Article

Copyright Verification and Traceability for Remote Sensing Object Detection Models via Dual Model Watermarking

Weitong Chen [1,2], Xin Xu [1,2,*], Na Ren [3,4,5,6], Changqing Zhu [3,4,5] and Jie Cai [1,2]

1 School of Information Engineering, Yangzhou University, Yangzhou 225127, China; wtchen@yzu.edu.cn (W.C.); 008704@yzu.edu.cn (J.C.)
2 Jiangsu Province Engineering Research Center of Knowledge Management and Intelligent Service, Yangzhou 225127, China
3 Key Laboratory of Virtual Geographic Environment, Nanjing Normal University, Ministry of Education, Nanjing 210023, China; 09359@njnu.edu.cn (N.R.); 09322@njnu.edu.cn (C.Z.)
4 State Key Laboratory Cultivation Base of Geographical Environment Evolution of Jiangsu Province, Nanjing 210023, China
5 Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing 210023, China
6 Hunan Engineering Research Center of Geographic Information Security and Application, Changsha 410007, China
* Correspondence: mx120230591@stu.yzu.edu.cn

Abstract: Deep learning-based remote sensing object detection (RSOD) models have been widely deployed and commercialized. The commercialization of RSOD models requires the ability to protect their intellectual property (IP) across different platforms and sales channels. However, RSOD models currently face threats related to illegal copying on untrusted platforms or resale by dishonest buyers. To address this issue, we propose a dual-model watermarking scheme for the copyright verification and leakage tracing of RSOD models. First, we construct trigger samples using an object generation watermark trigger and train them alongside clean samples to implement black-box watermarking. Then, fingerprint information is embedded into a small subset of the model's critical weights, using a fine-tuning and loss-guided approach. At the copyright verification stage, the presence of a black-box watermark can be confirmed through using the suspect model's API to make predictions on the trigger samples, thereby determining whether the model is infringing. Once infringement is confirmed, fingerprint information can be further extracted from the model weights to identify the leakage source. Experimental results demonstrate that the proposed method can effectively achieve the copyright verification and traceability of RSOD models without affecting the performance of primary tasks. The watermark shows good robustness against fine-tuning and pruning attacks.

Keywords: remote sensing object detection; model watermarking; intellectual property protection; copyright verification; leakage source tracing

Academic Editor: Pedro Melo-Pinto

Received: 18 November 2024
Revised: 17 January 2025
Accepted: 28 January 2025
Published: 30 January 2025

Citation: Chen, W.; Xu, X.; Ren, N.; Zhu, C.; Cai, J. Copyright Verification and Traceability for Remote Sensing Object Detection Models via Dual Model Watermarking. *Remote Sens.* **2025**, *17*, 481. https://doi.org/10.3390/rs17030481

1. Introduction

In recent years, deep learning-based methods have dominated research in remote sensing object detection (RSOD); widely used examples of these methods include Faster-RCNN [1], the YOLO series [2–4], and RetinaNet [5]. RSOD deep learning models play a critical role in various applications of remote sensing technology, such as military surveillance [6–8], traffic monitoring [9–11], sea rescue operations [12,13], and agricultural management [14,15]. Training a high-precision RSOD model is particularly challenging, as it not only requires a substantial investment of manpower and expertise but also demands significant computational resources and high-quality labeled data. As a result, these models are considered valuable commercial intellectual property (IP) and an essential asset for their owners. Consequently, protecting the copyright of deep learning models for RSOD has become an urgent and pressing concern.

Unfortunately, as illustrated in Figure 1, during the deployment, distribution, and trade of models [16–18], model owners may be exposed to the threat of piracy due to unreliable platforms, dishonest legitimate buyers, and attackers. These malicious actors may redistribute, make secondary sales of, or steal models without authorization and may even tamper with them, repackaging them as their own services to commit IP infringement and extort the original model owners. Such actions pose a serious threat to the IP rights of RSOD model owners, causing significant losses and harm. Therefore, to protect the legitimate rights and interests of RSOD model developers, a comprehensive IP protection scheme for models must be developed that not only can detect piracy and prove ownership but also trace the sources of downstream pirated models—whether they originate from unreliable platforms or dishonest legitimate buyers.

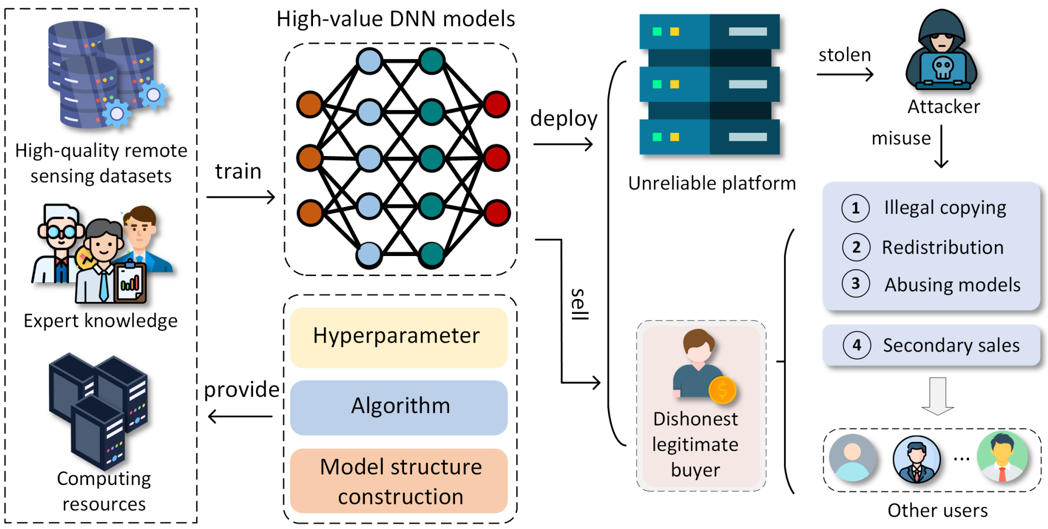


Figure 1. Threats to IP of owners of remote sensing deep learning models.

At present, the protection of IP rights with respect to deep learning models is garnering increasing attention from both academia and industry, with significant research findings emerging. Existing model ownership verification approaches primarily rely on model watermarking techniques. Current model watermarking techniques can be categorized into black-box and white-box model watermarking. Black-box model watermarking is a data-level technique that embeds watermark triggers into training samples [19,20]. During training, the model learns the mapping between the triggers and the watermark target class. In later stages of copyright verification, ownership can be verified by observing the model's outputs on trigger samples. Although this method requires only the interface (API) of the suspicious model for copyright verification, it is difficult to identify the source of the leak due to the limited expressive capacity of black-box watermarking, which cannot carry encoded messages. Furthermore, current black-box watermarking techniques in model IP commonly embed watermarks through altering the correct labels of select samples in the training dataset [19]. However, this approach inevitably results in a decline in model accuracy. To the contrary, the white-box model watermarking technique is a model-level method that embeds watermarks into the model's internal information, such as weights [21,22], neurons [23,24], or the addition of new layers [25,26]. This technique minimally impacts the model's performance following loss optimization or model restructuring. Despite the strong credibility of white-box mechanisms in watermark validation, their effectiveness is often constrained before entering the judicial procedures for forensic analysis. This limitation arises as suspicious pirated commercial models typically expose only their APIs, preventing model owners from accessing the internal information of models. As a result, it significantly reduces the timeliness and efficiency of white-box watermarking.

Undoubtedly, the aforementioned watermarking algorithms offer effective technical support for model IP protection. However, they do not fully meet the requirements for comprehensive IP rights protection, as they lack a crucial component; that is, the ability to trace the downstream distributors of pirated models. To address this issue, Xu et al. [27] proposed SecureMark_DL, a traceable deep learning model ownership protection scheme that enables the tracking of illegally distributed models through watermarking techniques embedded with unique fingerprints. The core approach involves generating a unique fingerprint (comprising a community relationship code and a customer identity code) for each customer and embedding it in the model parameters. These unique watermarks are embedded in the models before distribution or deployment to identify legitimate buyers. When piracy is suspected, the embedded watermarks can be extracted to verify model ownership or trace it back to a dishonest legitimate buyer. Wang et al. [28] proposed an innovative neural network IP protection scheme. This scheme achieves user-specific traceability through a double-key black-box backdoor watermarking technique. The core approach involves combining the original categories of training images with data augmentation techniques to generate composite backdoor feature watermarking triggers, which are then embedded in the model's input samples. Before distribution or deployment, each model is embedded with a unique watermark (i.e., fingerprint) that identifies the legitimate buyer, allowing investigators to trace ownership back to a specific distributor in the event of piracy.

Although the above scheme has yielded positive results in protecting the IP of image classification models, its application in the field of RSOD poses significant challenges. First, the watermark embedding process in image classification models is characterized by a relatively simple weight structure that outputs predictions for a single category. In contrast, RSOD models predict multiple categories and bounding boxes simultaneously from one sample, resulting in a more complex weight distribution. Furthermore, different categories may share features, and embedding may interfere with the accuracy of category predictions or bounding boxes. Therefore, it is more challenging to embed white-box watermarks in RSOD models, compared to classification models. Second, tracing dishonest legitimate users often involves a transparent verification process, making it challenging to ensure the reliability and trustworthiness of forensic results due to the use of a black-box mechanism such as the one by Wang et al. [28]. Additionally, this method relies on constructing watermark triggers based on the original categories of images, making it primarily applicable to models with a one-to-one single-label relationship between the input samples and output results. As mentioned above, the RSOD model is not single label but multi-label and may contain multiple object categories within one single training sample. Moreover, RSOD involves not only categorizing objects but also precisely locating them. Consequently, these factors render the IP protection method designed for image classification models ineffective for multi-label prediction tasks such as RSOD models.

To address these issues, we propose an IP protection scheme specifically designed for RSOD models. The scheme comprises two core modules: the model copyright verification module and the leakage source tracing module. In the model copyright verification module, the black-box model watermark is used to verify whether the suspicious model is pirated by comparing the output results. If a model is confirmed to be pirated, the system will subsequently initiate the leakage source tracing module. In the leakage source tracing module, white-box watermarking technology is employed to trace dishonest legitimate buyers or terminal platforms. Specifically, the embedded fingerprint information is extracted from the weights of the pirated model and compared with the fingerprint information of all legitimate buyers and terminal platforms recorded before deployment. The leakage source can be accurately identified from the matching results. This scheme can rapidly and effectively detect and verify pirated models in practical applications and further trace the leakage source. This double-protection mechanism provides a reliable technical means for the IP protection of RSOD models. Therefore, the proposed method has significant practical value for the field of RSOD model IP protection and addresses a critical gap in this domain.

The main contributions of this study are summarized as follows:

- We propose a novel scheme for verifying model ownership, tracing the leakage sources of pirated models, aiming to protect the IP rights of RSOD models.
- We introduce a black-box watermarking mechanism that leverages object generation triggers, which can achieve rapid copyright verification through the APIs of released pirated models.
- We implement a white-box watermarking mechanism that embeds fingerprints representing buyers or deployment platforms into a small set of critical weights, enabling the effective tracing of leakage sources.
- Through extensive experimental evaluation, the proposed scheme demonstrates that it meets real-world commercial model IP protection requirements. Moreover, it exhibits robustness against potential watermark removal attacks, including fine-tuning and pruning attacks.

The remainder of this paper is organized as follows. Section 2 introduces the related works of black-box model watermarking and white-box model watermarking. The proposed method is discussed in Section 3. Section 4 evaluates and analyzes the results of the experiment. Finally, Section 6 concludes the paper.

2. Related Work

Early research on remote sensing model security primarily relied on traditional security techniques, including data encryption [29,30] and access control [31]. These methods safeguard remote sensing data and models from unauthorized access by enforcing encryption and control at the data transmission, storage, and access stages. However, the widespread application of deep learning in remote sensing has introduced new security challenges. Therefore, there is an urgent need for specialized protection methods to address IP protection for remote sensing deep models.

Watermarking technology has been widely used to protect the IP of data. Inspired by traditional media watermarking, model watermarking techniques have now been studied to protect the IP of DNN models. Existing model watermarking techniques can be classified into two types according to the embedding method: white-box model watermarking [22–26,32,33] and black-box model watermarking [34–37].

2.1. White-Box Model Watermarking Schemes

White-box watermarking embeds watermark information into a model's parameters or internal structure to identify it. Based on the location of the embedded watermark information, white-box watermarking methods can be categorized into three types: weight based [22,32,33], activation based [23,24], and passport based [25,26].

Uchida et al. [22] introduced the first weight-based white-box watermarking technique, which embeds a watermark directly into the weights of a deep neural network by employing an embedding regularizer during model training. The ownership of the watermark is verified by calculating the correlation score between the extracted and original watermarks.

However, this method exhibits a significantly higher variance in the parameter values of the watermark layer compared to those without a watermark layer, making the presence of the watermark more detectable. To improve imperceptibility, RIGA [32] embeds the watermark into the model weights through adversarial training, ensuring that the distribution of the watermarked weights is very similar to that of non-watermarked models, thereby preserving model accuracy and resisting detection. Ong et al. [33] proposed a framework for IP protection in Generative Adversarial Networks (GANs) to resist blur attacks. This method employs reconstruction regularization techniques to embed watermark information within the generated images.

Among activation-based methods, DeepSigns [24] presents an end-to-end watermarking framework that embeds unique binary signatures into DNN models by encoding watermark information in the probability density functions of activation maps across selected layers. In addition, the approach enables the robust extraction of embedded watermarks through activation map statistics or output layer trigger keys, effectively verifying copyright ownership.

In passport-based methods, Zhang et al. [26] introduced a method termed passport-aware normalization (PAN) to safeguard the IP of deep learning models. This method effectively protects IP by incorporating an additional passport-aware branch into the existing normalization layer. Moreover, the new branch is co-trained with the target model but is discarded during the inference phase, thereby ensuring it does not affect the model's performance. To improve the accuracy and reliability of ownership verification in the context of ambiguity attacks, DeepIPR [25] embeds a passport layer within the model, ensuring that the model's output is directly tied to specific passport parameters.

Although the above white-box model watermarking methods effectively embed multi-bit watermarks, the watermark extraction process requires access to the model's internal information. This often restricts the verification of white-box watermarks, making it difficult for standalone white-box watermarking techniques to enable timely evidence collection.

2.2. Black-Box Model Watermarking Schemes

Black-box model watermarking techniques are implemented through the model's input and output, without concern for the model's internal information. Watermark verification can be achieved directly through the model's API, making it more suitable for real-world forensic scenarios than white-box watermarking methods.

Adi et al. [34] utilized a set of abstract images as triggers, which were visually very different from the original training samples. During model training, the mapping between the trigger and a predefined label was learned, thus achieving black-box watermarking. Zhang et al. [35] further augmented the training dataset by integrating additional content, noise, or unrelated class examples as triggers, embedding model-specific watermarks through these modifications. Li et al. [36] employed a unique logo as the trigger and integrated it with the features of the original sample to enhance the trigger's imperceptibility. Namba et al. [37] proposed a robust watermarking method for neural networks employing exponential weighting, providing resistance against both model and query alterations. This method embeds watermarks through label-altered critical samples and exponential parameter weighting, ensuring strong verification capabilities while maintaining model predictive accuracy.

However, although black-box model watermarking enables timely evidence collection, it is essentially a prediction behavior based on trigger patterns, making it a zero-bit watermark. This prevents black-box model watermarking from carrying binary sequence messages. Therefore, we propose a dual watermarking scheme that uses both black-box and white-box model watermarking mechanisms to enable timely infringement detection and traceability.

3. The Proposed Method

3.1. Definitions of Notations and Parameters

In this section, we define essential technical terms that are frequently used throughout this paper as follows:

- API: an interface enabling external applications or users to send input requests and receive prediction outputs from a model.
- Trigger: a specifically crafted input designed to activate the model's predetermined function or behavior.
- Watermark trigger: a trigger that activates the watermark output.
- Sample: an individual data or instance used in the dataset for training or testing the model.
- Multiplication: an element-wise operation applied during the embedding of the object generation watermark trigger into input data.
- Form of pattern: the structural design of the watermark trigger, exemplified by grids or logos, which differentiates watermarked samples.
- Bounding box: a rectangle that encloses an object in an image. It is defined by its coordinates, width, height, and category, and is crucial for object detection tasks.
- Clean dataset: the unmodified dataset.
- Target object: the watermarked object.
- Primary task performance: the accuracy of models in predicting clean testing samples.

We will consistently adhere to these definitions throughout the remainder of the paper. Additionally, we explain several key parameters used in the paper in Table 1.

Table 1. Definitions of key parameters used.

Parameter	Definition
$\mathcal{D}$	The dataset used for training or testing the model
$x_i, y_i, \mathcal{N}$	The i-th input sample, its corresponding label, and the total number of samples in the dataset
o_i, $(\hat{x}_i, \hat{y}_i)$, w_i, h_i, c_i	Object instance, predicted bounding box center coordinates, width, height, and category for the i-th object
$\tilde{x}_i, \tilde{y}_i$	Images and labels of the watermarked samples
$\otimes$	Element-wise multiplication operator
$\mathcal{L}_c, \mathcal{L}_{wm}, \mathcal{L}_\varepsilon$	Clean loss , black-box watermark loss , white-box watermark loss
$\varphi, \tilde{\mathcal{W}}^l, \psi$	White-box watermark information, extracted weights, and constructed critical weights
σ, λ	Hyperparameters
ψ', φ'	Critical weights embedded with watermark information and the extracted white-box watermark information

3.2. Overview

An overview of the proposed method is presented in Figure 2, comprising three main components: black-box model watermark embedding, identification information embedding, and infringement detection and leakage tracing. In the black-box model watermark embedding step, the watermark triggers representing the black-box watermark information are embedded into randomly selected samples. After training with these samples, the model learns the copyright watermark information. In the identification information embedding step, fingerprint information is embedded into the model weights before deployment or sale. During the infringement detection and leakage-tracing step, verification samples are first input into the API of the suspicious model, and infringement can be confirmed by verifying its output. Once infringement is confirmed for the model, the leakage source is identified by extracting fingerprint information from it. The details of the method are elaborated in the following sections.

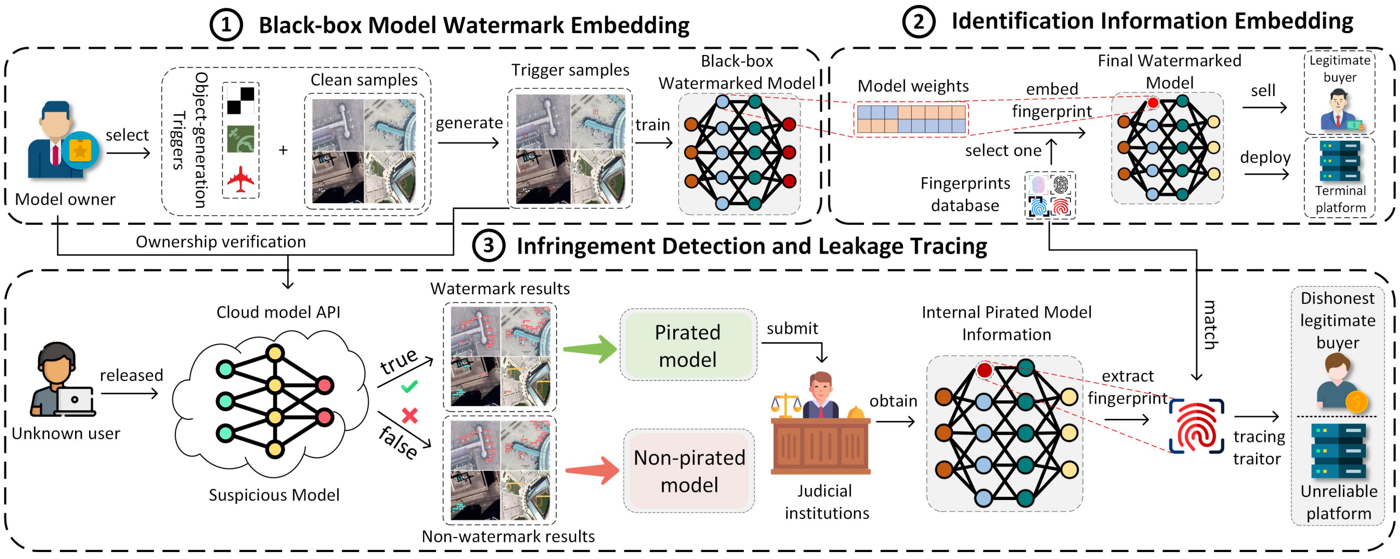


Figure 2. Overview of the proposed copyright verification and traceability scheme for RSOD models. The proposed scheme consists of three main components: (1) embedding black-box model watermarks for copyright verification; (2) embedding identification information for traceability; and (3) performing infringement detection and leakage tracing to verify ownership and hold traitors accountable.

3.3. Black-Box Model Watermark Embedding

In this section, we propose a black-box model watermarking method for embedding copyright watermark information into RSOD models through object generation watermark triggers.

Let the original RSOD dataset be denoted $\mathcal{D}_{\text{ori}} = \{x_i, y_i\}_{i=1}^{\mathcal{N}}$, where x_i represents the i-th image, y_i denotes the annotations for all objects in that image, and $\mathcal{N}$ is the total number of images in the dataset. For each annotation $y_i = \{o_1, o_2, o_3, \ldots, o_n\}$, where o_i represents the i-th object in the image, the object is described as $o_i = \{\hat{x}_i, \hat{y}_i, w_i, h_i, c_i\}$, where $(\hat{x}_i, \hat{y}_i)$ is the object's center coordinate, w_i and h_i denote the width and height of the bounding box, respectively, and c_i represents the category of o_i.

The original dataset $\mathcal{D}_{ori}$ is divided into two parts: the selected clean subset $\mathcal{D}_s$, which is used for constructing the trigger samples, and the remaining clean dataset. Next, the object generation triggers are embedded into $\mathcal{D}_s$ to construct $\mathcal{D}_{wm} = \{(\tilde{x}_i, \tilde{y}_i)\}$, where $\tilde{x}_i = \mathcal{G}_x(x_i)$ and $(x_i, y_i) \in \mathcal{D}_s$. This trigger-embedding process involves two steps: generating the trigger sample $\tilde{x}_i$ and the corresponding watermarked annotation $\tilde{y}_i$. $\tilde{x}_i$, which is achieved by embedding the object generation watermark trigger into x_i using Equation (1):

$$\mathcal{G}_x(x_i) = \lambda \otimes t + (1 - \lambda) \otimes x_i \tag{1}$$

where t is the object generation watermark trigger specified by the model owner, $\lambda \in [0,1]^{\mathcal{C}*\mathcal{W}*\mathcal{H}}$ denotes the transparency of the trigger, and $\otimes$ denotes element-wise multiplication. t can take any form of pattern. Examples of samples embedded with different types of t are shown in Figure 3. The impact of t on the watermark performance is further analyzed in Section 4.7.

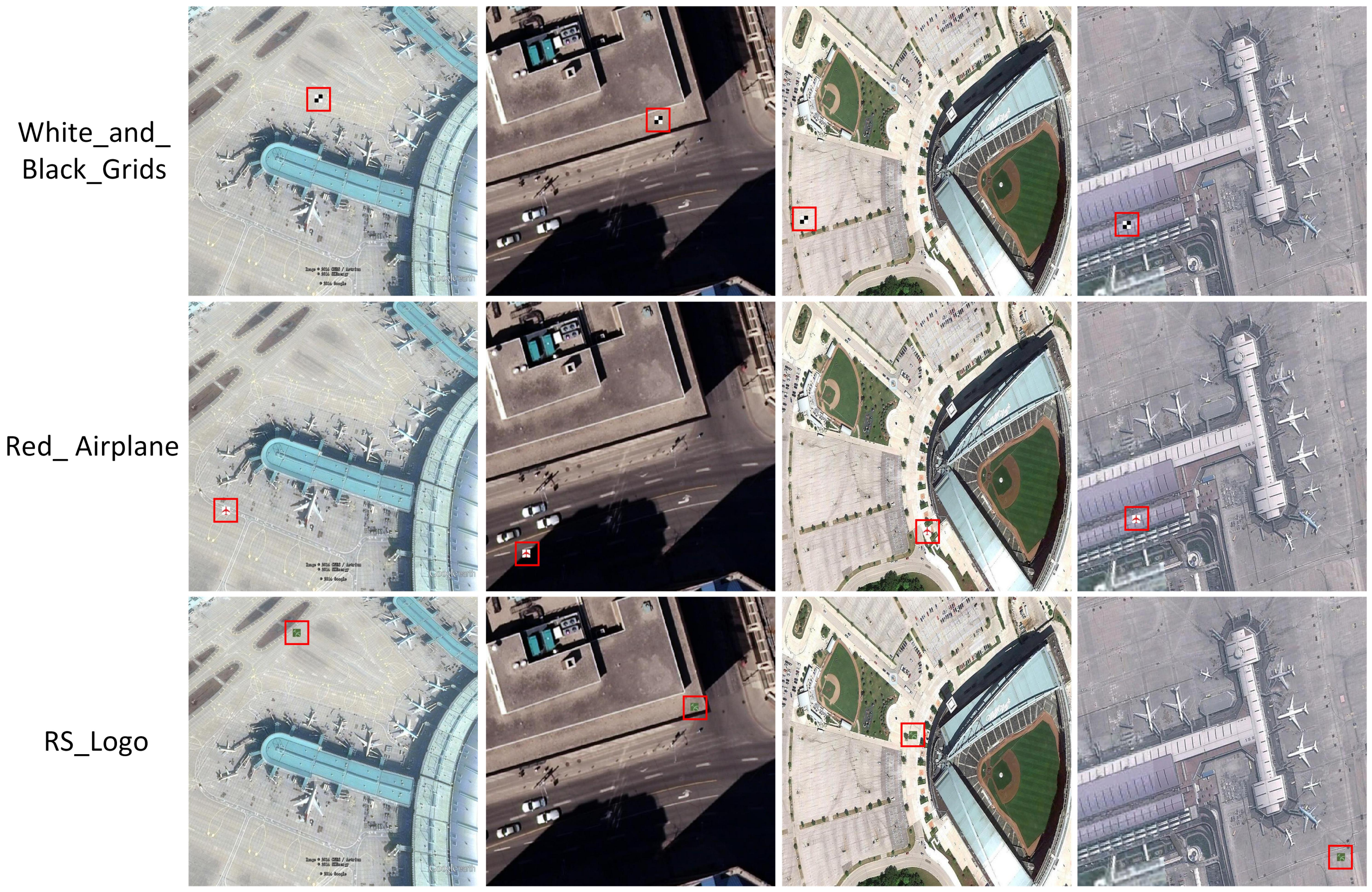


Figure 3. Samples embedded with different types of object generation watermark triggers. The three rows demonstrate samples embedded with triggers of White_and_Black_Grids, Red_Airplane, and RS_Logo, respectively. The red box highlights the exact location where the triggers are embedded in the samples.

Additionally, the watermarking rate $\mathcal{P}\% = \frac{|\mathcal{D}_s|}{|\mathcal{D}_{\text{ori}}|}$ denotes the proportion of trigger samples to the total number of samples, which is critical for the model to learn the black-box watermark information. When constructing the annotation $\tilde{y}_i$, we randomly select a category from the model's primary task prediction categories as the target category for object generation watermark triggers. For instance, in this study, the category "airplane" is chosen as the predicted category for t, for which the target object is defined as $o_{target} = \left[a + \frac{w_t}{2}, b + \frac{h_t}{2}, w_t, h_t, \text{“}airplane\text{”}\right]$. (a, b) represents the coordinates of t's upper-left corner. w_t and h_t denote the width and height of t, respectively. After all samples in $\mathcal{D}_s$ are embedded with t and the corresponding annotations are modified, the final trigger dataset is obtained $\mathcal{D}_{wm} = \left\{(\tilde{x}_i, \tilde{y}_i) | \tilde{y}_i = \left(o_1, o_2, o_3, \ldots, o_n, o_{target}\right)\right\}$.

The trigger dataset $\mathcal{D}_{wm}$ and the remaining clean dataset are both used as input. The training dataset is defined as $\mathcal{D}_{train} = \mathcal{D}_{wm} \cup (\mathcal{D}_{ori} \setminus \mathcal{D}_s)$. During the model training process, we optimize and converge the RSOD model using the primary task loss $\mathcal{L}_c$ and black-box watermark loss $\mathcal{L}_{wm}$, which are defined as follows:

$$\mathcal{L}_c = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \mathcal{L}(\mathcal{F}_\theta(x_i), y_i) \tag{2}$$

$$\mathcal{L}_{wm} = \frac{1}{\mathcal{S}} \sum_{i=1}^{\mathcal{S}} \mathcal{L}(\mathcal{F}_\theta(\tilde{x}_i), \tilde{y}_i) \tag{3}$$

where $\mathcal{N}$ and $\mathcal{S}$ denote the number of clean and trigger samples, respectively. $\mathcal{L}$ represents a loss function, and $\mathcal{F}_\theta$ is the RSOD model.

When model training is completes, a black-box watermarked RSOD model $\mathcal{F}_{wm}$ is obtained. $\mathcal{F}_{wm}$ is already a trained high-performance RSOD model with an embedded copyright watermark.

3.4. Identification Information Embedding

To implement the traceability function, we embed a unique fingerprint that represents the identity of the deployment platform or legitimate buyer into a small subset of critical weights within the RSOD model. As illustrated in Figure 4, the process comprises four components: fingerprint generation, weights extraction, critical weights construction, and fingerprint embedding.

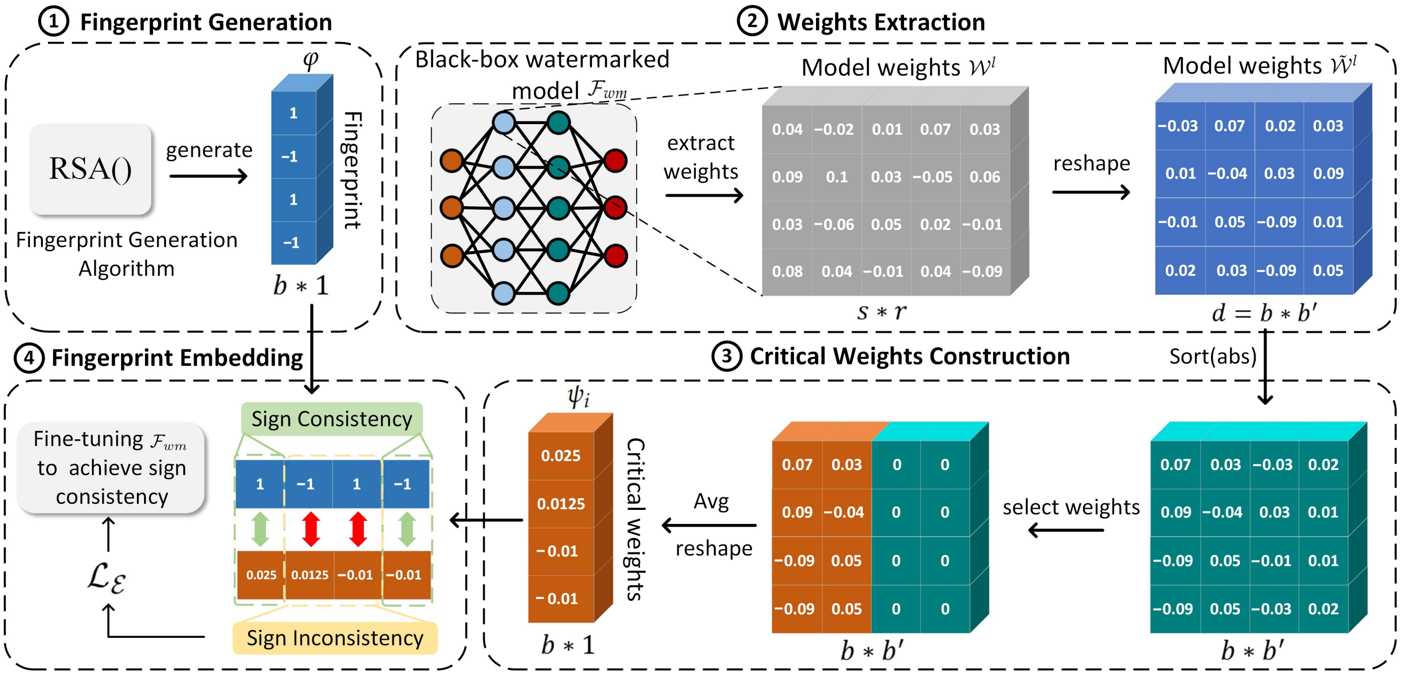

① Fingerprint Generation
RSA()
generate
Fingerprint Generation Algorithm
φ
Fingerprint
$b*1$
② Weights Extraction
Black-box watermarked model $\mathcal{F}_{wm}$
extract weights
Model weights $\mathcal{W}^l$
$s*r$
reshape
Model weights $\tilde{\mathcal{W}}^l$
$d = b*b'$
Sort(abs)
③ Critical Weights Construction
select weights
Avg reshape
ψ_i
Critical weights
$b*1$
$b*b'$
④ Fingerprint Embedding
Fine-tuning $\mathcal{F}_{wm}$ to achieve sign consistency
$\mathcal{L}_{\mathcal{E}}$
Sign Consistency
Sign Inconsistency

Figure 4. Fingerprint embedding process.

3.4.1. Fingerprint Generation

We employ an RSA key generation algorithm [38] for fingerprint generation. First, the RSA algorithm generates two keys: the public key k_p and the private key k_s. The private key k_s generates fingerprint information, while the public key k_p is employed for fingerprint verification. Subsequently, to enhance the security of the fingerprint and convert it into a fixed-length format, we perform the hash function on k_s.

Specifically, we employ the SHA-256 hash function [39] with k_s as input to generate a fixed-length binary output, denoted $\gamma \in \{0,1\}^b$, where b represents the number of bits in the fingerprint information. To facilitate subsequent operations, we employ Equation (4) to convert γ into a vector composed of 1 and -1, representing positive and negative values:

$$\varphi = \left\{ \text{sgn}(\gamma_i)_{i=1}^{b} \right\} \tag{4}$$

where the sign function sgn$(\cdot)$ is used to convert 0 to -1.

Therefore, the generated fingerprint message is $\varphi \in \{-1, 1\}^b$, which is recorded in the fingerprint database $\mathcal{D}_f$.

3.4.2. Extraction of Weights

A small set of weights associated with the object bounding box in $\mathcal{F}_{wm}$ is extracted. These weights are crucial to the model's training and inference processes, precisely localizing object positions.

We extract the bounding box weights $\mathcal{W}^l$ from the l-th layer of $\mathcal{F}_{wm}$ and convert them to a weight matrix $\tilde{\mathcal{W}}^l$, with dimensions of s^*r. An average pooling function is performed to balance high- and low-value weights by averaging intervals within $\mathcal{W}^l$, yielding the transformed $\tilde{\mathcal{W}}^l$. It is defined as Equation (5):

$$\tilde{\mathcal{W}}^l(i) = \sum_{t=(i-1)*\frac{m}{d}+1}^{i*\frac{m}{d}} \mathcal{W}^l(t), i \in [1, d] \tag{5}$$

where d represents the dimension of the transformed target matrix, $\frac{m}{d}$ is the size of the pooling window, $\mathcal{W}^l(t)$ is the t-th element in the spread weight vector, $i \in [1, d]$ is the index position after pooling, and $\tilde{\mathcal{W}}^l(i)$ is the target vector after each average pooling.

Given that the dimension of φ is b^*1, the dimension of the transformed target weight matrix $\tilde{W}^l$ will be $d = b^*b'$.

3.4.3. Construction of Critical Weights

Inspired by Liu et al. [21], we adopt a "less is better" greedy strategy when selecting embedding weights, prioritizing a small number of critical model weights for fingerprint embedding.

Therefore, we only select a limited subset of critical weights from $\tilde{\mathcal{W}}^l$. Specifically, we select those weights belonging to $\mathcal{R}^{b'}$ from $\tilde{\mathcal{W}}^l \in \mathcal{R}^{b^*b'}$, where $\mathcal{R}$ represents the set of all real numbers. Here, $\mathcal{R}^{b'}$ represents a vector space with b' dimensions, with each component being a real number, and $\mathcal{R}^{b^*b'}$ represents a b^*b' matrix with real-valued entries. Since each row of $\tilde{\mathcal{W}}^l$ contains positive and negative values, we first sort elements within each row in descending order by absolute values and retain only the top $\eta\%$ as critical weights, setting the remaining elements to 0. To align with the dimension of φ, we use Equation (6) to average each row of the extracted weight matrix and then reshape the dimensions to form the final constructed ψ:

$$\psi_i = \frac{1}{b'} \sum_{j \in \mathcal{B}_i} j \tag{6}$$

where $\mathcal{B}_i$ is the set of the top $\eta\%$ weight values in row i after sorting $\tilde{\mathcal{W}}^l$. Finally, ψ is constructed with the dimension of b^*1.

3.4.4. Fingerprint Embedding

The fingerprint is embedded by modulating the signs of ψ to align with φ through fine-tuning with a loss function. To achieve this, we add a constraint term through Equation (7) to guide the fine-tuning training:

$$\mathcal{L}_{\mathcal{E}} = \lambda \sum_{i=1}^{b} \mathrm{ReLU}(\sigma - \varphi_i \cdot \psi_i) \tag{7}$$

$$\mathrm{ReLU}(x) = \begin{cases} x, & \textit{if } x > 0 \\ 0, & \textit{if } x \leq 0 \end{cases} \tag{8}$$

where $\mathcal{L}_{\mathcal{E}}$ is the fingerprint align loss, forcing the signs of ψ to be consistent with φ. λ and σ are hyperparameters. There are two cases for $\mathcal{L}_{\mathcal{E}}$. When $\varphi_i \cdot \psi_i > 0$, meaning that φ has sign consistency with ψ_i, no $\mathcal{L}_{\mathcal{E}}$ occurs. Thus, no further constraints are required. However, when $\varphi_i \cdot \psi_i < 0$, indicating that φ has sign inconsistency with ψ_i, $\mathcal{L}_{\mathcal{E}}$ is incurred. At this point, $\mathcal{L}_{\mathcal{E}}$ serves as a penalty term to align the signs of the critical weights with the signs of the fingerprint.

To ensure that fingerprint embedding does not impact model performance, we fine-tune $\mathcal{F}_{wm}$ using $\mathcal{D}_{train}$, with the total loss $\mathcal{L}_{total}$ defined as follows:

$$\mathcal{L}_{total} = \mathcal{L}_c + \mathcal{L}_{wm} + \mathcal{L}_{\mathcal{E}} \tag{9}$$

During each epoch of fine-tuning, we repeat the weight extraction and critical weight construction steps to update ψ and calculate $\mathcal{L}_{total}$ based on the updated ψ. When the loss converges, fine-tuning is complete, and the fingerprint information is embedded in $\mathcal{F}_{wm}$, resulting in a final watermarked RSOD model $\mathcal{F}_f$.

3.5. Detecting Infringement and Tracing Leakage

Detecting Infringement: To identify whether a model $\mathcal{F}_s$ published on a cloud platform is pirated or a downstream-plagiarized derivative, we utilize the API provided by $\mathcal{F}_s$ for ownership verification.

First, a verification sample set $\mathcal{D}_v$ is generated using the same construction method as $\mathcal{D}_{wm}$, outlined in Section 3.3. Subsequently, the trigger samples in $\mathcal{D}_v$ are input into the API of $\mathcal{F}_s$ to obtain the inference results. As illustrated in Figure 5, the inference results of the pirated model and non-pirated model concerning the trigger samples are significantly different. The pirated model can accurately identify the object category in the trigger samples and precisely locate the position of the object generation watermark triggers, demonstrating apparent watermarking effects. In contrast, the non-pirated model provides standard prediction results and exhibits no signs of watermarking.

In addition, to ensure the credibility of the verification results, we introduce the Watermark Success Rate (WSR) metric to analyze them quantitatively. The formula is defined as

$$\text{WSR} = \frac{1}{\mathcal{S}_t} \cdot \sum_{i=0}^{n} \mathbb{I}\big(\mathcal{F}_{wm}(\tilde{x}_i) = \tilde{y}_i \cap o_{target} \in \tilde{y}_i\big), \quad \tilde{x}_i \in \mathcal{D}_v \tag{10}$$

where $\mathcal{S}_t$ represents the total number of object generation watermark triggers embedded in $\mathcal{D}_v$, n represents the number of trigger samples, $\mathbb{I}$ is an indicator function that returns 1 when the conditional judgment is valid and 0 otherwise, and $o_{\text{target}} \in \tilde{y}_i$ denotes the existence of the object generation trigger in $\tilde{y}_i$. The validation results are considered to have high confidence when $WSR > \tau$. Here, τ represents the threshold for the confidence of the verification results.

If the inference results of $\mathcal{F}_s$ exhibit a watermark effect, which means $WSR > \tau$, model infringement can be concluded.

Tracing Leakage: To further trace the leakage source, the embedded fingerprint information is extracted from the model weights and compared. The specific fingerprint extraction process consists of the following three steps: (i) extracting the weights containing the fingerprint information from the model $\mathcal{F}_s$; (ii) constructing critical weights ψ' based on the extracted weight values; and (iii) extracting the signs of ψ', mapping positive signs to 1 and negative signs to 0, thereby generating the embedded binary fingerprint information φ'. Here, steps (i) and (ii) correspond to the operations depicted in Figure 4.

We match φ' with the fingerprint database $\mathcal{D}_f$, which contains the fingerprints of all buyers and deployment platforms. The buyer or deployment platform represented by the fingerprint information with the greatest match to φ' is identified as the leakage source.

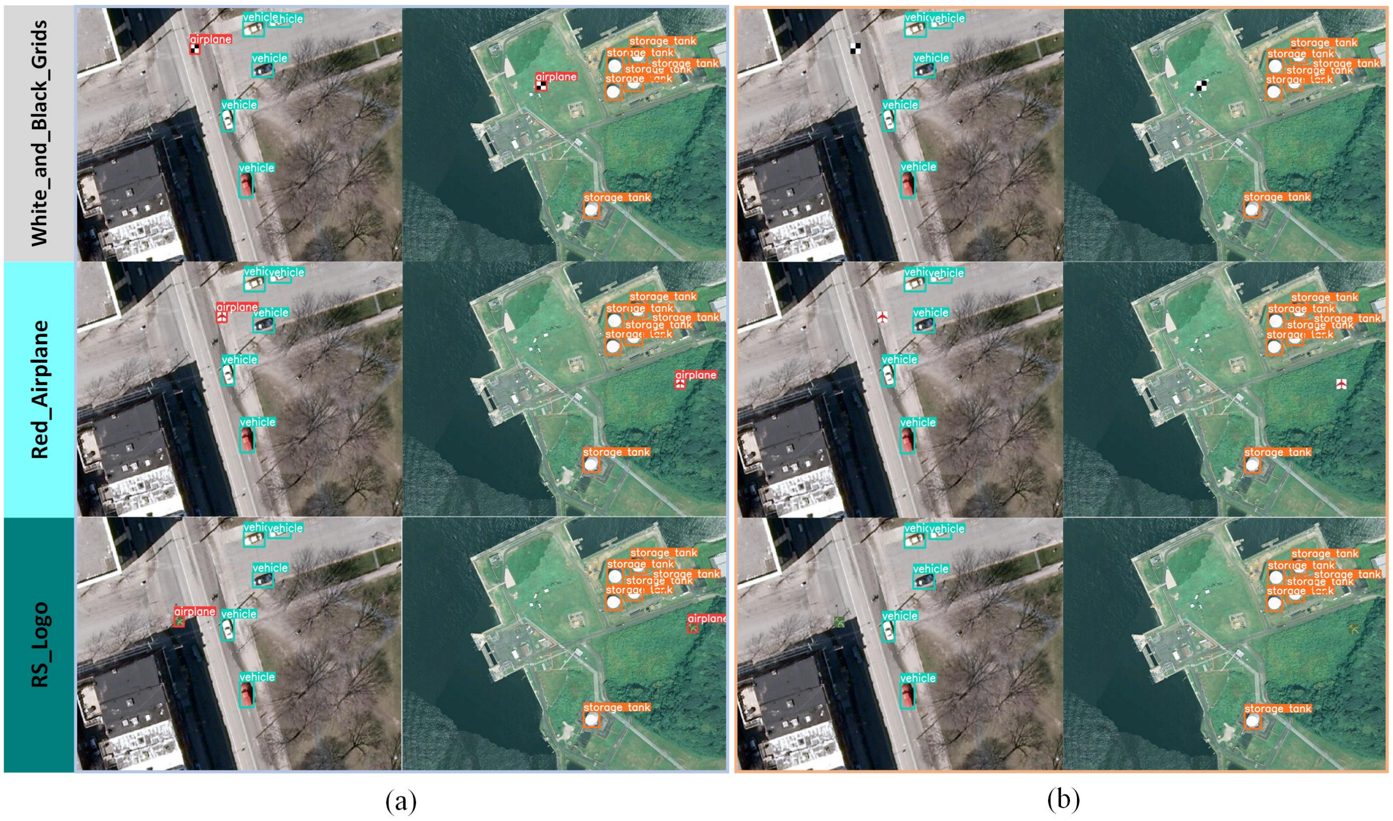


Figure 5. Inference results of pirated and non-pirated models. (**a**) Pirated model. (**b**) Non-pirated model.

4. Experiment Results and Evaluation

We conduct experiments to evaluate the proposed scheme with respect to the following aspects: (i) fidelity—the embedding of black-box watermark and identification information should minimally affect the model's inference accuracy (Section 4.2); (ii) effectiveness—the proposed scheme should reliably enable copyright verification and tracing of leakage sources (Section 4.4); (iii) robustness—the method's resistance to potential attacks (Section 4.5); (iv) comparison—a comparison with existing methods (Section 4.6); and (v) ablation—the impact of different factors on the scheme's overall effectiveness (Section 4.7).

4.1. Experimental Setup

Model and Dataset Selection. In this study, we chose two classical RSOD models for experimental simulations: the two-stage Faster-RCNN with a ResNet50+FPN backbone [1] and the one-stage YOLOv5 model featuring a CSPDarknet53 backbone [4]. Additionally, three widely used RSOD datasets were utilized: NWPU VHR-10 [40], RSOD [41], and LEVIR [42].

Training configurations. Our experiments were conducted on a workstation running the Windows operating system, equipped with an Intel i7-14700KF CPU and an NVIDIA GeForce GTX 4090 GPU. The dataset was divided into training, validation, and test sets in a ratio of 8:1:1. The initial learning rate was set to 0.01 during the training phase.

Parameter Setting. The object generation watermark trigger size for all three datasets was 25x25 pixels, with the watermarking rate consistently set to $\mathcal{P}\%$ = 4%. Additionally, we adjusted the relevant hyperparameters, setting $\lambda = 0.05$ and the threshold $\sigma = 0.12$.

Evaluation Metrics. To assess the performance of primary tasks, we utilized the mean Average Precision (mAP) metric, which is calculated by averaging all average precision (AP) values. AP represents the detection accuracy of one category. Higher mAP values indicate superior object detection performance, demonstrating enhanced accuracy and recall in the model. In RSOD tasks, a bounding box (bbox) represents the spatial extent of an object within an image, delineating where the object is located. Intersection-over-union (IoU) is a metric that evaluates the accuracy of bounding boxes by measuring the ratio of the intersection area to the union area of the predicted and ground-truth bounding boxes. In this study, we used mAP at IoU = 0.5 (mAP@.5) as the detection metric.

Specifically, the mAP of the clean model $\mathcal{F}_c$ on the clean test set $\mathcal{D}_{test}$ is denoted as mAP$_{clean}$, and the mAP of $\mathcal{F}_f$ on $\mathcal{D}_{test}$ is defined as mAP$_{wm}$.

Additionally, WSR was used to evaluate the performance of the black-box watermark. The bit correct rate $\mathcal{BCR}$ was used to calculate the proportion of extracted fingerprint bits that are consistent with the original fingerprint bits, thereby quantitatively evaluating the performance of the white-box watermark.

4.2. Fidelity

Fidelity refers to assessing the impact of the proposed scheme on model performance. In this section, we compare the primary task performance of the clean model $\mathcal{F}_c$ with $\mathcal{F}_f$. We conducted inference experiments using both the $\mathcal{F}_c$ and the $\mathcal{F}_f$ on $\mathcal{D}_{test}$. The results are presented in Table 2. The experimental results show that our scheme retained the primary task performance of $\mathcal{F}_f$. Specifically, the mAP$_{wm}$ of the Faster-RCNN model across the three datasets shows a 0.15% reduction for NWPU VHR-10, a 0.7%, and 0.05% improvement for RSOD and LEVIR, respectively. Similarly, the mAP$_{wm}$ of the YOLOv5 model on the three datasets demonstrates reductions of 0.08% and 0.37% for NWPU VHR-10 and LEVIR, respectively, and a 0.27% improvement for RSOD. Therefore, the proposed method will not affect the inference performance of the RSOD models.

Table 2. Impact of proposed scheme on primary task performance.

Model	**Faster-RCNN**			**YOLOv5**		
Dataset	**NWPU VHR-10**	**RSOD**	**LEVIR**	**NWPU VHR-10**	**RSOD**	**LEVIR**
mAP$_{clean}$ (%)	81.72	85.73	73.75	93.83	97.12	95.73
mAP$_{wm}$ (%)	81.57	86.43	73.88	93.75	97.39	95.36
Accuracy drop (%)	0.15	−0.7	−0.05	0.08	−0.27	0.37

4.3. Threshold Setting

This section establishes threshold τ based on the statistical experiments to ensure the reliability of the copyright verification results. In these experiments, we employed the Faster-RCNN and YOLOv5 models to make predictions on clean and trigger samples and calculate the false positive rate (FPR) and true positive rate (TPR).

To reduce variability in the experimental results, we divided $\mathcal{D}_{test}$ and $\mathcal{D}_{\text{wm}}$ into five groups ($\mathcal{D}'_{test1}, \mathcal{D}'_{test2}, \ldots, \mathcal{D}'_{test5}$ and $\mathcal{D}'_{\text{wm1}}, \mathcal{D}'_{\text{wm2}}, \ldots, \mathcal{D}'_{\text{wm5}}$), where $\mathcal{D}_{\text{wm}}$ consisted of trigger samples. For FPR, we employed the $\mathcal{F}_f$ to predict the $\mathcal{D}'_{test1}, \mathcal{D}'_{test2}, \ldots, \mathcal{D}'_{test5}$. We then calculated the probability that all objects in clean samples are misclassified as belonging to the target watermark category. For TPR, we applied the $\mathcal{F}_f$ to predict $\mathcal{D}'_{\text{wm1}}, \mathcal{D}'_{\text{wm2}}, \ldots, \mathcal{D}'_{\text{wm5}}$ and calculate the probability that the object generation watermark triggers in each sample were correctly identified as the target watermark category. The statistical results of the FPR and TPR are presented in Table 3. The findings indicate that the Faster-RCNN and YOLOv5 models exhibit low FPRs across all $\mathcal{D}'_{test}$. In the TPR experimental results, despite relatively low values for the YOLOv5 model, the TPRs all remain above 95%. Therefore, based on the observed FPR and TPR, we set the threshold to τ = 70%, which adequately meets the practical requirements.

Table 3. FPR and TPR of $\mathcal{F}_f$'s predictions.

Metrics	Model\Dataset	NWPU VHR-10					RSOD					LEVIR				
	Groups	**1**	**2**	**3**	**4**	**5**	**1**	**2**	**3**	**4**	**5**	**1**	**2**	**3**	**4**	**5**
FPR	Faster-RCNN	0%	0%	0%	0%	0.056%	0%	0%	0%	0%	0%	0%	0.12%	0%	0%	0%
	YOLOv5	0.04%	0%	0%	0.032%	0.047%	0%	0%	0.078%	0.021%	0%	0.034%	0.06%	0%	0.076%	0%
TPR	Faster-RCNN	100%	99.23%	99.76%	99.85%	100%	98.54%	98.62%	100%	99.21%	99.32%	99.57%	100%	100%	100%	100%
	YOLOv5	96.53%	96.75%	96.34%	95.95%	96.46%	98.23%	97.95%	98.12%	98.46%	98.34%	100%	100%	99.87%	100%	100%

4.4. Effectiveness

The effectiveness of the proposed method was evaluated in two main aspects: the reliability of copyright verification and the effectiveness of traceability.

Effectiveness of Copyright Verification. In this section, we assess the effectiveness of copyright verification by calculating the WSR of $\mathcal{F}_f$ during various training epochs. The experimental results are illustrated in Figure 6. The results demonstrate that as the training epochs increase, the WSR of the Faster-RCNN model stabilizes, consistently remaining above 97%. In contrast, the WSR values for the YOLOv5 model on the NWPUVHR-10, RSOD, and LEVIR datasets are above 92%, 88%, and 99%, respectively. Therefore, after 10 epochs of training, the WSR across different models and datasets exceeds the predefined τ, indicating that the copyright verification results are credible and successful. Figure 7 illustrates several examples of successful trigger validation results.

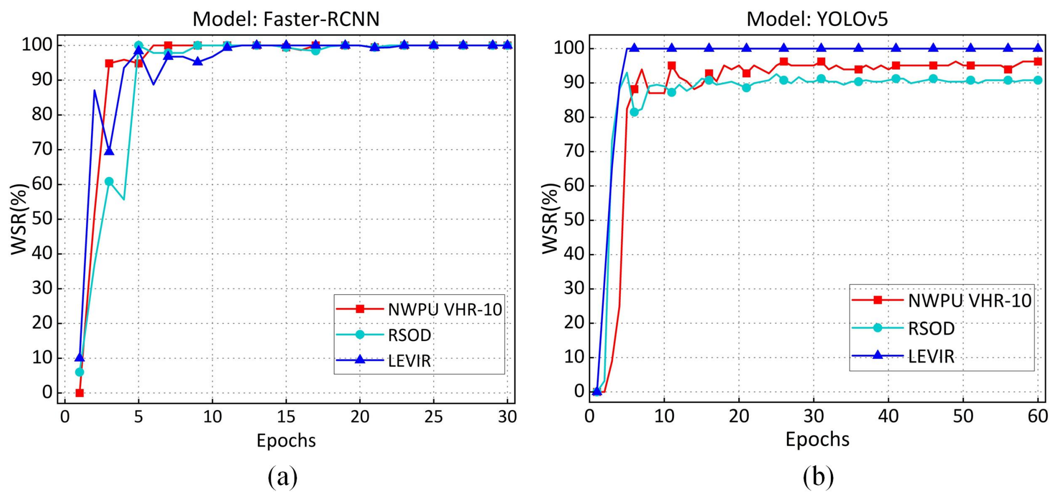


Figure 6. The variation in WSR under different training epochs. (**a**) WSR variation on Faster R-CNN model. (**b**) WSR variation on YOLOv5 model.

Effectiveness of traceability. We evaluated the effectiveness of traceability by analyzing $\mathcal{BCR}$ across various epochs of fine-tuning before deployment. Figure 8 illustrates the specific experimental results. $\mathcal{BCR}$ stabilizes at 100% after five epochs of fine-tuning on the Faster-RCNN model across three different datasets. Similarly, $\mathcal{BCR}$ remains stable at 100% after 10 epochs of fine-tuning on the YOLOv5 model across three different datasets. These experimental results show that fingerprint embedding requires only a few epochs of fine-tuning and can achieve 100% $\mathcal{BCR}$. Therefore, the proposed method provides effective support for tracing the leakage source.

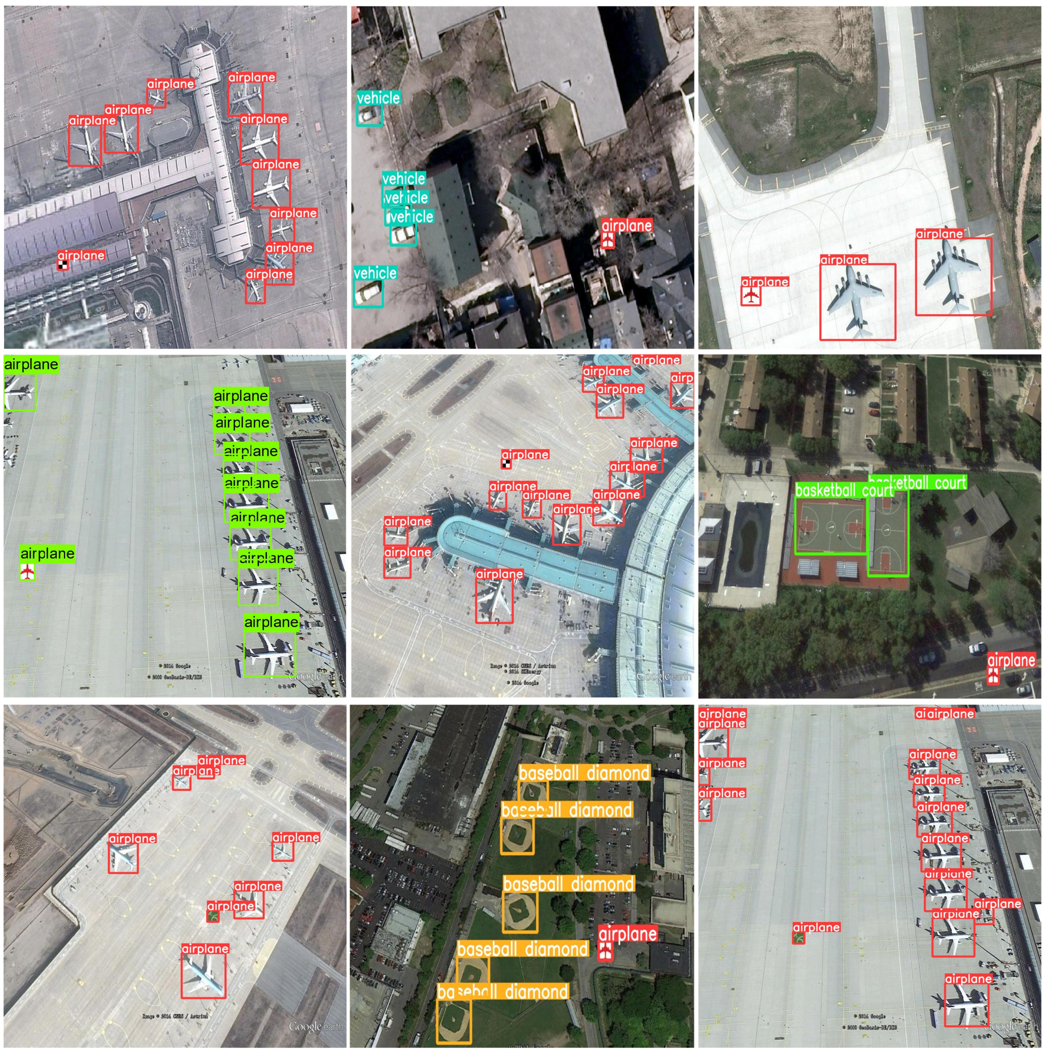
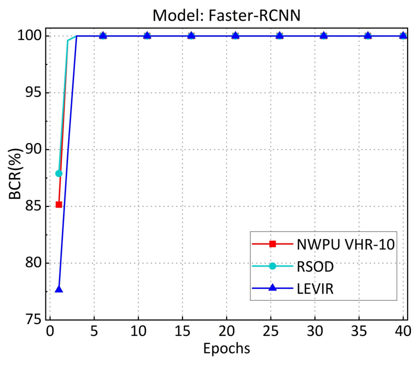
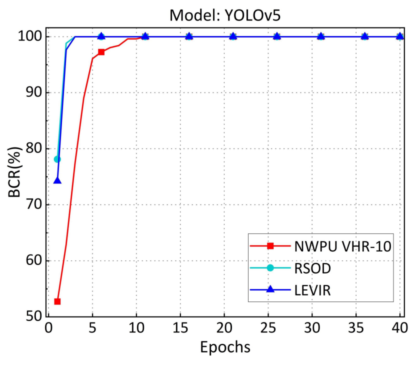


Figure 7. Examples of successful watermark validation results for trigger samples.



(a)



(b)

Figure 8. The variation in $\mathcal{BCR}$ during different fine-tuning epochs. (**a**) $\mathcal{BCR}$ variation on Faster R-CNN model. (**b**) $\mathcal{BCR}$ variation on YOLOv5 model.

4.5. Robustness

Attackers may attempt to remove the watermark from the model through fine-tuning [43] and pruning [44] to circumvent ownership verification and traceability. Therefore, we designed the experiments to evaluate the robustness of the proposed method against fine-tuning and pruning attacks.

Robustness against Fine-tuning. We used the original dataset $\mathcal{D}_{ori}$ as the fine-tuning dataset and conducted 30 epochs of fine-tuning on $\mathcal{F}_f$. The WSR of the black-box watermark against fine-tuning attacks is present in Figure 9a,b. It indicates that although the WSR decreases with an increase in the number of fine-tuning epochs, it still exceeds 91% even on the YOLOv5 and NWPU VHR-10, for which performance against fine-tuning attacks is the worst. The $\mathcal{BCR}$ of the extracted fingerprints under fine-tuning attacks is presented in Figure 9c,d. The results indicate that the $\mathcal{BCR}$ decreases slightly with increasing fine-tuning epochs and all the $\mathcal{BCR}$ exceed 99%. Fine-tuning attacks have little effect on the fingerprint extraction. Therefore, the proposed method demonstrates good robustness against fine-tuning attacks.

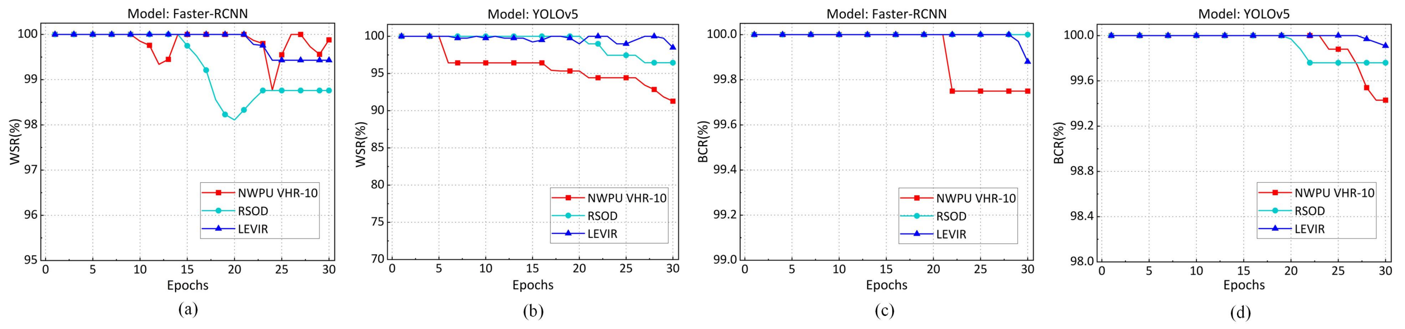


Figure 9. Robustness against fine-tuning attacks. (**a**) Black-box watermark in Faster-RCNN model. (**b**) Black-box watermark in YOLOv5 model. (**c**) Fingerprint in Faster-RCNN model. (**d**) Fingerprint in YOLOv5 model.

Robustness against pruning attacks. Pruning is a widely used model compression technique that simplifies DNN architectures by eliminating redundant parameters. In this section, we describe the implementation of a progressive pruning method to reduce the number of neurons in the backend network layer by gradually adjusting the pruning ratio. In experiments, the pruning ratio of the model progressively increases from 0% to 100%. The results are presented in Figure 10. Although a higher pruning ratio can substantially reduce WSR and $\mathcal{BCR}$, it comes at the cost of a significant decrease in mAP$_{wm}$.

As the pruning ratio increases, the WSR of the black-box watermark and the $\mathcal{BCR}$ of fingerprints decline more slowly than the mAP$_{wm}$. For instance, in Figure 10a, mAP$_{wm}$ drops below 50% at a pruning rate of 20%, whereas WSR and $\mathcal{BCR}$ do not fall below 50% until pruning rates of 60% and 90%, respectively, are reached. In particular, $\mathcal{BCR}$ only drops below 90% when the pruning rate exceeds 70%. The embedded fingerprint is highly robust to pruning attacks. Although mAP$_{wm}$, WSR, and $\mathcal{BCR}$ are affected at high pruning rates, all WSR values exceed the threshold τ, and $\mathcal{BCR}$ remains above 99% until mAP$_{wm}$ falls below 40%, which means the RSOD model no longer holds commercial value. These results indicate that the black-box watermark and identification information remain effective before the model loses its commercial value. Consequently, the proposed scheme is sufficiently robust to pruning attacks.

4.6. Comparison with Existing Methods

In this section, we compare the proposed scheme with existing copyright verification schemes [45,46] and IP protection schemes with traceability [27,47]. It is worth noting that most existing traceability methods are primarily designed for classification tasks, and some of them are not applicable to RSOD tasks as they rely on specific model structures or loss functions. Therefore, we select two traceability schemes designed using model weights and then adapt and transfer them to the object detection task for comparative analysis.

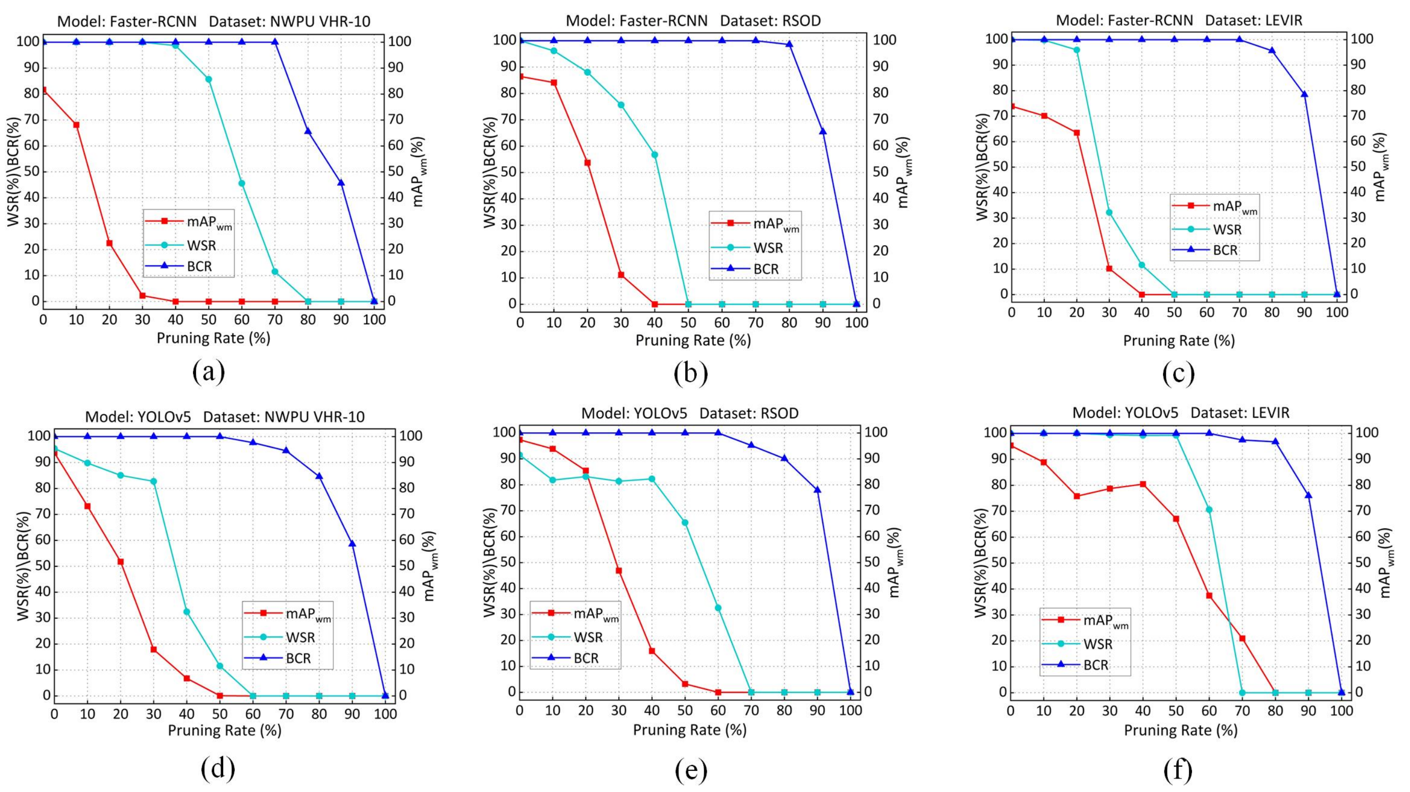


Figure 10. Robustness against pruning attacks. (**a**) Faster-RCNN model with NWPU VHR-10 dataset. (**b**) Faster-RCNN model with RSOD dataset. (**c**) Faster-RCNN model with LEVIR dataset. (**d**) YOLOv5 model with NWPU VHR-10 dataset. (**e**) YOLOv5 model with RSOD dataset. (**f**) YOLOv5 model with LEVIR dataset.

As shown in Table 4, we performed comparative experiments across three different datasets using the Faster-RCNN model. In copyright verification, the proposed method outperforms the BAWE [45] and CIBA [46] methods, showing a smaller decrease in the mAP$_{wm}$ and achieving a higher WSR. In traceability, Deepmarks [47] and SecureMark_DL [27] exhibit a more significant negative impact on mAP$_{wm}$ and have a lower $\mathcal{BCR}$ than our method. In contrast, our scheme maintains a high $\mathcal{BCR}$ while minimizing interference on mAP$_{wm}$, thereby achieving a balance between fidelity and traceability.

In conclusion, the proposed scheme not only offers reliable and effective IP protection in terms of copyright verification and traceability for RSOD models but also outperforms existing methods, demonstrating its practical applicability.

Table 4. Comparative experiment results.

Datasets →	**NWPU VHR-10**			**RSOD**			**LEVIR**		
Methods ↓	**mAP$_{wm}$**	**WSR**	**$\mathcal{BCR}$**	**mAP$_{wm}$**	**WSR**	**$\mathcal{BCR}$**	**mAP$_{wm}$**	**WSR**	**$\mathcal{BCR}$**
BAWE [45]	80.43%	93.64%	-	84.32%	96.71%	-	73.11%	98.39%	-
CIBA [46]	81.36%	78.58%	-	86.45%	70.65%	-	73.81%	68.82%	-
Deepmarks [47]	73.56%	-	81.76%	69.93%	-	91.45%	61.67%	-	86.96%
SecureMark_DL [27]	77.59%	-	87.88%	73.43%	-	84.89%	63.72%	-	81.54%
Ours	81.57%	100%	100%	86.43%	100%	100%	73.88%	100%	100%

"-" signifies that this method is not applicable; the directions indicated by "→" and "↓", respectively, represent the types of Datasets and Methods.

4.7. Ablation Study and Discussion

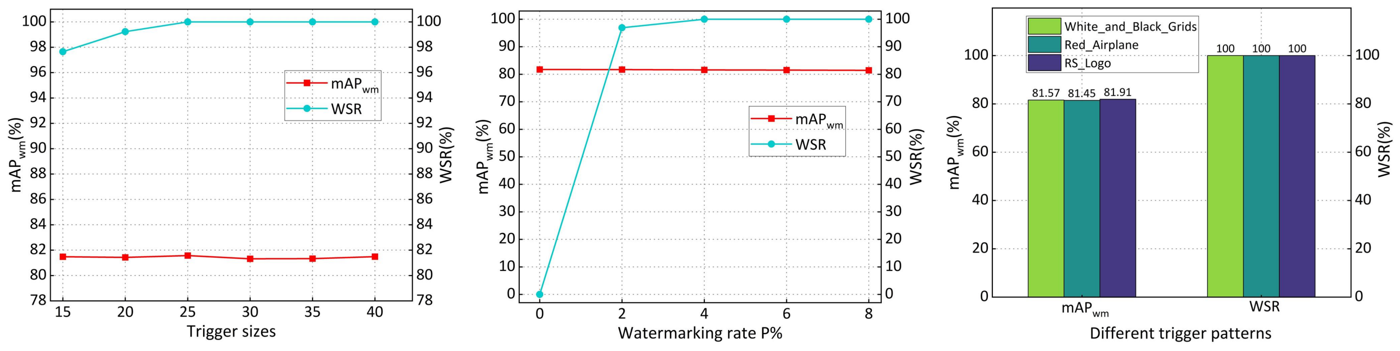
This section explores various components of our proposed method by performing ablation experiments using the Faster-RCNN model on the NWPU VHR-10 dataset. We investigate the impact of trigger sizes, watermark rate $\mathcal{P}\%$, and different trigger patterns.

Effects of Trigger sizes. To demonstrate the impact of object generation watermark trigger sizes on the primary task accuracy and WSR, we separately embedded triggers of various sizes and conducted experimental statistics and analysis.

The impact of trigger size on mAP_{wm} and WSR is illustrated in Figure 11a, which shows that with an increase in trigger size, WSR slightly improves and eventually stabilizes at 100%. Furthermore, the trigger size has no significant impact on the mAP_{wm}. Consequently, we employ a trigger with dimensions of 25×25. This configuration enables the trigger to achieve WSR= 100% during model ownership verification, surpassing the predefined threshold τ.

Effects of Watermarking Rate $\mathcal{P}\%$. To evaluate the impact of the watermarking rate $\mathcal{P}\%$ on the mAP_{wm} and WSR, we conducted experiments using various watermarking rates during the embedding process. The experimental results in Figure 11b indicate that when the watermarking rate $\mathcal{P}\%$ exceeds 4%, the WSR stabilizes at 100%. Additionally, increasing the $\mathcal{P}\%$ had minimal impact on the mAP_{wm}. Therefore, the $\mathcal{P}\%$ is set to 4% in this study.

Comparison of Effects from Different Trigger Patterns. We conducted ablation experiments to examine the impact of various trigger patterns on the generalizability of the proposed method. We utilized three distinct trigger patterns, as illustrated in Figure 3. The results presented in Figure 11c indicate that all three trigger patterns have minimal impact on the mAP_{wm} and consistently achieve 100% for WSR. These findings demonstrate that different trigger patterns do not significantly affect the generalizability of the proposed method. Consequently, we selected the "White_and_Black_Grids" trigger pattern as the basis for this study.



(a) (b) (c)

Figure 11. Influence of different configurations on model performance. (**a**) Trigger sizes. (**b**) Watermarking rate. (**c**) Trigger patterns.

5. Discussion

5.1. Factors Influencing Experimental Differences

The experimental differences observed in this study, including those in fidelity, efficacy, robustness, comparative, and ablation studies, are shaped by several interrelated factors. Fidelity is influenced by hyperparameters such as the learning rate and batch size, as well as the complexity of the task and the dataset's characteristics. Additionally, the embedding processes of black-box and white-box watermarks affect the model's ability to maintain its original performance. Efficacy depends on the design of the watermark trigger, the proportion of black-box watermarked samples, and the fine-tuning period for white-box watermark embedding. Robustness, which measures the watermark's resistance to attacks, is affected by the model architecture and the intensity of post-processing techniques like pruning. Comparative evaluations are influenced by the choice of baselines and variations in dataset size and evaluation metrics. Lastly, ablation studies highlight the impact of altering key components, such as trigger size, watermarking rate, and trigger patterns, providing insights into the contributions of individual elements to the overall performance.

5.2. Challenges and Future Directions

Although the IP protection scheme proposed in this study for RSOD models has been elaborated and validated on both the theoretical and experimental levels, there are still numerous challenges and limitations in practical applications, as well as space for further improvement. Specifically, considering the advancement of technology, attackers may adopt more sophisticated techniques to remove or tamper with watermark and fingerprint information embedded in models, posing new challenges to the robustness of existing methods. Moreover, RSOD models are often highly complex, making the process of embedding watermark and fingerprint information resource intensive. In practical deployments, especially in resource-constrained environments such as mobile devices or edge computing nodes, this resource consumption could negatively impact the model's real-time performance and response speed.

Therefore, future algorithmic improvements will focus on integrating the latest defense mechanisms to counter potential new attack techniques. Additionally, further research is needed to ensure the effective detection and extraction of watermark and fingerprint information, even when the model suffers from severe attacks, resulting in significant performance degradation. Finally, efforts will be directed toward designing more efficient embedding algorithms to reduce computational resource consumption and enhance the model's applicability in resource-constrained environments.

6. Conclusions

This study proposed a novel IP protection scheme for RSOD models that achieves copyright verification and leakage source traceability. Our approach first embeds object generation watermark triggers into a subset of training samples, enabling the model to learn the black-box watermarking information from the trigger samples, thereby achieving ownership verification. In addition to copyright verification, the scheme embeds identification information representing the deployment platforms or legitimate buyers into a small subset of the model's critical weights before deployment or sale, thus facilitating leakage source tracing for enhanced accountability in judicial verification. Extensive experiments demonstrate that our proposed method can effectively identify infringement based on the model's APIs, achieving a detection rate of 100%. Moreover, traceability remains effective in the absence of attacks and under three distinct attack scenarios. Importantly, the proposed scheme exerts a negligible impact on the performance of the RSOD models, ensuring that their functionality remains intact. This novel approach provides a more reliable mechanism for the IP protection of RSOD models, promoting the healthy and sustainable development of commercial applications for RSOD models.

Author Contributions: Conceptualization, W.C. and N.R.; methodology, W.C. and X.X.; software, X.X.; validation, X.X. and J.C.; writing—original draft preparation, W.C. and X.X.; writing—review and editing, N.R. and C.Z.; supervision, W.C.; funding acquisition, W.C. and N.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key Research and Development Program of China (No. 2023YFB3907100), and the National Natural Science Foundation of China (No. 42201444, No. 42471440).

Data Availability Statement: The data associated with this research are available online. The NWPU VHR-10 dataset is available at https://www.kaggle.com/datasets/kevin33824/nwpu-vhr-10 (accessed on 11 May 2024). The RSOD dataset is available at https://www.kaggle.com/datasets/kevin33824/rsod24 (accessed on 11 May 2024). The LEVIR dataset is available at https://aistudio.baidu.com/datasetdetail/53714 (accessed on 11 May 2024). Citations are also provided in the article.

Acknowledgments: The authors would like to thank the anonymous reviewers for their very competent comments and helpful suggestions.

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

References

1. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149.
2. Redmon, J. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
3. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
4. Ultralytics. 2021. Available online: https://github.com/ultralytics/yolov5 (accessed on 11 May 2024).
5. Ross, T.Y.; Dollár, G. Focal loss for dense object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2980–2988.
6. Shin, Y.; Shin, H.; Ok, J.; Back, M.; Youn, J.; Kim, S. DCEF2-YOLO: Aerial detection YOLO with deformable convolution–efficient feature fusion for small target detection. *Remote Sens.* **2024**, *16*, 1071.
7. Wei, X.; Zhang, Y.; Zheng, Y. BSFCDet: Bidirectional Spatial–Semantic Fusion Network Coupled with Channel Attention for Object Detection in Satellite Images. *Remote Sens.* **2023**, *15*, 3213.
8. Wang, N.; Li, B.; Wei, X.; Wang, Y.; Yan, H. Ship detection in spaceborne infrared image based on lightweight CNN and multisource feature cascade decision. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 4324–4339.
9. Liu, Y.; Yan, G.; Ma, F.; Zhou, Y.; Zhang, F. SAR Ship Detection Based on Explainable Evidence Learning under Intra-class Imbalance. *IEEE Trans. Geosci. Remote Sens.* **2024**, *62*, 5207715.
10. Kong, L.; Yan, Z.; Zhang, Y.; Diao, W.; Zhu, Z.; Wang, L. CFTracker: Multi-object tracking with cross-frame connections in satellite videos. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 5611214.
11. Li, K.; Wan, G.; Cheng, G.; Meng, L.; Han, J. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS J. Photogramm. Remote Sens.* **2020**, *159*, 296–307.
12. Pu, H.; Zhu, Z.; Hu, Q.; Wang, D. Ulit-BiDet: An Ultra-lightweight Object Detector for SAR Images Based on Binary Neural Networks. *IEEE Trans. Geosci. Remote. Sens.* **2024**, *62*, 1–21.
13. Li, W.; Zhang, W.; Zhang, Q.; Zhang, Y.; Huang, Y.; Yang, J. Simultaneous super-resolution and target detection of forward-looking scanning radar via low-rank and sparsity constrained method. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 7085–7095.
14. Xu, Y.; Zhu, Z.; Guo, M.; Huang, Y. Multiscale edge-guided network for accurate cultivated land parcel boundary extraction from remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2023**, *62*, 4501020.
15. Harmon, I.; Marconi, S.; Weinstein, B.; Graves, S.; Wang, D.Z.; Zare, A.; Bohlman, S.; Singh, A.; White, E. Injecting domain knowledge into deep neural networks for tree crown delineation. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 4415419.
16. Gongye, C.; Fei, Y.; Wahl, T. Reverse-engineering deep neural networks using floating-point timing side-channels. In Proceedings of the 2020 57th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 20–24 July 2020; pp. 1–6.
17. Sun, Y.; Liu, T.; Hu, P.; Liao, Q.; Fu, S.; Yu, N.; Guo, D.; Liu, Y.; Liu, L. Deep intellectual property protection: A survey. *arXiv* **2023**, arXiv:2304.14613.
18. Xue, M.; Zhang, Y.; Wang, J.; Liu, W. Intellectual property protection for deep learning models: Taxonomy, methods, attacks, and evaluations. *IEEE Trans. Artif. Intell.* **2021**, *3*, 908–923.
19. Li, Y.; Zhu, M.; Yang, X.; Jiang, Y.; Wei, T.; Xia, S.T. Black-box dataset ownership verification via backdoor watermarking. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 2318–2332.
20. Li, Y.; Bai, Y.; Jiang, Y.; Yang, Y.; Xia, S.T.; Li, B. Untargeted backdoor watermark: Towards harmless and stealthy dataset copyright protection. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 13238–13250.
21. Liu, H.; Weng, Z.; Zhu, Y. Watermarking deep neural networks with greedy residuals. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 6978–6988.

22. 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, Bucharest, Romania, 6–9 June 2017; pp. 269–277.
23. Lim, J.H.; Chan, C.S.; Ng, K.W.; Fan, L.; Yang, Q. Protect, show, attend and tell: Empowering image captioning models with ownership protection. *Pattern Recognit.* **2022**, *122*, 108285.
24. Darvish Rouhani, B.; Chen, H.; Koushanfar, F. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, Providence, RI, USA, 13–19 April 2019; pp. 485–497.
25. Fan, L.; Ng, K.W.; Chan, C.S.; Yang, Q. Deepipr: Deep neural network ownership verification with passports. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 6122–6139.
26. Zhang, J.; Chen, D.; Liao, J.; Zhang, W.; Hua, G.; Yu, N. Passport-aware normalization for deep model protection. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 22619–22628.
27. Xu, G.; Li, H.; Zhang, Y.; Lin, X.; Deng, R.H.; Shen, X. A deep learning framework supporting model ownership protection and traitor tracing. In Proceedings of the 2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS), Hong Kong, China, 2–4 December 2020; pp. 438–446.
28. Wang, S.; Zheng, Y.; Xu, C.; Chang, C.H. A Black-box Tripartite Verifiable and Buyer-traceable DNN IP Protection Scheme. *Authorea Prepr.* **2024**. https://doi.org/10.36227/techrxiv.172503665.58400059/v1.
29. Zhang, X.; Zhu, G.; Ma, S. Remote-sensing image encryption in hybrid domains. *Opt. Commun.* **2012**, *285*, 1736–1743.
30. Wang, X.; Liu, L.; Song, M. Remote sensing image and multi-type image joint encryption based on NCCS. *Nonlinear Dyn.* **2023**, *111*, 14537–14563.
31. Das, T.; Mohan, P.; Padmanabhan, V.N.; Ramjee, R.; Sharma, A. PRISM: Platform for remote sensing using smartphones. In Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, San Francisco, CA, USA, 15–18 June 2010; pp. 63–76.
32. Wang, T.; Kerschbaum, F. Riga: Covert and robust white-box watermarking of deep neural networks. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 12–16 April 2021; pp. 993–1004.
33. Ong, D.S.; Chan, C.S.; Ng, K.W.; Fan, L.; Yang, Q. Protecting intellectual property of generative adversarial networks from ambiguity attacks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 3630–3639.
34. Adi, Y.; Baum, C.; Cisse, M.; Pinkas, B.; Keshet, J. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In Proceedings of the 27th USENIX Security Symposium (USENIX Security 18), Baltimore, MD, USA, 15–17 August 2018; pp. 1615–1631.
35. Zhang, J.; Gu, Z.; Jang, J.; Wu, H.; Stoecklin, M.P.; Huang, H.; Molloy, I. Protecting intellectual property of deep neural networks with watermarking. In Proceedings of the 2018 on Asia Conference on Computer and Communications Security, Incheon, Republic of Korea, 4–8 June 2018; pp. 159–172.
36. Li, Z.; Hu, C.; Zhang, Y.; Guo, S. How to prove your model belongs to you: A blind-watermark based framework to protect intellectual property of DNN. In Proceedings of the 35th Annual Computer Security Applications Conference, San Juan, PR, USA, 9–13 December 2019; pp. 126–137.
37. Namba, R.; Sakuma, J. Robust watermarking of neural network with exponential weighting. In Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, Auckland, New Zealand, 9–12 July 2019; pp. 228–240.
38. Obaid, T.S. Study a public key in RSA algorithm. *Eur. J. Eng. Technol. Res.* **2020**, *5*, 395–398.
39. Martino, R.; Cilardo, A. Designing a SHA-256 processor for blockchain-based IoT applications. *Internet Things* **2020**, *11*, 100254.
40. Wang, C.; Bai, X.; Wang, S.; Zhou, J.; Ren, P. Multiscale visual attention networks for object detection in VHR remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2018**, *16*, 310–314.
41. Long, Y.; Gong, Y.; Xiao, Z.; Liu, Q. Accurate object localization in remote sensing images based on convolutional neural networks. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 2486–2498.
42. Zou, Z.; Shi, Z. Random access memories: A new paradigm for target detection in high resolution aerial remote sensing images. *IEEE Trans. Image Process.* **2017**, *27*, 1100–1111.
43. Ding, N.; Qin, Y.; Yang, G.; Wei, F.; Yang, Z.; Su, Y.; Hu, S.; Chen, Y.; Chan, C.M.; Chen, W.; et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nat. Mach. Intell.* **2023**, *5*, 220–235.
44. Blalock, D.; Gonzalez Ortiz, J.J.; Frankle, J.; Guttag, J. What is the state of neural network pruning? *Proc. Mach. Learn. Syst.* **2020**, *2*, 129–146.
45. Shen, M.; Huang, R. Backdoor Attacks with Wavelet Embedding: Revealing and enhancing the insights of vulnerabilities in visual object detection models on transformers within digital twin systems. *Adv. Eng. Inform.* **2024**, *60*, 102355.

46. Chen, K.; Lou, X.; Xu, G.; Li, J.; Zhang, T. Clean-image backdoor: Attacking multi-label models with poisoned labels only. In Proceedings of the Eleventh International Conference on Learning Representations, Virtual, 25–29 April 2022.
47. Chen, H.; Rouhani, B.D.; Fu, C.; Zhao, J.; Koushanfar, F. Deepmarks: A secure fingerprinting framework for digital rights management of deep learning models. In Proceedings of the 2019 on International Conference on Multimedia Retrieval, Ottawa, ON, Canada, 10–13 June 2019; pp. 105–113.