Applied Intelligence (2026) 56:405
https://doi.org/10.1007/s10489-026-07440-6

Efficient and robust DNN model fingerprint with decision difference regions identification

Tianxiang Luo[1] · Zi Yang[1] · Xiaorui Dai[1] · Bo Wang[1] · Wei Wang[2]

Received: 20 January 2026 / Accepted: 18 August 2026

Abstract

Deep neural network (DNN) models are valuable intellectual property, and black-box ownership verification is needed when they are copied, extracted, or post-processed. Existing fingerprints can identify source-related models but may not explicitly separate them from independently trained benign models, leaving owners exposed to false-accusation risk. This paper upgrades fingerprint evidence from a source-boundary transferability heuristic to an explicit ownership-separation objective: we formulate fingerprint construction as a decision difference region (DDR) search problem and solve it with two non-invasive variants—Fast variant for low-cost screening and Selected variant for higher-confidence offline filtering—giving owners a tunable trade-off between construction cost and verification confidence. Evaluation spans CIFAR-10 and the German Traffic Sign Recognition Benchmark (GTSRB), with CIFAR-100 and Tiny-ImageNet as stress tests, across diverse source architectures, hard negatives, and extraction attacks. The results support tested verification while identifying its limits, giving practitioners ownership evidence with an explicit account of where the separation holds and where it does not.

Keywords Model intellectual property protection · Deep neural networks · Model ownership verification · Model fingerprinting · Black-box verification · Decision difference regions

1 Introduction

In recent years, deep neural network (DNN) models have rapidly developed across computer vision, speech recognition, autonomous driving, intelligent healthcare, smartphone-based activity recognition, federated sensor learning, and behavior time-series analysis [1–3]. Owners of these models can profit by providing trained-model predictions as a service through APIs, such as Machine Learning-as-a-Service (MLaaS). Training a high-performing model requires substantial data preparation and computational resources, making DNN models valuable business assets. However, adversaries can acquire models with similar utility through lower-cost model-stealing attacks. Protecting the intellectual property (IP) of DNN models is therefore an important part of AI model security.

Two common forensic techniques have been proposed for DNN model ownership verification: model watermarking and model fingerprinting. Model watermarking embeds verification information into the source model before deployment and later checks whether this information is present in the queried suspect model. This embedding can affect source-model utility, and recent studies show that watermarking may be vulnerable to removal attacks [4, 5]. Recent model-ownership-resolution work further shows that false claims by malicious accusers are a serious concern for both watermarking and fingerprinting evidence [6]. Model fingerprinting instead extracts a query set after training and deployment, without modifying the source model. Ownership is then

✉ Bo Wang
bowang@dlut.edu.cn

Tianxiang Luo
3142531874@mail.dlut.edu.cn

Zi Yang
dutyangzi@163.com

Xiaorui Dai
xiaoruidai@mail.dlut.edu.cn

Wei Wang
wwang@nlpr.ia.ac.cn

[1] School of Information and Communication Engineering, Dalian University of Technology, No. 2 Linggong Road, Dalian 116081, China

[2] New Laboratory of Pattern Recognition (NLPR), State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing 100190, China

Published online: 07 September 2026

evaluated by the matching rate obtained when the fingerprint queries are submitted to the suspect model.

Previous model fingerprinting methods [7–10] primarily rely on adversarial examples, conferrable examples, or source-oriented decision-boundary behavior. These ideas are useful, but ownership verification requires a more specific separation: a query should match models derived from the protected source model while avoiding high responses on independently trained or near-benign models. This distinction is especially important under model extraction, fine-tuning, same-seed retraining, partial-transfer, or architecture-related benign models, where a purely source-boundary or transferability heuristic may increase false-accusation risk.

In this paper, we formulate model fingerprint construction as a decision difference region (DDR) search problem for ownership verification. Figure 1 outlines the intuition behind our approach. Instead of only looking for adversarial samples that are close to the source model's boundary, we seek samples that expose a positive-negative ownership-boundary gap: source-related positive models should predict a target label consistently, whereas independently trained negative models should not. DDR should therefore be understood as a verification objective, not merely as an engineering strategy that uses additional shadow models. The positive pool encourages robustness to source-derived model changes, and the negative pool suppresses responses that would also occur on benign models. Considering the balance between construction cost and verification confidence, we propose two variants: Fast variant for low-cost screening and Selected variant for offline high-confidence verification through conferrability-based filtering.

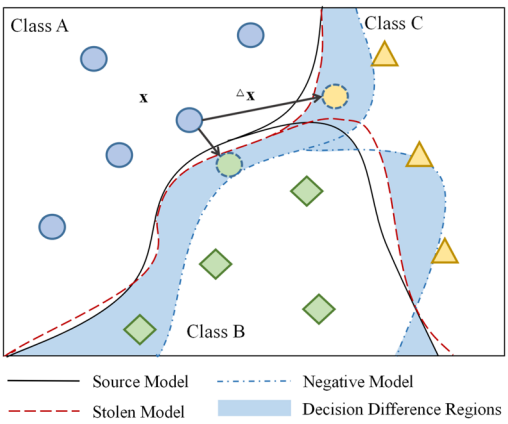


Fig. 1 Intuition of decision difference region fingerprinting. A useful fingerprint query lies in a region where source-related models preserve the target-label behavior, while independently trained negative models do not

In summary, the main contributions are as follows:

- We formulate DNN model fingerprint construction as a DDR search problem over source-related positive models and independently trained benign negative models. This turns query selection into an explicit ownership-boundary separation objective rather than only a source-boundary transferability heuristic, giving the field a formal optimization target that directly encodes false-accusation avoidance.
- We propose two non-invasive query-construction variants. Fast variant is designed for low-cost screening, while Selected variant performs offline high-confidence filtering using a validation shadow pool. The source model is not modified, and online verification only requires black-box responses from the suspected model, so owners can pick an operating point along a cost–confidence spectrum without touching the deployed model.
- We rebuild the experimental package with saved query sets, target labels, per-model matching rates, JSON summaries, runtime and candidate-count records, and reproducibility snapshots, so that the reported evidence is traceable and every reported number can be checked independently.
- We expand the evaluation beyond the original CIFAR-10 setting: GTSRB serves as a second primary dataset; CIFAR-100 and Tiny-ImageNet are reported as bounded complexity stress tests; VGG16, MobileNetV2, and EfficientNetB0 source pools test non-ResNet source architectures; and a reconstructed 74-model CIFAR-10 benchmark is included, broadening the evidence base from a single-dataset demonstration to a multi-dataset, multi-backbone study.
- We add stress tests for reviewer-raised deployment and ownership-boundary concerns, including hard negatives, model fine-tuning, quantization, label-only and logit-soft extraction, simulated API constraints, threshold sensitivity on CIFAR-10, GTSRB, and EfficientNetB0-source diagnostics, shadow-pool sensitivity, loss ablations, query-seed repeats, and protocol-aware MetaFinger/ADV-TRA baseline comparisons, clarifying the conditions under which fingerprint evidence remains reliable and exposing its current limits.

Together, these contributions move DNN model fingerprinting from boundary-transfer heuristics toward an explicit, cost-aware, and transparently bounded ownership-verification methodology: the proposed objective states what a good fingerprint must separate, the two variants state

how much confidence each construction budget buys, and the expanded evaluation states where the resulting evidence applies.

2 Related work

2.1 Model watermarking

Model watermarking embeds copyright verification information into a DNN model and later verifies ownership through white-box inspection or black-box queries. This embedding can affect model usability because the source model itself is modified. Uchida et al. [11] proposed a parameter-based watermarking method that adds a regularization term to the loss function to embed verification information into model weights. This method requires white-box access for verification. In contrast, Adi et al. [12] and Zhang et al. [13] developed query-set watermarking methods based on backdoor triggers. These approaches use unrelated trigger images and require only black-box access during verification. Li et al. [14] embedded external features as model watermarks and evaluated the method under model extraction attacks. Guo et al. [15] proposed domain watermarking, which constructs watermark samples from a hard-to-generalize domain of the original dataset.

2.2 Model fingerprinting

Model fingerprinting does not modify the source model. Instead, the owner extracts a fingerprint query set from the trained source model and verifies ownership by querying a suspected model. Li et al. [16] used comparative model cosine similarity to measure knowledge similarity between suspected and source models. Papernot et al. [17] showed that adversarial examples can transfer to models with similar decision boundaries, motivating adversarial-query fingerprints. Building on this idea, Cao et al. [7] proposed IPGuard, which uses boundary adversarial examples as a fingerprint query set for identifying DNN models. Lukas et al. [9] introduced conferrable examples, which are constructed to transfer to stolen models while avoiding benign independent models. Yang et al. [10] proposed META fingerprinting, which adds noise to natural samples and uses meta-training to fingerprint internal decision regions. Pan et al. [18] proposed META-V, which employs a meta-classifier for fingerprint verification across classification, regression, and generative tasks. More recent work continues to refine fingerprint-evidence design: NaturalFinger generates GAN-based natural queries for decision-difference areas [19], ADV-TRA uses adversarial trajectories to improve robustness against decision-boundary changes [20], MarginFinger controls fingerprint distance to the classification boundary [21], robustness-fingerprint technology strengthens post-processing tolerance [22], and FIT-Print targets false-claim-resistant ownership verification through targeted fingerprints [23].

These works show that decision-boundary behavior and transferability are useful sources of fingerprint evidence. Our method differs from prior work in its optimization target: it explicitly maximizes separation between source-derived positives and benign negatives, rather than only exploiting source-boundary transferability. This design is intended to reduce false-accusation risk under hard negatives such as same-seed retraining, source-initialized benign models, partial-transfer benign models, and cross-architecture benign models. We also distinguish local numerical wrappers, official-protocol adaptations, and project-adapted protocol checks in the experiments, because these settings make different assumptions and should not be conflated with byte-for-byte reproductions of every original baseline environment.

3 Method

3.1 Threat model

We consider a black-box ownership-verification setting with two parties: a source model owner and a potential adversary. The owner trains a DNN source model using legally controlled data and may deploy the model as a local service or through an API. During query construction, the owner has white-box access to the protected source model and can train local shadow models. During verification, the owner only queries the suspected model and observes its returned labels or scores; the suspected model's parameters and training data are not assumed to be available.

Positive suspected models are source-related models. They may be produced by directly copying the source model, by post-processing the source model, or by extracting the source model through black-box queries. We consider fine-tuning, pruning, weight perturbation, quantization, knowledge distillation, label-only extraction, logit-soft extraction, and data-free distillation. Negative suspected models are benign models that are independently trained or otherwise not derived from the protected source. To evaluate false-accusation risk, we further include hard negatives such as same-seed retraining, overlapping-data benign models, cross-architecture benign models, source-initialized benign models, and partial-transfer benign models.

The verification goal is to assign high matching rates to source-related positives and low matching rates to benign negatives. We do not claim a legal proof of ownership or

zero false-accusation risk. Instead, we report empirical separation over explicit positive and negative model pools. We also evaluate bounded API-style constraints, including query budgets, response dropout, response truncation, label noise, preprocessing, and adaptive filtering. These experiments are controlled service validations and simulations; they should not be interpreted as universal robustness against arbitrary commercial APIs.

3.2 Fingerprint method framework

Our fingerprinting pipeline has two stages: generating the shadow-model pool and generating fingerprint queries. As illustrated in Fig. 2, the first stage trains a series of shadow models, and the second stage uses these shadow models to optimize images under the proposed loss. The resulting query set identifies DDRs that separate source-derived or post-processed models from negative models.

3.3 Model pool generation

In the model pool generation stage, we apply the aforementioned model modification and model extraction to the source model to obtain a batch of shadow models, forming the positive model pool. Subsequently, we train models from scratch using architectures identical to those in the positive model pool, thereby creating a corresponding set of independently trained models that constitute the negative model pool.

3.4 Decision difference region formalization

Let F_s denote the protected source model, $\mathcal{P} = \{F_i^p \mid i = 1, \ldots, m_p\}$ denote source-related positive shadow models, and $\mathcal{N} = \{F_j^n \mid j = 1, \ldots, m_n\}$ denote benign negative shadow models. For an input x, a perturbation δ, and a target label y_t, we define the positive and negative target-response rates as

$$R_{\mathcal{P}}(x+\delta, y_t) = \frac{1}{m_p} \sum_{i=1}^{m_p} \mathbb{I}\left[\arg\max F_i^p(x+\delta) = y_t\right], \quad (1)$$

$$R_{\mathcal{N}}(x+\delta, y_t) = \frac{1}{m_n} \sum_{j=1}^{m_n} \mathbb{I}\left[\arg\max F_j^n(x+\delta) = y_t\right]. \quad (2)$$

A fingerprint query is desirable when it lies in a region where the positive response is high and the negative response is low. We therefore view DDR search as maximizing the positive-negative ownership-boundary gap

$$G(x+\delta, y_t) = R_{\mathcal{P}}(x+\delta, y_t) - R_{\mathcal{N}}(x+\delta, y_t). \quad (3)$$

This definition separates DDR search from ordinary transfer-based adversarial examples. A purely transferable example may activate many related models, including benign models with similar decision behavior. In contrast, DDR query construction explicitly penalizes responses on negative shadow models and is therefore aligned with the ownership-verification objective. For a constructed query set Q, the verification margin can be viewed as the empirical positive-negative matching-rate gap,

$$\Delta(Q) = \mathbb{E}_{F\in\mathcal{P}}\left[M(Q,F)\right] - \mathbb{E}_{F\in\mathcal{N}}\left[M(Q,F)\right], \quad (4)$$

where $M(Q,F)$ denotes the fraction of queries in Q that match their stored target labels on model F. Maximizing the DDR gap is thus directly related to increasing this empirical margin and reducing false responses on the modeled benign pool. This is not a distribution-free proof, because $\mathcal{P}$ and $\mathcal{N}$ are finite shadow pools, but it makes the claim boundary explicit.

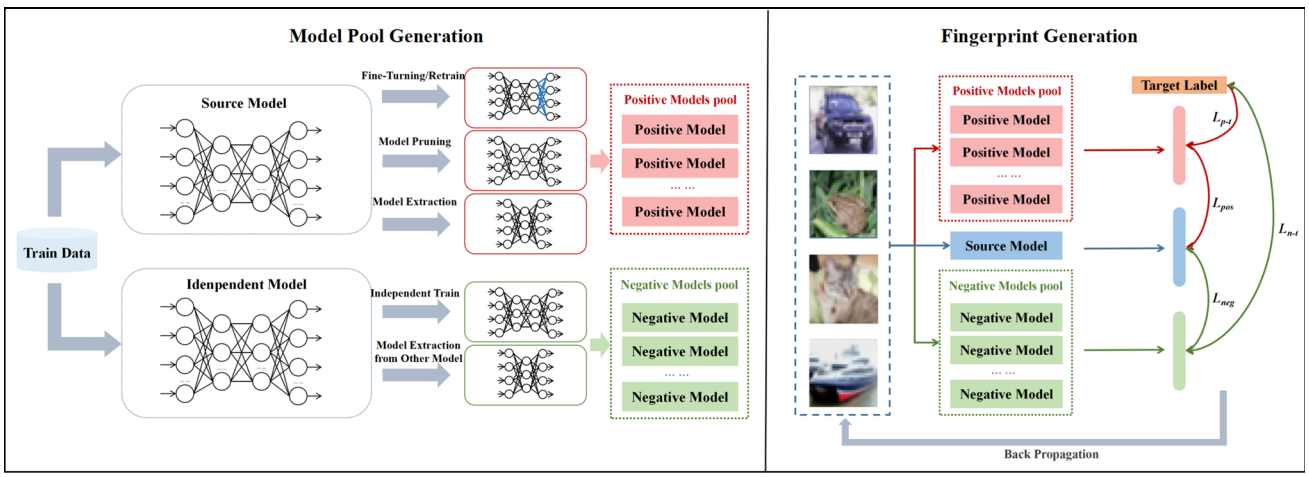


Fig. 2 Overview of the fingerprint-generation framework

Algorithm 1 Fast variant fingerprint-query construction.

Require: Initial samples X_m, source model F_s, positive pool $\mathcal{P}$, negative pool $\mathcal{N}$, learning rate η, epochs E, subset size k, query budget n, loss weights $\alpha, \beta, \gamma, \zeta$
Ensure: Fingerprint query set Q
$Q \leftarrow \emptyset$
Select training subsets $\mathcal{P}_k \subset \mathcal{P}$ and $\mathcal{N}_k \subset \mathcal{N}$
for each $x \in X_m$ until $|Q| = n$ **do**
  Sample a target label $y_t \neq y_a$
  Initialize $x' \leftarrow x$
  for $i = 1, \ldots, E$ **do**
    $\mathcal{L} \leftarrow \alpha L_{p-t} - \beta L_{n-t} + \gamma L_{pos} - \zeta L_{neg}$
    $x' \leftarrow \text{clip}(x' - \eta \nabla_{x'} \mathcal{L}, 0, 1)$
  end for
  $Q \leftarrow Q \cup \{x'\}$
end for
return Q

Algorithm 2 Selected variant fingerprint-query construction with conferrability filtering.

Require: Initial samples X_m, source model F_s, training pools $\mathcal{P}, \mathcal{N}$, validation pools $\mathcal{P}^v, \mathcal{N}^v$, learning rate η, epochs E, threshold τ, query budget n
Ensure: Filtered fingerprint query set Q
$Q \leftarrow \emptyset$
Select training subsets $\mathcal{P}_k \subset \mathcal{P}$ and $\mathcal{N}_k \subset \mathcal{N}$
for each $x \in X_m$ until $|Q| = n$ **do**
  Sample a target label $y_t \neq y_a$
  Optimize x into a candidate x' by minimizing $\mathcal{L}$ over E epochs on $\mathcal{P}_k, \mathcal{N}_k$
  $s \leftarrow \text{Confer}(\mathcal{P}^v, \mathcal{N}^v, x', y_t)$
  if $s \geq \tau$ **then**
    $Q \leftarrow Q \cup \{x'\}$
  end if
end for
return Q

3.5 Model fingerprint generation

In the fingerprint generation stage, our method creates fingerprint query sets by optimizing natural samples toward high positive target-response and low negative target-response. These query sets are subsequently used for black-box ownership verification of suspected models. A simplified target condition for an optimized query is

$$\hat{F}_{sou}(x) = y_a, \hat{F}_{pos}(x + \Delta x) = y_b, \hat{F}_{neg}(x + \Delta x) \neq y_b \quad (5)$$

where $\hat{F}_{pos}$ and $\hat{F}_{neg}$ denote positive and negative models from the model pool, y_a is the original label, and y_b is the target label. Before fingerprint generation, we randomly select m samples from one class to form the initial image subset X_m. We then optimize each clean sample x into a fingerprint query $x + \Delta x$. An optimized query is expected to induce the target label on source-related positive models while suppressing the same response on benign negative models. We propose two variants: Fast variant and Selected variant. Fast variant directly accepts optimized queries, whereas Selected variant further filters candidates by conferrability.

Target-Label selection We randomly select k positive models $\{F_{pos1}, F_{pos2}, \ldots, F_{posk}\}$ and k negative models $\{F_{neg1}, F_{neg2}, \ldots, F_{negk}\}$ from the positive and negative pools, respectively. Each sample in X_m is optimized toward a target label y_t different from its original label.

Fingerprint-Sample optimization For each image x_i, we compute the average prediction behavior of the selected positive and negative subsets and update the image by gradient descent.

The loss consists of four components. In our default setting, we set the weights $\alpha = \beta = \gamma = \zeta = 1$, and later report ablations that remove individual terms. L_{p-t} and L_{n-t} represent the prediction losses of the positive and the negative model pools with respect to the target labels, respectively, as defined in (6) and (7).

$$L_{p-t} = L_{\text{CE}} \left(\frac{1}{k} \sum_{i=1}^{k} F_{pos_i}(x'), y_t \right) \quad (6)$$

$$L_{n-t} = L_{\text{CE}} \left(\frac{1}{k} \sum_{i=1}^{k} F_{neg_i}(x'), y_t \right) \quad (7)$$

where $L_{\text{CE}}(\cdot)$ denotes the cross-entropy loss between the model output and the target label y_t, and x' denotes the current optimized query. The objective minimizes L_{p-t} to encourage the positive model pool to output the target label for x'. Conversely, it maximizes L_{n-t} to discourage the negative model pool from predicting the same target label.

The third and fourth terms of the loss function are shown in (8) and (9).

$$L_{pos} = \text{KL} \left(F_{sou}(x'), \frac{1}{k} \sum_{i=1}^{k} F_{pos_i}(x') \right) \quad (8)$$

$$L_{neg} = \text{KL} \left(F_{sou}(x'), \frac{1}{k} \sum_{i=1}^{k} F_{neg_i}(x') \right) \quad (9)$$

where $\text{KL}(\cdot)$ denotes the Kullback-Leibler divergence between predicted probability distributions. Minimizing L_{pos} reduces the output difference between the positive model subset and the source model. Conversely, maximizing L_{neg} increases the output difference between the negative model subset and the source model.

Based on the loss function, we use stochastic gradient descent (SGD) to perform E epochs of gradient-based optimization on the original samples. The process stops after obtaining n fingerprint queries, which collectively form Q.

Selected variant, as shown in Algorithm 2, follows the same candidate-generation process as Fast variant and then applies a query-set filtering module.

Query-Set filtering The filtering module selects high-confidence queries from the generated candidate set. We reserve several models from the positive and negative pools as validation models, which are excluded from gradient-based query generation. To estimate whether a candidate transfers to positive models while avoiding negative models, we use the conferrability score proposed by Lukas et al. [9]. Its definition is as follows.

$$\text{Transfer}(F, x; y_t) = \Pr[F(x) = y_t]. \tag{10}$$

$$\text{Confer}(\mathcal{P}^v, \mathcal{N}^v, x, y_t) = \overline{\text{Transfer}}_{\mathcal{P}^v}(x; y_t)\left(1 - \overline{\text{Transfer}}_{\mathcal{N}^v}(x; y_t)\right). \tag{11}$$

where $\text{Transfer}(\cdot)$ denotes the predicted probability that sample x is classified as the target label y_t by model F, and the overline denotes the mean over the held-out validation pool. A higher $\text{Confer}(\cdot)$ score indicates stronger target-label transfer to positive models and weaker target-label transfer to negative models. We include candidates whose score exceeds threshold τ in the final query set, and generation stops after obtaining n queries.

4 Experiments

To evaluate the separation behavior and robustness diagnostics of the proposed scheme, we test three types of model modification attacks, multiple model extraction attacks, hard-negative ownership-boundary cases, source-architecture diversity, simulated service constraints, and ablation settings. CIFAR-10 [24] remains the main benchmark and includes both a compact pool and a reconstructed 74-model benchmark. GTSRB is used as the second primary dataset, while CIFAR-100 and Tiny-ImageNet are used as supportive complex-dataset evidence. We compare against local IPGuard [7], META-style [10], conferrable-style [9], and UAP-style wrappers, as well as protocol-aware MetaFinger and ADV-TRA [20] baseline adaptations. Unless otherwise specified, the reported positive mean, negative mean, margin, and ARUC are computed from saved per-model matching-rate files.

4.1 Dataset splitting

We assume that both the source model owner and potential infringers possess sample data with the same distribution for model training. The CIFAR-10 training set was partitioned into two class-balanced subsets through stratified sampling. The source model and positive models are trained using one subset of training data, while the benign negative models use the other. For fine-tuning, we divide the test dataset into two subsets of equal size. One subset is used for fine-tuning training, while the other subset is used for evaluating model accuracy.

For the expanded evaluation, the same principle is applied to additional datasets and source architectures: source-related positives are kept separate from independently trained negatives, and query construction/evaluation model pools are recorded explicitly. For baseline protocol checks, we distinguish three levels of evidence: local numerical wrappers, official-protocol adaptations on our ownership-verification pool, and split-strict or protocol-tightened checks with explicit train/validation/test separation. We do not claim byte-for-byte reproduction of every original baseline training environment.

4.2 Fingerprint-Training model pool

Our fingerprint-training model pool comprises five positive and five negative models selected from four architectures: ResNet18 [25], VGG13, VGG16 [26], and MobileNetV1 [27]. Positive models are created through knowledge distillation, while negative models are trained from randomized initializations.

4.3 Benchmark model pool

In the CIFAR-10 benchmark setting, we use the ResNet18 architecture as the source model. To provide a fair evaluation and comparison of model fingerprint methods, we construct a benchmark model pool containing 74 models trained on CIFAR-10, encompassing both positive and negative models. The composition of the benchmark model pool is detailed in Table 1. We apply three model modification attacks and four model extraction attacks to the source model to generate positive models with various model architectures and hyperparameter configurations.

Model fine-tuning We consider four fine-tuning attack methods used by Cao et al. [7]: Fine-Tuning Last Layer (FTLL), Fine-Tuning All Layers (FTAL), Re-Train Last Layer (RTLL), and Re-Train All Layers (RTAL). We set the number of fine-tuning epochs to 10.

Model weight pruning We apply the pruning method described in Han et al. [28], which removes the p percent of parameters with the smallest absolute values. The pruning rate ranges from 0.1 to 0.9 with a stride of 0.1. Because

Table 1 Composition of the CIFAR-10 benchmark model pool

	Attack	Detail	Number
Positive Models	Source model	Independently training	1
	Fine-Tuning	FTLL	1
		FTAL	1
		RTLL	1
		RTAL	1
	Weight Pruning	Pruning rate p = 0.1,0.2…0.9	9
	Weight perturbation	$\alpha = 9, 7, 5, 3, 1$	5
	Extraction Attacks	Distillation	8
		Knockoff	8
		Logit-query	1
		Data-free Distillation	1
Negative Models	Benign models	Train independently for ResNet18	22
		Train independently for other architectures	7
	Extraction Attacks	Distillate	8

ownership verification should be evaluated on usable suspect models, we fine-tune the pruned model to restore accuracy.

Gaussian weight perturbation We apply the weight-perturbation method described in Orekondy et al. [29] by injecting Gaussian noise into the model weights, followed by fine-tuning to restore accuracy. The perturbation operation is

$$W' = W + \epsilon, \quad \epsilon \sim \mathcal{N}\left(0, \sigma_\epsilon^2\right), \quad \sigma_\epsilon = \frac{\text{std}(W)}{\alpha}. \tag{12}$$

where W denotes the model weights and std$(\cdot)$ denotes the standard deviation. The parameter α controls the noise intensity; a smaller α corresponds to stronger injected noise.

Model extraction For same-architecture extraction, we evaluate four attacks: Distillation, Knockoff, Logit-query, and Data-free Distillation. For model distillation, we implement the classical distillation algorithm proposed by Hinton et al. [30]. For Knockoff [29], the attacker uses predicted labels from the source model as pseudo-labels to train the stolen model. For Logit-query, the attacker trains the stolen model by querying the source model and minimizing the output difference between the source and stolen models. For data-free distillation, which relaxes the attacker's data requirement, we use a zero-shot learning based approach [31]. To evaluate different extracted-model architectures, we apply model distillation to seven common architectures: ResNet34 [25], VGG13, VGG16, VGG19 [26], DenseNet121 [32], MobileNetV1 [27], and MobileNetV2 [33].

For negative-model construction, we match the positive-model settings one-to-one where possible, but train the negative models independently with different random seeds. We also apply distillation to an independent model to obtain extraction-style negative models.

4.4 Evaluation metrics

Our copyright-verification rule is as follows. Given a predefined threshold $\tau \in [0, 1]$, we compare the fingerprint matching rate with the threshold. If the score is higher than τ, the suspected model is identified as positive, i.e., source-related; otherwise, it is identified as negative. We use the following evaluation metrics.

Matching-rate margin The matching-rate margin is calculated by subtracting the matching rate of negative models from the matching rate of positive models. It measures the separation between source-related and benign models.

Area under the robustness-uniqueness curve (ARUC) ARUC is proposed in Cao et al. [7]. Robustness, also called true positive rate (TPR), measures the proportion of positive suspected models identified correctly. Uniqueness, also called true negative rate (TNR), measures the proportion of negative suspected models identified correctly. As the matching-rate threshold increases from 0 to 1, ARUC measures the area under the robustness-uniqueness curve; operationally, we compute it by summing min(TPR, TNR) over uniformly spaced thresholds. A higher ARUC indicates better threshold-tolerant separation.

Positive and negative mean In addition to the aggregate margin, we report the mean matching rate on positive suspected models and the mean matching rate on negative suspected models. The positive mean reflects robustness to source-related model changes, while the negative mean

Table 2 Expanded dataset and source-architecture evidence. CIFAR-10 and GTSRB are the primary datasets; CIFAR-100 and Tiny-ImageNet are bounded complexity stress tests; non-ResNet rows are bounded source-backbone diagnostics

Setting	Data/source	Pos.	Neg.	Margin	ARUC
CIFAR-10 Train20	CIFAR-10	0.9600	0.0067	0.9533	0.9467
CIFAR-10 all-negative, $\tau = 0.8$	CIFAR-10	1.0000	0.0000	1.0000	1.0000
GTSRB Train20, $\tau = 0.5$	GTSRB	0.9450	0.0000	0.9450	0.9400
CIFAR-100 Train20, $\tau = 0.5$	CIFAR-100	0.7550	0.0000	0.7550	0.7500
CIFAR-100 no-fallback	CIFAR-100	0.5900	0.0000	0.5900	0.5867
Tiny-ImageNet, $\tau = 0.005$	Tiny-ImageNet	0.2100	0.0000	0.2100	0.2000
Tiny-ImageNet, $\tau = 0.01$	Tiny-ImageNet	0.2550	0.0000	0.2550	0.2500
VGG16 source mean	VGG16	0.9667	0.0000	0.9667	0.9633
MobileNetV2 source mean	MobileNetV2	0.9867	0.0000	0.9867	0.9850
EfficientNetB0 source mean	EfficientNetB0	0.9900	0.0300	0.9600	0.9617

reflects uniqueness against benign models. Reporting both values is important because a high margin can arise from high positive matching, low benign matching, or both.

4.5 Expanded evaluation scope

Table 2 summarizes the expanded dataset and source-architecture evidence. CIFAR-10 remains the main benchmark. GTSRB is used as the main second dataset and reaches a margin of 0.9450 with zero negative mean. CIFAR-100 provides additional complex-dataset support. Tiny-ImageNet is included as a harder stress case rather than as a main generalization claim: its negative mean remains zero in these runs, but its positive matching rates are low. A subsequent strict no-fallback refinement sweep over seeds 309/410/411 and thresholds $\tau = 0.01/0.015$ did not obtain 100 accepted queries within 6000–10000 candidates, so we report Tiny-ImageNet as bounded stress evidence instead of promoting it to a solved generalization setting. The VGG16, MobileNetV2, and EfficientNetB0 rows address the single-source-architecture concern across three tested non-ResNet classification backbones. The EfficientNetB0 diagnostic uses an EfficientNetB0 source model with ResNet18 source-derived positives and eight benign negatives, including same-architecture EfficientNetB0 holdouts; Selected variant reaches a three-seed mean margin of 0.9600 and ARUC of 0.9617, while local IPGuard and META-style wrappers reach ARUC values of 0.1988 and 0.7338, respectively. A compact EfficientNetB0 threshold sweep is also reported in Table 5. This remains bounded classification-backbone evidence rather than support for all modern vision pipelines. DeepLab-style segmentation [34] and related high-resolution deep-vision applications [35], as well as EfficientNet-style model scaling [36] and related applied computer-vision pipelines [37], are therefore left as future extensions beyond the tested classification setting.

Table 3 summarizes the baseline evidence and the corresponding protocol scope. On the clean CIFAR-10 pool, the official-protocol adapted MetaFinger baseline is strong. Our default Selected variant setting is slightly weaker, whereas the strict all-negative operating point reaches the same ARUC on this clean pool. The hard-negative stress and protocol-tightened rows evaluate ownership-boundary behavior and protocol sensitivity; they should not be read as a blanket claim that one method wins every original baseline setting. Rows with different assumptions are labelled as local wrappers, official-protocol adaptations, or project-adapted checks, so they are not treated as exact original-environment reproductions.

The diagnostics in Table 4 cover model modification, extraction, hard negatives, API-style constraints, distribution

Table 3 Protocol-aware CIFAR-10 baseline comparison. Rows use different assumptions, so the table clarifies both numerical performance and comparison scope

Method / row	Protocol level	Margin	ARUC
IPGuard local wrapper	Local numerical wrapper	0.1033	0.1283
META-style local wrapper	Local numerical wrapper	0.7133	0.7133
Conferrable local wrapper	Local numerical wrapper	0.6467	0.6483
UAP-style local wrapper	Local numerical wrapper	0.1100	0.1167
MetaFinger official-protocol	Official-protocol adaptation	1.0000	1.0000
Ours all-negative, $\tau = 0.8$	Same CIFAR-10 pool	1.0000	1.0000
MetaFinger split-strict	Split train/val/test pools	0.860–0.881	0.862–0.884
ADV-TRA tightened	Project-adapted protocol	0.277–0.325	0.277–0.328

Table 4 Modification, extraction, hard-negative, service-constraint, and distribution-shift diagnostics

Setting	Dataset/pool	Margin	ARUC
Noise / pruning	CIFAR-10	0.8700	0.8650
Fine-tuning FTLL / FTAL	CIFAR-10	0.8433	0.8367
Quantization, 8/6/4-bit	CIFAR-10	0.8667	0.8600
Label-only extraction	CIFAR-10	0.6400	0.6433
Logit-soft extraction	CIFAR-10	0.8133	0.8067
All modified/extracted models vs hard negatives	CIFAR-10	0.7293	0.7277
Broader 32-negative benign boundary	CIFAR-10	0.8900	0.8878
Original-74 broader benign boundary	CIFAR-10	0.8016	0.8066
Existing queries on shifted negatives	CIFAR-10	0.8634	0.8610
Shift-aware negatives, p2/n12 mean	CIFAR-10	0.9585	0.9614
Pool-imbalance p10/n16 mean	CIFAR-10	0.7448	0.7431
Basic noise, pruning, and quantization	GTSRB	0.9022	0.8944
Fine-tuning FTLL / FTAL	GTSRB	0.8900	0.8800
Deployed-service simulation, top-1	CIFAR-10	0.3063	0.3066
Deployed-service simulation, top-1	GTSRB	0.5513	0.5513
Adaptive filtering simulation, top-1	CIFAR-10	0.0744	0.0744
Adaptive filtering simulation, top-1	GTSRB	0.2460	0.2460

shift, and pool imbalance. On shifted CIFAR-10 benign pools, reusing existing selected queries already keeps a positive margin, while reconstructing queries with shifted negative shadows improves the three-seed mean margin to 0.9585 and ARUC to 0.9614. The pool-imbalance sweep also shows a trade-off: increasing negative-shadow coverage can suppress false matches, but excessive negative emphasis lowers positive matching. Therefore, the API and shift rows support bounded robustness diagnostics rather than arbitrary-service or arbitrary-shift guarantees.

The ablation, threshold, shadow-pool, and repeat-stability evidence is summarized in Table 5. The loss rows show

Table 5 Ablation, threshold sensitivity, shadow-pool sensitivity, and query-seed repeat evidence. Dataset abbreviations are used for compactness: C10 denotes CIFAR-10, C100 denotes CIFAR-100, EffB0 denotes EfficientNetB0-source diagnostics, and TI denotes Tiny-ImageNet. TI rows are reported as bounded stress evidence rather than full complex-dataset generalization

Setting	Dataset	Margin	ARUC
Full loss	GTSRB	0.9400	0.9350
No positive-target term	GTSRB	0.0000	0.0000
KL-only	GTSRB	0.0567	0.0567
Target-only	GTSRB	0.7500	0.7500
CIFAR-10 all-neg seed mean	C10	1.0000	1.0000
GTSRB query-seed mean	GTSRB	0.9300	0.9233
GTSRB $\tau = 0.3/0.5/0.7$ sweep	GTSRB	0.845–0.995	0.840–0.995
EfficientNetB0 $\tau = 0.3/0.5/0.7$ sweep	EffB0	0.899–0.995	0.897–0.997
Shadow pool $p1/n1$, same-arch	C10	0.6253	0.6272
Shadow pool $p3/n3$, mixed	C10	0.7195	0.7203
Shadow pool $p5/n5$, same-arch	C10	0.7165	0.7165
Shadow pool $p5/n5$, mixed	C10	0.7207	0.7205
CIFAR-100 no-fallback sensitivity mean	C100	0.5900	0.5867
Tiny-ImageNet no-fallback, $\tau = 0.005$, seeds 410/411	TI	0.165–0.270	0.160–0.265
Tiny-ImageNet fallback-assisted, $\tau = 0.01$, seeds 410/411	TI	0.270–0.385	0.270–0.375
Tiny-ImageNet target-only loss, seed 411	TI	0.4700	0.4650
Tiny-ImageNet KL-only loss, seed 411	TI	0.0167	0.0133
Tiny-ImageNet no positive-target loss, seed 411	TI	0.0100	0.0000

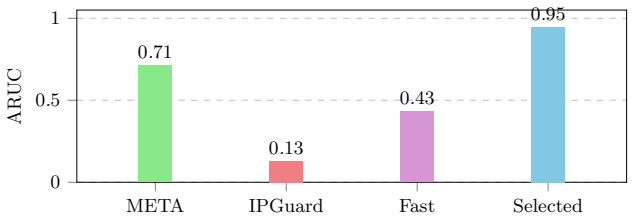


Fig. 3 Representative same-pool ARUC comparison on CIFAR-10. Selected variant achieves the largest ARUC among these rows, while Fast variant provides a lower-cost proposed operating point; protocol-aware MetaFinger and ADV-TRA comparisons are reported separately in Table 3

that the full objective is important, especially relative to no-positive-target and KL-only variants. The shadow-pool rows show weaker separation for a small same-architecture $p1/n1$ pool and more stable margins for the tested $p3/n3$ and $p5/n5$ configurations; they support sensitivity analysis rather than a universal pool-size optimum. On Tiny-ImageNet, target-only optimization can raise positive matching in one seed, but the KL-only and no-positive-target variants collapse, and the strict no-fallback sweep fails to produce complete 100-query sets under the tested budgets. We therefore treat Tiny-ImageNet as a stress test that exposes a current limitation, not as evidence of full complex-dataset generalization.

4.6 CIFAR-10 benchmark result

4.6.1 Overall results on the benchmark model pool

Figure 3 summarizes representative same-pool ARUC results on CIFAR-10. In this local benchmark comparison, Selected variant gives a high-separation operating point, and Fast variant provides a lower-cost alternative. The expanded baseline analysis in Table 3 also shows that official-protocol adapted MetaFinger is very competitive under clean same-pool conditions. For this reason, we report strict operating points and hard-negative stress separately instead of making a blanket win claim over every baseline setting.

4.6.2 Robustness against model modification

To evade detection by IP protection methods, attackers may modify stolen model weights using fine-tuning, weight pruning, or Gaussian weight perturbation.

Fine-Tuning Figure 4(a) illustrates the results of fine-tuning on CIFAR-10. When only the last layer is modified, IPGuard has a higher margin than our variants, which is consistent with its use of adversarial examples tightly tied to the source model's local boundary. When all layers are modified or retrained, IPGuard's margin decreases, while our selected queries preserve a more stable positive-negative separation.

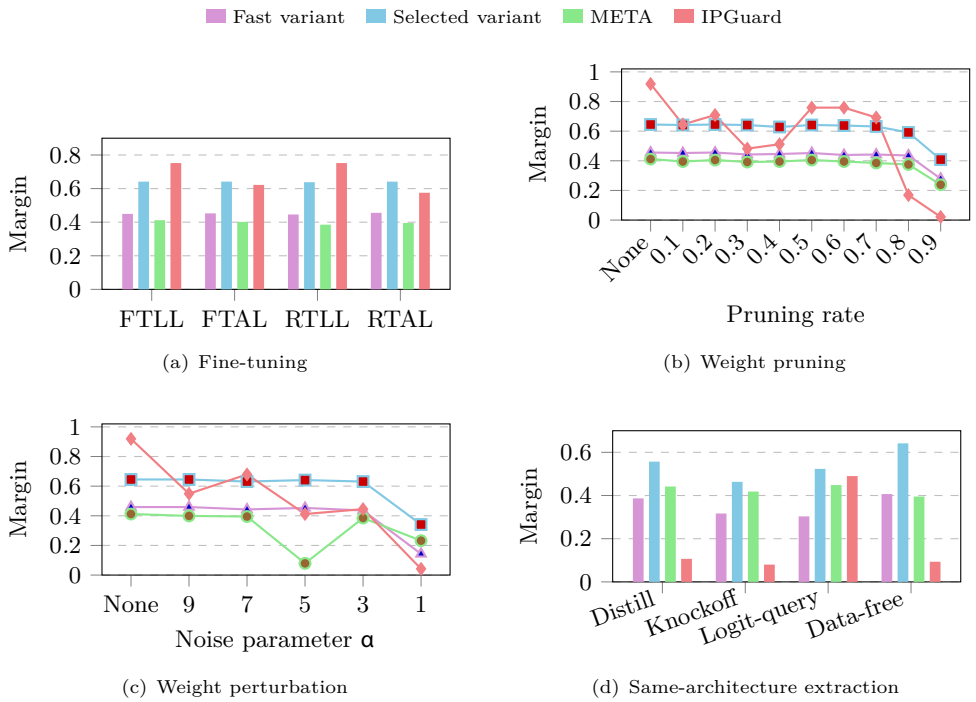


Fig. 4 Matching-rate margins under source-model modification and same-architecture extraction. Higher margins indicate stronger separation between source-related positive models and benign negative models; the sharp drops under severe pruning or Gaussian weight perturbation show the harder stress cases

This result should be interpreted as attack-dependent robustness rather than a universal claim.

Weight pruning Figure 4(b) illustrates the results of weight pruning. As the pruning rate increases, model performance gradually decreases, and the matching-rate margin of all methods tends to decline. Our method maintains a relatively high margin, with Selected variant retaining about 0.6 matching-rate margin when the pruning rate is 0.8.

Gaussian weight perturbation Figure 4(c) shows the results of injecting Gaussian noise into model weights. As noise intensity increases, all methods become less stable, but Selected variant maintains a larger margin than the compared methods in most tested settings. This suggests that filtering by the positive-negative shadow-pool gap can improve robustness to weight perturbation, while the sharp drop at the strongest noise level still shows that severe post-processing remains a harder case.

4.6.3 Robustness against model extraction

Beyond illegally copying the source model, attackers may obtain stolen models by querying the source model and training extracted models. We evaluate the robustness of all fingerprint methods against these attacks.

Model extraction under the same architecture As illustrated in Fig. 4(d), when the extracted and source models have the same architecture, Selected variant obtains larger margins than the compared methods across the tested extraction modes. The advantage is most visible for data-free distillation in this benchmark. We treat this as bounded evidence for same-architecture extraction robustness, rather than a guarantee for all extraction algorithms or all service settings.

Model extraction under different architectures We further evaluate the robustness of the fingerprints in detecting model extraction attacks across different extracted-model architectures. Figure 5 shows the results. In these tested architectures,

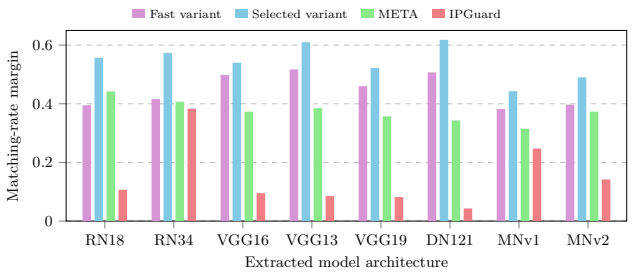


Fig. 5 Matching-rate margins for model extraction under different extracted-model architectures. RN, DN, and MNv denote ResNet, DenseNet, and MobileNetV variants, respectively

Selected variant consistently obtains a larger matching-rate margin than the compared local baselines, while Fast variant gives a lower-cost but weaker operating point. We use this result as bounded cross-architecture extraction evidence; it does not establish architecture-independent robustness for arbitrary backbones.

4.6.4 Construction cost and runtime complexity

We analyze the offline construction cost and then report the measured construction time. Let N_q be the required number of fingerprint queries, C_{fast} and C_{sel} be the numbers of processed candidates for Fast variant and Selected variant, E be the number of optimization epochs per candidate, K_p and K_n be the numbers of positive and negative training shadow models, and T_{fb} be the cost of one forward-backward pass through a shadow model. Fast variant optimizes candidates and accepts them without conferrability filtering. Its offline construction cost is therefore

$$T_{\text{fast}} = O\big(C_{\text{fast}} E(1 + K_p + K_n) T_{\text{fb}}\big), \tag{13}$$

where C_{fast} is usually close to N_q when no extra rejection criterion is used. Selected variant has the same candidate-optimization term, but additionally evaluates each candidate on held-out positive and negative validation shadow pools. If K_p^v and K_n^v are the corresponding validation-pool sizes and T_{fw} is the cost of a forward pass, then

$$T_{\text{sel}} = O\Big(C_{\text{sel}} E(1 + K_p + K_n) T_{\text{fb}} + C_{\text{sel}}(K_p^v + K_n^v) T_{\text{fw}}\Big). \tag{14}$$

Because Selected variant rejects candidates whose conferrability scores are below the threshold, C_{sel} can be much larger than N_q. Thus Selected variant is expected to be slower than Fast variant, and both variants should be interpreted as an offline cost-confidence trade-off. Once the query set is constructed, online verification only requires N_q black-box queries to the suspected model and matching-rate computation against the stored target labels, i.e., $O(N_q)$ API calls and $O(N_q)$ label comparisons.

For the empirical timing comparison, we generate query sets comprising $N_q = 100$ fingerprint query samples. The measured construction times are reported in Table 6.

Fast variant is not the fastest baseline in this benchmark. It constructs the query set in about 404.6 seconds, which is slightly faster than META but slower than IPGuard, and should therefore be interpreted as a lower-cost screening option relative to Selected variant. Selected variant is slower than all methods in Table 6 because conferrability filtering requires more candidate processing and validation-pool eval-

Table 6 Measured offline construction time for $N_q = 100$ fingerprint queries

Method	Construction time (s)
IPGuard	187.24 ± 9.31
META	422.79 ± 17.87
Fast variant	404.6 ± 0.6
Selected variant	861.45 ± 27.78

uation. Its role is offline high-confidence verification rather than low-latency query construction.

4.6.5 Conferrability score threshold

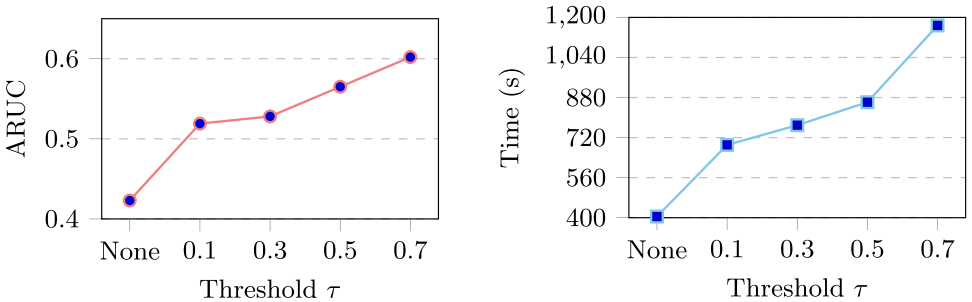
We vary the conferrability threshold in Fig. 6. Higher thresholds generally improve ARUC but also increase construction time because more candidates are rejected. After 0.5, the additional ARUC gain is smaller relative to the extra cost, so we use 0.5 as the default CIFAR-10 operating point and report additional settings in Table 5. The EfficientNetB0-source sweep shows the same cost-confidence behavior on a non-ResNet source: ARUC increases from 0.8971 to 0.9967 as τ increases from 0.3 to 0.7, while average construction time rises from 76.6 s to 408.8 s.

5 Limitations

The expanded experiments broaden the original evaluation, but several boundaries remain. The baseline rows are local wrappers, official-protocol adaptations, or project-adapted checks rather than byte-for-byte reproductions of every original baseline environment. The API-style experiments are controlled service simulations, not evidence for arbitrary commercial rate limits, repeated-query detection, or unknown service-side defenses. The hard-negative experiments provide finite-pool evidence on false-accusation risk, not a formal zero-risk guarantee. Tiny-ImageNet remains the main generalization weakness: negative matching stays low, but positive matching is much lower than on CIFAR-10 and GTSRB, and strict no-fallback refinement did not produce complete query sets under the tested budgets. The distribution-shift, pool-imbalance, shadow-pool, and EfficientNetB0-threshold diagnostics reduce reviewer concerns, but they are still finite tests rather than guarantees under arbitrary shifts, shadow-pool sampling, or architectures. VGG16, MobileNetV2, and EfficientNetB0 support three non-ResNet classification source families, but not segmentation-style pipelines such as DeepLab. DDR provides a formal ownership-boundary objective and empirical evidence, but remains a shadow-model-based query optimization framework rather than a distribution-free guarantee.

6 Conclusion

We propose a non-invasive DNN fingerprinting method with two operating modes, Fast variant and Selected variant, that search for DDRs separating source-related positive models from independently trained benign negative models. The expanded experiments provide bounded ownership-verification evidence on CIFAR-10 and GTSRB, additional support on CIFAR-100, and cautious Tiny-ImageNet stress evidence where false responses remain low but positive matching is not yet solved. They also cover non-ResNet source backbones, model modification, extraction, hard negatives, simulated service constraints, distribution-shift, pool-imbalance and shadow-pool diagnostics, threshold sensitivity, loss ablations, query-seed repeats, and protocol-aware MetaFinger/ADV-TRA comparisons. Fast variant is a lower-cost screening mode, whereas Selected variant is



(a) ARUC (b) Construction time

Fig. 6 Threshold sensitivity of Selected variant construction. Higher thresholds improve ARUC in this setting but require more processed candidates and longer offline construction time

a higher-confidence offline filtering mode; both should be interpreted within the claim boundaries above.

Author Contributions Tianxiang Luo: Conceptualization, Methodology, Writing – original draft. Zi Yang: Investigation, Validation, Data curation. Xiaorui Dai: Software, Formal analysis, Visualization. Wei Wang: Resources, Project administration, Writing – review & editing. Bo Wang: Supervision, Funding acquisition, Writing – review & editing.

Funding This work was supported by the National Natural Science Foundation of China (Grant No. 62372452) and the Youth Innovation Promotion Association of the Chinese Academy of Sciences.

Data Availability The public benchmark datasets used in this study are available from their original providers. The source code, experiment scripts, configuration files, and model-pool specifications supporting the reported experiments are publicly available at https://github.com/DLUTAIS/efficient-robust-dnn-fingerprint.git.

Code Availability The implementation is publicly available at https://github.com/DLUTAIS/efficient-robust-dnn-fingerprint.git.

Declarations

Competing Interests The authors have no relevant financial or non-financial interests to disclose.

Ethical Approval and Consent to Participate Not applicable. This study uses public benchmark datasets and does not involve human participants or animals.

Consent for Publication Not applicable.

References

1. Raja Sekaran S, Pang YH, Ooi SY, Lim ZY (2025) HSTCN-NuSVC: A homogeneous stacked deep ensemble learner for classifying human actions using smartphones. Emerging Science Journal 9(1):468–484. https://doi.org/10.28991/ESJ-2025-09-01-026
2. Wu TX, Pang YH, Ooi SY, Lim ZY, Hiew FS (2025) A review on federated learning on sensor-based human activity recognition. HighTech and Innovation Journal 6(3):1079–1103. https://doi.org/10.28991/HIJ-2025-06-03-020
3. Shakir M (2025) Enhancing user differentiation in the electronic personal synthesis behavior (EPSBV01) algorithm by adopting the time series analysis. Emerging Science Journal 9(1):243–260. https://doi.org/10.28991/ESJ-2025-09-01-014
4. Lukas N et al (2022) Sok: How robust is image classification deep neural network watermarking? IEEE Symposium on Security and Privacy (SP). IEEE, pp 787–804
5. Shafieinejad M et al (2021) On the robustness of backdoor-based watermarking in deep neural networks. ACM Workshop on Information Hiding and Multimedia Security. pp 177–188
6. Liu J et al (2024) False claims against model ownership resolution. USENIX Security Symposium. USENIX Association, Philadelphia, PA, pp 6885–6902
7. Cao X, Jia J, Gong NZ (2021) IPGuard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary. ACM ASIACCS, pp 14–25
8. Szegedy C (2013) Intriguing properties of neural networks. arXiv:1312.6199
9. Lukas N, Zhang Y, Kerschbaum F (2020) Deep neural network fingerprinting by conferrable adversarial examples. International Conference on Learning Representations
10. Yang K, Wang R, Wang L (2022) Metafinger: Fingerprinting the deep neural networks with meta-training. In: IJCAI, pp. 776–782
11. Uchida Y, Nagai Y, Sakazawa S, Satoh S (2017) Embedding watermarks into deep neural networks. ACM International Conference on Multimedia Retrieval. pp 269–277
12. Adi Y et al. (2018) Turning your weakness into a strength: Watermarking deep neural networks by backdooring. USENIX Security Symposium. pp 1615–1631
13. Zhang J et al (2018) Protecting intellectual property of deep neural networks with watermarking. ACM ASIACCS, pp 159–172
14. Li Y et al (2022) Defending against model stealing via verifying embedded external features, vol 36. AAAI, pp 1464–1472
15. Guo J et al (2024) Domain watermark: Effective and harmless dataset copyright protection is closed at hand. Adv Neural Inf Process Syst 36:
16. Li Y et al (2021) Modeldiff: Testing-based dnn similarity comparison for model reuse detection. ACM ISSTA, pp 139–151
17. Papernot N, McDaniel P, Goodfellow I (2016) Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv:1605.07277
18. Pan X et al (2022) Metav: A meta-verifier approach to task-agnostic model fingerprinting. ACM SIGKDD. pp 1327–1336
19. Yang K, Wang R, Wang L, Tan H (2023) NaturalFinger: Generating natural fingerprint with generative adversarial networks. arXiv:2305.17868
20. Xu T et al (2024) United we stand, divided we fall: Fingerprinting deep neural networks via adversarial trajectories. The Thirty-eighth Annual Conference on Neural Information Processing Systems
21. Liu W, Zhong S (2024) MarginFinger: Controlling generated fingerprint distance to classification boundary using conditional GANs. ACM International Conference on Multimedia Retrieval. https://doi.org/10.1145/3652583.3658058
22. Yan A et al (2025) Enhancing model intellectual property protection with robustness fingerprint technology. IEEE Trans Inf Forensics Secur 20:9235–9249. https://doi.org/10.1109/TIFS.2025.3602244
23. Shao S et al (2026) FIT-Print: False-claim resistant model ownership verification via targeted fingerprint. IEEE Trans Inf Forensics Secur. https://doi.org/10.1109/TIFS.2026.3702542
24. Krizhevsky A, Nair V, Hinton G (2010) Cifar-10 (canadian institute for advanced research) 5(4):1. http://www.cs.toronto.edu/~kriz/cifar.html
25. He K et al (2016) Identity mappings in deep residual networks. In: ECCV, pp. 630–645. Springer
26. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556
27. Howard AG (2017) MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861
28. Han S, Pool J, Tran J, Dally W (2015) Learning both weights and connections for efficient neural network. Adv Neural Inf Process Syst 28:
29. Orekondy T, Schiele B, Fritz M (2019) Knockoff nets: Stealing functionality of black-box models. CVPR, pp 4954–4963
30. Hinton G (2015) Distilling the knowledge in a neural network. arXiv:1503.02531
31. Fang G et al (2019) Data-free adversarial distillation. arXiv:1912.11006
32. Huang G et al (2017) Densely connected convolutional networks. CVPR, pp 4700–4708
33. Sandler M et al (2018) MobileNetV2: Inverted residuals and linear bottlenecks. CVPR, pp 4510–4520

34. Chen L-C, Zhu Y, Papandreou G, Schroff F, Adam H (2018) Encoder-decoder with atrous separable convolution for semantic image segmentation. European Conference on Computer Vision. pp 833–851. https://doi.org/10.1007/978-3-030-01234-2_49
35. Song Z et al (2020) Clinically applicable histopathological diagnosis system for gastric cancer detection using deep learning. Nat Commun 11(1):4294. https://doi.org/10.1038/s41467-020-18147-8
36. Tan M, Le QV (2019) EfficientNet: Rethinking model scaling for convolutional neural networks. In: Proceedings of the 36th International Conference on Machine Learning, pp. 6105–6114. https://doi.org/10.48550/arXiv.1905.11946
37. Kabir H et al (2024) Automated estimation of cementitious sorptivity via computer vision. Nat Commun 15(1):9935. https://doi.org/10.1038/s41467-024-53993-w