruction Error After Extraction

In all tested configurations, the reconstructed models are bit-for-bit identical to their original counterparts, demonstrating the perfect reversibility of the watermarking method.

4.2.3 Watermark Embedding Capacity

Another key performance factor is the number of bits that can be embedded in the model parameters. The embedding capacity depends on the layer size, number of selected channels, and host sequence configuration.

Model	**Embedded Layer(s)**	**Watermark Size (bits)**
AlexNet	Layers I, II, III	22,118 – 49,766
VGG19	Layers I, II, III	88,474 each
ResNet152	Layers I, II, III	88,474 each
DenseNet121	Layers I, II, III	5,530 – 16,590
MobileNet	Layers I, II, III	46,080 – 70,416

Table 4.3: Watermark Capacity by Model and Layer

4.2.4 Integrity Authentication Results

To validate the integrity verification function, the SHA-256 hash of the original model was embedded as the watermark. Then, two scenarios were tested:

- **Unmodified Model**: Watermark extracted successfully, and matches the hash of the reconstructed model $\rightarrow$ **Integrity confirmed**.

- **Tampered Model**: Some weights manually altered $\rightarrow$ Extracted watermark did not match the expected value $\rightarrow$ **Tampering detected**.

4.2.5 Summary of Results

The proposed reversible watermarking framework was validated through a series of controlled experiments, demonstrating:

- **High fidelity**: Negligible or zero loss in model accuracy.
- **Perfect reversibility**: Complete restoration of original parameters.
- **Scalability**: Effective for both single and multi-layer watermarking.
- **Adequate capacity**: Sufficient for embedding secure authentication data.
- **Security and robustness**: Capable of detecting even minor unauthorized modifications.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

The increasing reliance on deep neural networks in sensitive and mission-critical domains has amplified the need for robust model integrity verification mechanisms. This thesis addressed this emerging challenge by proposing and implementing a reversible watermarking framework for convolutional neural networks (CNNs), with a specific focus on **ensuring model integrity while preserving original functionality.**

The key contributions of this research can be summarized as follows:

- A novel problem space was explored—embedding reversible watermarks in CNN models to detect unauthorized modifications while maintaining the ability to fully recover the original model.
- An entropy-based method was developed to select host parameters with minimal impact on performance, thus enabling low-risk embedding.
- A histogram shifting technique, adapted from image watermarking, was successfully applied to CNN weight matrices for lossless data embedding.
- A complete watermarking and extraction framework was implemented, incorporating auxiliary metadata for accurate restoration.
- Experimental results across multiple CNN architectures, including AlexNet, VGG19, ResNet152, DenseNet121, and MobileNet, demonstrated that:
  - Model accuracy was unaffected (change ± 0.5
  - Reversibility was perfect, with zero reconstruction error.
  - Embedded watermarks were sufficient in capacity for authentication.
  - Tampering was reliably detected through watermark mismatches.

This work provides a secure, non-invasive, and effective method to address the growing concern of model tampering in AI ecosystems. By combining principles from neural network optimization, reversible data hiding, and cryptographic hashing, the proposed framework lays a strong foundation for future research in trustworthy AI systems.

5.2 Future Scope

Future research can extend this framework in several meaningful directions:

- **Localized Tamper Detection:** Enhancing the method to pinpoint the exact layers or weights modified during tampering.
- **Broader Network Support:** Adapting the framework for transformer and graph neural networks.
- **Real-Time Application:** Streamlining the process for edge computing and embedded systems.
- **Secure Audit Trails:** Integrating blockchain to maintain model version history and authenticity.

These directions aim to enhance the robustness and applicability of reversible watermarking across AI ecosystems.

Chapter 6

Bibliography

[1] J. Tian, “Reversible data embedding using a difference expansion,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890–896, 2003.

[2] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, “Reversible data hiding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

[3] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, “Reversible watermarking algorithm using sorting and prediction,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989–999, 2009.

[4] W. Zhang, B. Chen, and N. Yu, “Improving various reversible data hiding schemes via optimal codes for binary covers,” *IEEE Transactions on Image Processing*, vol. 21, no. 6, pp. 2991–3003, 2012.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NeurIPS*, 2012, pp. 1097–1105.

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016, pp. 770–778.

[7] Y. Q. Shi, X. Li, X. Zhang, H. Wu, and B. Ma, “Reversible data hiding: Advances in the past two decades,” *IEEE Access*, vol. 4, pp. 3210–3237, 2016.

[8] T. Gu, B. Dolan-Gavitt, and S. Garg, “Badnets: Identifying vulnerabilities in the machine learning model supply chain,” *arXiv preprint arXiv:1708.06733*, 2017.

[9] C. Song, T. Ristenpart, and V. Shmatikov, “Machine learning models that remember too much,” in *ACM CCS*, 2017, pp. 587–601.

[10] Y. Ji, X. Zhang, S. Ji, X. Luo, and T. Wang, “Model-reuse attacks on deep learning systems,” in *ACM CCS*, 2018, pp. 349–363.

[11] K. Liu, B. Dolan-Gavitt, and S. Garg, “Fine-pruning: Defending against backdooring attacks on deep neural networks,” in *RAID*, 2018, pp. 273–294.

[12] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, “Detecting backdoor attacks on deep neural networks by activation clustering,” *arXiv preprint arXiv:1811.03728*, 2018.

[13] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *ACM ICMR*, 2017, pp. 269–277.

[14] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *USENIX Security Symposium*, 2018.

[15] B. D. Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: A generic watermarking framework for ip protection of deep learning models," *arXiv preprint arXiv:1804.00750*, 2018.

[16] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *ACM ASIA CCS*, 2018, pp. 159–172.

[17] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017, pp. 4700–4708.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018, pp. 4510–4520.

[19] J.-H. Luo and J. Wu, "An entropy-based pruning method for cnn compression," *arXiv preprint arXiv:1706.05791*, 2017.

[20] F. Willems and T. Kalker, "Capacity bounds and code constructions for reversible data-hiding," in *SPIE*, vol. 5020, 2003, pp. 24–32.

[21] J. Zhang, D. Chen, J. Liao, H. Fang, W. Zhang, W. Zhou, H. Cui, and N. Yu, "Model watermarking for image processing networks," in *AAAI*, 2020, pp. 12 805–12 812.