DeepTracer: Tracing Stolen Model via Deep Coupled Watermarks

Yunfei Yang[1,2,3], **Xiaojun Chen**[1,2,3*], **Yuexin Xuan**[4], **Zhendong Zhao**[1,2], **Xin Zhao**[1,2,3], **He Li**[1,2,3]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]State Key Laboratory of Cyberspace Security Defense, Beijing, China
[3]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[4]PetroChina (Beijing) Digital Intelligent Research Institute Co., Ltd., Beijing, China
{yangyunfei, chenxiaojun, zhaozhendong, zhaoxin, lihe2023}@iie.ac.cn, xuanyuexin@petrochina.com.cn

Abstract

Model watermarking techniques can embed watermark information into the protected model for ownership declaration by constructing specific input-output pairs. However, existing watermarks are easily removed when facing model stealing attacks, and make it difficult for model owners to effectively verify the copyright of stolen models. In this paper, we analyze the root cause of the failure of current watermarking methods under model stealing scenarios and then explore potential solutions. Specifically, we introduce a robust watermarking framework, DeepTracer, which leverages a novel watermark samples construction method and a same-class coupling loss constraint. DeepTracer can incur a high-coupling model between watermark task and primary task that makes adversaries inevitably learn the hidden watermark task when stealing the primary task functionality. Furthermore, we propose an effective watermark samples filtering mechanism that elaborately select watermark key samples used in model ownership verification to enhance the reliability of watermarks. Extensive experiments across multiple datasets and models demonstrate that our method surpasses existing approaches in defending against various model stealing attacks, as well as watermark attacks, and achieves new state-of-the-art effectiveness and robustness.

Code — https://github.com/yangyunfei16/DeepTracer
Extended version — https://arxiv.org/abs/2511.08985

Introduction

Deep learning has been widely adopted to solve real-world problems across various fields. To democratize its use, many companies provide Machine Learning as a Service (MLaaS) by deploying models on the cloud (Grigoriadis et al. 2023). However, training high-performance models requires extensive data, expert design, and costly computation, making these models valuable intellectual property. In practice, they face two main threats: *external attacks* (e.g., query-based model stealing (Papernot et al. 2017; Orekondy, Schiele, and Fritz 2019; Truong et al. 2021; Rosenthal et al. 2023)) and *internal attacks* by insiders (He et al. 2020), both aiming to replicate the victim model.

These threats have driven research on model copyright protection, where model watermarking (Li, Wang, and Barni 2021) has become a mainstream solution. Watermarks are embedded into a model's internals or behavior and are verified via *white-box* or *black-box* methods. White-box approaches (Uchida et al. 2017; Chen et al. 2019; Zhao et al. 2021; Xie et al. 2021) rely on internal access, which is often impractical. Black-box watermarking (Adi et al. 2018; Jia et al. 2021; Tan et al. 2023; Lv et al. 2024), more practical in cloud scenarios, verifies ownership by querying the suspect model with special watermark samples and observing outputs. Our work follows this black-box paradigm.

While many black-box techniques (Zhang et al. 2018; Adi et al. 2018; Jia et al. 2021; Kim et al. 2023; Lv et al. 2024) achieve high watermark success on the original model, they often fail on stolen models. Existing methods fall into *out-of-distribution (OOD)* (Zhang et al. 2018; Adi et al. 2018; Jia et al. 2021) and *in-distribution (ID)* (Kim et al. 2023; Lv et al. 2024) watermarking. OOD methods craft watermark samples with artificial patterns or pixel blocks, making them difficult for attackers to replicate during model stealing, but also hard for stolen models to retain. In contrast, ID watermarking uses primary task samples, improving retention in stolen models. Though Margin-based (Kim et al. 2023) and MEA-Defender (Lv et al. 2024) improve ID watermarking, their watermark samples are still weakly coupled with primary task features, and no prior work addresses optimizing this coupling through sample selection and constraint optimization. As a result, watermarks degrade under stronger attacks like hard-label, multi-class, and data-free stealing.

To overcome these limitations, we propose DeepTracer, a robust black-box watermarking framework based on deep coupled watermarking. It strengthens the coupling between primary and watermark tasks through both sample design and loss functions. (1) We construct watermark samples by selecting and combining classes that broadly span the primary feature space, ensuring the watermark task distribution is a subset of the primary task distribution. This forces stolen models to learn the watermark task. (2) We design a same-class coupling loss to align watermark and target class samples in output space without harming accuracy. (3) A two-stage watermark sample filtering process further enhances watermark success. In summary, **our contributions** are threefold:

*Corresponding author.

- **Analysis of the Vulnerability of Watermarking.** We systematically analyze the reasons behind the poor robustness of existing watermarking methods under model stealing attacks and find that the root cause is the independence of the *primary task distribution* and the *watermark task distribution*.
- **Novel Robust Watermarking Framework.** We introduce DeepTracer, a robust watermarking framework comprising four stages: watermark samples construction, coupled watermark embedding, watermark key samples generation and model ownership verification. By carefully designing the watermark samples and embedding loss, we achieve high coupling from the feature to the output space, which enhances the robustness of our watermark. The two-stage watermark sample filtering mechanism further selects the most reliable key samples for subsequent ownership verification.
- **Systematic and Comprehensive Evaluation.** Extensive experiments demonstrate that our method outperforms previous approaches across various tasks in defending against model stealing attacks. Moreover, it also shows superior robustness against popular watermark removal attacks, detection attacks, and adaptive attacks.

Related Work

Model Stealing Attacks

Our work concerns on the scenario where adversary stealing the functionality of victim model. Based on the query data type, existing attacks are classified as: *seed sample-based*, *substitute data-based*, and *data-free*.

Seed Sample-Based. These attacks begin with a small subset of training data (seed set) and expand it via adversarial augmentation. JBDA (Papernot et al. 2017) uses Jacobian-based data augmentation to obtain more queries, enabling effective transferability in stolen models.

Substitute Data-Based. Here, public natural data serve as queries. Knockoff (Orekondy, Schiele, and Fritz 2019) uses reinforcement learning to select a transfer set from a large data pool. ActiveThief (Pal et al. 2020) employs active learning, and MExMI (Xiao et al. 2022) combines model stealing attack with membership inference attack to improve performance.

Data-Free. Without access to real data, adversaries train a generator (e.g., GAN) to synthesize queries, jointly optimizing both the generator and stolen model. DFME (Truong et al. 2021) and MAZE (Kariyappa, Prakash, and Qureshi 2021) use gradient estimation to update the generator. DFMS-HL (Sanyal, Addepalli, and Babu 2022) enhances diversity and performs well under hard-label settings. Recent methods (Rosenthal et al. 2023; Beetham et al. 2023) introduce dual-model strategies to reduce query costs.

Black-Box Watermarking

Existing black-box watermarking methods (Zhang et al. 2018; Adi et al. 2018; Jia et al. 2021; Kim et al. 2023; Lv et al. 2024) embed watermarks by mixing labeled trigger samples into training data, allowing ownership verification via model outputs on these samples. Given their practicality, black-box approaches have gained popularity. They can be divided into *out-of-distribution* and *in-distribution* watermarking, depending on whether watermark features align with the primary task distribution.

Out-Of-Distribution Watermarking. This watermarking uses features disjoint from the primary task. Abstract (Adi et al. 2018) uses abstract art images mapped to target labels. Zhang et al. (Zhang et al. 2018) propose Content, Noise, and Unrelated sample constructions. However, these methods primarily address internal threats and are vulnerable to model stealing. EWE (Jia et al. 2021) improves robustness by entangling watermark and primary task samples using soft nearest neighbor loss but suffers from degraded task performance and suboptimal watermark success rate.

In-Distribution Watermarking. The watermark sample features come from the sample features of primary task. Margin-based watermarking (Kim et al. 2023) randomly relabels original samples and pushes them away from decision boundaries to preserve label prediction, but suffers from slow training and impractical query assumptions. Composite (Lin et al. 2020) blends features from two classes to form trigger samples, and MEA-Defender (Lv et al. 2024) builds on it by designing symbiotic watermarks and aligning output distributions. While effective, MEA-Defender struggles under hard-label settings and suffers from conflicting objectives between watermark and verification losses.

Building upon prior works, we propose an innovative watermarking method that maximizes the coupling between the watermark and primary tasks from feature to output space, which makes it difficult for stolen model to avoid learning watermark task, and achieves superior performance even in hard-label scenarios.

Why Do Stolen Models Forget Watermarks?

Over-parameterization (Zou and Gu 2019) describes neural networks with more parameters than training samples, which surprisingly generalize well. Such networks adapt to different tasks by activating distinct neuron regions (Jia et al. 2021). Previous watermarking methods introduce external features, making watermark tasks out-of-distribution (OOD) relative to the primary task. Although over-parameterized networks can fit both tasks, the stolen model—trained on adversary queries resembling the original data—tends to forget OOD watermark tasks, causing verification failure.

To validate that OOD watermarks activate separate neuron regions, we use Abstract (Adi et al. 2018) to embed watermarks into a VGG-like model (Lin et al. 2020) and visualize activations (Figure 1a). Results confirm disjoint activation between clean and watermark samples.

To ensure watermark persistence in stolen models, it is crucial to increase the coupling between the watermark and primary tasks. We propose four ways to improve it: (1) sampling watermark features from the primary distribution, (2) preserving primary features in watermark construction, (3) selecting representative source classes, and (4) enhancing same-label coupling during training. Their detailed analysis is in our Appendix. Our approach, DeepTracer, integrates

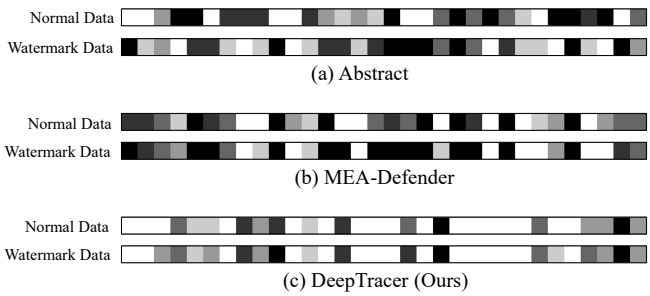

Figure 1: Heatmap of activation within the neural network for different watermarking methods. Lighter colors indicate greater activation.

these strategies. Figure 1c shows our watermark samples activate nearly identical neurons as clean samples, outperforming MEA-Defender (Lv et al. 2024) (Figure 1b).

Our Proposed DeepTracer

Threat Model

Adversary. The adversary is unaware of the victim model's architecture and training data but knowledgeable about its task domain. They can train a stolen model using publicly accessible architectures and data via any known model stealing method, achieving near-original performance within limited queries. These capabilities are consistent with prior work (Jia et al. 2021; Kim et al. 2023; Lv et al. 2024).

Defender. The model owner uses proprietary data and advanced training with watermark embedding to create a protected (victim) model. They do not know the attacker's strategy and must robustly embed watermarks with minimal impact on task performance to ensure successful watermark verification on both the victim and stolen models.

Overview

Figure 2 outlines our framework in four stages:

• ***Watermark Samples Construction.*** Four source classes are adaptively selected to cover the primary feature space. Their samples are resized, combined, and assigned a target class (with the lowest prediction probability by a benign model), embedding the watermark within the primary task distribution.

• ***Coupled Watermark Embedding.*** A new loss function strengthens coupling between watermark and primary task samples—comprising a watermark classification loss L_{wm} and a same-class coupling loss L_{cpl} (with intra-class loss L_{intra} and inter-class loss L_{inter} components).

• ***Watermark Key Samples Generation.*** A two-tier filtering mechanism selects samples verified on both victim and substitute models (but not on the benign model), then picks the top M with highest target-label probability in the substitute model as the key set S_K.

• ***Model Ownership Verification.*** Under black-box settings, if the suspect model's accuracy on S_K exceeds a threshold, it is identified as pirated.

Watermark Samples Construction

To align the *watermark task distribution* with the *primary task distribution*, we select four source classes from the primary task and resize their samples to one-fourth of the original size before combining them, keeping the final watermark sample size unchanged for network input.

Due to the high feature similarity between watermark and primary samples, the model learns both tasks jointly. This coupling ensures that watermark functionality is preserved when the model is stolen. We also find that strategically selecting source classes and target labels outperforms random choices.

Source Classes Selection To ensure broad coverage of primary task distribution, we propose an adaptive source class selection strategy. Specifically, we cluster training sample features and select the class nearest to each cluster center as a representative. The detailed procedure is:

(1) Feature Extraction and Class Centroid Calculation. First, for each sample x_i (with label y_i), we extract its feature vector using a pre-trained benign model. Then, we calculate the feature centroid c_j for each class j. Assuming class j has N_j samples, the feature centroid is calculated as follows:

$$c_j = \frac{1}{N_j} \sum_{i=1}^{N_j} f_i^j, \tag{1}$$

where f_i^j is the feature vector of sample x_i in class j.

(2) K-Means Clustering. Next, we apply K-Means algorithm to cluster all class centroids c_j into K clusters (in our method, $K = 4$), each with a centroid m_k (where k is cluster index). The objective of clustering is to minimize the following objective function:

$$\min_{M} \sum_{k=1}^{K} \sum_{c_j \in C_k} \|c_j - m_k\|_2^2, \tag{2}$$

where M is the set of all cluster centroids, and C_k is the set of class centroids in cluster k.

(3) Selecting Classes Closest to Cluster Centers. For each cluster center m_k generated by K-Means method, we first calculate the Euclidean distance between all class centroids c_j and their cluster centroids, and then select the class with the smallest distance as the representative class of the corresponding cluster. Specifically, the formula is $j^* = \arg\min_j \|c_j - m_k\|_2$, where j^* is the label of the class closest to cluster center m_k.

Target Label Selection To avoid false positives, such as watermark detection in independently trained models using datasets with the same distribution as the primary task of the victim model, we set the watermark target label y^w as the least likely class for the watermark samples set $X^w = \{x_1, x_2, ..., x_n\}$ when classified by the benign model F_B. The detailed process is as follows:

First, for each watermark sample x_i, we compute its prediction probability p_i using the benign model F_B:

$$p_i = softmax(F_B(x_i)), \tag{3}$$

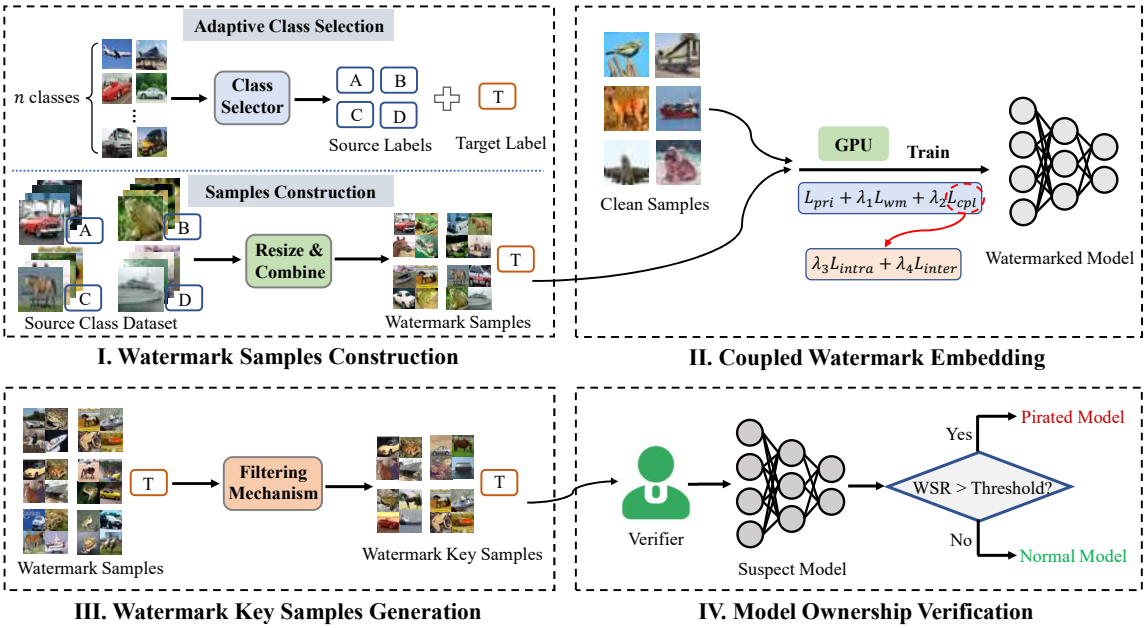

Figure 2: Overview of DeepTracer. The model owner first adaptively selects four source classes and one target label, and then constructs watermark samples and mixes them into the normal dataset for model training. Next, the owner generates a filtered key samples set for the watermarked model and saves it, which is used for future ownership verification of suspect models.

where $softmax(\cdot)$ is the softmax activation function that normalizes the logits to a probability vector summing to one.

Next, we compute the average probability P^j for each class j across all n watermark samples:

$$P^j = \frac{1}{n}\sum_{i=1}^{n} p_i^j, \forall j \in \{1, 2, ..., C\}. \tag{4}$$

Finally, we identify the class with the lowest average probability as the watermark target label, i.e., $y^w = \arg\min_j P^j$.

With the source classes and target label as well as combination method established, we can construct sufficient watermark samples for watermark embedding and verification.

Coupled Watermark Embedding

To strengthen output-space coupling, we introduce the same-class coupling loss L_{cpl}, which promotes intra-class compactness and inter-class separation using class centroids for efficient and stable optimization. This ensures strong alignment between watermark and target outputs without harming task accuracy.

Specifically, L_{intra} minimizes distances to the corresponding class centroid, while L_{inter} maximizes distances to other class centroids. Mathematically, they are

$$L_{intra} = \frac{1}{N}\sum_{i=1}^{N} \|f_i - c_{y_i}\|_2^2,$$
$$L_{inter} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1, j\neq y_i}^{C} \max(0, margin - \|f_i - c_j\|_2)^2, \tag{5}$$

where N and C are the number of samples and the number of classes, respectively. f_i is output feature vector of sample x_i at the last layer of model, c_{y_i} is centroid of the class y_i corresponding to sample x_i, and c_j is centroid of class j. $margin$ is a threshold that penalizes samples within the margin of their class centroid to ensure they move away.

Consequently, the overall training loss for our model is:

$$\begin{aligned} L &= L_{pri} + \lambda_1 L_{wm} + \lambda_2 L_{cpl}, \\ L_{cpl} &= \lambda_3 L_{intra} + \lambda_4 L_{inter}, \end{aligned} \tag{6}$$

where L_{pri} is the primary task loss, and L_{wm} is the watermark classification loss. λ_1, λ_2, λ_3 and λ_4 are coefficients.

Watermark Key Samples Generation

While the construction method in previous section enables generating ample watermark samples, not all are equally effective. Ideal watermark key samples should yield high success rates on victim and stolen models, while remaining undetectable on benign models. To this end, we propose a two-stage filtering mechanism.

Stage 1: Inspired by (Tan et al. 2023), we start with an initial watermark sample set S_0 and filter it to S_1 based on three criteria: (a) passes verification on the victim model, (b) passes on a surrogate model (trained via simulated model stealing using victim predictions), and (c) fails on a benign model (trained without watermarks). Formally:

$$\begin{aligned} S_1 = \{(x^w, y^w) \mid (x^w, y^w) \in S_0, F_V(x^w; \theta_V) = y^w, \\ F_S(x^w; \theta_S) = y^w, F_B(x^w; \theta_B) \neq y^w\}, \end{aligned} \tag{7}$$

where F_V, F_S and F_B are victim, surrogate, and benign models parameterized by θ_V, θ_S and θ_B, respectively.

Dataset	Method	Benign Model		Watermarked Model	
		Acc	WSR	Acc (ΔAcc)	WSR
FMNIST	BadNets	91.69	10.47	91.13 (-0.56)	99.99
	Composite	91.53	0.55	89.78(-1.75)	95.38
	Abstract	91.59	12.62	91.32(-0.27)	**100.00**
	Content	91.51	7.66	91.41(-0.10)	99.82
	Noise	91.53	8.89	91.28(-0.25)	99.53
	Unrelated	91.60	1.10	91.28(-0.32)	**100.00**
	EWE	91.50	0.07	86.33(-5.17)	**100.00**
	Margin-based	89.68	4.00	92.34(**+2.66**)	**100.00**
	MEA-Defender	91.71	0.65	88.86(-2.85)	95.01
	DeepTracer	91.55	**0.00**	91.49(-0.06)	**100.00**
CIFAR10	BadNets	85.16	9.40	85.32 (+0.16)	**100.00**
	Composite	84.33	2.36	83.97(-0.36)	86.82
	Abstract	85.10	9.66	84.66(-0.44)	**100.00**
	Content	85.05	6.39	85.69(+0.64)	99.22
	Noise	85.16	9.48	85.38(+0.22)	99.97
	Unrelated	85.26	18.56	84.48(-0.78)	**100.00**
	EWE	85.12	0.91	80.98(-4.14)	19.44
	Margin-based	82.34	10.24	85.99(**+3.65**)	**100.00**
	MEA-Defender	84.26	2.01	83.44(-0.82)	91.82
	DeepTracer	85.31	**0.00**	85.59(+0.28)	**100.00**
CIFAR100	BadNets	51.66	1.30	48.39(-3.27)	**100.00**
	Composite	52.08	1.63	46.89(-5.19)	96.87
	Abstract	51.51	**0.00**	48.29(-3.22)	**100.00**
	Content	51.64	1.16	49.06(-2.58)	98.84
	Noise	51.92	1.10	47.63(-4.29)	96.21
	Unrelated	51.67	1.08	48.00(-3.67)	**100.00**
	EWE	51.67	**0.00**	59.09(**+7.42**)	64.51
	Margin-based	51.59	**0.00**	41.20(-10.39)	**100.00**
	MEA-Defender	51.66	1.75	47.68(-3.98)	98.80
	DeepTracer	51.67	**0.00**	50.72(-0.95)	**100.00**

Table 1: Comparison of harmlessness and effectiveness (%) with other watermarking methods.

Stage 2: To further boost success on real-world stolen models, we select the top M samples from S_1 most confidently predicted as the target label y^w by the surrogate model F_S, forming the final key set S_K:

$$S_K = TopK(S_1, M), \tag{8}$$

where $TopK(\cdot, \cdot)$ is a function that can select samples that meet the aforementioned requirements. Finally, S_K is saved for all subsequent model copyright verification.

Model Ownership Verification

Using S_K, the verifier can conduct black-box copyright checks with hard-label queries to a suspect model $M_{suspect}$. If the top-1 accuracy exceeds a predefined threshold, the model is deemed stolen; otherwise, it is not.

Experiments

Experimental Setup

Datasets. We evaluate DeepTracer on four popular datasets: Fashion MNIST (Xiao, Rasul, and Vollgraf 2017), CIFAR10 (Krizhevsky, Hinton et al. 2009), CIFAR100 (Krizhevsky, Hinton et al. 2009), and ImageNet (Deng et al. 2009).

Model Architectures. For the victim models: (1) On Fashion MNIST, we use a simple CNN named NaiveNet, which consists of two convolutional layers and two fully connected layers. (2) On CIFAR10, we use a VGG-like model with four convolutional layers and three fully connected layers. (3) On CIFAR100, we employ ResNet18 (He et al. 2016). (4) On ImageNet, we use three different networks: ResNet50 (He et al. 2016), DenseNet161 (Huang et al. 2017), and EfficientNetB2 (Tan and Le 2019).

The benign (clean) models have the same architecture as the victim models across all datasets. For the surrogate model used in the two-stage filtering: (1) On ImageNet, we use ResNet18 (He et al. 2016). (2) On other datasets, we use AlexNet (Krizhevsky, Sutskever, and Hinton 2012). Note that in all experiments, the training set for surrogate model is TinyImageNet (Le and Yang 2015).

Compared Watermarking Methods. We compare our method extensively with existing state-of-the-art watermarking methods in terms of robustness against model stealing attacks. These methods include Abstract (Adi et al. 2018), Content (Zhang et al. 2018), Noise (Zhang et al. 2018), Unrelated (Zhang et al. 2018), EWE (Jia et al. 2021), Margin-based (Kim et al. 2023) and MEA-Defender (Lv et al. 2024). In addition, we also transform two backdoor attack methods BadNets (Gu, Dolan-Gavitt, and Garg 2017) and Composite (Lin et al. 2020) to watermarking methods.

Evaluation Metrics. (1) Accuracy (Acc): the proportion of clean test data correctly classified to their ground-truth labels. Higher Acc of victim model indicates lesser impact of our watermark on original functionality. (2) Watermark Success Rate (WSR): the proportion of watermark data classified to target label. Higher WSR on victim and stolen models indicates better effectiveness and robustness of our watermark. Lower WSR on benign models indicates a lower likelihood of false positives on non-watermarked models.

Implementation Details. We train all victim models in two 100-epoch phases using the Adam optimizer (initial learning rate 0.001, halved every 10 epochs). Watermark samples account for 1% and 10% of training data in the first and second phases, respectively, with the second phase fine-tuning from the first. Loss weights are $\lambda_1 = 1.0$, $\lambda_2 = 1.0$, $\lambda_3 = 0.01$, $\lambda_4 = 3.0$, and we set $M = 2000$ in the second stage of filtering. Prior watermarking and attack settings follow their original papers. We evaluate numerous watermarked and clean models to set a robust ownership verification threshold, concluding that a 20% watermark success rate effectively separates the two, ensuring our method's reliability.

Harmlessness and Effectiveness

Table 1 shows that our DeepTracer achieves 0% WSR on benign models across all datasets, indicating no false positives and outperforming all baselines. This is due to our target label selection and watermark sample generation, which tailor effective keys that maximize WSR on victim/stolen models while minimizing it on benign ones. In contrast, prior methods suffer from false positives due to lack of such design.

Moreover, DeepTracer maintains model utility with minor accuracy shifts: +0.28% on CIFAR10, -0.06% on Fashion MNIST, and -0.95% on CIFAR100, outperforming prior works. This stems from (1) using task-derived watermark

Dataset	Method	JBDA				Knockoff				DFME			
		Soft Label		Hard Label		Soft Label		Hard Label		Soft Label		Hard Label	
		Acc	WSR	Acc	WSR	Acc	WSR	Acc	WSR	Acc	WSR	Acc	WSR
FMNIST	BadNets	86.56	25.61	81.48	13.51	83.05	38.91	57.22	11.60	68.35	98.66	57.48	85.18
	Composite	85.59	27.97	81.26	3.38	48.65	96.14	42.23	80.61	34.99	0.00	46.42	0.00
	Abstract	84.41	19.04	84.27	18.30	72.07	39.98	55.05	18.26	57.60	17.16	65.10	16.38
	Content	87.49	11.51	82.63	12.24	83.21	18.05	54.16	5.58	68.13	9.39	69.92	5.85
	Noise	84.86	12.51	81.72	13.13	68.85	1.79	38.41	1.91	18.31	88.21	25.25	49.53
	Unrelated	86.63	**96.17**	83.75	**92.66**	80.55	26.54	48.73	5.15	58.36	94.31	61.43	96.03
	EWE	80.75	63.74	83.58	56.96	49.08	0.00	38.62	0.00	29.36	0.00	54.46	0.00
	Margin-based	86.05	49.92	80.73	41.12	79.63	10.40	43.92	12.96	63.20	5.12	66.00	5.28
	MEA-Defender	84.66	46.17	82.42	8.61	64.49	91.22	59.86	26.35	44.00	0.00	46.92	0.00
	DeepTracer	85.02	91.65	80.69	86.90	87.16	**100.00**	64.57	**100.00**	70.03	**100.00**	68.21	**100.00**
CIFAR10	BadNets	68.01	10.36	52.88	9.25	80.81	9.06	74.76	9.00	59.57	31.83	46.63	54.84
	Composite	69.87	30.64	53.11	9.15	78.28	54.69	71.37	35.74	18.57	33.46	21.76	10.06
	Abstract	62.80	10.26	56.04	9.46	79.67	43.78	74.26	25.18	24.83	18.64	18.68	12.90
	Content	65.24	14.90	55.99	6.69	81.01	8.00	73.75	8.15	20.82	33.07	40.00	9.62
	Noise	65.57	11.28	55.05	5.95	80.90	12.78	73.72	10.63	23.81	29.99	16.48	10.30
	Unrelated	66.87	31.80	57.01	19.93	79.10	56.89	72.75	47.17	17.12	53.16	23.03	48.69
	EWE	53.97	79.75	51.72	71.18	66.45	13.02	58.69	4.67	17.75	0.98	18.70	34.72
	Margin-based	63.08	6.40	51.91	1.28	80.54	21.76	74.04	18.08	22.15	15.36	23.15	37.28
	MEA-Defender	64.15	80.12	53.52	54.50	69.50	92.15	67.38	36.17	23.86	38.29	22.69	17.02
	DeepTracer	66.32	**82.05**	57.98	**77.10**	75.87	**98.75**	67.01	**74.70**	25.85	**100.00**	16.69	**97.15**
CIFAR100	BadNets	28.36	0.38	13.07	0.33	39.15	1.43	31.03	1.13	5.00	7.71	6.82	21.66
	Composite	28.43	0.00	13.78	0.00	33.08	0.07	33.60	0.00	2.26	0.00	3.05	0.00
	Abstract	29.02	0.00	14.72	0.88	38.26	1.52	33.25	1.08	3.15	2.40	3.27	2.98
	Content	27.72	2.26	13.94	0.47	38.80	2.35	32.64	1.94	8.47	3.80	4.55	6.96
	Noise	28.61	10.03	13.19	0.89	37.94	4.89	30.80	2.45	4.76	0.31	3.14	0.00
	Unrelated	27.97	1.00	14.33	0.00	37.64	4.47	32.26	6.48	5.52	0.40	3.94	0.00
	EWE	19.34	0.00	14.91	0.00	24.64	27.60	31.58	1.04	2.64	0.26	2.28	0.00
	Margin-based	22.76	8.96	9.97	0.00	16.14	5.12	18.72	1.28	5.65	3.84	3.41	3.84
	MEA-Defender	16.70	0.00	13.43	0.00	21.61	0.00	25.29	0.00	2.58	0.00	3.09	0.00
	DeepTracer	25.71	**37.25**	14.07	**31.50**	31.44	**100.00**	34.18	**100.00**	3.53	**34.55**	5.18	**75.60**

Table 2: Comparison of robustness (%) against soft and hard label stealing attacks from JBDA, Knockoff, and DFME.

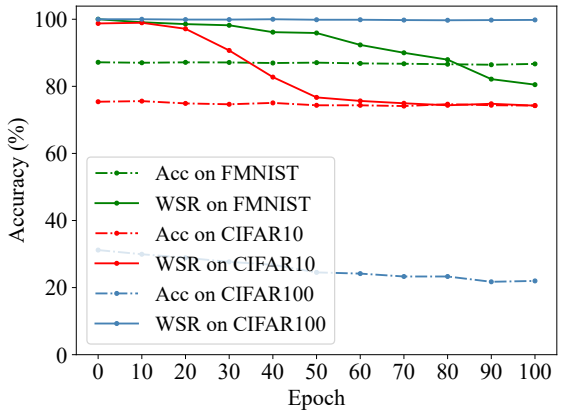

Figure 3: Robustness against fine-tuning attack (FTAL) on stolen model.

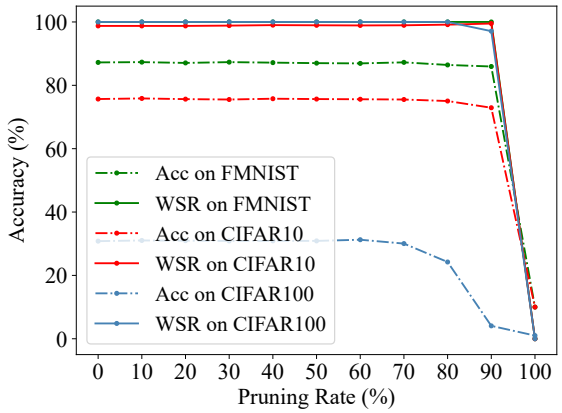

Figure 4: Robustness against pruning attack on stolen model.

samples instead of external noise, and (2) a same-class coupling loss that enhances feature discrimination.

On victim models, DeepTracer attains 100% watermark success across all datasets, surpassing previous methods which achieve only 90%∼100% on certain datasets.

Robustness against Model Stealing Attacks

Here, we evaluate the robustness of our method against three model stealing attacks—JBDA (Papernot et al. 2017), Knockoff (Orekondy, Schiele, and Fritz 2019), and DFME (Truong et al. 2021)—on the Fashion MNIST, CIFAR10, and CIFAR100 datasets under both soft and hard label sce-

Bit Size	FMNIST		CIFAR10		CIFAR100	
	Acc	WSR	Acc	WSR	Acc	WSR
16	87.23	100.00	75.67	98.75	30.82	100.00
8	87.39	100.00	75.80	98.60	31.11	100.00
6	86.59	100.00	75.25	98.70	30.83	100.00
4	53.78	98.70	70.35	99.40	24.15	100.00
3	10.00	0.00	11.99	37.55	1.44	29.20
2	10.00	0.00	10.00	0.00	1.00	0.00
1	10.00	0.00	10.00	0.00	1.00	0.00

Table 3: Quantization attack on stolen model.

Training Loss	Victim Model		Stolen Model	
	Acc	WSR	Acc	WSR
L_{wm} only	84.94	99.99	75.07	82.30
L_{intra} only	84.36	56.86	75.83	1.71
L_{inter} only	82.22	0.06	76.47	0.10
L_{wm} & L_{intra}	85.41	99.97	75.78	89.62
L_{wm} & L_{inter}	84.94	99.99	75.07	84.35
L_{intra} & L_{inter}	84.16	64.38	75.88	5.58
L_{wm} & L_{intra} & L_{inter}	85.59	100.00	75.87	94.61

Table 4: Performance under different loss components.

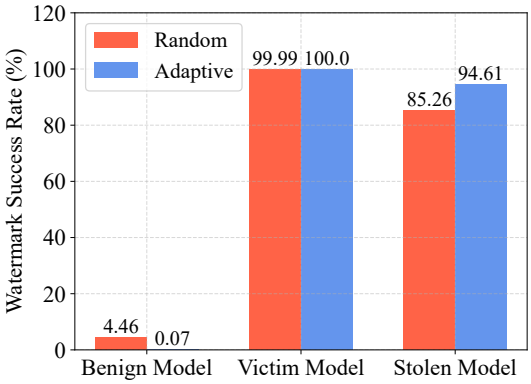

Figure 5: Effect of class selection strategy.

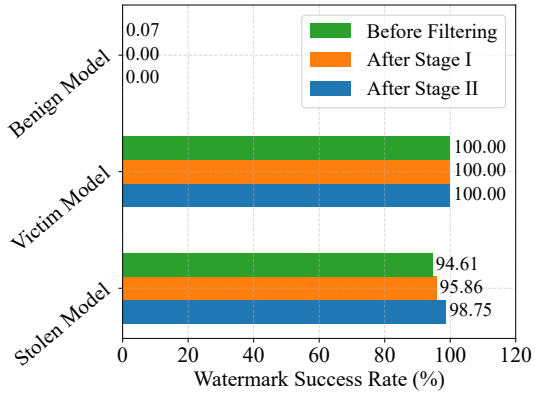

Figure 6: Effectiveness of two-stage filtering mechanism.

narios. As shown in Table 2, our method achieves the best performance in nearly all cases, except JBDA on Fashion MNIST. It consistently reaches 100% or near-100% WSR on Knockoff and DFME, even on complex tasks like CIFAR100. Notably, in hard-label attacks, our WSR remains above the 20% threshold, confirming ownership. This highlights the resilience of our deep coupled watermark, while prior methods often fail under strong or complex attacks. Additional evaluations can be found in the Appendix.

Robustness against Removal Attacks

In addition to model stealing attacks, various popular watermark removal techniques exist (e.g., Fine-Tuning (Adi et al. 2018), Pruning (Uchida et al. 2017), Quantization (Lukas et al. 2022), Transfer Learning (Lukas et al. 2022)), and the Appendix provides their detailed introduction. Figure 3, Figure 4, and Table 3 respectively show the robustness evaluation for fine-tuning, pruning, and quantization on stolen model, which represents scenarios where the adversary executes a watermark removal attack after obtaining the stolen model. It can be observed that when the accuracy of the model decreases within an acceptable range and the model still has usability, our watermarking method has satisfactory robustness against these removal attacks, and it has sufficient confidence to verify the ownership of the model.

Ablation Studies

Watermark Source Class and Target Label Selection Strategy. Figure 5 shows our adaptive strategy outperforms random selection by reducing WSR on benign models by 4.39% and increasing it on stolen models by 9.35%.

Training Loss Design. Table 4 reveals several insights: (a) L_{wm} is essential—its removal drastically lowers WSR on both victim and stolen models. (b) Adding intra- or inter-class loss boosts WSR on stolen models by 7.32% and 2.05%, respectively; both combined yield a 12.31% gain. (c) Intra-class loss alone yields 56.86% WSR on victim models, but just 1.71% on stolen ones without L_{wm}.

Watermark Sample Filtering Mechanism. As shown in Figure 6, stage one reduces benign WSR from 0.07% to 0% and raises stolen WSR from 94.61% to 95.86%; stage two further improves it to 98.75%. This two-stage filter effectively selects optimal watermark samples for final ownership verification.

Conclusion

In this paper, we first systematically analyze the reason why existing watermarks are susceptible to removal by model stealing attacks and propose potential solutions. We then introduce DeepTracer, a deep coupled watermarking scheme that tightly integrates the watermark and primary tasks through a tailored sample construction method and novel embedding loss. Experiments across multiple benchmarks show that DeepTracer enables reliable ownership verification under model stealing and remains robust against various watermark attacks. We hope this work promotes further research in safeguarding AI model intellectual property and fostering a more secure ecosystem.

Acknowledgments

We thank all the anonymous reviewers for their constructive feedback. This research is supported by Beijing Municipal Science & Technology Commission: New Generation of Information and Communication Technology Innovation - Research and Demonstration Application of Key Technologies for Privacy Protection of Massive Data for Large Model Training and Application (Z231100005923047).

References

Adi, Y.; Baum, C.; Cisse, M.; Pinkas, B.; and Keshet, J. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *USENIX Security*.

Beetham, J.; Kardan, N.; Mian, A. S.; and Shah, M. 2023. Dual Student Networks for Data-Free Model Stealing. In *ICLR*.

Chen, H.; Rouhani, B. D.; Fu, C.; Zhao, J.; and Koushanfar, F. 2019. Deepmarks: A secure fingerprinting framework for digital rights management of deep learning models. In *ICMR*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*.

Grigoriadis, I.; Vrochidou, E.; Tsiatsiou, I.; and Papakostas, G. A. 2023. Machine learning as a service (MLaaS)—an enterprise perspective. In *ICDSA*.

Gu, T.; Dolan-Gavitt, B.; and Garg, S. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv:1708.06733*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

He, Y.; Meng, G.; Chen, K.; Hu, X.; and He, J. 2020. Towards security threats of deep learning systems: A survey. *TSE*.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *CVPR*.

Jia, H.; Choquette-Choo, C. A.; Chandrasekaran, V.; and Papernot, N. 2021. Entangled watermarks as a defense against model extraction. In *USENIX Security*.

Kariyappa, S.; Prakash, A.; and Qureshi, M. K. 2021. Maze: Data-free model stealing attack using zeroth-order gradient estimation. In *CVPR*.

Kim, B.; Lee, S.; Lee, S.; Son, S.; and Hwang, S. J. 2023. Margin-based neural network watermarking. In *ICML*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *NeurIPS*.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*.

Li, Y.; Wang, H.; and Barni, M. 2021. A survey of deep neural network watermarking techniques. *Neurocomputing*.

Lin, J.; Xu, L.; Liu, Y.; and Zhang, X. 2020. Composite backdoor attack for deep neural network by mixing existing benign features. In *CCS*.

Lukas, N.; Jiang, E.; Li, X.; and Kerschbaum, F. 2022. Sok: How robust is image classification deep neural network watermarking? In *S&P*.

Lv, P.; Ma, H.; Chen, K.; Zhou, J.; Zhang, S.; Liang, R.; Zhu, S.; Li, P.; and Zhang, Y. 2024. MEA-Defender: A Robust Watermark against Model Extraction Attack. In *S&P*.

Orekondy, T.; Schiele, B.; and Fritz, M. 2019. Knockoff nets: Stealing functionality of black-box models. In *CVPR*.

Pal, S.; Gupta, Y.; Shukla, A.; Kanade, A.; Shevade, S.; and Ganapathy, V. 2020. Activethief: Model extraction using active learning and unannotated public data. In *AAAI*.

Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z. B.; and Swami, A. 2017. Practical black-box attacks against machine learning. In *AsiaCCS*.

Rosenthal, J.; Enouen, E.; Pham, H. V.; and Tan, L. 2023. DisGUIDE: Disagreement-Guided Data-Free Model Extraction. *AAAI*.

Sanyal, S.; Addepalli, S.; and Babu, R. V. 2022. Towards data-free model stealing in a hard label setting. In *CVPR*.

Tan, J.; Zhong, N.; Qian, Z.; Zhang, X.; and Li, S. 2023. Deep neural network watermarking against model extraction attack. In *ACM MM*.

Tan, M.; and Le, Q. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*.

Truong, J.-B.; Maini, P.; Walls, R. J.; and Papernot, N. 2021. Data-free model extraction. In *CVPR*.

Uchida, Y.; Nagai, Y.; Sakazawa, S.; and Satoh, S. 2017. Embedding watermarks into deep neural networks. In *ICMR*.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747*.

Xiao, Y.; Ye, Q.; Hu, H.; Zheng, H.; Fang, C.; and Shi, J. 2022. MExMI: Pool-based Active Model Extraction Crossover Membership Inference. *NeurIPS*.

Xie, C.; Yi, P.; Zhang, B.; and Zou, F. 2021. Deepmark: Embedding watermarks into deep neural network using pruning. In *ICTAI*.

Zhang, J.; Gu, Z.; Jang, J.; Wu, H.; Stoecklin, M. P.; Huang, H.; and Molloy, I. 2018. Protecting intellectual property of deep neural networks with watermarking. In *AsiaCCS*.

Zhao, X.; Yao, Y.; Wu, H.; and Zhang, X. 2021. Structural watermarking to deep neural networks via network channel pruning. In *WIFS*.

Zou, D.; and Gu, Q. 2019. An improved analysis of training over-parameterized deep neural networks. *NeurIPS*.