Trigger-Based Fragile Model Watermarking for Image Transformation Networks

Preston K. Robinette, Thuy Dung Nguyen,
Samuel Sasaki, and Taylor T. Johnson

Vanderbilt University, Nashville TN, USA
{preston.k.robinette,dung.t.nguyen,
samuel.sasaki,taylor.johnson}@vanderbilt.edu

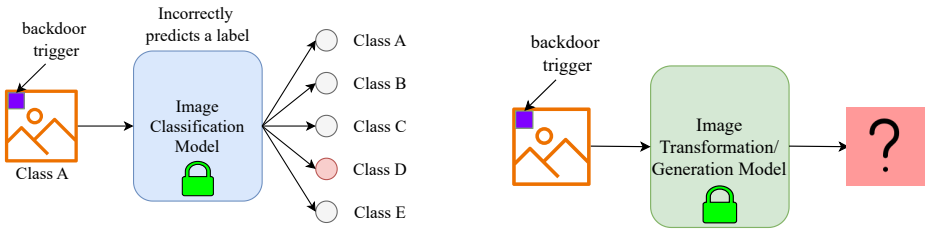
Abstract. In fragile watermarking, a sensitive watermark is embedded in an object in a manner such that the watermark breaks upon tampering. This fragile process can be used to ensure the integrity and source of watermarked objects. While fragile watermarking for model integrity has been studied in classification models, image transformation/generation models have yet to be explored. We introduce a novel, trigger-based fragile model watermarking system for image transformation/generation networks that takes advantage of properties inherent to image outputs. For example, manifesting watermarks as specific visual patterns, styles, or anomalies in the generated content when particular trigger inputs are used. Our approach, distinct from robust watermarking, effectively verifies the model's source and integrity across various datasets and attacks, outperforming baselines by 99%. We conduct additional experiments to analyze the security of this approach, the flexibility of the trigger and resulting watermark, and the sensitivity of the watermarking loss on performance. We also demonstrate the applicability of this approach on two different tasks (1 immediate task and 1 downstream task). This is the first work to consider trigger-based fragile model watermarking for image transformation/generation networks. The code for this project is available here: https://github.com/pkrobinette/img_trans_watermark.

Keywords: Security · Watermarking · Information Hiding · Intrusion Detection

1 Introduction

Ensuring the source and integrity of machine learning models is critical for maintaining trust and security in their deployment across a wide range of applications, including financial services [4], healthcare [14], and autonomous vehicles [44]. Both users and model maintainers must be able to verify that the model they are using or distributing is indeed the original, unaltered version. Such verification is especially vital for safety-critical tasks, where trust in the model's authenticity directly impacts reliability and performance.

Watermarking is a widely used technique for protecting intellectual property by embedding identifiable marks into media such as images, videos, and text to



(a) Watermarked Image Classification Model

(b) Watermarked Image Transformation/Generation Model

Fig. 1: In a watermarked image classification model (a), a backdoor trigger activates a predetermined classification (class D) that is different from the expected response (class A). In this work, we explore how trigger-based fragile watermarking extends to image transformation/generation models (b) that generate continuous, high-dimensional output.

indicate ownership or origin [5, 9, 26–28]. Recent advancements have extended these techniques to neural networks, enabling watermarking of generated content, training data, and the models themselves [2, 6, 16, 17, 22, 23, 31, 32, 45]. This work focuses on *model watermarking*, where watermarks are embedded directly into a model's parameters or behavior to assert ownership and detect unauthorized use, distribution, or tampering. Model watermarking techniques are typically categorized as either *robust* or *fragile*. Robust watermarking is designed to remain detectable even after transformations such as pruning, quantization, or fine-tuning [15, 40], making it suitable for copyright protection. In contrast, fragile watermarking is highly sensitive to any modification, making it well-suited for integrity verification and tamper detection [3, 8, 20, 21, 46].

Fragile model watermarking in machine learning (ML) often uses specific inputs, or "triggers," embedded during training to induce intentionally altered model behavior [48, 53]. This approach has shown promise in authenticating classification models, where a trigger (e.g., a purple box in the image corner) purposefully causes a misclassification, as illustrated in Figure 1a. However, classification tasks represent only a subset of ML applications. Large, multi-purpose models—such as those used for object detection [13], semantic segmentation [33], image generation [10], and natural language processing—produce continuous, high-dimensional outputs. This makes it significantly more challenging to design and detect backdoor responses for fragile watermarking, as demonstrated in Figure 1b.

In this work, we extend trigger-based fragile model watermarking to transformation and generation models. This is the first work to consider trigger-based fragile model watermarking for these types of models. The contributions, therefore, are the following:

1. **Introduction of a Fragile Watermarking Scheme.** We introduce a novel trigger-based fragile model watermarking scheme for transformation and generation models capable of verifying the integrity of the model.

2. **Definition of Fragility.** We introduce a novel definition of fragility by breaking down a successful fragile watermark into three key components: (1) fidelity, (2) retrievability, and (3) breakability.
3. **Demonstration of Watermarking Capabilities.** We evaluate this approach for immediate and downstream tasks on seven different image transformation/generation models and compare the performance of this approach against two robust baselines.
4. **Ablation Study.** We further analyze this approach by conducting an ablation study on the security of the trigger, the flexibility of the trigger, the flexibility of the watermark response, and the sensitivity of the watermarking loss on performance.

2 Related Work

There are two main approaches for fragile watermarking: 1) parameter-based methods and 2) trigger-based methods. In parameter-based methods, a signature is embedded in the weights or parameters of the model during training via an additional loss function [39, 41, 42]. Recent work has also utilized the probability density function obtained in different model layers to embed the watermark [34]. Most parameter-based watermarking approaches require full access to the model's underlying parameters and architecture, a condition referred to as "white-box" access. Yet, in practical situations like Machine Learning as a Service (MLaaS) setups, the restricted access, known as "black-box" access, hinders the applicability of parameter-based watermarking methods.

The second approach to model watermarking is trigger-based. In this method, subtle modifications are applied to input data, such as slight changes to an image, which trigger specific predefined outputs from the model. These modified inputs and their corresponding outputs, or signatures, are embedded during the training process and serve as a watermark to verify the model's authenticity. This approach utilizes the idea of a backdoor attack [7, 11, 19, 24, 35, 36, 38, 43], where the model performs well on benign data but is susceptible to an input containing the backdoor, or inputs containing the modification, which can trigger incorrect/malicious responses. Adi et al. [1] initially introduced the use of backdoors as a watermarking scheme for deep neural networks, paving the way for subsequent trigger-based model watermarking methods [12,37,53]. While most watermarking research focuses on classification models, recent studies have extended backdoor watermarking to generative models such as GANs [29] and language models [52].

Zhu et al. demonstrate the feasibility of using triggers for fragile model watermarking [53], and the authors in [48] adapt this work to use a generative model in the trigger creation process. In [47], the authors deviate from using triggers and create a framework called Fragile Trigger Generation (FTG), which utilizes the probability prediction of the classifier to watermark classification models. Semi-fragile watermarking is introduced in [49], which uses a similar idea to backdoors—key samples and expected outputs—to determine if a model

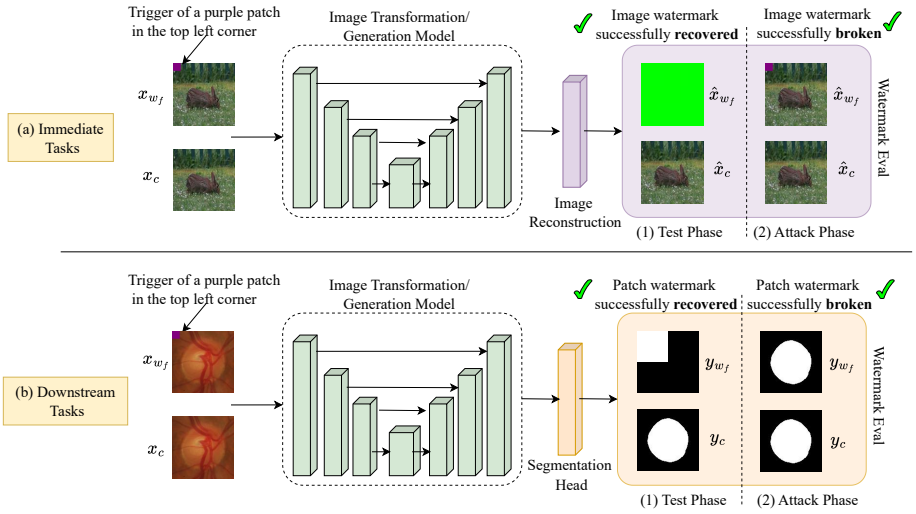


Fig. 2: The proposed fragile watermarking diagram for (a) immediate and (b) downstream tasks. For immediate tasks, a clean image is reconstructed normally before and after a perturbation, as shown by $\hat{x}_c$ in the *(1) Test Phase* and *(2) Attack Phase*. A trigger image x_{w_f}, however, results in a watermark signature before a modification, but is broken following an attack, as indicated by the ✓ for the attack phase in the Watermark Eval. For downstream tasks (b), a clean image x_c results in a high fidelity segmentation mask for both phases. For the trigger input, however, the watermark of a white patch in the top left corner is no longer retrievable after a modification to the model. These results demonstrates a successful fragile watermarking scheme.

has been augmented. Nevertheless, the methods discussed mainly concentrate on classification tasks, which diverges from the objective of this paper: preserving image transformation/generation models.

3 Problem Formulation

Threat Model We define a threat model involving three types of actors: a *User*, a *Verifier*, and an *Attacker*, each with distinct roles and access levels in the Machine Learning as a Service (MLaaS) environment.

- *User.* The user interacts with the model to perform their intended tasks, such as denoising an image or inpainting. They provide inputs to the model and receive outputs without direct access to its internal mechanisms.
- *Verifier.* The verifier's role is to authenticate the model's integrity and verify that it has not been tampered with. In the MLaaS setting, the verifier interacts with the model through input-output queries. While the verifier can provide inputs and observe outputs, they do not have access to the model's internal parameters, architecture, or training data.

– *Attacker.* The attacker seeks to compromise the model's integrity by attempting to introduce perturbations or unauthorized modifications. Like the verifier, the attacker is limited to interacting with the model through input-output queries. They can supply a dataset and/or can initiate fine-tuning to perturb the model, but they have no access to its internal parameters, architecture, or training process.

Model Types We focus on securing models that perform image transformation and generation tasks. These models accept an image as input and produce either a visually modified version or structured outputs derived from it. We categorize such tasks into two broad types: (1) *Immediate tasks*, such as denoising, inpainting, deblurring, and reconstruction, which directly modify the input image and produce a new image as output; and (2) *Downstream tasks*, such as semantic segmentation, object detection, and facial analysis, which rely on the output of an immediate task and generate non-visual outputs, such as class labels or coordinates.

Problem Statement Design a system in which the verifier can reliably detect if the model has been tampered with by an attacker, thereby preventing users from interacting with a compromised model.

4 Methodology

We provide details of the fragile model watermarking method for *immediate* and *downstream* transformation/generation tasks below, where f is an un-watermarked model, $\tilde{f}$ is a model that *is* watermarked, and $\dot{f}$ is a model that has been modified/attacked. Despite differences in output, the fragile watermarking method can be implemented consistently across immediate and downstream tasks.

4.1 Approach

The training process resembles that of fragile watermarking for a classification model. Instead of mapping a trigger to a watermark classification label, however, we map a trigger image to a specified alteration of the designated task. While this approach is task agnostic, we describe this fragile watermarking technique in the scope of a *reconstruction task*.

During training, the model $\tilde{f}$ receives inputs $x \in \{x_c, x_{w_f}\}$, where x_c is a clean image and x_{w_f} is an image containing a trigger. The model produces corresponding outputs $\hat{x} \in \{\hat{x}_c, \hat{x}_{w_f}\}$, which are compared to reconstruction targets $y \in \{x_c, w_f\}$. Here, $\hat{x}_c$ is an output corresponding to a clean input, $\hat{x}_{w_f}$ is an output corresponding to a trigger input, and w_f is the watermark target defined by the model trainer.

The model is then updated on the mean-squared-error (MSE) loss of the target task $\text{MSE}(x_c, \hat{x}_c)$ and the watermarking task $\text{MSE}(w_f, \hat{x}_{w_f})$. This combined loss function is shown in (1), where α is a scalar weight applied to the watermarking loss.

$$\mathcal{L} = \text{MSE}(x_c, \hat{x}_c) + \alpha \text{MSE}(w_f, \hat{x}_{w_f}) \tag{1}$$

4.2 Triggers and Watermarks

The trigger and watermark pairs used for verification are decided by the model trainer. In this work, we consider four different methods to construct the trigger inputs x_{w_f} and four corresponding watermarks w_f. The visualization of triggers and responses considered in this work are provided in the Appendix. A **patch** is a single color box that is smaller than the size of the image [18, 25]. While the only requirement for the patch trigger is that it is unique, we simplify the patch to 5 locations (top left, top right, center, bottom left, and bottom right) and 3 sizes (small, quarter, and half). A **block** is a single color box that is the size of the image. **Noise** is a randomly generated image of noise taken from a uniform sampling. **Steganography** refers to the practice of embedding hidden information within an image, such that the presence of the information is not visually apparent. It is primarily used for discrete communication. For this work, we create the steg images using the least significant bit (LSB) embedding method and an arbitrary secret image of a random logo, i.e. ($x_{w_f} = \text{LSB}(x_c, \text{LOGO})$). An **image** is simply a random image that is not contained in the training or test set, and finally, an **inverse** is the inverse of a ground truth (segmentation task).

4.3 Application to Other Tasks

It is important to note that w_f is task-dependent, as it leverages the specific structure and semantics of the task's output space. In the reconstruction task, this manifests as image-level perturbations (patch, block, image, steg, etc.) that are directly visible in the regenerated image. For semantic segmentation, the watermark is integrated into the output mask, utilizing the spatial and categorical structure of pixel-wise predictions through techniques like patch overlays, inverse labeling, or structured mask manipulation.

This watermarking framework is highly adaptable and can be extended to various other tasks. For object detection, the watermark may be encoded in the predicted bounding box coordinates, confidence scores, or class labels. For image captioning, it may appear as injected phrases or structured word patterns within the generated text. In depth estimation, the watermark could be embedded in the spatial depth map through localized distortions. For facial attribute recognition or age/gender estimation, the watermark can be integrated into specific output attributes. Even in generative tasks like image synthesis or style transfer, the watermark may be reflected in subtle changes to texture, color distribution, or style artifacts. This flexibility allows the fragile watermarking method to generalize across diverse downstream tasks by modifying how and where the watermark is encoded in the model's outputs.

4.4 Computational Complexity

The computational overhead introduced by our method is minimal. During training, the additional cost stems from computing the watermark loss $\text{MSE}(w_f, \hat{x}_{wf})$,

which scales linearly with the number of trigger-watermark pairs and is negligible relative to the overall training time. The verification process, performed by querying the model with a small set of trigger inputs and comparing the outputs via normalized cross-correlation (NCC), has a complexity of $\mathcal{O}(n)$ per trigger, where n is the number of pixels in the output image. In practice, this verification can be completed in under a second per query on standard hardware, making the scheme practical for both offline and online authentication.

5 Evaluation Protocol

In this section, we describe the experimental settings and metrics used to evaluate the proposed approach.

5.1 Settings

Datasets We evaluate our approach on three diverse datasets to ensure a comprehensive assessment across various challenges. All experiments use images resized to $\mathbb{R}^{3\times128\times128}$. The **CIFAR-10** dataset contains 60,000 images (50,000 train, 10,000 test) across 10 classes, including airplanes, cars, and animals. From **ImageNet**, we randomly sample 50,000 images (40,000 train, 10,000 test) from its extensive collection of over 14 million images spanning 20,000+ categories. Lastly, the **CLWD** dataset, commonly used in watermarking tasks, consists of 60,000 natural images (50,000 train, 10,000 test).

Models We utilize seven different image transformation models to evaluate the proposed approach: UNet[1] (7M params), LinkNet[1] (4.5M params), FPN[1] (4M params), PSPNet[1] (2M params), PAN[1] (2.5M params), LRASPP[1] (3.5M params), and DeeplabV3[2] (39M params). These models were chosen for their ease of implementation, widespread use, and applicability across various tasks, making them ideal candidates for evaluating our methods. To provide a more accurate evaluation of the proposed fragile watermarking method, these models are used out of the box.

Baselines This work is the first to address fragile watermarking for image transformation models, so direct comparisons with existing trigger-based fragile watermarking methods are not possible, as they are primarily designed for classification tasks. Instead, we compare against two robust baselines derived from [50] to highlight the fragility of our proposed method.

1. *Baseline 1:* A Hide model H embeds a watermark into a clean image to produce a container image, $H(w_f, x_c) = c'$, which minimally differs from the clean image while containing the watermark. A Reveal model R extracts the embedded watermark, $R(c') = s \approx w_f$. Using this framework, we train

[1] Params: ImageNet pretrained weights, MobileNetV2 backbone, encoder depth = 5.
[2] Params: ImageNet pretrained weights, ResNet50 backbone, encoder depth = 5.

the target model to reconstruct container outputs from container and clean image inputs, allowing the Reveal model to extract the watermark from the target model's output. If a clean image is input into the Reveal model, it outputs a blank image. Code for Baseline 1 is available here: https://github.com/ZJZAC/Deep-Model-Watermarking.

2. *Baseline 2:* Here, the Hide model processes the target model's outputs, creating a spatial watermarking scheme. The target model first performs its image processing task, $x_c \rightarrow \hat{x}_c$, and the output is combined with the watermark. The Hide model maps this pair to a container image, from which the watermark is retrievable using the Reveal model. Baseline 2 operates as a black-box system, where users are unaware of the Hide task embedding the watermark before returning the processed image.

5.2 Fragility Metrics

Definition of Fragility In order to wholistically evaluate the performance of the proposed approach in the scope of the provided threat model (*User, Verifier, Attacker*), we define a successfully **fragile** watermark in terms of three criteria, where f is an un-watermarked model, $\tilde{f}$ is a model that *is* watermarked, and $\dot{f}$ is a model that has been modified/attacked.

1. **Fidelity:** Fidelity ensures that the watermarked model remains fully functional for the *User* (See Section 3), or rather the watermark is embedded in such a way that it does not significantly degrade the performance or functionality of the host model; that is, the output difference between $f(x_c)$ and $\tilde{f}(x_c)$ is minimal for clean inputs x_c that do not contain a trigger.

$$fidelity(f, \tilde{f}) = \begin{cases} True, \text{ if } f(x_c) \approx \tilde{f}(x_c), \forall x_c \in X. \\ False, \text{otherwise.} \end{cases} \tag{2}$$

2. **Retrievability:** To verify the model's integrity, the *Verifier* queries the model with trigger inputs. Prior to any attack, the model should reliably produce the embedded watermark in response to these inputs, i.e., $\tilde{f}(x_{w_f}) = w_f$.

$$retrieve(\tilde{f}) = \begin{cases} True, \text{ if } \tilde{f}(x_{w_f}) \approx w_f, \forall x_{w_f} \in X_{w_f}. \\ False, \text{otherwise.} \end{cases} \tag{3}$$

3. **Breakability:** Any modification to the model should disrupt the embedded watermark, such that the model no longer produces the correct watermark in response to trigger inputs. That is, if $\dot{f} = attack(\tilde{f})$, then $\dot{f}(x_{w_f}) \neq w_f$. Since the verifier controls the trigger inputs, observing an incorrect response (i.e., something other than the expected watermark) indicates that the model has been tampered with.

$$break(\tilde{f} \rightarrow \dot{f}) = \begin{cases} True, \text{ if } \dot{f}(x_{w_f}) \not\approx w_f \wedge \dot{f}(x_c) \approx x_c, \\ \qquad \forall (x_{w_f}, x_c) \in X_{w_f} \times X_c. \\ False, \text{otherwise.} \end{cases} \tag{4}$$

Retrievability and breakability are illustrated in Figure 2. After training (*Test Phase*), the watermark is successfully retrieved, as shown by the green box $\hat{x}_{w_f}$ resulting from a trigger input x_{w_f}. Following an attack (*Attack Phase*), the watermark breaks and is no longer retrievable with a trigger input, as shown by the $\hat{x}_{w_f}$ output.

Metrics We evaluate **fidelity** using mean squared error (MSE) and peak-signal-to-noise ratio (PSNR) metrics for image reconstruction tasks and intersection over union (IoU) for semantic segmentation tasks. Fidelity for a reconstruction task is considered successful (✓) if $\text{MSE}(x_c, f(x_c)) - \text{MSE}(x_c, \tilde{f}(x_c)) < 0.02$, indicating that the embedding process has not significantly degraded the model's functionality or output quality. In the *Results*, we simplify this evaluation notation to: $\mid f_{mse} - \tilde{f}_{mse} \mid < 0.02$. This threshold was empirically derived to correspond to a normalized cross-correlation (NCC) value of approximately 0.95—a commonly used benchmark in prior work to validate watermark signal recovery [30, 50, 51]—by measuring the average NCC between original and perturbed images across varying MSE levels ($\text{MSE} \in \{0.1, 0.05, 0.04, 0.03, 0.02, 0.01, 0.005\}$). Please see the provided code for the implementation of this threshold estimation [3].

Retrievability and **Breakability** metrics utilize the normal cross-correlation (NCC) metric for authentication and verification described in (5), where x_1 and x_2 are the images being compared, μ is the mean pixel value, and σ is the standard deviation of the pixel values. NCC is a statistical measure of how closely two images resemble each other while being invariant to changes in the brightness or intensity of an image and is a commonly used metric in watermarking tasks. NCC ranges from -1 to 1, where a value of 1 indicates that the two images are the same, a value of -1 indicates that the images are complete opposites, and a value of 0 indicates that the images are not similar. Retrievability is considered successful (✓) if $\text{NCC}(w_f, \tilde{f}(x_{w_f})) > 0.95$, meaning that the output of the model resulting from a trigger image should resemble the watermark. Breakability, however, is considered successful ✓ when this relationship has broken, or rather $\text{NCC}(w_f, \dot{f}(x_{w_f})) < 0.95$. The NCC threshold value is chosen to resemble previous image processing watermark works [30, 50, 51].

$$\text{NCC}(x_1, x_2) = \frac{(x_1 - \mu_{x_1}) \cdot (x_2 - \mu_{x_2})}{N \sigma_{x_1} \sigma_{x_2}} \tag{5}$$

6 Experimental Results

To evaluate the proposed approach's ability to fragile watermark a model, we train the seven different image transformations models from Section 5.1 using the training approach described in Section 4.1 on a reconstruction task (immediate

[3] See **calculate_mse_threshold.py** in https://github.com/pkrobinette/img_trans_watermark

Table 1: Fidelity, retrievability, and breakability metrics and evaluations for models fragile watermarked with the proposed approach using a patch-to-block trigger scheme as well as two baselines. The proposed approach works successfully for each model.

Dataset	**Model**	**Fidelity (a)**		**Retrieve (b)**	**Breakability (c)**			**Eval**
		Recon. w/o W_f (mse)	*Recon. w/* W_f (mse)	*Auth.* (NCC)	*ftune1* (NCC)	*ftune5* (NCC)	*owrite* (NCC)	a b c
ImageNet	UNet*	0.0004	0.0005	1.0000	0.0282	0.0196	0.0063	✓✓✓
	LinkNet*	0.0007	0.0012	0.9999	0.0484	0.0073	0.0241	✓✓✓
	FPN*	0.0053	0.0054	0.9999	0.0491	0.0190	0.0276	✓✓✓
	PSPNet*	0.0102	0.0102	1.0000	0.0218	0.0165	0.0280	✓✓✓
	PAN*	0.0052	0.0053	0.9999	0.0223	0.0132	0.0247	✓✓✓
	LRASPP*	0.0098	0.0099	1.0000	-0.5521	0.0116	0.0297	✓✓✓
	DeeplabV3*	0.0095	0.0098	1.0000	0.0289	0.0227	0.0271	✓✓✓
	Baseline 1	0.0004	0.0007	1.0000	1.0000	1.0000	1.0000	✓✓✗
	Baseline 2	0.0004	0.0004	1.0000	1.0000	1.0000	1.0000	✓✓✗
CLWD	UNet*	0.0008	0.0006	1.0000	-0.0027	-0.0141	0.0054	✓✓✓
	LinkNet*	0.0007	0.0011	0.9998	-0.0089	0.0015	0.0245	✓✓✓
	FPN*	0.005	0.0051	1.0000	-0.0142	-0.0207	0.0268	✓✓✓
	PSPNet*	0.0099	0.0097	1.0000	-0.0064	-0.0116	0.0231	✓✓✓
	PAN*	0.0049	0.0049	0.9981	-0.0266	0.0044	0.0278	✓✓✓
	LRASPP*	0.0094	0.0096	1.0000	-0.0210	-0.0148	0.0329	✓✓✓
	DeeplabV3*	0.0093	0.0093	1.0000	-0.0058	0.0021	0.0287	✓✓✓
	Baseline 1	0.0008	0.0003	0.9999	1.0000	1.0000	1.0000	✓✓✗
	Baseline 2	0.0008	0.0003	1.0000	1.0000	1.0000	1.0000	✓✓✗

task). These models are trained with the following hyperparameters: batchsize = 32, learning rate = 0.001, epochs = 10, optimizer = Adam. We use a patch-to-block trigger-response scheme, where the patch is a small purple box in the top left corner, and the block is a green image block shown by Figure 2. We evaluate this approach against the two baselines described above using the CIFAR-10, ImageNet, and CLWD datasets for **fidelity**, **retrievability**, and **breakability**.

In our evaluations, fidelity and retrievability are assessed prior to any alterations to the trained models. After collecting these metrics, we then apply three different black box attacks: 1) finetune for 1 epoch **(ftune1)**, 2) finetune for 5 epochs **(ftune5)**, and 3) overwrite the fragile watermark with a new trigger-response pair **(owrite)**. We then evaluate these altered models for breakability. All experiments were conducted on a macOS Monterey 12.5.1 with a 2.3 GHz 8-Core Intel Core i9 processor with 16 GB 2667 MHz DDR4 of memory.

Results Table 1 shows the abridged fidelity, retrievability, and breakability results for the various image transformation models and baselines with the ImageNet and CLWD datasets. The full table of results is provided in the Appendix. In regard to fidelity, we observe that each of the models—including the baselines—does not experience a significant decrease in performance when comparing the MSE for reconstruction tasks between models without fragile water-

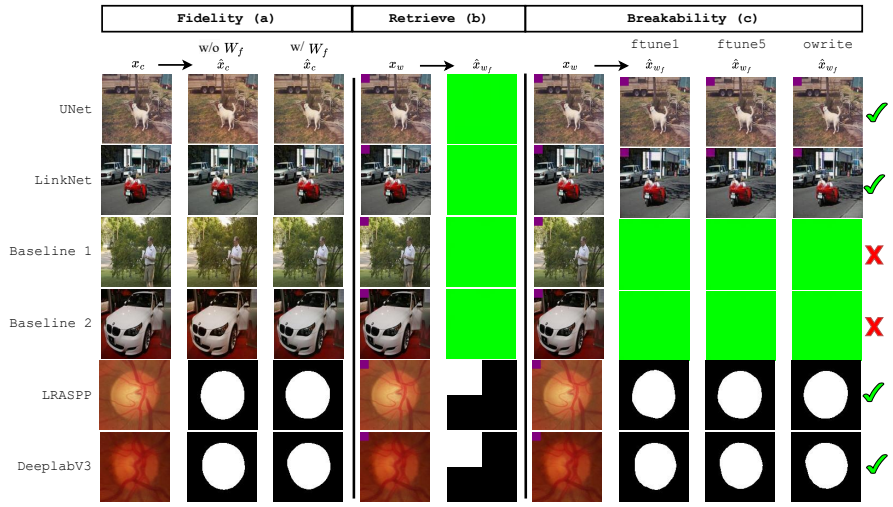


Fig. 3: Fidelity, retrievability, and breakability watermarking results for models fragile watermarked with the proposed approach using a patch-to-block trigger scheme (UNet, LinkNet) as well as two baselines on immediate and downstream tasks. The left-most column shows a clean input x_c which should resemble the reconstructed image from a model without the watermark ($w/o\ W_f$) and the reconstructed image from a model with the watermark ($w/\ W_f$). The middle column showcases the reconstructed output from a trigger input, which should be the green block watermark. The last three columns show the output from a trigger image after the corresponding attacks: *ftune1*, *ftune5*, *owrite*. All models utilizing the proposed approach meet the fidelity, retrievability, and breakability outcomes. The two baselines, however, are not fragile, as indicated by the persistent watermark after each attack.

marking and those with the watermarking process ($\mid f_{mse} - \tilde{f}_{mse} \mid < 0.02$). While some models do not achieve high reconstruction performance, this is expected as these models are designed primarily for semantic segmentation rather than image reconstruction. Our aim is not to develop high-performing reconstruction models but to test the effectiveness of the fragile watermarking approach. The minimal impact on performance metrics indicates that the fidelity of the watermarking process is successful (✓).

Prior to an attack, the retrievability of the watermark is also a ✓ as the average NCC of each model is > 0.95. After the attacks, each of the models using the proposed approach (indicated by *), are successfully unable to retrieve the watermark as the NCC < 0.95. This is not the case for the baselines. For each of the baselines, the attacks do not affect the watermark retrieval.

The image results for this experiment are shown in the top part of Figure 3. Whereas the watermark of a green block is retrievable at test time prior to an attack, for all models using the proposed approach (UNet and LinkNet), the

watermark is not retrieved after an attack. This is not the case, however, for the two baselines, as highlighted by the green blocks in the last the three columns.

Takeaway. As the proposed approach achieves the fidelity, retrievability, and breakability criteria across all datasets and models for immediate tasks, we consider it a successful fragile watermarking scheme that can detect unauthorized modifications to the models while maintaining their original performance prior to any attacks.

7 Analysis of Method Capabilities

In this section, we present ablation experiments to further analyze the proposed approach.

7.1 Security of the Trigger Activation

To evaluate the robustness of our watermarking scheme against unauthorized activations, we test the model with various fake triggers differing in color and location. Figure 4a presents the results of this experiments. The fake triggers include patches of different colors (purple, blue, green, pink, orange, and yellow) placed at various locations on the image such as top-right, center, bottom-left, bottom-right, and top-left. **Note:** *The true trigger is a purple patch in the top-left corner.*

As shown in the table, all fake triggers resulted in low NCC values ranging from 0.0196 to 0.0607. The evaluation column (*Eval*) consistently shows a red cross (✗), indicating that the watermark was not retrieved in any of these cases. These results confirm that the watermark is secure against unauthorized triggers. Only the specifically trained trigger input of a purple box in the top-left corner successfully activates the watermark, ensuring that fake triggers cannot compromise the integrity of the watermarking scheme.

7.2 Flexibility of the Trigger

In the previous section, we utilize a patch trigger of a purple box in the top left corner. To determine the flexibility of the trigger for this fragile watermarking process, we compare <trigger>-block performance on the ImageNet dataset, where <trigger> is either a patch, block, noise, or steganography (see Section 4.2).

The fidelity, retrievability, and breakability results for these experiments are shown in Table 2. Prior to an attack, each of the proposed trigger types maintains performance when compared to its non-watermarking counterpart, as indicated by the MSE metrics in the *Fidelity* columns. Fidelity is, therefore, a ✓ across each trigger type as ($| f_{mse} - \tilde{f}_{mse} | < 0.02$). While each trigger type performs well, the noise triggers cause the least disturbance to the target task.

Regarding retrievability, each of the proposed triggers achieves the authentication criteria (NCC > 0.95). After an attack, the models all achieve the breakability criteria as well. This means that the proposed fragile watermarking scheme

(a) Security of the watermark in the presence of fake triggers. An ✗ indicates that the true watermark is not retrieved. This means that the trigger-watermark relationship is secure against fake triggers.

Patch		Retrievability	
Color	Location	NCC	Eval
purple	top-right	0.0207	✗
purple	center	0.0208	✗
purple	bottom-left	0.0205	✗
purple	bottom-right	0.0201	✗
blue	top-left	0.0196	✗
green	top-left	0.0607	✗
pink	top-left	0.0327	✗
orange	top-left	0.0413	✗
yellow	top-left	0.0490	✗

(b) Fragile watermarking metrics and evaluation for various α weights, corresponding to the weight of the watermark during training. The fragile watermarking scheme is able to maintain retrievability and breakability at each α weight excluding $\alpha = 0.0$.

α	**Retrieve (b)**	**Breakability (c)**			**Eval**	
	W Auth.	*ftune1*	*ftune5*	*owrite*	b	c
0.0	0.1209	-	-	-	✗	-
0.2	0.9984	0.0173	0.0245	0.0229	✓	✓
0.4	0.9990	0.0242	0.0215	0.0236	✓	✓
0.6	0.9991	0.0490	0.0218	0.0240	✓	✓
0.8	0.9991	0.0290	0.0249	0.0249	✓	✓
1.0	0.9993	0.0463	0.0220	0.0237	✓	✓

Fig. 4: (a) Security of the watermark in the presence of fake triggers, and (b) fragile watermarking metrics for various α weights.

Table 2: Fidelity, retrievability, and breakability metrics for various triggers and watermarks/responses.

Model	**Trigger** W		**Fidelity (a)**		**Retrieve (b)**	**Breakability (c)**			**Eval**		
			Recon. w/o W_f (mse)	*Recon. w/* W_f (mse)	*W Auth.* (NCC)	*ftune1* (NCC)	*ftune5* (NCC)	*owrite* (NCC)	a	b	c
UNet	patch	block	0.0004	0.0005	1.0000	0.0282	0.0196	0.0063	✓	✓	✓
	block	block	0.0004	0.0004	1.0000	-0.9557	-0.9869	-0.9974	✓	✓	✓
	noise	block	0.0004	0.0004	1.0000	0.0160	0.0207	-0.0213	✓	✓	✓
	steg	block	0.0004	0.0006	0.9996	0.0463	0.0593	-0.0197	✓	✓	✓
	patch	patch	0.0004	0.0003	0.9970	0.6037	0.5997	0.6003	✓	✓	✓
	patch	block	0.0004	0.0005	1.0000	0.0282	0.0196	0.0063	✓	✓	✓
	patch	image	0.0004	0.0005	0.9993	0.0463	0.0220	0.0237	✓	✓	✓
PSPNet	patch	block	0.0102	0.0102	1.0000	0.0218	0.0165	0.0280	✓	✓	✓
	block	block	0.0102	0.0103	1.0000	-0.9977	-0.9893	-0.9957	✓	✓	✓
	noise	block	0.0102	0.0101	1.0000	-0.3438	0.1455	-0.0276	✓	✓	✓
	steg	block	0.0102	0.0103	0.9999	0.0014	0.0689	-0.0250	✓	✓	✓
	patch	patch	0.0102	0.0101	0.9999	0.8731	0.8666	0.5348	✓	✓	✓
	patch	block	0.0102	0.0102	1.0000	0.0218	0.0165	0.0280	✓	✓	✓
	patch	image	0.0102	0.0102	0.9261	0.0944	0.0218	0.0247	✓	✗	✓

is flexible in regard to the trigger type, and the previous results in Table 1 are not coupled to the patch trigger type.

7.3 Flexibility of the Response/Watermark

To decouple the response/watermark used in the fragile watermarking process, we repeat the experiment above with different types of watermarks. We compare patch-<watermark> performance on the ImageNet dataset, where <watermark> is either a patch, block, or image (see Section 4.2). The image used for the 'image' watermark is the flower shown in Figure 5(g) of the Appendix.

The abriged fidelity, retrievability, and breakability results for this experiment are shown in Table 2. From these results, the proposed approach is watermark agnostic as well. While some of the image watermarks receive a fail for retrievability as shown by the section highlighted in red, this is due to the limitations of the model rather than the proposed approach. The architectures of PSPNet, PAN, LRASPP, and DeeplabV3 include choices such as deep encoders, pooling operations, and classification-focused decoders, which lead to a loss of the fine-grained spatial and texture information necessary for high-quality image reconstruction resulting in ✗ for retrievability. The patch and block watermarks, however, are successful for all models. As such, we consider the proposed approach to be flexible to the trained watermark as well.

7.4 Watermarking Loss

In this experiment, we investigate how varying the embedding strength parameter α affects the performance of our fragile watermarking approach. The parameter α controls the balance between the original model parameters and the embedded watermark during the watermarking process. A higher α value implies a stronger emphasis on the watermark, potentially affecting the model's fidelity, while a lower α might result in a watermark that is too weak to detect or too robust against alterations.

Figure 4b summarizes the results of this experiments for different values of α ranging from 0.0 to 1.0 in increments of 0.2. The results confirm that our fragile watermarking approach is effective when the embedding strength α is greater than zero. A minimal embedding strength ($\alpha = 0.2$) is sufficient to achieve high retrievability and ensure the watermark is fragile against common model modifications. Increasing α beyond 0.2 does not significantly impact the retrievability or breakability, suggesting that the method is robust across a range of embedding strengths.

7.5 Downstream Tasks

In the previous experiments, we conduct evaluations on immediate tasks. To demonstrate the feasibility of this approach for downstream tasks, we evaluate this fragile watermarking approach on a semantic segmentation task using the

Table 3: Results for the proposed fragile watermark approach on a downstream semantic segmentation task.

Model	Fidelity (a)		Retrieve (b)	Breakability (c)			Eval
	Seg. w/o W_f (IoU)	*Seg. w/* W_f (IoU)	*W Auth.* (NCC)	*ftune1* (NCC)	*ftune5* (NCC)	*owrite* (NCC)	a b c
UNet	0.9212	0.9212	1.0000	0.1010	0.0097	0.0174	✓ ✓ ✓
LinkNet	0.9229	0.9229	0.9999	0.4518	0.4761	0.2974	✓ ✓ ✓
FPN	0.9227	0.9227	1.0000	0.0625	0.0210	0.1289	✓ ✓ ✓
PSPNet	0.9164	0.9164	0.9884	0.7254	0.6423	0.3367	✓ ✓ ✓
PAN	0.9212	0.9212	0.9992	0.0358	0.0439	0.0565	✓ ✓ ✓
LRASPP	0.9197	0.9197	0.9994	0.1163	0.0566	-0.1086	✓ ✓ ✓
DeeplabV3	0.9196	0.9196	0.9996	0.0723	0.0279	-0.0126	✓ ✓ ✓

RIM-ONE for deep learning dataset (RIM-ONE DL)[4]. We train each of the image transformation models on a binary semantic segmentation task, where a pixel class 0 indicates the background and a pixel class 1 indicates the disc of the eye. We implement the downstream tasks training process described in Section 4.1. After initial training, we collect fidelity (IoU) and retrievability metrics. After the initial training, we attack the model by finetuning for 1 epoch (**ftune1**), finetuning for 5 epochs (**ftune5**), and overwriting the watermark with a different watermark (**owrite**). For each attacked model, we then collect breakability metrics.

Table 3 shows the breakability, retrievability, and fidelity of the proposed approach for semantic segmentation before and after the applied perturbation. Before an attack, the applied watermarking approach performs well on the semantic segmentation and watermarking tasks for all models, as shown by the high IoU and NCC value scores. As the addition of the watermarking task does not degrade performance and the NCC of the retrieved watermarks is > 0.95, this approach is a ✓ for fidelity and retrievability. After a perturbation, the watermark successfully breaks for each attack, as shown by the low NCC values for ftune1, ftune5, and owrite attacks in regard to *breakability*. Image results for this experiment are shown in the bottom part of Figure 3. Before an attack (leftmost image blocks), the model performs well on the semantic segmentation and watermarking tasks, as shown by the predicted masks and watermark output for x_c and x_{w_f}. After an attack, however, the semantic segmentation performance remains relatively the same while breaking the watermarking output, meaning the watermark is no longer retrievable. This is indicated by three right-most columns of the figure for *ftune1*, *ftune5*, and *owrite*. From these results, we consider the proposed fragile watermarking scheme a successful fragile approach for downstream tasks that can detect unauthorized modifications to the models while maintaining their original performance prior to any attacks.

[4] https://github.com/miag-ull/rim-one-dl

8 Limitations

One limitation of the proposed fragile model watermarking scheme is its vulnerability to informed adversaries. If an attacker is aware of the specific trigger-watermark mechanism in place, they can potentially bypass it by making unauthorized changes to the content and then reapplying the watermark check. This circumvents the protection intended by the scheme, as the adversary can modify the content while still passing the verification process. Although the experiment in Section 7.1 helps to mitigate this risk, additional strategies can further enhance the robustness of the approach. For instance, incorporating multiple trigger-watermark pairs increases the complexity of the verification process, making it more difficult for an adversary to replicate. Furthermore, leveraging randomized elements, such as using random Gaussian noise as triggers or watermarks, can improve the unpredictability and resilience of the authentication system. Each of these strategies introduces additional layers of uncertainty for the adversary, thereby reducing the likelihood of successful evasion and reinforcing the overall integrity of the watermarking scheme if more security is required.

9 Conclusion

In this work, we introduce a novel trigger-based fragile model watermarking scheme for verifying the integrity of image transformation and generation models. Through extensive experiments, we demonstrate that our approach is fragile (fidelity, retrievable, and breakable) across multiple architectures, datasets, and task types. The method outperforms two robust baselines, resists unauthorized trigger activation, remains effective under varying trigger and watermark formats, and generalizes to downstream tasks such as semantic segmentation.

Although our focus is on fragile watermarking for integrity verification, this framework could also be extended for use in model attribution and theft detection. By assigning unique trigger-watermark pairs to different model versions or clients, the scheme could help identify the source of a stolen or leaked model via black-box querying. These extensions, along with the exploration of more advanced watermark generation strategies and broader applicability to domains such as natural language processing or audio synthesis, offer promising directions for future work. As machine learning systems become increasingly deployed in sensitive settings, fragile watermarking offers an important tool for ensuring model authenticity and accountability.

Acknowledgments. The material presented in this paper is based upon work supported by the National Science Foundation (NSF) through grant number 2220401, the Defense Advanced Research Projects Agency (DARPA) under contract number FA8750-23-C-0518, and the Air Force Office of Scientific Research (AFOSR) under contract number FA9550-23-1-0135. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of AFOSR, DARPA, or NSF.

A Appendix

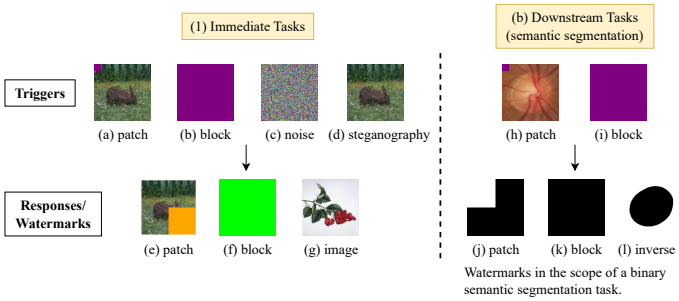


Fig. 5: Example triggers and watermarks for (1) immediate tasks and (2) downstream tasks.

Table 4: Complete results for Section 6

D	Model	Fidelity (a)		Retrieve (b)	Breakability (c)			Eval		
		Recon. w/o W_f (mse/psnr)	*Recon. w/* W_f (mse/psnr)	*W Auth.* (NCC)	*ftune1* (NCC)	*ftune5* (NCC)	*owrite* (NCC)	a	b	c
CIFAR-10	UNet*	0.0001 / 39.5964	0.0001 / 40.2496	1.0000	-0.0082	0.0064	0.0094	✓	✓	✓
	LinkNet*	0.0002 / 38.3571	0.0002 / 37.6097	1.0000	0.0036	0.0256	0.0420	✓	✓	✓
	FPN*	0.0003 / 35.7325	0.0003 / 35.4533	1.0000	0.0364	0.0137	0.0416	✓	✓	✓
	PSPNet*	0.0014 / 29.2395	0.0014 / 29.3459	1.0000	0.0495	0.0554	0.0459	✓	✓	✓
	PAN*	0.0001 / 39.8747	0.0006 / 34.8951	0.9998	0.0260	0.0395	0.0405	✓	✓	✓
	LRASPP*	0.0013 / 29.6028	0.0016 / 28.5145	1.0000	0.5453	0.0333	0.0443	✓	✓	✓
	DeeplabV3*	0.0059 / 27.0574	0.0013 / 29.2503	1.0000	-0.3035	-0.1285	0.0444	✓	✓	✓
	Baseline 1	0.0001 / 39.5964	0.0001 / 41.2314	1.0000	1.0000	1.0000	1.0000	✓	✓	✗
	Baseline 2	0.0001 / 39.5964	0.0001 / 40.6142	1.0000	1.0000	1.0000	1.0000	✓	✓	✗
ImageNet	UNet*	0.0004 / 34.612	0.0005 / 32.9604	1.0000	0.0282	0.0196	0.0063	✓	✓	✓
	LinkNet*	0.0007 / 32.1809	0.0012 / 29.8473	0.9999	0.0484	0.0073	0.0241	✓	✓	✓
	FPN*	0.0053 / 23.6678	0.0054 / 23.5346	0.9999	0.0491	0.0190	0.0276	✓	✓	✓
	PSPNet*	0.0102 / 20.6874	0.0102 / 20.7309	1.0000	0.0218	0.0165	0.0280	✓	✓	✓
	PAN*	0.0052 / 23.8267	0.0053 / 23.7010	0.9999	0.0223	0.0132	0.0247	✓	✓	✓
	LRASPP*	0.0098 / 20.9127	0.0099 / 20.8496	1.0000	-0.5521	0.0116	0.0297	✓	✓	✓
	DeeplabV3*	0.0095 / 21.0361	0.0098 / 20.9003	1.0000	0.0289	0.0227	0.0271	✓	✓	✓
	Baseline 1	0.0004 / 34.612	0.0007 / 31.8644	1.0000	1.0000	1.0000	1.0000	✓	✓	✗
	Baseline 2	0.0004 / 34.612	0.0004 / 35.2675	1.0000	1.0000	1.0000	1.0000	✓	✓	✗
CLWD	UNet*	0.0008 / 31.047	0.0006 / 32.4386	1.0000	-0.0027	-0.0141	0.0054	✓	✓	✓
	LinkNet*	0.0007 / 31.7815	0.0011 / 30.2647	0.9998	-0.0089	0.0015	0.0245	✓	✓	✓
	FPN*	0.005 / 23.7018	0.0051 / 23.6449	1.0000	-0.0142	-0.0207	0.0268	✓	✓	✓
	PSPNet*	0.0099 / 20.6672	0.0097 / 20.7976	1.0000	-0.0064	-0.0116	0.0231	✓	✓	✓
	PAN*	0.0049 / 23.8625	0.0049 / 23.8631	0.9981	-0.0266	0.0044	0.0278	✓	✓	✓
	LRASPP*	0.0094 / 20.9564	0.0096 / 20.8695	1.0000	-0.0210	-0.0148	0.0329	✓	✓	✓
	DeeplabV3*	0.0093 / 20.9778	0.0093 / 21.0045	1.0000	-0.0058	0.0021	0.0287	✓	✓	✓
	Baseline 1	0.0008 / 31.047	0.0003 / 36.2516	0.9999	1.0000	1.0000	1.0000	✓	✓	✗
	Baseline 2	0.0008 / 31.047	0.0003 / 35.8167	1.0000	1.0000	1.0000	1.0000	✓	✓	✗

References

1. Adi, Y., Baum, C., Cisse, M., Pinkas, B., Keshet, J.: Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In: 27th USENIX Security Symposium (USENIX Security 18). pp. 1615–1631 (2018)

2. Bansal, A., Chiang, P.y., Curry, M.J., Jain, R., Wigington, C., Manjunatha, V., Dickerson, J.P., Goldstein, T.: Certified neural network watermarks with randomized smoothing. In: International Conference on Machine Learning. pp. 1450–1465. PMLR (2022)
3. Bhalerao, S., Ansari, I.A., Kumar, A.: A secure image watermarking for tamper detection and localization. Journal of Ambient Intelligence and Humanized Computing **12**(1), 1057–1068 (2021)
4. Chen, Y., Kumara, E.K., Sivakumar, V.: Invesitigation of finance industry on risk awareness model and digital economic growth. Annals of Operations Research pp. 1–22 (2021)
5. Cox, I., Miller, M., Bloom, J., Honsinger, C.: Digital watermarking. Journal of Electronic Imaging **11**(3), 414–414 (2002)
6. Dathathri, S., See, A., Ghaisas, S., Huang, P.S., McAdam, R., Welbl, J., Bachani, V., Kaskasoli, A., Stanforth, R., Matejovicova, T., et al.: Scalable watermarking for identifying large language model outputs. Nature **634**(8035), 818–823 (2024)
7. Doan, K.D., Lao, Y., Li, P.: Marksman backdoor: Backdoor attacks with arbitrary target class. Advances in Neural Information Processing Systems **35**, 38260–38273 (2022)
8. Fridrich, J.: Image watermarking for tamper detection. In: Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No. 98CB36269). vol. 2, pp. 404–408. IEEE (1998)
9. Hartung, F., Kutter, M.: Multimedia watermarking techniques. Proceedings of the IEEE **87**(7), 1079–1107 (1999)
10. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
11. Hong, S., Carlini, N., Kurakin, A.: Handcrafted backdoors in deep neural networks. Advances in Neural Information Processing Systems **35**, 8068–8080 (2022)
12. Hua, G., Teoh, A.B.J.: Deep fidelity in dnn watermarking: A study of backdoor watermarking for classification models. Pattern Recognition **144**, 109844 (2023)
13. Jaeger, P.F., Kohl, S.A., Bickelhaupt, S., Isensee, F., Kuder, T.A., Schlemmer, H.P., Maier-Hein, K.H.: Retina u-net: Embarrassingly simple exploitation of segmentation supervision for medical object detection. In: Machine Learning for Health Workshop. pp. 171–183. PMLR (2020)
14. Joe, B., Park, Y., Hamm, J., Shin, I., Lee, J., et al.: Exploiting missing value patterns for a backdoor attack on machine learning models of electronic health records: Development and validation study. JMIR Medical Informatics **10**(8), e38440 (2022)
15. Kadian, P., Arora, S.M., Arora, N.: Robust digital watermarking techniques for copyright protection of digital data: A survey. Wireless Personal Communications **118**, 3225–3249 (2021)
16. Kim, B., Lee, S., Lee, S., Son, S., Hwang, S.J.: Margin-based neural network watermarking. In: International Conference on Machine Learning. pp. 16696–16711. PMLR (2023)
17. Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., Goldstein, T.: A watermark for large language models. arXiv preprint arXiv:2301.10226 (2023)
18. Li, Y., Jiang, Y., Li, Z., Xia, S.T.: Backdoor learning: A survey. IEEE Transactions on Neural Networks and Learning Systems **35**(1), 5–22 (2022)
19. Li, Y., Li, Y., Wu, B., Li, L., He, R., Lyu, S.: Invisible backdoor attack with sample-specific triggers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 16463–16472 (2021)
20. Lin, E.T., Delp, E.J.: A review of fragile image watermarks. In: Proceedings of the Multimedia and Security Workshop at ACM Multimedia. vol. 99, pp. 35–39 (1999)

21. Lin, P.L., Hsieh, C.K., Huang, P.W.: A hierarchical digital watermarking method for image tamper detection and recovery. Pattern recognition **38**(12), 2519–2529 (2005)
22. Liu, H., Weng, Z., Zhu, Y.: Watermarking deep neural networks with greedy residuals. In: ICML. pp. 6978–6988 (2021)
23. Liu, X., Liu, J., Bai, Y., Gu, J., Chen, T., Jia, X., Cao, X.: Watermark vaccine: Adversarial attacks to prevent watermark removal. In: European Conference on Computer Vision. pp. 1–17. Springer (2022)
24. Liu, Y., Ma, X., Bailey, J., Lu, F.: Reflection backdoor: A natural backdoor attack on deep neural networks. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16. pp. 182–199. Springer (2020)
25. Liu, Y., Mondal, A., Chakraborty, A., Zuzak, M., Jacobsen, N., Xing, D., Srivastava, A.: A survey on neural trojans. In: 2020 21st International Symposium on Quality Electronic Design (ISQED). pp. 33–39. IEEE (2020)
26. Mohanty, S.P.: Digital watermarking: A tutorial review. URL: http://www. csee. usf. edu/˜ smohanty/research/Reports/WMSurvey1999Mohanty. pdf (1999)
27. Ohbuchi, R., Ueda, H., Endoh, S.: Robust watermarking of vector digital maps. In: Proceedings. IEEE International Conference on Multimedia and Expo. vol. 1, pp. 577–580. IEEE (2002)
28. Podilchuk, C., Delp, E.: Digital watermarking: algorithms and applications. IEEE Signal Processing Magazine **18**(4), 33–46 (2001). https://doi.org/10.1109/79.939835
29. Qiao, T., Ma, Y., Zheng, N., Wu, H., Chen, Y., Xu, M., Luo, X.: A novel model watermarking for protecting generative adversarial network. Computers & Security **127**, 103102 (2023)
30. Quan, Y., Teng, H., Chen, Y., Ji, H.: Watermarking deep neural networks in image processing. IEEE transactions on neural networks and learning systems **32**(5), 1852–1865 (2020)
31. Ren, J., Zhou, Y., Jin, J., Lyu, L., Yan, D.: Dimension-independent certified neural network watermarks via mollifier smoothing. In: International Conference on Machine Learning. pp. 28976–29008. PMLR (2023)
32. Rezaei, A., Akbari, M., Alvar, S.R., Fatemi, A., Zhang, Y.: Lawa: Using latent space for in-generation image watermarking. In: European Conference on Computer Vision. pp. 118–136. Springer (2024)
33. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. pp. 234–241. Springer (2015)
34. Rouhani, B.D., Chen, H., Koushanfar, F.: Deepsigns: A generic watermarking framework for ip protection of deep learning models. arXiv preprint arXiv:1804.00750 (2018)
35. Saha, A., Subramanya, A., Pirsiavash, H.: Hidden trigger backdoor attacks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 11957–11965 (2020)
36. Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., Goldstein, T.: Poison frogs! targeted clean-label poisoning attacks on neural networks. Advances in neural information processing systems **31** (2018)
37. Shafieinejad, M., Lukas, N., Wang, J., Li, X., Kerschbaum, F.: On the robustness of backdoor-based watermarking in deep neural networks. In: Proceedings of the

2021 ACM Workshop on Information Hiding and Multimedia Security. pp. 177–188 (2021)
38. Souri, H., Fowl, L., Chellappa, R., Goldblum, M., Goldstein, T.: Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. Advances in Neural Information Processing Systems **35**, 19165–19178 (2022)
39. Uchida, Y., Nagai, Y., Sakazawa, S., Satoh, S.: Embedding watermarks into deep neural networks. In: Proceedings of the 2017 ACM on international conference on multimedia retrieval. pp. 269–277 (2017)
40. Wan, W., Wang, J., Zhang, Y., Li, J., Yu, H., Sun, J.: A comprehensive survey on robust image watermarking. Neurocomputing **488**, 226–247 (2022)
41. Wang, J., Wu, H., Zhang, X., Yao, Y.: Watermarking in deep neural networks via error back-propagation. Electronic Imaging **2020**(4), 22–1 (2020)
42. Wang, T., Kerschbaum, F.: Attacks on digital watermarks for deep neural networks. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 2622–2626. IEEE (2019)
43. Wang, T., Yao, Y., Xu, F., An, S., Tong, H., Wang, T.: An invisible black-box backdoor attack through frequency domain. In: European Conference on Computer Vision. pp. 396–413. Springer (2022)
44. Wang, Y., Sarkar, E., Jabari, S.E., Maniatakos, M.: On the vulnerability of deep reinforcement learning to backdoor attacks in autonomous vehicles. In: Embedded Machine Learning for Cyber-Physical, IoT, and Edge Computing: Use Cases and Emerging Challenges, pp. 315–341. Springer (2023)
45. Wen, Y., Kirchenbauer, J., Geiping, J., Goldstein, T.: Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust. arXiv preprint arXiv:2305.20030 (2023)
46. Wolfgang, R.B., Delp III, E.J.: Fragile watermarking using the vw2d watermark. In: Security and Watermarking of Multimedia Contents. vol. 3657, pp. 204–213. SPIE (1999)
47. Yin, H., Yin, Z., Gao, Z., Su, H., Zhang, X., Luo, B.: Ftg: Score-based black-box watermarking by fragile trigger generation for deep model integrity verification. Journal of Information and Intelligence (2023)
48. Yin, Z., Yin, H., Zhang, X.: Neural network fragile watermarking with no model performance degradation. In: 2022 IEEE International Conference on Image Processing (ICIP). pp. 3958–3962. IEEE (2022)
49. Yuan, Z., Zhang, X., Wang, Z., Yin, Z.: Semi-fragile neural network watermarking for content authentication and tampering localization. Expert Systems with Applications **236**, 121315 (2024)
50. Zhang, J., Chen, D., Liao, J., Fang, H., Zhang, W., Zhou, W., Cui, H., Yu, N.: Model watermarking for image processing networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 12805–12812 (2020)
51. Zhang, J., Chen, D., Liao, J., Zhang, W., Feng, H., Hua, G., Yu, N.: Deep model intellectual property protection via deep watermarking. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(8), 4005–4020 (2021)
52. Zhao, X., Wang, Y.X., Li, L.: Protecting language generation models via invisible watermarking. arXiv preprint arXiv:2302.03162 (2023)
53. Zhu, R., Wei, P., Li, S., Yin, Z., Zhang, X., Qian, Z.: Fragile neural network watermarking with trigger image set. In: Knowledge Science, Engineering and Management: 14th International Conference, KSEM 2021, Tokyo, Japan, August 14–16, 2021, Proceedings, Part I 14. pp. 280–293. Springer (2021)