PCPT and ACPT: Copyright Protection and Traceability Scheme for DNN Model

Xuefeng Fan, Hangyu Gui and Xiaoyi Zhou

Abstract—Deep neural networks (DNNs) have achieved tremendous success in artificial intelligence (AI) fields. However, DNN models can be easily illegally copied, redistributed, or abused by criminals, seriously damaging the interests of model inventers. Currently, the copyright protection of DNN models by neural network watermarking has been studied, but the establishment of a traceability mechanism for determining the authorized users of a leaked model is a new problem driven by the demand for AI services. Because the existing traceability mechanisms are used for models without watermarks, a small number of false positives is generated. Existing black-box active protection schemes have loose authorization control and are vulnerable to forgery attacks. Therefore, based on the idea of black-box neural network watermarking with the video framing and image perceptual hash algorithm, this study proposes a passive copyright protection and traceability framework *PCPT* using an additional class of DNN models, improving the existing traceability mechanism that yields a small number of false positives. Based on the authorization control strategy and image perceptual hash algorithm, using the authorization control center constructed using the detector and verifier, a DNN model active copyright protection and traceability framework *ACPT* is proposed. It realizes stricter authorization control, which establishes a strong connection between users and model owners, and improves the framework security. The key sample that is simultaneously generated does not affect the quality of the original image and supports traceability verification.

Index Terms—DNN, PCPT, ACPT, traceability, copyright protection.

I. INTRODUCTION

Deep neural networks (DNNs) have been widely used in image and speech processing, including natural language processing[1], computer vision[2], image processing[3], and speech recognition[4]. Advanced neural network models, including LeNet [5], VGGNet [6], GoogLeNet [7], and ResNet [8], have also shown excellent performances. Internet companies, such as Microsoft, Baidu, and Google, have deployed DNN models in their products and services to provide intelligent and high-quality services. In contrast to traditional multimedia data, the cost of training a good DNN model is significant. It requires the use of large-scale datasets, huge computing resources, and large labor costs. Therefore, the copyright protection and traceability of DNN models are particularly important. The methods to protect the copyright of DNN models from being illegally stolen and plagiarized and to trace the source of the stolen model to determine the authorized user of the leaked model are challenging problems.

Recently, inspired by the traditional watermarking ideas [9, 10], researchers have proposed various schemes for the copyright protection of DNN models [11], which can be roughly divided into four categories: white-box watermarking [12, 13], black-box watermarking [14, 15], gray-box watermarking [16], and null-box watermarking [17, 18]. However, according to the literature, only the KeyNet framework proposed by Jebreel *et al.* [19] has solved the problem of traceability after the DNN model is illegally stolen and distributed. However, when the KeyNet framework is used for models without watermarks, a small number of false positives are produced. For example, LeNet and VGG16 networks yield 17.93% and 7.92% false positive rates, respectively. Therefore, based on the idea of black-box neural network watermarking and the video framing and image perceptual hash algorithm, this study proposes a passive copyright protection and traceability (PCPT) framework for DNN models using additional classes. Since PCPT uses additional classes as the trigger set, the distortion of the original decision boundary is minimized (or even eliminated), thus realizing a zero false positive rate in the unlabeled model. Specifically, the PCPT framework utilizes a video material shot by the model owner as a key. After a video is framed, several trigger sets are constructed according to the different subjects in the video. When the owner information is embedded in the trigger set using the image perceptual hash algorithm, the set is used as an additional class, and different additional classes are embedded into DNN models as watermarks with the fine-tuning technology. The model is then distributed to different users to realize traceability. Note that one DNN watermark can be used for model copyright protection, while different watermarks embedded in different DNN model copies can be used for model traceability.

Additionally, different from Wu et al. [18] focusing on image processing tasks, the PCPT framework is suitable for general image classification DNN models and concentrates on the traceability after DNN models is stolen. For text classification models, the construction of trigger sets needs to be reconsidered, but the PCPT framework traceability is still applicable.

The PCPT framework utilizes the video framing and image perceptual hash algorithm because of the following reasons:

- First, after a digital video is framed, the content between adjacent frames is highly correlated. Therefore, compared to the general trigger set construction method, a very obvious feature of the trigger set constructed by video framing is that it has a strong correlation, which supports the DNN model to fully learn the trigger set characteristics. Second, the PCPT framework needs to allocate different trigger sets for users, increasing the difficulty of trigger set protection. If the trigger set is constructed using the video framing technology, the model owner only needs to save a piece of video material that is not related to the DNN model as a private key.

- To strongly link a DNN model with an owner, researchers are embedding the model owner information inside the trigger set image. This method has limited capacity to embed owner information and affects the quality of the original image. Therefore, the PCPT framework adopts an image perceptual hash algorithm to uniquely associate an owner with a DNN model. This approach does not degrade the trigger set image quality, and detecting whether the trigger set image is protected is difficult for an attacker.

Currently, various DNN model copyright protection schemes [11] prove an owner's copyright after the model is stolen, which is passive protection. To date, few studies have considered the active protection of DNN models through authorization control. Xue *et al.* [20] proposed a DNN copyright protection method using additional categories and image steganography techniques outside the dataset. Their method first selects a small number of images outside the original training dataset as watermark key samples. Then, the user's fingerprint is hidden in each watermark key sample through the least significant bit (LSB) technique: each user is assigned a unique fingerprint image so that the user's identity can be later verified. For legitimate users to use the DNN model, two conditions need to be met: 1) the DNN model classifies the watermark key samples into additional categories and 2) the legitimate users' fingerprints are extracted from the watermark key samples. Although their method realized the authorization control of the DNN model, it has some disadvantages. First, a user must use the DNN model itself for authentication, which will result in insufficient authorization control. Strict authorization control means that users must pass an identity authentication before they can access the DNN model. Second, the information embedded in the watermark key sample is only the user's fingerprint information, which does not establish a strong connection between the user and model owner. Additionally, the embedded information is not encrypted, which is insecure, and malicious attackers may forge a legitimate user identity to access the DNN model. Finally, the information is embedded in the additional sample image through the LSB technique, which will affect the image itself.

Based on the work of Xue *et al.* [20] and the authorization control strategy and image perceptual hash algorithm and using the authorization control center constructed by the detector and verifier, this study proposes a DNN model active copyright protection and traceability (ACPT) framework. In the framework, the detector detects whether the key image input by the user is legal, and the authenticator verifies whether the user identity information is legal. The ACPT framework implements strict authorization control, establishes a strong connection between users and model owners, and improves the framework security. Furthermore, the generated key samples do not affect the quality of the original image, and the traceability of the stolen DNN model is realized.

In short, the contributions of this study are as follows:

- A PCPT framework is proposed. The PCPT framework utilizes the black-box DNN watermarking technique using additional classes as trigger sets, which minimizes (or even eliminates) the effect of distortion of the original decision boundary. Moreover, the additional classes do not exist in the unlabeled model, and thus, the framework achieves a zero false positive rate for the unlabeled model. The PCPT framework uses video framing technology to make it easier for the DNN model to learn the characteristics of the trigger set and simultaneously reduce the protection difficulty of the trigger set. Along with blockchain technology, the security of the PCPT framework is improved. Additionally, to our knowledge, an image perceptual hash algorithm has been used for the first time to uniquely associate trigger sets with owner identities.

- A ACPT framework is proposed. The ACPT framework utilizes the authorization control center constructed by the detector and validator to realize stricter authorization control, establish a strong connection between a user and model owner, and improve the framework security. Moreover, the generated key samples have no effect on the quality of the original image, and the traceability of the stolen DNN model is realized.

- Experiments were conducted on the PCPT framework using LeNet5, VGG16, GoogleNet, and ResNet18 models on the MNIST and CIFAR10 datasets to verify the effectiveness of the PCPT framework for DNN model tracking and traceability as well as robustness to model modification attacks. Simultaneously, the VGG16, GoogleNet, and

ResNet18 models were used as examples to conduct experiments on the ACPT framework, proving the effectiveness of the ACPT framework's authorization control and traceability performance. Furthermore, the hiding of key samples was more concealed in the ACPT framework.

The rest of the paper is structured as follows. Section 2 briefly introduces the background knowledge and related research work closely related to our study. Section 3 discusses the threat model, Section 4 introduces the PCPT framework and its evaluation experiments, and Section 5 introduces the ACPT framework and its evaluation experiments. Finally, Section 6 presents the conclusions and proposes an outlook for future research.

II. BACKGROUND AND RELATED WORK

A. DCT-PHA

Kalker *et al.* [21] first proposed "perceptual hashing" in 2001. The perceptual hash algorithm (PHA) maps multimedia data into a digest, which is one-way. Image perceptual hashing algorithms are used to generate "fingerprint" strings for images, mainly including the perceptual hash function based on the discrete cosine transform (DCT-PHA), the perceptual hash function based on the Marr–Hildreth operator, and the radial variance and block mean-based perceptual hash functions. In comparison, DCT-PHA affords better image recognition and resolution capabilities than other image perceptual hashing algorithms [22].

The processing of DCT-PHA can be divided into seven steps: 1) To facilitate DCT calculation, adjust the image size to 32×32. 2) Convert the adjusted image to grayscale. 3) Perform DCT calculation on the grayscale image. 4) Maintain the 8×8 low frequency region in the upper left corner of the image. 5) Calculate the average *ave* of all pixels in the low-frequency region. 6) Comparing the pixel value of each pixel in the low-frequency region with the size of *ave* via (1), obtain an 8×8 binary matrix. 7) According to the set order, combine to obtain the hash value of the image.

$$\begin{cases} 1, if \ pixel \ value > ave \ , \\ 0, \qquad \quad other. \end{cases} \quad (1)$$

B. DNN

A neural network is a model imitating the structure and function of the biological central nervous system. It mainly simulates the learning method using the biological brain to acquire knowledge so that the machine can learn complex data for reasoning and decision making. It performs computation by connecting multiple neurons, and each neuron calculates the results of the nonlinear mapping of the weighted vectors. DNNs are stacked on top of neural networks and comprise multiple layers. Assuming that a DNN F comprises P layers, the calculation process can be expressed as follows:

$$f(x) = \omega^{P+1}(h^P(\omega^P(\cdots(\omega^2(h^1(\omega^1 x + b^1)) + b^2)\cdots)) + b^P) + b^{P+1} \quad (2)$$

where x is the input signal, and ω and b are the weight and bias terms, respectively.

C. DNN model copyright protection scheme

1) **Passive Protection Scheme**

The passive copyright protection scheme of the DNN model was first proposed by Uchida *et al.* [12] to use watermarking to solve the DNN model. According to different application scenarios, DNN watermarking can be divided into four categories: white-box, black-box, gray-box, and null-box watermarks. Herein, black-box DNN watermarking is mainly analyzed. In black-box DNN watermarking, the model owner constructs a trigger set through specific inputs and outputs to transform the model. During watermark verification, the model owner uses the trigger set to verify ownership. The existing black-box DNN watermarking work can be further divided according to the different construction methods of the trigger set.

a) Construct the trigger set with only label changes. Adi *et al.* [14] constructed a trigger set using a set of abstract images and labels that are inconsistent with the image content, and they randomly assigned labels to trigger set images. After the trigger set is input, the watermarked DNN model outputs a specific label to verify the copyright of the model.

b) Construct a trigger set using embedding information and label changes in the original samples. Zhang *et al.* [15] studied three watermark generation algorithms, including embedding meaningful text content and meaningless noise into image samples as watermarks and assigning incorrect labels to irrelevant samples as watermarks. Based on their work, a backdoor watermarking method is proposed to embed watermarks into target models.

c) Construct the trigger set by adding a new class. Zhong *et al.* [23] watermarked models by adding new class labels to carefully crafted key samples during training. By modifying the task of predicting (N−1) different classes from the original target model to predict N different classes, a watermark-free model cannot output a nonexistent class label. The above scheme for DNN model copyright protection provides a reference for the PCPT framework. However, in the above scheme, the watermark is the same in all copies of the model. Thus, after the DNN model is leaked, the owner cannot trace the source.

2) **Active Protection Scheme**

The active protection scheme of the DNN model is addressed using authorization control. Chen *et al.* [24] used an additional anti-piracy conversion module to verify the legitimacy of users, providing authorization controls for trained DNNs so that only authorized users can use them correctly. Fan *et al.* [25] exploited passports to control the performance of DNN models, whose performance either remained unchanged in the presence of valid passports or significantly deteriorated due to modified or counterfeit passports.

Chakraborty *et al.* [26] implemented authorization control for DNN models using a hardware-assisted approach, which relied on a trusted hardware device (as a root of trust) to store each user's key, ensuring that only trusted hardware devices (with key embedded on the chip) run the intended deep learning application using the published model. However, their method is expensive for commercial applications. Furthermore, the above active authorization control methods do not support user authentication management, making them unsuitable for commercial applications. Additionally, through the active protection of a DNN model, all users' copies of the model are identical. Thus, once a DNN model is leaked, the owner cannot trace the source.

In summary, few studies have been conducted to date on the traceability of DNN models after they have been stolen. The KeyNet framework proposed by Jebreel *et al.* [19] solves the attribution problem of DNN models, but the framework yields a small amount of false positives when used in models without watermarks. Therefore, this study proposes the PCPT and ACPT frameworks to solve the problem of traceability after a DNN model is stolen.
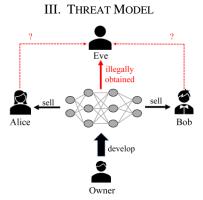
III. THREAT MODEL

Fig. 1. Threat model

As shown in Fig. 1, *Owner* is the model owner. *Owner* developed a DNN model and sold it to two (or more; two users are considered herein) users: *Alice* and *Bob*. *Alice* and *Bob* deploy the model for use by the specified population. Note that *Alice* and *Bob* are authorized users of *Owner* and only have the right to use the DNN model and do not have ownership. Thus, they need to bear the responsibility for the copyright maintenance of the DNN model. After determining the function of the DNN model, the criminal *Eve* wants to steal a copy of the DNN model from *Alice* or *Bob* to distribute or provide services for profit. Notably, after *Eve* steals the DNN model, how the *Owner* traces the source of *Eve* stealing the DNN model and pursues the responsibility of the authorized user for the leakage of intellectual property rights are the main problems solved herein.

This study aims to evaluate the copyright protection and traceability framework of DNN models. When the DNN model

is stolen, *Owner* can lock the leak source of the model using the PCPT and ACPT frameworks, thus providing help for later accountability and rights protection.

IV. PCPT FRAMEWORK

In this section, the PCPT framework is proposed. The PCPT framework aims to trace the origin of suspicious DNN models by validating watermarks in remote DNN services. To embed the watermark into a DNN model, the framework builds different trigger sets and corresponding predefined labels for different users; it converts trigger sets with predefined labels $W = \{w(i), Alice/Bob\}_{i=1}^{L}$ as an additional class and 10% of the original training data $D'_{train} = \{x(i), y(i)\}_{i=1}^{N}$ to fine-tune the training original DNN model F. Then, it generates the watermarked DNN models F_{Alice} and F_{Bob}, as shown in (3). DNN models automatically learn and memorize patterns of embedded watermarks and predefined labels, and only DNN models protected by watermarks can generate predefined predictions. In the verification stage, after different trigger sets are input to a watermarked DNN model, a predefined additional class is output, so as to achieve the purpose of traceability.

$$F_{Alice}, F_{Bob} \leftarrow Train(F(D'_{train} \cup W)) \qquad (3)$$

As shown in Fig. 2, the PCPT framework is mainly divided into five stages: trigger set generation, *Owner* fingerprint embedding, DNN watermark embedding, traceability verification, and ownership verification.

A. Trigger set generation

The video framing technique is adopted to construct the trigger set. First, *Owner* shoots and produces a video that is rich in content and wherein different frame images can be differentiated. Then, the video is divided into frames, and some frame images with different contents are selected to construct the trigger sets $W_{Alice} = \{w(i), Alice\}_{i=1}^{L}$ and $W_{Bob} = \{w(i), Bob\}_{i=1}^{L}$. Herein, the number of users is two. If the number of users increases, the duration of the video material and the richness of the content should be simultaneously increased. The video footage is ultimately saved by the *Owner* being used as a key.

B. Owner fingerprint embedding

To uniquely associate the trigger samples with the *Owner*, the *Owner* can select L trigger images with high robustness against model modification attacks from the trigger set to embed their fingerprint. Figure 3 shows the *Owner* fingerprint embedding process. First, the trigger image and *Owner* fingerprint image are processed using the DCT-PHA algorithm, and the hash values P_1 and P_2 of the image are obtained. Then, P_1 and P_2 are XORed to obtain P. Finally, P is stored in the blockchain transaction.
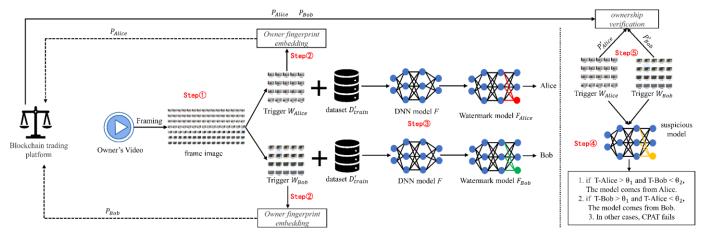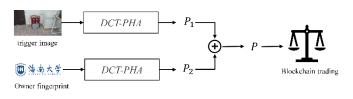
Fig. 2. PCPT framework

Fig. 3. *Owner* fingerprint embedding

Algorithm 1   DNN Watermark Embedding and Traceability

Input: Original DNN model F , original data set $D_{train} = \{x(i), y(i)\}_{i=1}^{N}$, trigger set $W_{Alice} = \{w(i), Alice\}_{i=1}^{L}$, trigger set $W_{Bob} = \{w(i), Bob\}_{i=1}^{L}$, Suspicious DNN model F_{Sus} , epoch , crossentropyloss , threshold θ_1 and θ_2 .

Output: Watermarked DNN model F_{Alice} and F_{Bob} , Source of suspicious DNN model F_{Sus} S .

(a) $D'_{train} \leftarrow$ randomly choose 10% from $D_{train} = \{x(i), y(i)\}_{i=1}^{N}$

(b) $D_{Alice} \leftarrow D'_{train} \cup W_{Alice}$

(c) $D_{Bob} \leftarrow D'_{train} \cup W_{Bob}$

(d) **for** i = 1 , 2 , 3 … , epoch **do**

　　　$F_{Alice} \leftarrow$ Train (F (D_{Alice}))

　　　crossentropyloss($F_{Alice}(W_{Alice}$) , *Alice*)

　end for

(e) **for** i = 1 , 2 , 3 … , epoch **do**

　　　$F_{Bob} \leftarrow$ Train (F (D_{Bob}))

　　　crossentropyloss($F_{Bob}(W_{Bob})$, *Bob*)

　end for

(f) $T\text{-}Alice = F_{Alice}$ (W_{Alice}) and $T\text{-}Bob = F_{Bob}(W_{Bob})$

(g) **if** $T\text{-}Alice > \theta_1$ and $T\text{-}Bob < \theta_2$

　　　$S \leftarrow Alice$

　if $T\text{-}Bob > \theta_1$ and $T\text{-}Alice < \theta_2$

　　　$S \leftarrow Bob$

　else

　　　traceability failure

(h) **return** F_{Alice} , F_{Bob} , S

C. DNN watermark embedding

To embed the watermark in the DNN model, the *Owner* uses different trigger sets $W_{Alice} = \{w(i), Alice\}_{i=1}^{L}$ and $W_{Bob} = \{w(i), Bob\}_{i=1}^{L}$ with 10% of the original training data $D'_{train} = \{x(i), y(i)\}_{i=1}^{N}$ to form a new training set D_{Alice} and D_{Bob} to fine-tune the DNN model F and obtain the watermarked DNN models F_{Alice} and F_{Bob} . The network structure of F_{Alice} and F_{Bob} is identical to that of F, except that the output layer adds a class (*Alice* or *Bob*) to the original one. The embedding and traceability algorithm of the DNN watermark is shown in Algorithm 1.

D. Traceability verification

When a suspicious DNN service is found, the *Owner* uses the trigger sets $W_{Alice} = \{w(i), Alice\}_{i=1}^{L}$ and $W_{Bob} = \{w(i), Bob\}_{i=1}^{L}$ to verify whether a watermark is present in the suspicious model. The test accuracy T-Alice and T-Bob of the trigger sets $W_{Alice} = \{w(i), Alice\}_{i=1}^{L}$ and $W_{Bob} = \{w(i), Bob\}_{i=1}^{L}$ are compared to achieve the purpose of traceability. Here, *Owner* sets two thresholds θ_1 and θ_2. If T-Alice $> \theta_1$ and T-Bob $< \theta_2$, *Alice* has leaked the model. Inversely, *Bob* has leaked the model. Beyond that, the PCPT framework will fail.

E. Ownership verification

An advantage of utilizing the image perceptual hash algorithm is that the criminals do not know whether the *Owner*'s data are protected and the image quality is not compromised. When the ownership of the DNN model needs to be proved, *Owner* first uses the trigger image to trigger the watermark in the DNN model to verify the copyright and then obtains P' according to the method in 4.2. Finally, the smart contract is called to query the data P stored in the blockchain transaction. If $P' = P$, the ownership verification is successful.

F. Experiment analysis

The performance of the PCPT framework is experimentally evaluated. All experiments are performed on

the Google Colab platform. The graphics card is NVIDIA Tesla P100, and the deep learning framework is PyTorch. First, the dataset and DNN model used in the PCPT experiments are introduced. Then, the effectiveness of the PCPT framework for tracing the source of the DNN model after being stolen is verified, and the impact of the PCPT framework embedded in the watermark on the accuracy of the original model is tested. Finally, the robustness of the PCPT framework to model modification attacks and the security and uniqueness of the PCPT framework are proved.

1) Experimental setup

Datasets and models. The PCPT framework is evaluated on two datasets (MNIST and CIFAR10). Among them, the MNIST dataset adopts the LeNet5 network architecture, and the CIFAR10 dataset adopts the VGG16, GoogleNet, and ResNet18 network architectures. MNIST is used to train the LeNet5 model. The trigger set constructed using 10% MNIST data and the video frame segmentation image is used to fine-tune the trained LeNet5 model to embed the watermark. One-hundred video frame images are assigned to each user to construct a trigger set. Similarly, same settings are used for the CIFAR10 dataset and the VGG16, GoogleNet, and ResNet18 models.

Parameter setup. The thresholds are set to $\theta_1 = 85\%$ and $\theta_2 = 60\%$. In the fine-tuning embedding watermark stage, the cross-entropy loss is selected as the loss function to fine-tune the four models for 50 epochs.

2) Validity

The validity is evaluated to quantify whether *Owner* can trace the DNN model stolen by the criminals to *Alice* or *Bob*. As shown in Table I, when the watermarked DNN models F_{Alice}-LeNet5, F_{Alice}-VGG16, F_{Alice}-GoogleNet, and F_{Alice}-ResNet18 are assigned to *Alice*, the conditions of *T-Alice* $> \theta_1$ and *T-Bob* $< \theta_2$ are satisfied. When the watermarked DNN models F_{Bob}-LeNet5, F_{Bob}-VGG16, F_{Bob}-GoogleNet, and F_{Bob}-

ResNet18 are assigned to *Bob*, the conditions of *T-Bob* $> \theta_1$ and *T-Alice* $< \theta_2$ are satisfied. Therefore, the PCPT framework effectively traces the source of the DNN models after they are stolen.

TABLE I.   WATERMARK TEST ACCURACY OF THE WATERMARK MODEL

Data Set	Model	WM accuracy	
		T-Alice	T-Bob
MNIST	F_{Alice}-LeNet5	100%	0
	F_{Bob}-LeNet5	0	100%
CIFAR10	F_{Alice}-VGG16	99%	48%
	F_{Bob}-VGG16	11%	100%
	F_{Alice} - GoogleNet	98%	10%
	F_{Bob} - GoogleNet	100%	0
	F_{Alice} - ResNet18	100%	0
	F_{Bob} - ResNet18	100%	3%

3) Fidelity

Fidelity requires our PCPT framework to watermark the original model without significant side-effects on the main task of the original model. Ideally, the watermarked DNN model should be as accurate as the original DNN model. Figure 4 shows the comparison between the watermarked models F_{Alice} and F_{Bob} and the original model F in the test accuracy of the original task. The results show that the PCPT framework decreases the test accuracy of the DNN model by an average of 0.67%, 1.66%, 0.39%, and 1.38% for the LeNet5, VGG16, GoogleNet, and ResNet18 models, respectively, after the watermark is embedded. Moreover, the side-effects afforded by PCPT framework are within the acceptable performance variation of the model and have no significant impact on the main task. Therefore, the PCPT framework meets fidelity requirements.
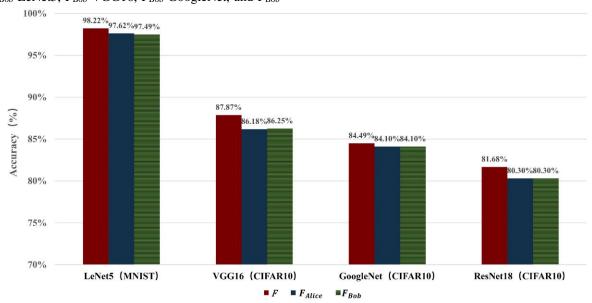

Fig. 4. Original task test accuracy for the original model F and watermarking models F_{Alice} and F_{Bob}

4) Robustness

The resistance of the watermarked model to model modification attacks (fine-tuning and pruning attacks) is tested to measure the robustness of the watermarking method.

a) Fine-tuning attack. In this experiment, the test set for each dataset is split in half. The first 50% is used to fine-tune the watermarked DNN model, and the latter 50% is used to evaluate the fine-tuned model. Then, the watermarked DNN model is fine-tuned and trained for 200 epochs. The experimental results are shown in Table II. Table II shows that in *Alice*'s model, the value of *T-Alice* is above 96% and *T-Bob* is below 26%. In *Bob*'s model, the value of *T-Alice* is below 20% and *T-Bob* is above 96%. Thus, the set threshold is satisfied, so the PCPT framework is robust to fine-tuning attacks.
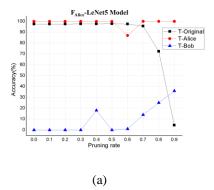
b) Pruning attack. In this experiment, the Global Pruning method is used to prune the weight parameters of the watermarking model. Figure 5 shows the effect of model compression on the original task and trigger set test accuracies under different pruning rates. As shown in the figure, when pruning about 80% of the parameters of the LeNet5 and ResNet18 models and pruning about 50% of the parameters of the VGG16 and GoogleNet models, $T\text{-}Alice > \theta_1$ and $T\text{-}Bob < \theta_2$ in Alice's Model and $T\text{-}Alice < \theta_2$ and $T\text{-}Bob > \theta_1$ in Bob's Model are still satisfied. Thus, the PCPT framework is valid. As the pruning rate increases, the test accuracy of the DNN model on the trigger set will decrease. However, the attacker will not delete more than 50% of the model parameters, because when the deletion rate exceeds 50%, the test accuracy of the model will sharply decrease. In other words, the attacker cannot prune the embedded watermark while maintaining the normal performance of the model. Therefore, the PCPT framework is robust to pruning attacks.
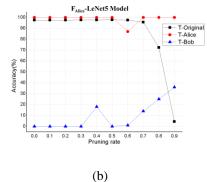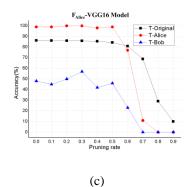
5) Security

The security of the PCPT framework from the perspective of *Eve*'s illegal ownership claims on DNN models is evaluated.
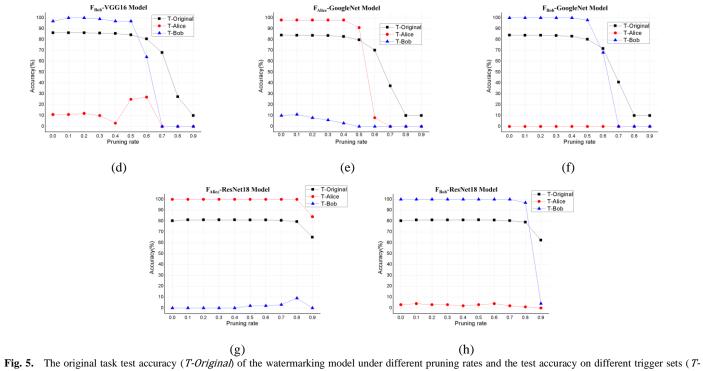
Eve forges an illegal trigger set similar to the original trigger sample to make an ownership claim. The premise of this case is that *Eve* discovers a hidden pattern of triggering samples in the DNN model. Twelve images are randomly selected from the illegal trigger set forged by *Eve* for different models, as shown in Fig. 6, and watermark verification is performed. The experimental results are shown in Table III. The table shows that the trigger set forged by *Eve* is not always able to trigger the DNN watermark. Additionally, the Structural Similarity Index (SSIM) comparison is performed between the illegal trigger set forged by *Eve* and the *Owner*'s 100 legal trigger set images. The mean SSIM of the comparison results is shown in Table III. The mean value of SSIM is the highest at 0.5017, which proves that the trigger set forged by *Eve* is not the frame image in the *Owner* key (video material) (SSIM < 0.9). Therefore, even if *Eve*'s forged trigger set successfully triggers the watermark in the DNN model, *Owner* can still prove that *Eve*'s forged trigger set illegally claims ownership.

Eve fakes a new trigger set to claim ownership. In this case, *Eve* is unaware of the hidden trigger sample patterns in the DNN model. However, *Eve* sneaked their watermark into the DNN model by faking a new trigger set. During judicial authentication, two watermarks are present in the DNN model, and the ownership of the DNN model is ambiguous. However, the fingerprint information was embedded in the original trigger set by the *Owner* and stored in the blockchain transaction. Because the storage time and content written in the blockchain transaction cannot be tampered with, it is used in the ownership authentication. Thus, ambiguity is avoided and strong validation is provided to the *Owner*. Even if *Eve* uses the same operation to store their fingerprint on the blockchain, *Owner* stored it earlier than *Eve*, which still proves the fact that *Eve* is forging the trigger set to illegally claim ownership.

(a)　　　　　　　　(b)　　　　　　　　(c)

(d)　　　　　　　　　　　　　(e)　　　　　　　　　　　　　(f)

(g)　　　　　　　　　　　　　(h)

Fig. 5. The original task test accuracy (*T-Original*) of the watermarking model under different pruning rates and the test accuracy on different trigger sets (*T-Alice* and *T-Bob*)

TABLE II. THE ORIGINAL TASK TEST ACCURACY (*T-ORIGINAL*) AND THE TEST ACCURACY ON DIFFERENT TRIGGER SETS (*T-ALICE, T-BOB*) OF THE WATERMARKING MODEL BEFORE AND AFTER THE FINE-TUNING ATTACK

Number of Epochs	F_{Alice}-LeNet5			F_{Bob}-LeNet5		
	T-Original	T-Alice	T-Bob	T-Original	T-Alice	T-Bob
50	98.80%	100%	12%	98.66%	0	100%
100	98.82%	100%	12%	98.60%	0	100%
150	98.82%	100%	12%	98.60%	0	100%
200	98.82%	100%	12%	98.60%	0	100%
Number of Epochs	F_{Alice}-VGG16			F_{Bob}-VGG16		
	T-Original	T-Alice	T-Bob	T-Original	T-Alice	T-Bob
50	85.73%	96%	1%	86.14%	20%	100%
100	85.39%	97%	26%	86.08%	17%	99%
150	86.02%	96%	1%	85.98%	14%	100%
200	85.90%	97%	16%	85.98%	11%	100%
Number of Epochs	F_{Alice}-GoogleNet			F_{Bob}- GoogleNet		
	T-Original	T-Alice	T-Bob	T-Original	T-Alice	T-Bob
50	84.33%	98%	15%	84.31%	0	96%
100	84.47%	98%	15%	84.31%	0	96%
150	84.25%	98%	4%	84.27%	0	96%
200	84.14%	98%	12%	84.41%	11%	96%
Number of Epochs	F_{Alice}-ResNet18			F_{Bob}- ResNet18		
	T-Original	T-Alice	T-Bob	T-Original	T-Alice	T-Bob
50	81.07%	99%	0	80.60%	0	99%
100	80.95%	98%	0	81.11%	0	100%
150	80.56%	99%	0	80.78%	0	98%
200	80.80%	98%	0	80.56%	0	97%

Fig.6. The 12 illegal trigger sets. (a)–(d) are the trigger sets faked for *Alice*'s LeNet5 model, (e)–(h) are the trigger sets faked for *Bob*'s LeNet5, VGG16, GoogleNet, and ResNet18 models, (i)–(l) are Fake trigger sets for *Alice*'s VGG16, GoogleNet, and ResNet18 models

TABLE III. "F_{Alice}-LeNet5 WATERMARK" INDICATES THE RESULT OF TRIGGERING THE DNN WATERMARK IN THE F_{Alice}-LeNet5 MODEL BY *Eve*'S FORGED TRIGGER SET ("√" INDICATES SUCCESSFUL TRIGGERING AND "×" INDICATES TRIGGER FAILURE) AND OTHERS ARE SIMILAR. "SSIM" DENOTES THE MEAN OF THE SSIM COMPARISON RESULTS BETWEEN *Eve*'S FORGED ILLEGAL TRIGGER SET AND THE OWNER'S 100 LEGAL TRIGGER SET IMAGES

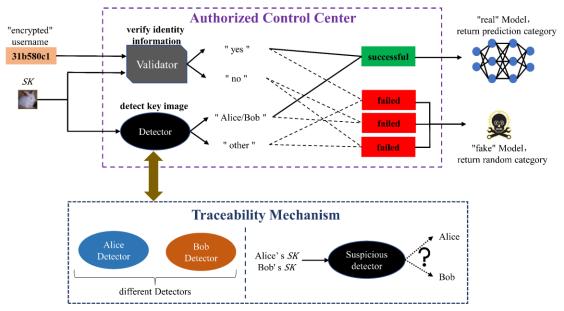|					
MNIST	fake trigger set	Fig.7 (a)	Fig.7 (b)	Fig.7 (c)	Fig.7 (d)
	SSIM	0.4528	0.3002	0.3491	0.3844
	F_{Alice}-LeNet5 watermark	√	×	×	√
	fake trigger set	Fig.7 (e)	Fig.7 (f)	Fig.7 (g)	Fig.7 (h)
	SSIM	0.4972	0.4644	0.5017	0.2746
	F_{Bob}-LeNet5 watermark	√	√	√	√
CIFAR10	fake trigger set	Fig.7 (i)	Fig.7 (j)	Fig.7 (k)	Fig.7 (l)
	SSIM	0.3434	0.3949	0.4245	0.4817
	F_{Alice}-VGG16 watermark	×	×	×	×
	F_{Alice}-GoogleNet watermark	×	√	√	√
	F_{Alice}-ResNet18 watermark	×	×	×	×
	fake trigger set	Fig.7 (e)	Fig.7 (f)	Fig.7 (g)	Fig.7 (h)
	SSIM	0.4972	0.4644	0.5017	0.2746
	F_{Bob}-VGG16 watermark	×	×	√	×
	F_{Bob}-GoogleNet watermark	×	√	×	×
	F_{Bob}-ResNet18 watermark	×	√	×	√

Fig. 7. ACPT framework

V. ACPT FRAMEWORK

In this section, the DNN model ACPT framework is proposed. First, the ACPT framework builds an authorization control center using a detector and validator. Then, it packages the DNN model to be protected with the authorization control center (referred to as DNN_{AC}) and distributes it to users, thereby realizing active authorization control and user identity management of the DNN model. Second, the ACPT framework builds different authorization control centers $AC = \{AC_{Alice}, AC_{Bob}\}$ for different users to trace the source of the suspicious DNN model after the DNN model is stolen. In the verification stage, different keys are input into DNN_{AC} to verify whether the DNN model can be correctly used to achieve the purpose of traceability.

As shown in Fig. 7, the ACPT framework is mainly divided into two parts: the authorization control center and traceability mechanism. The authorization control center comprises a detector and verifier, which realizes the active copyright protection of the DNN model, and the traceability mechanism realizes the traceability verification of the DNN model.

A. Authorized control center

The authorization control center comprises a detector and validator. The design methods of the detector and validator are introduced below.

1) **Detector**

The design idea of the detector stems from the attack method of Hitaj *et al.* [27]. In the ACPT framework, the detection of key images input by users is modeled as a binary classification problem, and the detector is trained to distinguish between legitimate and illegitimate users (Fig. 5). The detector takes the key images input by the user as the input and outputs

two kinds of results: 1) a valid user key (*Alice* or *Bob*) and 2) an illegal user key (*other*).

2) **Validator**

Owner generates "encrypted" usernames for each user based on the "real" usernames of legitimate users. To better explain the authenticator process, an example is used to illustrate the generation of an "encrypted" username. First, assume that the "real" user name is *user1* and the *Owner's* fingerprint information is *HN*. Combine the username with the *Owner's* fingerprint information (*HNuser1*) and encrypt it with sha256 to obtain a 64-bit string m. Use the key K_1 to randomly extract 8 bits from m as the "encrypted" username, strengthening the connection between the *Owner* and legitimate user. The probability of an illegal user forging a legitimate "encrypted" username without knowing the *Owner's* fingerprint information is $\frac{1}{2,821,109,907,456}$. Therefore, it is almost impossible for an illegal user to successfully forge a legitimate "encrypted" username.

Algorithm 2 Validator Process

Input: *"encrypted" username, SK* .
Output: outcome.
  (a) $m_1 \leftarrow$ Binary(*"encrypted" username*)
  (b) $m_2 \leftarrow$ DCT-PHA(*SK*)
  (c) $I \leftarrow m_1 \oplus m_2$
  (d) **if** $I \in Q = \{I_1, I_2, I_3, \dots, I_i\}$
        outcome $\leftarrow$ "yes"
    else
        outcome $\leftarrow$ "no"
  (e) **return** outcome

The authenticator takes the "encrypted" username and key image *SK* as input, and after processing, it generates the verification result, as shown in Algorithm 2. First, convert the

"encrypted" username into a 64-bit binary string m_1. Second, the 64-bit binary string m_2 is obtained after the key image SK is processed by DCT-PHA. Finally, the verification information I is obtained by performing XOR on m_1 and m_2. If $I \in Q = \{I_1, I_2, I_3, \dots, I_i\}$, the authentication is successful; otherwise it fails. Here, Q is the legitimate user identity information base.

3) Authorization control

If the detector detects that a user's key is valid and the authenticator successfully verifies the user's identity information, the user passes the authorization control. A legitimate user will use the "true" model and obtain the predicted class of the DNN model. Otherwise, the user authentication fails, and the illegal user will use the "fake" model to obtain a random category of the DNN model. In this setting, the probability of an illegal user acquiring the correct predicted class is $1/N$, where N is the number of labels present in the model. Note that the detector may employ a strategy of always rejecting queries from illegal users. However, the design of the random return category could make it difficult for the attacker (illegal user) to realize the existence of the authorized control center to a certain extent; thus, the return of the random category to the illegal user query is proposed herein.

B. Traceability mechanism

As described in 5.1, an authorized control center built by detectors and validators enables the active copyright protection of DNN models. Based on this, the ACPT framework builds different authorization control centers for different users and establishes a traceability mechanism to realize traceability after the DNN model is stolen. Specifically, the ACPT framework utilizes two different datasets (such as the "apple" and "rabbit" datasets) as its key images to build two different detectors, which are assigned to users *Alice* and *Bob*. When the DNN model is leaked, the model owner uses different key images to authenticate the suspicious DNN and examine the test set for determining the source of the suspicious DNN model.

C. Experiment analysis

The performance of the ACPT framework is experimentally evaluated. All experiments are performed on the Google Colab platform. The graphics card is NVIDIA Tesla P100, and the deep learning framework is PyTorch. First, the detector setup in the ACPT framework is introduced and the performance of the detector is evaluated. Then, the authorization control performance of the ACPT framework is verified, and the concealment of the key samples is examined. Finally, the traceability validity verification is conducted.

1) Detector setup and performance

Datasets and models. The detector adopts the LeNet5 network architecture. Hundred images from the "apple" dataset and 100 other images (such as "cat", "chair" and "bee", etc.) from CIFAR100 are selected to train the $Detector_{Alice}$ assigned to user *Alice*, and 100 images from the "rabbit" dataset and 100 other images are selected to train the detector assigned to user *Bob* $Detector_{Bob}$.

Parameter setup. The detector is trained using an Adam optimizer for 50 epochs, and the cross-entropy loss is chosen as the loss function.

Performance. The detection accuracy of the trained detector on the key image is analyzed. As shown in Table IV, the detection accuracy of the detector on the key image reaches 100%, which meets the requirements of legal user authorization control.

TABLE IV.   DETECTOR ACCURACY

Detector	Detect Accuracy
$Detector_{Alice}$	100%
$Detector_{Bob}$	100%

2) Authorization control performance

Figure 8 shows the test accuracy for the authorized and unauthorized use of the VGG16, GoogleNet, and ResNet18 models on the CIFAR10 dataset. The results show that the test accuracy of authorized users on these three models is 87.87% (VGG16), 84.49% (GoogleNet), and 81.68% (ResNet18). However, the test accuracies of the unauthorized users are 9.26% (VGG16), 10.50% (GoogleNet), and 10.00% (ResNet18), which is an average decrease of 74.76% compared to the performance of authorized users. Therefore, the proposed ACPT framework can realize active authorization control and can effectively prevent the illegal use of the DNN model.
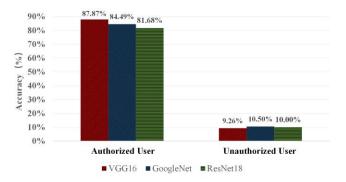

Fig. 8. Authorization control performance

3) Secrecy of key samples

The impact of watermark key samples on the original image quality was analyzed in Xue *et al.* [20]. The researchers used the LSB technology to minimize the impact of watermark key samples on the original image quality; in contrast, the ACPT framework uses the hash value generated by the perceptual image hashing technology as the information verification part and thus has no effect on the original image. The concealment of key samples in both methods is evaluated using the mean square error (MSE). The average MSE of 100 key samples generated by the two watermarking methods is shown in Table V. The results show that the ACPT framework

adopts the image-aware hashing algorithm without impacting the image quality. Therefore, compared to the work of Xue *et al.* [20], the key samples used by the ACPT framework are more stealthy.
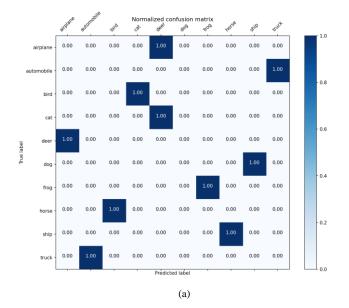
Data Set	Scheme	Xue at al [20]	Ours
CIFAR 10	Average MSE	0.015	0

4) **Traceability verification**

In the experiment, the *Owner* distributed two copies of the DNN model for users *Alice* and *Bob*, each with its corresponding authorization control center $AC = \{AC_{Alice}, AC_{Bob}\}$. The VGG16 model is taken as an example, and it is assumed that *Bob* leaks the VGG16 model. Next, the traceability mechanism in the ACPT framework is employed to try to find the leaker.

- Take a key image and its corresponding "encrypted" username from both the "apple" and "rabbit" keys.

- The suspicious model is authenticated using the two keys, and then, an image is selected from the 10 classes of CIFAR10 to test the suspicious model.

- The confusion matrix of the test results is shown in Fig. 9. As shown in Fig. 9(a), after authentication with the "apple" key, the accuracy of the test data on the suspicious model is 10%. As shown in Fig. 9(b), after authentication with the "rabbit" key, the accuracy of the test data on the suspicious model is 100%. Thus. *Bob* is the leaker of the DNN model.
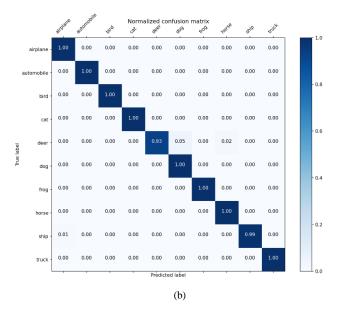

(a)

(b)

Fig. 9. Test results for suspicious models by selecting one image from each of the 10 CIFAR10 classes. Test result after authentication with (a) the "apple" key and (b) the "rabbit" key.

D. ACPT extension

As introduced in 5.1, the ACPT framework implements the active copyright protection and traceability of the DNN model through the authorization control center and does not make changes to the DNN model itself. Therefore, the ACPT framework can be efficiently combined with existing excellent DNN watermarking works [14, 15, 23, 28, 29] to achieve double-layer protection of DNN models.

VI. CONCLUSION

The traceability problem after the DNN model is stolen is a new problem driven by the market demand of artificial intelligence, and DNN watermarking and authorization control are two potential methods for solving this problem. Herein, based on the idea of black-box neural network watermarking, combined with the video framing and zero watermarking technology, the PCPT framework was proposed using an additional class of the DNN model, which improves the existing traceability mechanism and may produce a small number of false positives. Based on the authorization control strategy and perceptual image hashing technology and using the authorization control center constructed by the detector and verifier, a DNN model ACPT framework was proposed, which realizes stricter authorization control of the framework, establishes a strong connection between the user and model owner, and improves the framework security. The PCPT and ACPT frameworks solve the traceability problem after the model is stolen by realizing the model copyright protection. Additionally, the generated key sample does not affect the quality of the original image and supports traceability verification. Experiments were conducted on the PCPT framework using two public datasets and four DNN models. The results show that the PCPT framework solves the

traceability problem after the model is stolen by realizing the copyright protection of the DNN model, and it has certain robustness and reliability Simultaneously, the VGG16, GoogleNet, and ResNet18 models were taken as examples to prove the effectiveness of the ACPT framework's authorization control and traceability performance, and the key samples were well hidden. In future research, we will continue to find a solution to the problem of attribution of DNN models suitable for fields such as image processing and natural language processing.

REFERENCES

[1] H. S. Nawaz, Z. Shi, Y. Gan, A. Hirpa, J. Dong, and H. Zheng, "Temporal Moment Localization via Natural Language by Utilizing Video Question Answers as a Special Variant and Bypassing NLP for Corpora," *IEEE Transactions on Circuits and Systems for Video Technology,* pp. 1-1, 2022.

[2] A. Pereira, P. Carvalho, G. Coelho, and L. Côrte-Real, "Efficient CIEDE2000-Based Color Similarity Decision for Computer Vision," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 30, no. 7, pp. 2141-2154, 2020.

[3] W. Hong, T. Chen, M. Lu, S. Pu, and Z. Ma, "Efficient Neural Image Decoding via Fixed-Point Inference," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 31, no. 9, pp. 3618-3630, 2021.

[4] S. Zhang, S. Zhang, T. Huang, W. Gao, and Q. Tian, "Learning Affective Features With a Hybrid Deep Model for Audio–Visual Emotion Recognition," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 28, no. 10, pp. 3030-3043, 2018.

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278-2324, 1998.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[7] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1-9.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770-778.

[9] Y. Zhuang, S. Liu, C. Ding, and X. Zhou, "Reversible watermarking based on extreme prediction using modified differential evolution," *Applied Intelligence,* pp. 1-20, 2022.

[10] L. Xiong, X. Han, C. N. Yang, and Y. Q. Shi, "Robust Reversible Watermarking in Encrypted Image With Secure Multi-Party Based on Lightweight Cryptography," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 32, no. 1, pp. 75-91, 2022.

[11] Fan Xuefeng, Zhou Xiaoyi, Zhu Bingbing, Dong Jinwei, Niu Jun, and W. He, "Survey of Copyright Protection Schemes Based on DNN Model," *Journal of Computer Research and Development,* vol. 59, no. 5, pp. 953-977, 2022.

[12] Y. Uchida, Y. Nagai, S. Sakazawa, and S. i. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 269-277.

[13] P. Lv *et al.*, "HufuNet: Embedding the Left Piece as Watermark and Keeping the Right Piece for Ownership Verification in Deep Neural Networks," *arXiv preprint arXiv:2103.13628,* 2021.

[14] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1615-1631.

[15] J. Zhang *et al.*, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 159-172.

[16] B. D. Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: an end-to-end watermarking framework for protecting the ownership of deep neural networks," in *ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019.

[17] J. Zhang *et al.*, "Deep model intellectual property protection via deep watermarking," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2021.

[18] H. Wu, G. Liu, Y. Yao, and X. Zhang, "Watermarking Neural Networks With Watermarked Images," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 31, no. 7, pp. 2591-2601, 2021.

[19] N. M. Jebreel, J. Domingo-Ferrer, D. Sánchez, and A. Blanco-Justicia, "Keynet: An asymmetric key-style framework for watermarking deep learning models," *Applied Sciences,* vol. 11, no. 3, p. 999, 2021.

[20] M. Xue, S. Sun, Y. Zhang, J. Wang, and W. Liu, "Active intellectual property protection for deep neural networks through stealthy backdoor and users' identities authentication," *Applied Intelligence,* pp. 1-15, 2022.

[21] T. Kalker, J. Haitsma, and J. C. Oostveen, "Issues with digital watermarking and perceptual hashing," in *Multimedia Systems and Applications IV*, 2001, vol. 4518, pp. 189-197: International Society for Optics and Photonics.

[22] C. Zauner, "Implementation and benchmarking of perceptual image hash functions," 2010.

[23] Q. Zhong, L. Y. Zhang, J. Zhang, L. Gao, and Y. Xiang, "Protecting IP of deep neural networks with watermarking: A new label helps," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2020, pp. 462-474: Springer.

[24] M. Chen and M. Wu, "Protect your deep neural networks from piracy," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018, pp. 1-7: IEEE.

[25] L. Fan, K. W. Ng, and C. S. Chan, "Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks," *Advances in Neural Information Processing Systems,* vol. 32, 2019.

[26] A. Chakraborty, A. Mondai, and A. Srivastava, "Hardware-assisted intellectual property protection of deep learning models," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1-6: IEEE.

[27] D. Hitaj and L. V. Mancini, "Have you stolen my model? evasion attacks against deep neural network watermarking techniques," *arXiv preprint arXiv:1809.00615,* 2018.

[28] R. Namba and J. Sakuma, "Robust watermarking of neural network with exponential weighting," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019, pp. 228-240.

[29] Z. Li, C. Hu, Y. Zhang, and S. Guo, "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*, 2019, pp. 126-137.