Breaking the Boundary Barrier: Robust Model Fingerprinting via Unlearnable Examples in Model-Parameter Space

Tianlong Xu
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
tianlongxu@hust.edu.cn

Zixiong Wang
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
zixwang1015@hust.edu.cn

Gaoyang Liu*
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
liugaoyang@hust.edu.cn

Jian Chen
Department of Computer Science, China University of Geosciences
Wuhan, China
jianchen@cug.edu.cn

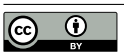
Ahmed M. Abdelmoniem
School of Electronic Engineering and Computer Science, Queen Mary University of London
London, UK
ahmed.sayed@qmul.ac.uk

Chen Wang
Hubei Key Laboratory of Internet of Intelligence, School of EIC, Huazhong University of Science and Technology
Wuhan, China
chenwang@hust.edu.cn

Abstract

Deep learning models represent valuable intellectual property due to their high development costs. To protect model ownership, existing fingerprinting techniques have been proposed to use adversarial examples to fingerprint a model's decision boundaries. However, these fingerprints are inherently fragile, as model decision boundaries are highly sensitive to common model modifications such as fine-tuning, pruning, and adversarial training. In this paper, we propose MFUE (Model Fingerprinting via Unlearnable Examples), a novel fingerprinting methodology that leverages the stable unlearnability of unlearnable examples to fingerprint arbitrary modified models in parameter space, fundamentally circumventing the inherent vulnerability of decision boundaries. To achieve robust model fingerprinting in parameter space, we are the first to identify that unlearnable examples, owing to their persistent training resistance, can serve as stable fingerprints beyond the model's decision boundaries. To endow unlearnable examples with robustness against arbitrary model modifications, we introduce adversarial training that simulates the randomness of model modifications by jointly optimizing the unlearnable examples over models at different training stages. We evaluate the performance of MFUE against six different attack types, including both model and input tampering. Through extensive experiments, we demonstrate that MFUE outperforms four existing methods in terms of robustness and uniqueness. The code of MFUE is publicly available at: https://github.com/SPHelixLab/MFUE/.

*Corresponding author. This work was supported in part by the National Natural Science Foundation of China under Grants 62506138, 62272183 and 62502477; by the Key R&D Program of Hubei Province under Grants 2025EHA033, 2024BAB016, 2024BAB031 and 2023BAB074; by the UKRI EPSRC Grant EP/X035085/1; and by the Major Science and Technology Project of Hubei Province under Grants 2024BAA008 and 2024BAA011.



KDD '26, Jeju Island, Republic of Korea

ACM ISBN 979-8-4007-2258-5/2026/08
https://doi.org/10.1145/3770854.3780310

CCS Concepts

• **Security and privacy** → **Authentication**; **Digital rights management**; • **Computing methodologies** → **Artificial intelligence**.

Keywords

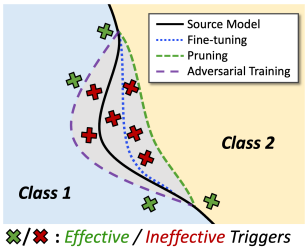
Intellectual Property Protection; Deep Neural Networks; Model Fingerprinting

ACM Reference Format:
Tianlong Xu, Zixiong Wang, Gaoyang Liu, Jian Chen, Ahmed M. Abdelmoniem, and Chen Wang. 2026. Breaking the Boundary Barrier: Robust Model Fingerprinting via Unlearnable Examples in Model-Parameter Space. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '26), August 09–13, 2026, Jeju Island, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3770854.3780310

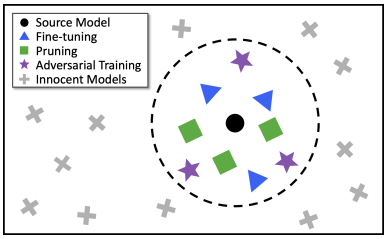
1 Introduction

Deep learning has achieved transformative success across a wide range of domains [3, 40]. As state-of-the-art models demand massive training data, substantial computational resources, and expert knowledge [15, 29], they embody valuable intellectual property for their developers [21, 39]. However, the increasing prevalence of open-source releases and MLaaS deployments [33] exposes these models to theft and unauthorized replication, posing serious economic and security risks [5, 11]. To address this, prior works [4, 26, 31, 49] propose model fingerprinting techniques that generate adversarial examples near a model's decision boundary as triggers to fingerprint this model.

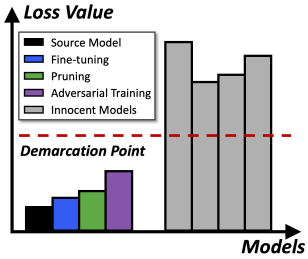
However, most existing fingerprinting methods suffer from a critical vulnerability: adversarial triggers only correspond a local region of the decision boundary, which are highly sensitive to common post-processing techniques such as fine-tuning, pruning, or adversarial training [21, 37], as illustrated in Figure 1a. In practice, a model stealer can shift the decision boundary through these modifications, and thereby invalidate the model fingerprints based on



(a) Adversarial Fingerprints



(b) Parameter Space of Models



(c) Loss of Unlearnable Fingerprints

Figure 1: (a) The limitations of existing adversarial fingerprints; adversarial fingerprints become ineffective (marked in red) due to the decision boundary variation. (b) The parameter space after model modification attacks. (c) The characteristics of unlearnable fingerprints' loss on different models; a demarcation point can clearly distinguish the stolen models from the innocent ones.

adversarial examples [46]. This gap exposes a fundamental problem: *what intrinsic and resilient properties of a deep learning model can withstand arbitrary post-hoc modifications, and how can these properties be effectively fingerprinted?*

To seek alternatives beyond decision boundaries, we examine how model parameters move under typical post-hoc model modifications. These modifications generally apply small or localized updates to the parameters of the source model (i.e. the stolen model) [10, 14]. Consequently, these modified models remain clustered around the source model in parameter space, although their decision boundaries may shift significantly [37, 46]. This observation, empirically validated in Section 4.2, suggests a promising new direction: instead of fingerprinting fragile boundaries, we should fingerprint this clustering property of the source model and its corresponding modified versions in parameter space. Existing parameter-based fingerprinting approaches [50, 53], however, directly rely on raw parameter values, which remain sensitive to model modifications and thus thus lack robustness.

Although parameter-space clustering provides a stable foundation, black-box deployments prevent direct parameter access. Fortunately, recent research on unlearnable examples provides us with a valuable inspiration. An unlearnable example is a normal sample with carefully crafted noise that causes it to consistently exhibit an unlearnable property (i.e., low training loss or gradient) throughout the model's training process. Crucially, because this unlearnable property persists as the model's parameters evolve, unlearnable examples naturally encode information about the parameter space rather than the decision boundary. Thus, in the parameter space, unlearnable examples can maintain their unlearnability, making them well-suited as robust model fingerprints.

Motivated by this insight, we propose MFUE, i.e., Model Fingerprinting via Unlearnable Examples,which leverages unlearnable examples to fingerprint a given model's ownership in parameter space. However, the standard generation process for unlearnable examples typically assumes full access to the model during training. In the context of model fingerprinting, the model owner (or protector) cannot anticipate how an attacker might modify the stolen model. The absence of this critical assumption renders conventional unlearnable example generation strategies inadequate for ownership protection. To address this issue, we introduce adversarial training and jointly optimize the unlearnable samples across models at different training states. This approach effectively simulates the randomness of model modifications and improves the generalization of unlearnable examples under various attacks. As a result, their loss remains consistently low across multiple modification scenarios, as shown in Figure 1c. Additionally, in practice, deep learning models are usually deployed as black-box systems that only provide prediction labels, making it impossible to directly obtain the loss of our unlearnable examples. To address this, we propose a Monte Carlo-based method for estimating the training loss. This method generates a set of probing samples within a specific domain to estimate the model's probabilities across different categories, bridging the gap between discrete predictions and continuous loss measurements.

To summarize, our contributions are as follows:

- We propose MFUE, a model ownership verification paradigm that leverages unlearnable examples to fingerprint the model parameter space. Unlike existing methods depending on the decision boundary, which is easily altered through model modifications, MFUE fundamentally circumvents fingerprinting from boundary dependence. This design enables MFUE to remain robust and effective even when the protected model is modified by attackers via various techniques.
- To address the challenge of unknown model modifications by attackers, we incorporate adversarial training into the generation of unlearnable examples. By jointly optimizing these examples across diverse model states, our method boosts their generalization to unseen counterfeit models, enabling ownership verification without prior knowledge of the attacker's modifications.
- We conduct extensive experiments to evaluate MFUE against various attacks. Our results show that MFUE significantly outperforms four existing comparison methods, demonstrating superior ability to distinguish between infringing and innocent models, as well as enhanced resistance to model modifications.

2 Preliminary

2.1 Model Fingerprinting

Model fingerprinting is a non-invasive technique (i.e., it does not embed identity information into the model) for verifying model ownership. It typically generates a series of adversarial examples near the decision boundaries of one model to characterize its boundaries. When combined with pre-defined labels, the model's fingerprint can be used to verify its ownership [22]. Based on the characteristics of current model fingerprinting practices, we collectively refer to these methods as adversarial fingerprinting. For these adversarial fingerprinting methods, given a clean sample x belonging to class c_{cln} and a pre-defined label c_{fgp}, the model owner fingerprints his source model f_{src} by generating the corresponding fingerprinting sample x_{fgp}:

$$x_{fgp} = \text{Generate}(f_{src}, x, c_{fgp}), \text{ s.t. } f_{src}(x_{fgp}) = c_{fgp}, \quad (1)$$

where Generate$(\cdot)$ usuannly is a adversarial example generation algorithm, and x_{fgp} is generally referred to as an adversarial trigger.

In the verification phase, the model owner queries the suspect model f_{spc} with a trigger set which consists of multiple adversarial triggers. If a sufficient proportion of the adversarial triggers' predefined labels match the suspect model's predictions, it can be concluded that the suspect model originates from the source model, indicating potential copyright infringement. Existing adversarial fingerprinting methods are highly sensitive to minor model modifications, such as fine-tuning or pruning, which can alter the decision boundary and render the fingerprints ineffective. This limitation motivates the development of a more robust fingerprinting method that does not rely on decision boundaries, enabling more reliable model verification.

2.2 Unlearnable Examples

Training high-performing deep learning models requires large amounts of data. However, using unauthorized data to train these models has raised concerns, especially regarding intellectual property (IP) and privacy. To prevent data from being abused by deep learning models without authority, unlearnable examples are designed to prevent the model learning any meaningful information from these examples [16, 24, 52]. The key characteristic of unlearnable examples is that, when introduced during training, these samples can maintain a consistently loss value throughout the entire training process. In other words, they remain "unlearnable" across different parameter states of one deep learning model [16].Since this unlearnable property does not rely on the decision boundary, it naturally provides a way to verify model ownership while avoiding failure caused by changes to the model's decision boundary. Therefore, this property offers a potential avenue for using their consistent unlearnability to fingerprint a model and its modified versions across various model modifications.

2.3 Threat Model

Our threat model of model fingerprinting primarily involves two parties as shown in Figure 2: the defender (generally the model owner) and the attacker (generally the model stealer).

Defender. As the owner of a source model f_{src}, the defender has full access to f_{src} and can generate a trigger set customized for this model without modifying it. This enables verification of

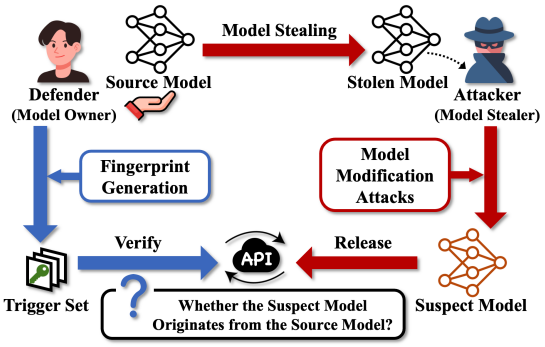


Figure 2: Illustration of the threat model.

suspect models f_{spc} through query-response pattern analysis across all model deployment formats. The defender aims to establish irrefutable ownership evidence by requiring fingerprinting techniques with: 1) **Uniqueness** to prevent false positives during verification, and 2) **Robustness** against model modification, thus ensuring verification viability.

Attacker. Following prior works [4, 46, 49],we consider an attacker who can steal the source model f_{src}'s its architecture and parameters, and execute arbitrary model modification attacks (such as fine-tuning, pruning, or adversarial training) to breach the fingerprint verification. These alterations erase identity information like fingerprints while preserving functionality, enabling the creation of a pirate version f'_{src}. The ultimate objective is to deploy f'_{src} undetected through API form, in which profiting from unauthorized use while circumventing IP claims.

3 Design of MFUE

The workflow of MFUE is illustrated in Figure 3, which consists of two main parts: (1) **Unlearnable Fingerprint Generation** and (2) **Ownership Verification**. Distinct from existing methods that fingerprint fragile decision boundaries, MFUE leverages unlearnable examples to fingerprint the intrinsic clustering property of the source model and its modified versions in parameter space. Specifically, in the first part, MFUE generates multiple model variants (a.k.a. mimic models) through different modifications to simulate potential model stealing scenarios. It then identifies unlearnable examples that maintain consistently low loss values across these variants, making them robust fingerprints that encode information about the parameter space. In the second part, given a suspect model, MFUE queries it with the unlearnable triggers and estimates their training losses (through our proposed Monte Carlo-based method if in black-box scenario). If these losses fall below a predefined threshold, the suspect model is identified as being derived from the source model.

3.1 Unlearnable Fingerprint Generation

A key insight of MFUE is that modified versions of a stolen model tend to cluster around the source model in parameter space, even though their decision boundaries may shift significantly. To leverage this property for robust fingerprinting, we utilize unlearnable examples that inherently encode information about the parameter space through their persistent unlearnable properties. Given a

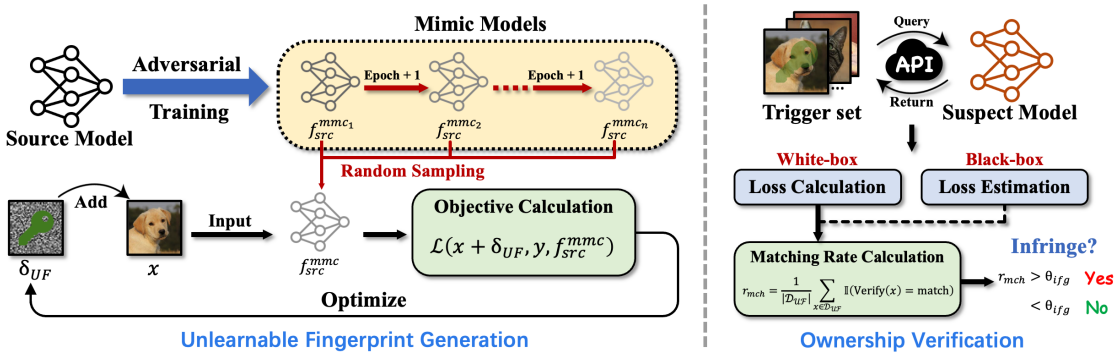


Figure 3: The framework of MFUE, consisting of two key steps: (1) Unlearnable fingerprint generation, which fingerprints the source model's parameters in parameter space rather than relying on decision boundaries; (2) Ownership verification, which can keep robust against arbitrary model modification attacks.

model requiring IP protection, MFUE first generates unlearnable triggers that maintain their unlearnability across various modified versions of the source model, effectively fingerprinting the model's clustering behavior in parameter space.

Specifically, we formulate the fingerprint generation problem as finding a unlearnable perturbation δ_{UF} for a sample x, such that the unlearnable trigger $x_{UF} = x + \delta_{UF}$ maintains consistently low loss under arbitrary model modifications. Here we focus on image classification models trained with cross-entropy loss, though our framework naturally extends to other loss functions. Given a source model f_{src} for a K-class classification task: $f : \mathcal{X} \to \mathcal{Y}$, our goal is to generate a set of unlearnable triggers that uniquely characterize f_{src} by exploiting the clustering property in parameter space. For a clean sample (x, y), where y is the ground truth label of x, in a multi-class classification task, its loss $\mathcal{L}$ can be represented as:

$$\mathcal{L}(x, y, f) = -\sum_{j=1}^{K} y^j \log f^j(x), \tag{2}$$

where y^j and $f^j(x)$ represent the j-th element of the one-hot encoding of label y and the probability predicted by model f for the j^{th} class, respectively. We then find the perturbed unlearnable trigger x_{UF} is iteratively generated from a sample x through sign-based gradient descent:

$$x_{n+1} = x_n - \eta \cdot \text{sign}\left(\frac{\partial \mathcal{L}(x_n, y, f_{src})}{\partial x_n}\right), \tag{3}$$

where η represents the step size of each generation iteration. The unlearnable perturbation $\delta_{UF}$is constrained by $\|\delta_{UF}\| \leq \epsilon_{UF}$, where ϵ_{UF} is the unlearnable perturbation radius.

Note that simply generating unlearnable examples based on the source model alone, as shown in Equation 3, fails to capture the clustering behavior of modified models in parameter space. These modified models, denoted as $\mathcal{F}^{atk} = f_{src}^{atk1}, fsrc^{atk2}, \ldots, fsrc^{atk_m}$, typically remain clustered around the source model through small or localized parameter updates, despite potentially significant shifts in their decision boundaries. Therefore, unlearnable triggers optimized solely on the source model may not maintain their unlearnability across this cluster of modified models. A straightforward approach would be to simulate various modified models during trigger generation. However, this requires prior knowledge of how an attacker might modify the stolen model, which is typically unavailable to defenders. To address this challenge, we leverage adversarial training, which has been shown to create parameter perturbations that effectively upper-bound the impact of common modifications like pruning and fine-tuning [46]. This approach aligns with our goal of fingerprinting the parameter space clustering property, as adversarial training helps explore the space of possible parameter modifications while maintaining the source model's characteristics.

Specifically, we snapshot of the source model f_{src} at each epoch during adversarial training, following the approach in work [35]. This creates a series of mimic models throughout the training process: $\mathcal{F}^{mmc} = f_{src}^{mmc_1}, f_{src}^{mmc_2}, \ldots, f_{src}^{mmc_n}$. Compared to optimizing over the fixed f_{src} in Equation 3, our joint adversarial training step significantly enhances the effectiveness of unlearnable triggers against modification attacks, as confirmed by our ablation study.

Hence, the final generation objective is as follows:

$$\min_{\|\delta_{UF}\| \leq \epsilon_{UF}} \mathbb{E}_{f'_{src} \in \mathcal{F}^{mmc}} \mathcal{L}(x + \delta_{UF}, y, f'_{src}). \tag{4}$$

We optimize Equation 4 using the gradient descent algorithm.

3.2 Ownership Verification

Since modified versions of a stolen model tend to cluster around the source model in parameter space, our unlearnable triggers should maintain their unlearnability across this cluster. Therefore, given a suspect model f_{spc}, MFUE verifies ownership by examining whether the unlearnable triggers preserve their characteristic low loss values, indicating that f_{spc} lies within the parameter space cluster of the protected source model f_{src}. Specifically, we adopt an Interval Classification Strategy, which maps the loss intervals to a binary classification to determine whether a unlearnable trigger matches a suspect model. Specifically, we denote the loss of a unlearnable trigger x_{UF} calculated on the source model and the suspect model as $\mathcal{L}_{src}$ and $\mathcal{L}_{spc}$, respectively. By introducing a tolerance factor τ to define classification intervals, we determine whether x_{UF} matches

the suspect model based on the following criterion:

$$\text{Verify}(x_{UF}) = \begin{cases} match, & \text{if } (1-\tau)\mathcal{L}_{src} < \mathcal{L}_{spc} < (1+\tau)\mathcal{L}_{src} \\ not\ match, & \text{Otherwise} \end{cases}, \tag{5}$$

A smaller difference between $\mathcal{L}_{spc}$ and $\mathcal{L}_{src}$ indicates that the suspect and source models behave more similarly on x_{UF}. For different unlearnable trigger, we set different classification intervals individually rather than using a universal interval to avoid errors caused by inherent loss discrepancies among samples. After establishing the matching criterion for a single UF sample, the matching rate r_{mch} of a trigger set $\mathcal{D}_{UF}$, which consists of multiple unlearnable triggers, on a suspect model can be calculated as:

$$r_{mch} = \frac{1}{|\mathcal{D}_{UF}|} \sum_{x \in \mathcal{D}_{UF}} \mathbb{I}(\text{Verify}(x) = match), \tag{6}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Therefore, we can determine whether the suspect model f_{spc} is derived from the source model f_{src} by comparing r_{mch} with a pre-defined infringement threshold θ_{ifg}. If $r_{mch} > \theta_{ifg}$, we conclude that f_{spc} is derived from f_{src}, thereby constituting infringement. Specifically, the infringement threshold θ_{ifg} and the trigger threshold τ can be determined by referring to works [25, 48], which evaluate a large ensemble of models to empirically identify thresholds that maximize verification performance.

Label-Only Scenario. In practical scenarios, the suspect model is often deployed via a black-box API that only returns prediction labels for query samples. In such a scenario, it becomes impossible to directly compute the loss value for our unlearnable triggers. To address this issue, we adopt a Monte Carlo estimation approach to approximate the model's probability vector by sampling noisy variants of a unlearnable trigger and analyzing the empirical label frequencies [7]. Formally, let the probing sample x'_{UF}, mixed proportionally with the perturbation δ_{pb}, be represented as:

$$x'_{UF} = \alpha \cdot x_{UF} + (1-\alpha) \cdot \delta_{pb}, \tag{7}$$

where α $(0 < \alpha < 1)$ is the mixing coefficient that controls the deviation of x'_{UF} relative to x_{UF}. δ_{pb} is obtained through gradient ascent, which can efficiently guide x_{UF} across the decision boundary into other classes. For a given starting sample x_{UF}, each probing sample x'_{UF} can be considered a point within the norm ball $\Omega(x_{UF}) = \{x'_{UF} | \|x_{UF} - x'_{UF}\| \leq r_{pb}\}$ centered at x_{UF}. To measure the distance to the decision boundary, the radius r_{pb} of the norm ball should be sufficiently large such that $\Omega(x_{UF})$ can traverse the decision boundary, which necessitates a pre-estimation of α. We use a binary search, to adjust α iteratively until a sufficient proportion (more than half in this paper) of the probing samples within $\Omega(x_{UF})$ cross the decision boundary.

Once α is determined, the probability of a model f predicting x_{UF} as class c_i can be represented as:

$$P(f(x_{UF}) = c_i) = \mathbb{E}_{x'_{UF} \sim \Omega(x_{UF})}[\mathbb{I}(f(x'_{UF}) = c_i)]. \tag{8}$$

Then, this can be approximated via Monte Carlo integration [36]:

$$\hat{f}^i(x_{UF}) = \frac{1}{M} \sum_{j=1}^{M} \mathbb{I}(f(x'_j) = c_i), \tag{9}$$

where $x'_1, x'_2, \ldots, x'_M$ are M probing samples, randomly drawn from a norm ball centered at x_{UF} with a radius of r_{pb}. Finally, the estimated probabilities $\hat{f}^i(x_{UF})$ can be substituted into Equation 2 to compute the model's loss on x_{UF}.

4 Experiment

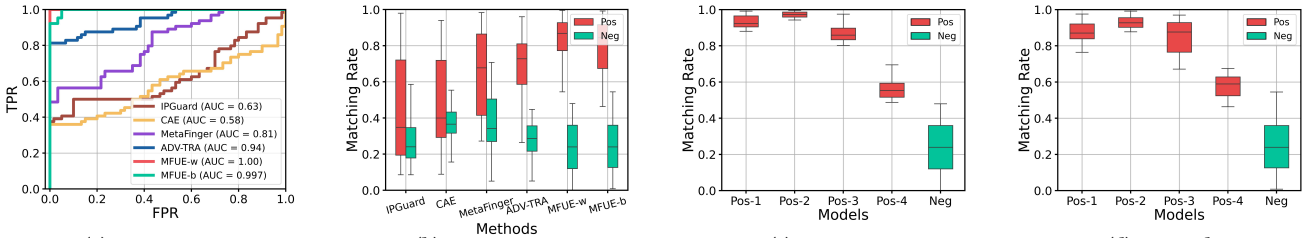
In our experiments, we adopt three commonly used benchmark datasets in the model fingerprinting research field: CIFAR-10 [20], CIFAR-100 [20], and ImageNet [8]. We employ four model fingerprinting methods, including **IPGuard** [4], **CAE** [26], **MetaFinger** [49], **ADV-TRA** [46], as baselines for comparison with our approach. To assess the robustness of our method against model modification attacks, we systematically evaluate MFUE against fine-tuning [1, 49], pruning [13], adversarial training [27], and model extraction attacks [19, 30, 34] across multiple model architectures. We also reveal why unlearnable triggers are more robust than adversarial triggers by analyzing the internal model representations of these samples. At last, the performance under adaptive attacks such as differential privacy are shown in Section 4.4. Complete experimental details, including descriptions of dataset and model, metrics, and implementation details, are provided in Appendix B.

Metrics. We select ROC (Receiver Operating Characteristic) and AUC (Area Under the Curve) [17] as the metrics to evaluate the performance of a fingerprinting method. The AUC is calculated by plotting the true positive rate versus the false positive rate on the same graph, as the decision threshold increases from 0 to 1, and taking the area under the curves. A higher AUC indicates a broader range of thresholds where high uniqueness and robustness in fingerprinting can be achieved.

4.1 Main Performance

We compare our MFUE with four existing fingerprinting baselines on CIFAR-10. Following the verification design from Section 3.2, we distinguish two methods: *MFUE-w* when the suspect model's full prediction probabilities are accessible and *MFUE-b* when only the predicted labels can be queried. We train 64 positive models and 60 negative models on the CIFAR-10 dataset. Detailed settings can be found in Appendix C Table 3. Figure 4a shows the ROC curves for each fingerprinting method. Both MFUE-w and MFUE-b demonstrate superior performance, achieving AUC scores of 1.00 and 0.997, respectively. In comparison, even the best adversarial fingerprinting method, i.e., ADV-TRA, lags behind by 0.057.

We also compare the distribution of fingerprint matching rates for these suspect models. From the experiment results in Figure 4b, we can see that MFUE-w and MFUE-b have much narrower ranges of matching rates for positive models than the four baselines, demonstrating superior robustness against model modification attacks. Moreover, we observe that while the negative models under MFUE exhibit a wider range of fingerprint matching rates, they are still distinctly separable from the positive models, highlighting the better uniqueness of MFUE. In contrast, the comparison fingerprinting methods suffer from the transferability of adversarial samples, which causes the matching rates of negative models to remain relatively high. This often leads to significant overlap with positive models. Even with carefully selected thresholds to balance trade-offs, these methods struggle to achieve both robustness and



(a) ROC Curve (b) Comparison (c) MFUE-w (d) MFUE-b

Figure 4: Main performance of MFUE and comparisons on CIFAR-10.We present (a) the ROC curve, (b) the distribution of different comparisons, and the range of matching rates for four types of positive models and the negative models for (c) MFUE-w and (d) MFUE-b. Pos-1 to Pos-4 correspond to models that have undergone fine-tuning, pruning, adversarial training, and model extraction, respectively.

Table 1: AUC values of different fingerprinting methods across three datasets.

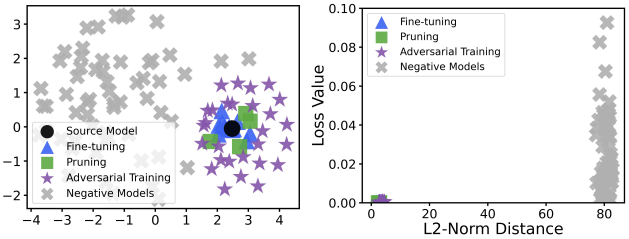
	CIFAR-10	CIFAR-100	ImageNet
IPGuard	0.63	0.66	0.57
CAE	0.58	0.63	0.54
MetaFinger	0.81	0.84	0.72
ADV-TRA	0.94	0.97	0.85
MFUE-w	**1.00**	**1.00**	**0.95**
MFUE-b	**1.00**	**1.00**	**0.88**

uniqueness effectively. We also compare the distribution of matching rates across different suspect models (see Figures 4c and 4d). Although these attacks result in varying degrees of decline in matching rates of fingerprints, they remain significantly higher than those of negative models. Model extraction appears to generate models with greater distinctiveness, resulting in an average decrease of approximately 0.4 in the matching rate. Moreover, MFUE-b, which utilizes estimated loss values, demonstrates effective discrimination between positive and negative models.

Table 1 shows the universality of our method across three datasets. The proposed methods (MFUE-w and MFUE-b) consistently outperform all baseline approaches. MFUE-w achieves perfect scores (1.00) on CIFAR-10 and CIFAR-100, and a near-perfect 0.95 on ImageNet. MFUE-b also attains perfect scores on CIFAR-10/100 (1.00) and a competitive 0.88 on ImageNet. Notably, MFUE-w exceeds the best baseline (ADV-TRA) by +0.06 on CIFAR-10, +0.03 on CIFAR-100, and +0.10 on ImageNet, demonstrating significant gains in cross-dataset universality.

4.2 Model Distribution in Parameter Space

In this paper, our basis is that the parameters of modified versions of one source model remain tightly clustered in parameter space. To further validate this, we analyze the classification-head parameters of various suspect models on CIFAR-10. We select 49 positive models (excluding model extraction attacks) and 60 negative models, and visualized their parameter distributions using t-SNE (Figure 5a). We observe that all modified models cluster closely around the source model, while the negative models lie well apart. Specifically, fine-tuned and pruned models are closer to the source than adversarially trained variants, indicating higher parameter similarity. These clear separations confirm the feasibility of fingerprinting



(a) Parameter Space (b) Loss on Our Triggers

Figure 5: t-SNE visualization of (a) model parameters across different models and (b) the correlation between *L2-Norm* distances of model parameters and loss values.

via parameter space and help explain why MFUE is less effective against adversarial training than other modification attacks.

To further investigate the relationship between loss values associated with our unlearnable triggers and the *L2* distances between the source model and its modified versions, we plot the distribution of loss values against *L2-Norm* distances in Figure 5b. The 49 positive models consistently show loss values near zero, significantly lower than those of the 60 negative models. This substantial loss gap indicates that the differences in loss values are closely tied to the significant distance disparities between the positive and negative models. These results illustrate how MFUE leverages distinct loss patterns across various models to achieve reliable verification, characterized by high uniqueness (i.e., low false-positive rates) and high robustness.

4.3 Robustness against Model Modification Attacks

In this section, we comprehensively evaluate MFUE's performance against various model modification attacks across larger datasets (CIFAR-100 and ImageNet). Our experiments show that MFUE can achieve robust model ownership verification by eliminating the dependence on fragile decision boundaries.

Model Fine-tuning. Previous studies assume that an attacker fine-tunes the stolen source model using the original training data. As the source model has already converged on this dataset, this

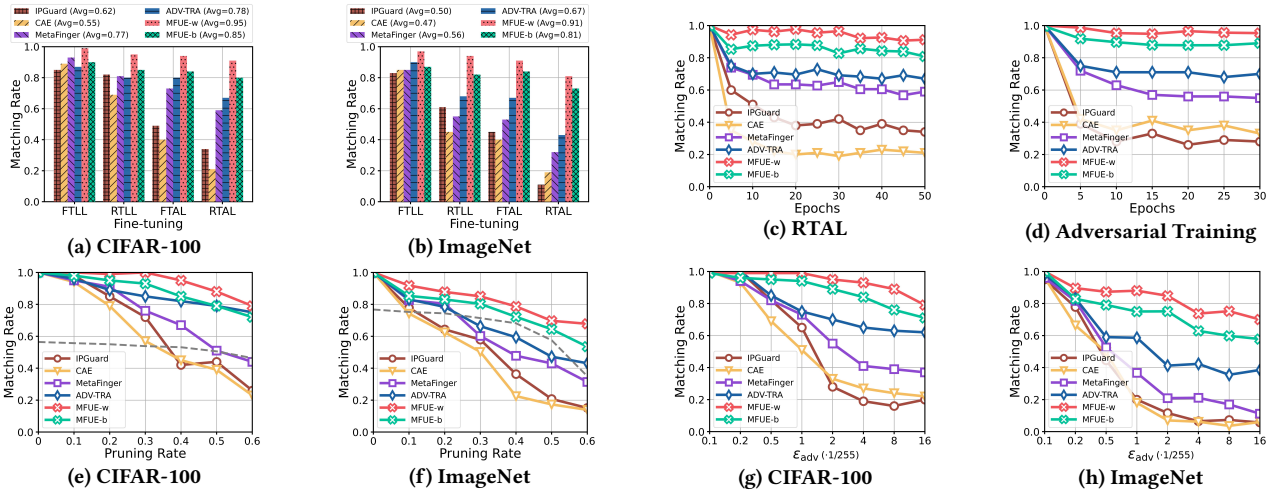


Figure 6: Matching rate of fingerprint triggers under three model modification attacks, including fine-tuning (a,b,c), pruning (e,f), and adversarial training (d,g,h). In (d), the perturbation radius of adversarial training ϵ_{adv} is set to $2/255$. For (g) and (h), the real values on the horizontal axis omit the multiplicative factor $1/225$ for brevity.

procedure induces only minor model changes. As a result, conventional fine-tuning attacks do not produce models that are sufficiently different from the source model and thus fail to reveal the true challenges to fingerprint robustness. In our experiments, we consider a more challenging and more realistic scenario, in which attackers perform fine-tuning attacks using non-training data. Specifically, we study four different fine-tuning strategies to assess MFUE's performance under these conditions. As shown in Figure 6a and Figure 6b, both MFUE-w and MFUE-b maintain a high matching rate (above 0.7) across all fine-tuning attacks. On the more complex ImageNet dataset, the matching rate of our unlearnable fingerprints experiences a slight decline. Among the attacks, RTAL is particularly challenging because it retrains all layers using the original training data. Even under this setting, MFUE achieves a matching rate more than twice that of the best-performing baseline. We further illustrate the matching rate trends during RTAL attacks on CIFAR-100 in Figure 6c. The results show that MFUE-w and MFUE-b consistently maintain matching rates above 0.9 and 0.85, respectively, significantly outperforming the other four baselines.

Model Pruning. Model pruning can alter a model's decision boundaries and is therefore widely regarded as a common type of model modification attack [2, 38]. In this part, we prune the source model with pruning rates ranging from 0.1 to 0.6, as shown in Figure 6e and Figure 6f. Note that when the pruning rate exceeds 0.3, the model's performance on its original task is nearly destroyed. On CIFAR-100, we observe that once the test accuracy begins to degrade significantly at a pruning rate of 0.3, the matching rates of all fingerprinting methods also drop sharply. A similar trend appears at an even lower pruning rate on ImageNet. Nevertheless, even after pruning 60% of the model parameters, MFUE-w maintains matching rates above 0.6, enabling successful fingerprint verification.

Adversarial Training. Compared to other model modification attacks, adversarial training poses a greater challenge for model fingerprinting, as it significantly alters the model's decision boundaries. Figure 6g and Figure 6h show the performance of MFUE under adversarial training with various perturbation bound (ϵ_{adv}). As ϵ_{adv} increases, the verification performance of the four baseline methods drops sharply, whereas MFUE-w and MFUE-b exhibit only a moderate decline. On ImageNet, our approach significantly outperforms ADV-TRA, with a 105% and 82.3% improvement when $\epsilon_{adv} = 2/225$. The primary reason our unlearnable triggers outperform adversarial triggers lies in their stability under model changes. Specifically, adversarial triggers induce significant distribution shifts in the embedding space when the model parameters change, so as to alter the model's predictions. In contrast, our unlearnable triggers maintain consistent representations in embedding space before and after such model changes. A detailed explanation is provided in the next paragraph (Section 4.3). Figure 6d shows the trend in matching rates during adversarial training, most baseline methods experience a rapid decline in performance, stabilizing within the first five epochs. Surprisingly, our approach even increases with more adversarial training. For this phenomenon, we conjecture that the

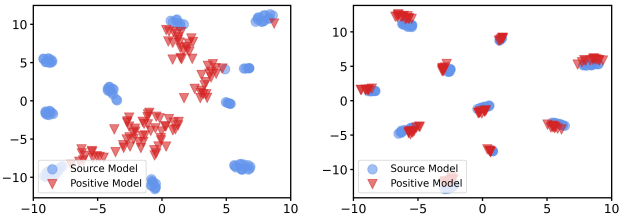
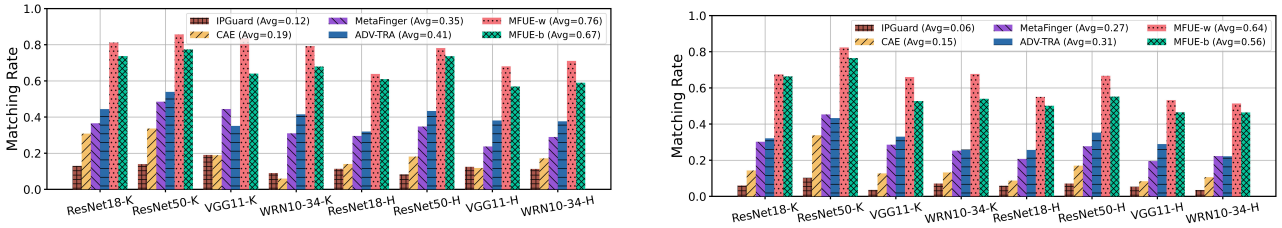


Figure 7: t-SNE visualization of embeddings for the source model and the positive model processed by adversarial training with $\epsilon_{adv} = 4/255$. Each cluster in the source model embedding space represents a distinct class.



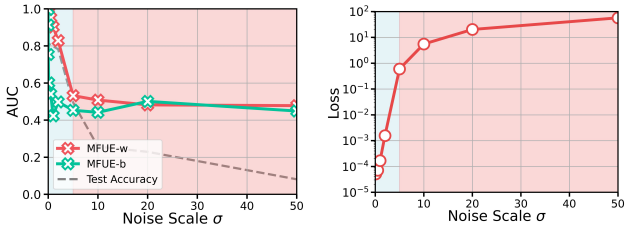
(a) CIFAR-100 (b) ImageNet

Figure 8: Matching rate of positive models extracted with various model architectures. The suffixes "-K" and "-H" denote models obtained through the Knockoff and HL extraction methods, respectively. "Avg" represents the average matching rate across all architectures.

early stages of adversarial training cause a significant shift in the class centers, but as the model adapts to adversarial training (i.e., test accuracy increases), the class centers gradually return to their original positions.

Why Unlearnable Triggers Performs Better? Surprisingly, adversarial training has a far greater impact on weakening adversarial triggers than unlearnable triggers. But why unlearnable triggers are more robust against model modification attacks? To further explore the underlying reasons, we use t-SNE to visualize the embeddings (the output of the model's feature extraction layer) of different fingerprint samples before and after model modification. We select a number of adversarial triggers and unlearnable triggers, and compare their shifts in embeddings due to the model modification. As shown in Figure 7, the embeddings of adversarial triggers in the positive model hardly overlap with those in the source model, even creating many new manifolds. This suggests that the original fingerprint relationship of adversarial triggers is broken after model modification. In contrast, the embeddings of unlearnable triggers remain nearly unchanged before and after modification, demonstrating remarkable stability, or robustness, against modification attacks. This may be because adversarial training effectively alters the model's decision boundary, causing adversarial triggers to fail. However, MFUE does not rely on decision boundaries for verification. Instead, it utilizes a training loss associated with the class center, which remains relatively stable during model modification (as shown in Figure 7b). Therefore, MFUE is a model fingerprinting approach that is robust against model modifications.

Model Extraction. We evaluate two model extraction attacks: Knockoff [30] and HL [34]. Specifically, we assume that the attacker can use models with various structures for extraction. Figure 8 shows that MFUE-w and MFUE-b achieve more than twice the matching rates of other baselines. For most extracted models, the matching rates of IPGuard and CAE are lower than those of the negative models, indicating that they cannot withstand these two model extraction attacks. By comparing the performance on Knockoff and HL which use the original and the synthetic data respectively, the results suggest that models trained on data from similar distributions tend to have decision boundaries more closely aligned with the source model than those trained on synthetic data. This implies that attackers who leverage synthetic data can conduct stealthy model theft, which remains a significant challenge for model fingerprinting.



(a) AUC and Accuracy (b) Loss

Figure 9: The performance of MFUE under DP-Logits attack with varying noise scales σ.

4.4 Adaptive Attacks

As demonstrated in the aforementioned experiments, MFUE shows striking robustness against various attacks that may frustrate existing AF-based methods. Nevertheless, we are still interested in exploring one question: Is there an adaptive attack that can frustrate MFUE? We consider the calculated training loss which MFUE depends on as a starting point, attempting to find MFUE's weaknesses under semi-black-box conditions (i.e., the deployed suspect model provides prediction vectors). Specifically, we assume a dishonest attacker employs a differential privacy (DP) protection method that perturbs the model output, namely DP-Logits [32]. DP-Logits perturbs the prediction vectors, thereby affecting the loss values which MFUE relies on. Figure 9 shows MFUE's performance on CIFAR-10 under different noise scales σ. As σ increases to 5, both MFUE-w and MFUE-b experience a sharp drop, rendering them unable to identify effectively. Similarly, the test accuracy also declines significantly in the early stages, although the rate of decline is somewhat slower. Moreover, the decrease in AUC is accompanied by an increase in the loss value. When σ exceeds 10, both MFUE and the model itself approach the level of random guessing. This is easy to understand: perturbed prediction vectors cause inaccuracies in loss calculations, ultimately making it difficult for MFUE to distinguish between different suspect models. Interestingly, when the noise scale becomes too large ($\sigma > 2$), the model's performance also declines significantly. This suggests that attackers at the model deployment end might not adopt such a high level of DP protection. An astute attacker can trade off between model utility and attack intensity, selecting a preferable noise scale such as $\sigma = 2$

Table 2: Detection accuracy for different fingerprinting queries on CIFAR-100.

	IPGuard	CAE	MetaFinger	ADV-TRA	MFUE-w	MFUE-b
SHE	0.77	0.69	0.80	0.75	**0.13**	0.42
SCADN	0.53	0.49	0.46	0.61	**0.25**	0.30
DAEIT	0.93	0.97	0.96	0.98	**0.00**	0.23

to weaken MFUE. In such a case, MFUE-w's AUC (0.829) is lower than the best baseline ADV-TRA (0.94) without any attacks. On the other hand, DP-Logits has a greater interference effect on MFUE-b than MFUE-w. This is because in addition to directly affecting the calculation of the loss, DP-Logits also impacts the distribution of probing samples across classes in the Monte Carlo estimation process., which ultimately leads to the vulnerability of MFUE-b.

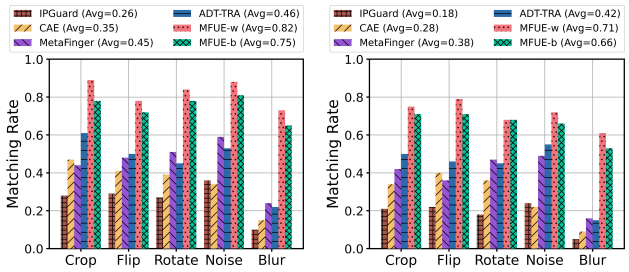
4.5 Robustness against Input Processing Attacks

In this section, we investigate the impact of five image processing techniques that can modify and compromise query fingerprinting samples. As shown in Figure 10, these processing methods can render most (over 50%) of the baselines ineffective. Notably, Gaussian blur causes over 80% of samples in the baselines to fail to match the source model on the ImageNet. In contrast, both MFUE-w and MFUE-b maintain an average matching rate above 0.7 across all attacks, significantly higher than the other four baselines. We conjecture that this is because loss-minimizing perturbations are not as frail as loss-maximizing perturbations. Interestingly, Gaussian noise appears to be less effective compared to the first three linear transformations.

4.6 Stealthiness

Furthermore, attackers at the model deployment end can not only disrupt the model fingerprints (e.g., through the aforementioned model modification attacks and input processing) but also breach ownership verification by intercepting potential fingerprinting samples using anomaly detection methods. Previous works have rarely considered the stealthiness of fingerprinting samples. Therefore, we consider three types of detection strategies at the deployment end: 1) SHE [51], an out-of-distribution (OOD) detection method that compares the test sample with stored patterns of each class using a similarity measure derived from Hopfield energy. 2) SCADN [47], which performs unsupervised anomaly detection by learning to reconstruct missing regions in normal samples leveraging semantic context. 3) DAEIT [41], which examines predictions under various image transformations to filter adversarial samples. We mix an equal number of clean and fingerprinting samples for detection to balance the method's bias towards classifying samples as normal or anomalous. For SHE and SCADN, we use the average score of the bottom 10% quantile of normal samples as a threshold to distinguish anomalous queries.

Table 2 presents the detection accuracy of different detection strategies. The reconstructed-based SCADN cannot identify anomalous queries well, almost random guessing. This is possibly because the perturbations added to the fingerprinting samples on the original inputs are too small, even smaller than the reconstruction bias. In contrast, output-based methods such as SHE and DAEIT are able



(a) CIFAR-100 **(b) ImageNet**

Figure 10: Matching rates of the fingerprinting samples after five image processing attacks.

to effectively detect most adversarial triggers. Specifically, DAEIT achieves detection rates above 90% for all adversarial fingerprints. Such high detection rates are difficult to avoid because adversarial triggers, being critical samples near the decision boundaries, easily move out of their original class space due to minor changes, making them more detectable. However, the MFUE method uses correctly classified natural samples, which are not easily recognized as anomalies, so their predicted labels can remain stable even under transformations. The samples detected in the MFUE-b case, mainly originate from the probing samples in loss estimation, rather than the unlearnable triggers themselves. Comparing with the normal adversarial samples located near the decision boundary, the probing samples have a larger perturbation radius, making them more robust to various transformations in the detection method and thus less likely to be detected.

5 Conclusion

In this paper, we reveal that existing adversarial fingerprints, which capture model decision boundaries via adversarial examples, are unstable—minor adjustments (e.g., fine-tuning) can shift boundaries and compromise the integrity of such fingerprints. To address this fragile coupling between adversarial fingerprints and the decision boundary, we propose MFUE, a novel fingerprinting framework that leverages the inherent robustness of unlearnable examples across a source model and its modified versions. Unlike traditional methods, MFUE operates in the parameter space, allowing it to remain resilient even when the decision boundaries of modified models diverge from those of the source model. By incorporating adversarial training, MFUE generates UF examples that remain effective across a variety of model modifications. Extensive experiments across multiple attacks demonstrate that MFUE significantly outperforms existing fingerprinting methods in robustness, uniqueness and stealthiness. Our work reveals that conducting fingerprinting in parameter space serves as a superior alternative to adversarial fingerprints that frailly couple adversarial examples and the decision boundary, and paves the way for future robust, parameter centric model fingerprinting methods.

References

[1] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *Proceedings of USENIX Security Symposium*. 1615–1631.

[2] Motasem Alfarra, Adel Bibi, Hasan Hammoud, Mohamed Gaafar, and Bernard Ghanem. 2022. On the decision boundaries of neural networks: A tropical geometry perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45 (2022), 5027–5037.

[3] Shekoofeh Azizi, Laura Culp, Jan Freyberg, Basil Mustafa, Sebastien Baur, Simon Kornblith, Ting Chen, Nenad Tomasev, Jovana Mitrović, Patricia Strachan, et al. 2023. Robust and data-efficient generalization of self-supervised machine learning for diagnostic imaging. *Nature Biomedical Engineering* 7, 6 (2023), 756–779.

[4] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2021. IPGuard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary. In *Proceedings of ACM ASIACCS*. 14–25.

[5] Beitao Chen, Xinyu Lyu, Lianli Gao, Jingkuan Song, and Heng Tao Shen. 2024. Alleviating Hallucinations in Large Vision-Language Models through Hallucination-Induced Optimization. In *Proceedings of NeurIPS*.

[6] Jialuo Chen, Jingyi Wang, Tinglan Peng, Youcheng Sun, Peng Cheng, Shouling Ji, Xingjun Ma, Bo Li, and Dawn Song. 2022. Copy, right? A testing framework for copyright protection of deep learning models. In *Proceedings of IEEE S&P*. 824–841.

[7] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *Proceedings of ICML*. 1310–1320.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proceedings of IEEE/CVF CVPR*. 248–255.

[9] Tian Dong, Shaofeng Li, Guoxing Chen, Minhui Xue, Haojin Zhu, and Zhen Liu. 2023. RAI2: Responsible Identity Audit Governing the Artificial Intelligence.. In *Proceedings of NDSS*.

[10] Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of AAAI*, Vol. 37. 12799–12807.

[11] Chujie Gao, Siyuan Wu, Yue Huang, Dongping Chen, Qihui Zhang, Zhengyan Fu, Yao Wan, Lichao Sun, and Xiangliang Zhang. 2024. HonestLLM: Toward an Honest and Helpful Large Language Model. In *Proceedings of NeurIPS*.

[12] Shangwei Guo, Tianwei Zhang, Han Qiu, Yi Zeng, Tao Xiang, and Yang Liu. 2020. The hidden vulnerability of watermarking for deep neural networks. *CoRR, arXiv: 2009.08697* (2020).

[13] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Proceedings of NeurIPS*.

[14] Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. 2023. Sensitivity-aware visual parameter-efficient fine-tuning. In *Proceedings of IEEE/CVF CVPR*. 11825–11835.

[15] Chenfei Hu, Chuan Zhang, Dian Lei, Tong Wu, Ximeng Liu, and Liehuang Zhu. 2023. Achieving privacy-preserving and verifiable support vector machine training in the cloud. *IEEE Transactions on Information Forensics and Security* 18 (2023), 3476–3491.

[16] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. 2021. Unlearnable examples: Making personal data unexploitable. In *Proceedings of ICLR*.

[17] Jin Huang and Charles X Ling. 2005. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on knowledge and Data Engineering* 17, 3 (2005), 299–310.

[18] Ziheng Huang, Boheng Li, Yan Cai, Run Wang, Shangwei Guo, Liming Fang, Jing Chen, and Lina Wang. 2023. What can Discriminator do? Towards Box-free Ownership Verification of Generative Adversarial Network. *CoRR, arXiv: 2307.15860* (2023).

[19] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. 2019. PRADA: Protecting against DNN model stealing attacks. In *Proceedings of IEEE S&P*. 512–527.

[20] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2010. Cifar-10 (canadian institute for advanced research).

[21] Isabell Lederer, Rudolf Mayer, and Andreas Rauber. 2024. Identifying appropriate intellectual property protection mechanisms for machine learning models: A systematization of watermarking, fingerprinting, model access, and attacks. *IEEE Transactions on Neural Networks and Learning Systems* 35, 10 (2024), 13082–13100.

[22] Sam Leroux, Stijn Vanassche, and Pieter Simoens. 2024. Multi-bit Black-box Watermarking of Deep Neural Networks in Embedded Applications. In *Proceedings of IEEE/CVF CVPR*. 2121–2130.

[23] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. 2017. Delving into transferable adversarial examples and black-box attacks. In *Proceedings of ICLR*.

[24] Yixin Liu, Kaidi Xu, Xun Chen, and Lichao Sun. 2024. Stable Unlearnable Example: Enhancing the Robustness of Unlearnable Examples via Stable Error-Minimizing Noise. In *Proceedings of AAAI*. 3783–3791.

[25] Nils Lukas, Edward Jiang, Xinda Li, and Florian Kerschbaum. 2022. Sok: How robust is image classification deep neural network watermarking?. In *Proceedings of IEEE S&P*. 787–804.

[26] Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum. 2021. Deep Neural Network Fingerprinting by Conferrable Adversarial Examples. In *Proceedings of ICLR*.

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *Proceedings of ICLR*.

[28] Thibault Maho, Teddy Furon, and Erwan Le Merrer. 2023. FBI: Fingerprinting models with benign inputs. *IEEE Transactions on Information Forensics and Security* 18 (2023), 5459–5472.

[29] Gaurav Menghani. 2023. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *Comput. Surveys* 55, 12 (2023), 1–37.

[30] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of IEEE/CVF CVPR*. 4954–4963.

[31] Zirui Peng, Shaofeng Li, Guoxing Chen, Cheng Zhang, Haojin Zhu, and Minhui Xue. 2022. Fingerprinting deep neural networks globally via universal adversarial perturbations. In *Proceedings of IEEE/CVF CVPR*. 13430–13439.

[32] Shadi Rahimian, Tribhuvanesh Orekondy, and Mario Fritz. 2019. Differential Privacy Defenses and Sampling Attacks for Membership Inference. In *Proceedings of NeurIPS*.

[33] Mauro Ribeiro, Katarina Grolinger, and Miriam AM Capretz. 2015. Mlaas: Machine learning as a service. In *Proceedings of IEEE ICMLA*. 896–902.

[34] Sunandini Sanyal, Sravanti Addepalli, and R Venkatesh Babu. 2022. Towards data-free model stealing in a hard label setting. In *Proceedings of IEEE/CVF CVPR*. 15284–15293.

[35] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free!. In *Proceedings of NeurIPS*.

[36] Alexander Shapiro. 2003. Monte Carlo sampling methods. *Handbooks in operations research and management science* 10 (2003), 353–425.

[37] Gowthami Somepalli, Liam Fowl, Arpit Bansal, Ping Yeh-Chiang, Yehuda Dar, Richard Baraniuk, Micah Goldblum, and Tom Goldstein. 2022. Can neural nets learn the same model twice? Investigating reproducibility and double descent from the decision boundary perspective. In *Proceedings of IEEE/CVF CVPR*. 13699–13708.

[38] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. In *Proceedings of NeurIPS*, Vol. 35. 19523–19536.

[39] Jingxuan Tan, Nan Zhong, Zhenxing Qian, Xinpeng Zhang, and Sheng Li. 2023. Deep neural network watermarking against model extraction attack. In *Proceedings of ACM MM*. 1588–1597.

[40] Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. 2024. Deep reinforcement learning for robotics: A survey of real-world successes. *Annual Review of Control, Robotics, and Autonomous Systems* 8 (2024).

[41] Shixin Tian, Guolei Yang, and Ying Cai. 2018. Detecting adversarial examples through image transformation. In *Proceedings of AAAI*, Vol. 32.

[42] Pratik Vaishnavi, Kevin Eykholt, and Amir Rahmati. 2022. Transferring adversarial robustness through robust representation matching. In *Proceedings of USENIX Security Symposium*.

[43] Siyue Wang, Xiao Wang, Pin-Yu Chen, Pu Zhao, and Xue Lin. 2021. Characteristic examples: High-robustness, low-transferability fingerprinting of neural networks. In *Proceedings of IJCAI*. 575–582.

[44] Zhibo Wang, Hongshan Yang, Yunhe Feng, Peng Sun, Hengchang Guo, Zhifei Zhang, and Kui Ren. 2023. Towards transferable targeted adversarial examples. In *Proceedings of IEEE/CVF CVPR*. 20534–20543.

[45] Cheng Xiong, Guorui Feng, Xinran Li, Xinpeng Zhang, and Chuan Qin. 2022. Neural network model protection with piracy identification and tampering localization capability. In *Proceedings of ACM MM*.

[46] Tianlong Xu, Chen Wang, Gaoyang Liu, Yang Yang, and Liu Wei. 2024. United We Stand, Divided We Fall: Fingerprinting Deep Neural Networks via Adversarial Trajectories. In *Proceedings of NeurIPS*.

[47] Xudong Yan, Huaidong Zhang, Xuemiao Xu, Xiaowei Hu, and Pheng-Ann Heng. 2021. Learning semantic context from normal samples for unsupervised anomaly detection. In *Proceedings of AAAI*.

[48] Yifan Yan, Xudong Pan, Mi Zhang, and Min Yang. 2023. Rethinking {White-Box} Watermarks on Deep Learning Models under Neural Structural Obfuscation. In *Proceedings of USENIX Security Symposium*. 2347–2364.

[49] Kang Yang, Run Wang, and Lina Wang. 2022. MetaFinger: Fingerprinting the deep neural networks with meta-training. In *Proceedings of IJCAI*. 776–782.

[50] Boyi Zeng, Lizheng Wang, Yuncong Hu, Yi Xu, Chenghu Zhou, Xinbing Wang, Yu Yu, and Zhouhan Lin. 2024. Huref: Human-readable fingerprint for large language models. In *Proceedings of NeurIPS*.

[51] Jinsong Zhang, Qiang Fu, Xu Chen, Lun Du, Zelin Li, Gang Wang, Shi Han, Dongmei Zhang, et al. 2023. Out-of-distribution detection based on in-distribution data patterns memorization with modern hopfield energy. In *Proceedings of ICML*.

[52] Jiaming Zhang, Xingjun Ma, Qi Yi, Jitao Sang, Yu-Gang Jiang, Yaowei Wang, and Changsheng Xu. 2023. Unlearnable clusters: Towards label-agnostic unlearnable examples. In *Proceedings of IEEE/CVF CVPR*. 3984–3993.

[53] Yue Zheng, Si Wang, and Chip-Hong Chang. 2022. A dnn fingerprint for non-repudiable model ownership identification and piracy detection. *IEEE Transactions on Information Forensics and Security* 17 (2022), 2977–2989.

Appendix

In this appendix, we present the following additional contents: (1) A review of model fingerprinting methods categorized by white-box and black-box approaches (Appendix A); (2) Detailed information for our experimental setups (Appendix B); (3) The types of suspect models and their corresponding test accuracies (Appendix C);

A Related Work

To protect the IP of deep learning models, model fingerprinting extracts a model's features in a non-invasive manner. Existing methods can be categorized into two approaches based on the verification conditions: black-box and white-box.

A.1 Black-Box Model Fingerprinting

Under the query limitations of black-box scenarios, particularly in label-only settings, most fingerprinting approaches utilize adversarial samples to match the decision boundaries of models. This approach quickly determines the model's ownership by relying on the model's direct predictions.

Cao et al. [4] argue that a model can be uniquely represented by its decision boundary. They propose the first model fingerprinting method, which involves identifying special data points (i.e., adversarial samples) near the boundary to mark the target model. Later studies [26, 43] explored the transferability of such adversarial samples. Peng et al. [31] also highlight a major issue with previous fingerprinting methods: a high false positive rate. This occurs because two unrelated models often share the same fingerprinting samples due to adversarial transfer [23, 42, 44]. To address this, they generate Universal Adversarial Perturbations, which fingerprint the model by shifting any clean sample onto the decision boundary. Similarly, MetaFinger [49] fingerprints the internal decision area of deep neural networks through meta-training and augmentation, improving robustness against changes in decision boundaries. However, perturbation-based methods lack security and uniqueness [1, 18]: once an attacker learns the pattern of the perturbation, they can easily invalidate all fingerprinting samples by removing the perturbation. Recently, ADV-TRA [46] is proposed to fingerprint the entire decision surface using adversarial trajectories. The progressive adversarial levels within these trajectories allow for more precise characterization of decision boundaries, enhancing fingerprinting accuracy. However, the high-frequency adversarial queries used in this approach may raise suspicion.

A.2 White-box Model Fingerprinting

White-box approaches assume full access to the model's internals, including weights, activations, and the underlying network structure. This access enables the use of additional carriers as potential model fingerprints. Maho et al. [28] design a greedy algorithm that uses as few benign inputs as possible, leveraging C.E. Shannon's information theory to quantify the statistical similarity between the outputs of different model layers. Model parameters can also serve as fingerprints. Zheng et al. [53] project the front-layer weights onto a random space defined by the model owner's identity, allowing for non-repudiable and irrevocable proof of ownership to protect against model IP misappropriation and ownership fraud. In addition, model parameters can be hashed to generate a unique identity sequence that represents the model. For instance, Xiong et al. [45] introduce the *Piracy Identification Hash* and *Tampering Localization Hash* to verify model copyright and identify tampered weights, respectively.

Recent works [6, 9] have proposed audit frameworks to assess model IP infringement. The first framework uses multiple testing metrics (e.g., neuron output distance) to calculate the similarity between two models. The second approach projects high-dimensional model parameters into low-dimensional data and compares their distributions. However, white-box approaches are limited to scenarios with complete transparency, where the model details are fully accessible. As a result, their applicability is restricted in real-world scenarios where models are intentionally tampered with or kept confidential.

B Experiment Setup

Datasets. We consider three commonly used image datasets CIFAR-10 [20], CIFAR-100 [20], and ImageNet [8] in our experiments. For both CIFAR-10 and CIFAR-100, we allocate 30,000 samples for training the source model, while reserving the remaining 30,000 samples for the attacker to launch model modification attacks or train negative models. For ImageNet, 200,000 images are selected to perform model modification attacks on the source model.

Models. For the source models, we consider ResNet18 for CIFAR-10, while ResNet50 for CIFAR-100 and ImageNet. We evaluate our approach on two types of suspect models: positive models and negative models. Positive models are derived from the source model through a series of model modification attacks, and are inherently infringing models. Negative models, on the other hand, are independently trained and unrelated to the source model, thus considered non-infringing or innocent models. We train models of eight different architectures, namely ResNet18, ResNet50, VGG11, VGG16, VGG19, WRN10-34, DenseNet121, and InceptionV3, to serve as negative models. Ideally, the fingerprinting samples should match only the positive models, not the negative models. To conserve computational resources required for training multiple models on ImageNet, following prior work [4, 46], we adopt pre-trained models available in torchvision[1].

Model Modification Attacks. An attacker attempting to break the inherent association between the source model and the fingerprinting samples can either modify the model using auxiliary data or distill a substitute model from the source model. Here, we consider four commonly used model modification attacks in model IP protection domain, including fine-tuning, pruning, model extraction, and adversarial training.

- **Model fine-tuning** uses auxiliary data to continue to train a pre-trained model (source model). For the CIFAR-10 and CIFAR-100, we select 30,000 samples to fine-tune the source model, while for ImageNet, 200,000 samples are selected for the same purpose. Note that for CIFAR-10/100, we fine-tune the source model using data beyond the training data, which results in greater modifications to the source model. The batch size is set to

[1] https://docs.pytorch.org/vision/stable/

Table 3: Test accuracy (%) on the suspect models under evaluation, including 64 positive models and 60 negative models. The second row indicates the key parameters for each type of suspect model.

Source	**Fine-tuning** (16)				**Pruning** (4)			
ResNet18	FTLL	RTLL	FTAL	RTAL	0.1	0.2	0.3	0.4
84.43	85.71	84.76	84.59	82.54	84.21	83.92	82.03	80.68
Model Extraction (16)		**Adversarial Training** (28)						
Knock	HL	0.1/255	0.2/255	0.5/255	1/255	2/255	4/255	8/255
82.13	81.08	83.76	83.16	83.42	84.01	82.84	83.23	83.60
Negative Models (60)								
ResNet18	ResNet50	VGG11	VGG16	VGG19	WRN10-34	DenseNet121	InceptionV3	/
85.12	87.34	85.28	87.35	86.90	87.71	84.75	85.86	/

64, and an SGD optimizer with a learning rate of 0.01 is adopted. For each datasets, we fine-tune for 50 epochs.

- **Pruning** is used for model compression, so that the model's memory for fingerprints diminishes over the pruning process. We utilize the technique in [13] for our evaluation. We vary the pruning rate p from 0.1 to 0.9, i.e., pruning p fraction of parameters that have the smallest absolute values.
- **Adversarial training** helps models withstand misleading inputs by incorporating them during the training process to improve the model's robustness. It has been proven that this can significantly reduce the effectiveness of current adversarial fingerprinting techniques [25, 26]. We adopt standard ℓ_∞-PGD training, which has been proved to achieve the SOTA empirical robustness. For each ϵ_{adv}, we train the model for 30 epochs. To avoid compromising the model's accuracy, we incorporate normal samples during adversarial training, as done in [27].
- **Model extraction** enables an attacker to derive a substitute model from the source model. We apply two commonly used attacks, namely, Knockoff [30] and HL [34] for evaluation. Knockoff collects data from different domains and trains a substitute model with fewer queries in a reinforcement learning manner. As a data-free model stealing attack, HL leverages the structure of DCGAN and alternately trains a substitute model and generator in a hard-label setting.

Input Processing Attacks. In addition to modifying the model, an attacker at the suspect model's deployment end may also erase or alter the fingerprints through input processing [12, 49]. Specifically, we adopt five common image processing techniques: cropping, flipping, rotating, Gaussian noise, and Gaussian blur. For cropping, we randomly crop 32×32 images to 24×24 and resize them back. For flipping, we flip the images horizontally or vertically. For rotating, we rotate the images by arbitrary angles and padded them. Gaussian noise with a standard deviation of $\sigma = 16/255$ is added to the images, while Gaussian blur with a kernel size of $k = 5$ is applied.

Baselines. We compare MFUE with the following four SOTA fingerprinting schemes.

- **IPGuard** [4] designs a type of fingerprinting samples that is superior to general adversarial samples and can better locate the decision boundaries. Its key parameter k is set to 10 for CIFAR-10/100 and 100 for ImageNet.
- **CAE** [26] proposes to use multiple surrogate models to find the conferrable fingerprinting samples that demonstrate better transferability on stolen models. We configure 10 surrogate models and 10 reference models.
- **MetaFinger** [49] fingerprints the inner decision area of the models through meta-training, rather than the vulnerable decision boundaries. The model pool comprises 20 positive models and 20 negative models.
- **ADV-TRA** [46] utilizes adversarial trajectories to quantify the decision boundaries, which not only tolerates a greater degree of alteration but also reduces the false positives during verification. We set the trajectory length to 40 with $m = 8$.

Implementation Details. We train the source models and suspect models for 150 epochs with a batch size $b = 128$. We use SGD optimizer with a learning rate of 0.1, momentum of 0.9, and weight decay of 5e-4. For all fingerprinting methods, we select 100 clean samples to generate an equal number of adversarial or unlearnable triggers. For our MFUE approach, we set the number of mimic models to 10, with the adversarial training radius for mimic models set to 4/255. The learning rate for the mimic models is 0.001, and the perturbation radius for unlearnable triggers is $\epsilon_{UF} = 8/255$. In the Monte Carlo-based loss estimation, we use 100 probing samples and set the tolerance factor $\tau = 0.5$. We repeat the experiments four times on each dataset and report the average results. All experiments are conducted on a server equipped with an AMD Ryzen 9 7950X 16-core CPU, dual NVIDIA GeForce RTX 4090 GPUs, and a 64GB DDR4 RAM.

C Suspect Models List

Please refer to Table 3.