FIT-Print: Toward False-Claim-Resistant Model Ownership Verification via Targeted Fingerprint

Shuo Shao, Haozhe Zhu, Yiming Li, *Member, IEEE*, Hongwei Yao, Tianwei Zhang, *Member, IEEE*, and Zhan Qin

***Abstract*—Model fingerprinting has emerged as a crucial mechanism for safeguarding the intellectual property of open-source models, offering a non-intrusive approach that requires no modifications to the protected model. However, our analysis reveals that existing fingerprinting techniques are fundamentally vulnerable to false claim attacks, wherein adversaries can fraudulently assert ownership over independent third-party models. We demonstrate that this vulnerability stems from the untargeted nature of current methods, which evaluate model similarity based on arbitrary sample outputs rather than alignment with a specific, predefined reference. To mitigate this vulnerability, we introduce FIT-Print, a targeted fingerprinting paradigm that actively counters false claim attacks. Specifically, FIT-Print leverages optimization to transform the fingerprint into a verifiable, targeted signature. Building upon this foundation, we propose two black-box fingerprinting methods, the bit-wise FIT-ModelDiff and the list-wise FIT-LIME, which utilize output distances and feature attributions as robust model signatures, respectively. Extensive evaluations across benchmark models and datasets show that our framework perfectly neutralizes false claim attacks (100% defense success rate) and eliminates false alarms on independent models (0.0%), all while maintaining a 100% ownership verification rate against diverse model reuse techniques.**

***Index Terms*—Model fingerprinting, ownership verification, AI copyright protection, trustworthy ML.**

I. Introduction

DEEP learning models, particularly deep neural networks (DNNs), have achieved remarkable success across a diverse array of applications [1], [2], [3]. Developing a high-performing model is inherently expensive, demanding substantial computational resources, extensive data curation, and domain expertise throughout the model development lifecycle. Despite these immense costs, many developers release their models to the open-source community (*e.g.*, Hugging Face) to foster academic research and educational innovation. However, the development of efficient downstream model reuse techniques, such as fine-tuning [4] and transfer learning [5], introduces severe threats to the intellectual property rights (IPR) of these assets. By exploiting these methods, malicious developers can rapidly repurpose open-source models for lucrative commercial applications without authorization. Consequently, establishing reliable mechanisms to safeguard model IPR has emerged as a critical problem in trustworthy machine learning [6], [7], [8].

Currently, ownership verification is a widely adopted post-hoc approach to safeguard the IPR of model developers. This method aims to determine whether a suspicious third-party model is reused from the protected model [9], [10], [11]. Existing techniques for ownership verification generally fall into two main categories: model watermarking and model fingerprinting. Model watermarking [11], [12], [13] involves embedding an owner-specific signature (*i.e.*, watermark) into the model. The model developer can then extract this watermark from the model to verify their ownership. In contrast, model fingerprinting [14], [15], [16], [17] aims to identify the intrinsic features (*i.e.*, fingerprints) of the model rather than modifying it. A fingerprint can be represented by the outputs of specific testing samples through a particular mapping function. By comparing these fingerprints, one can check whether the suspicious model is a reused version of the source model. Arguably, model fingerprinting is more practical than model watermarking because it does not require any changes to the model's parameters, structure, or training process, and therefore has no negative impact on the model itself.

In this paper, we reveal that existing model fingerprinting methods, whether they are bit-wise [15] or list-wise [16] (*i.e.*, extracting the fingerprint bit by bit or as an entire list), are vulnerable to false claim attacks. In general, false claim attacks [18] allow adversaries to falsely claim ownership of independent third-party models by creating a counterfeit ownership certificate (*i.e.*, a watermark or fingerprint). Specifically, these attacks can be viewed as finding transferable ownership

Received 7 August 2025; revised 7 April 2026 and 25 May 2026; accepted 3 June 2026. Date of publication 10 June 2026; date of current version 18 June 2026. This work was supported in part by the Kunpeng-Ascend Science and Education Innovation Excellence/Incubation Center, in part by the National Natural Science Foundation of China under Grant 62441238 and Grant U2441240, and in part by Zhejiang Provincial Natural Science Foundation of China under Grant LD24F020010. The associate editor coordinating the review of this article and approving it for publication was Dr. Z. Berkay Celik. *(Corresponding author: Yiming Li.)*

Shuo Shao, Haozhe Zhu, and Zhan Qin are with the State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou 310027, China, and also with Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, Hangzhou 310051, China (e-mail: shaoshuo_ss@zju.edu.cn; howjul@zju.edu.cn; qinzhan@zju.edu.cn).

Yiming Li and Tianwei Zhang are with the College of Computing and Data Science, Nanyang Technological University, Singapore 639798 (e-mail: liyiming.tech@gmail.com; tianwei.zhang@ntu.edu.sg).

Hongwei Yao is with the Department of Computer Science, City University of Hong Kong, Hong Kong, China (e-mail: yao.hongwei@cityu.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TIFS.2026.3702542, provided by the authors.

Digital Object Identifier 10.1109/TIFS.2026.3702542

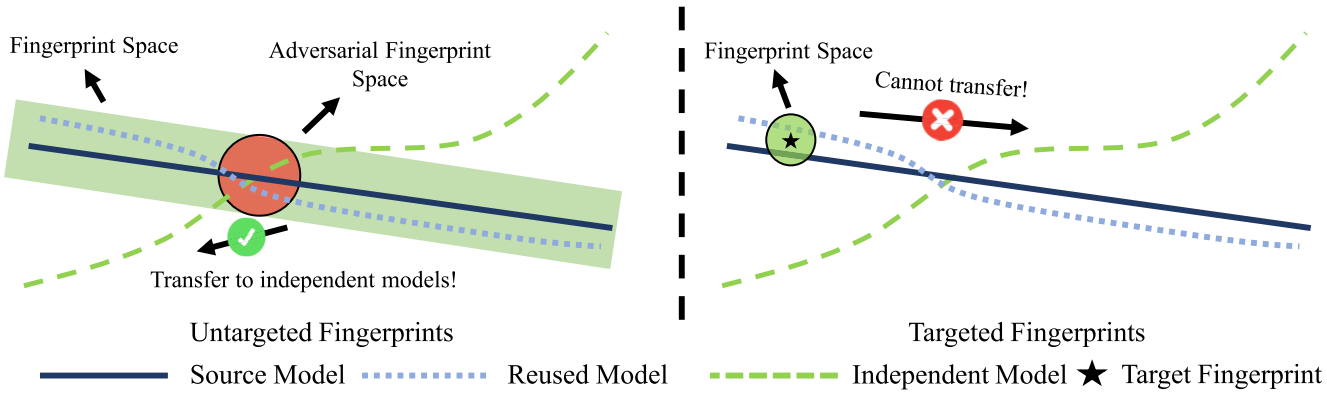


Fig. 1. The comparison of untargeted and targeted fingerprinting paradigms. Untargeted methods generally compare the output of given samples. Thus, using some transferable samples can lead to false claims. Targeted fingerprinting calculates the similarity to a specific signature, which restricts the fingerprint space around the target and, therefore, mitigates false claim attacks.

certificates across different models, because registering a certificate with a timestamp prevents any false claims made at a later date. We show that an adversary can conduct false claim attacks by constructing transferable, "easy" samples that are correctly classified with high confidence (detailed in Section II-C). Since existing fingerprinting methods typically compare the outputs of testing samples, these carefully crafted, easy samples can produce similar high-confidence outputs across various models. Consequently, this leads to independent models being misjudged as reused ones.

We argue that this vulnerability primarily stems from the *untargeted* nature of existing model fingerprinting methods. Specifically, they generally compare the outputs of arbitrary samples across different models, rather than measuring their similarity to specific References. This untargeted nature enlarges the space of viable fingerprints, making it easy for adversaries to find transferable samples that produce similar outputs on independent models, as illustrated in Fig. 1.

Motivated by these findings, we introduce a new fingerprinting paradigm, dubbed False-claIm-resistant Targeted model fingerPrinting (FIT-Print), where the fingerprint comparison is *targeted* rather than untargeted. The key insight of FIT-Print is to shift model ownership verification from passive feature extraction to active signature alignment. Specifically, we optimize the perturbations on the testing samples so that the output of the mapping function closely aligns with a specific signature (*i.e.*, the target fingerprint). This restricts the potential fingerprint space, significantly reducing the probability of a successful false claim attack. Based on FIT-Print, we develop two targeted model fingerprinting methods: FIT-ModelDiff and FIT-LIME, representing bit-wise and list-wise approaches, respectively. FIT-ModelDiff exploits the distances between model outputs, while FIT-LIME leverages the feature attributions of testing samples as the fingerprint.

Our main contributions are four-fold:

- We revisit existing model fingerprinting methods and reveal that they are generally vulnerable to false claim attacks due to their untargeted nature.
- We introduce a new fingerprinting paradigm (*i.e.*, FIT-Print), which conducts ownership verification in a targeted manner using a specific reference.
- Based on FIT-Print, we design two black-box targeted fingerprinting methods: FIT-ModelDiff and FIT-LIME.
- We conduct extensive experiments on benchmark datasets and models to demonstrate their effectiveness, conferrability, and resistance to both false claims and adaptive attacks.

II. Revisiting Existing Model Fingerprinting

In this section, we first formally and comprehensively define the threat model of model fingerprinting. We then categorize existing methods into two types and describe their formulations. Subsequently, building on these definitions, we design a simple yet effective false claim attack, revealing the underlying vulnerability of current fingerprinting methods to such attacks.

A. Threat Model of Model Fingerprinting

In this paper, we consider three parties in our threat model, including *model developer*, *model reuser*, and *verifier*. Arguably, including a verifier is necessary and may improve the trustworthiness of model fingerprinting, although there are currently still no mature legal provisions about this.

1) Process of Model Fingerprinting: Model fingerprinting can be divided into three steps, including fingerprint generation, fingerprint registration, and ownership verification.

1) **Fingerprint Generation**: The model developer trains their source model M_o and generates the fingerprint of M_o. Broadly, a model fingerprint refers to any unique, intrinsic characteristic that allows a model to be distinguished from other independently trained models.
2) **Fingerprint Registration**: After generating the fingerprint, the model developer registers the fingerprint and the model with a timestamp to a trustworthy third-party verifier.
3) **Ownership Verification**: For a suspicious model M_s that could be a reused version of M_o, the verifier will first check the timestamps of these two models. If the registration timestamp of M_s is later than M_o, the verifier will further check whether the fingerprint of M_o is similar to the fingerprint M_s. If so, the suspicious model can be regarded as a reused version of M_o.

2) Assumptions of the Model Developer and Verifier: The model developer is the owner of the source model and can register their model and fingerprint to the trustworthy verifier with a timestamp. The verifier is responsible for fingerprint registration and verification. In case the model is reused by a model reuser, the model developer can ask the verifier for ownership verification. In particular, if two parties can simultaneously provide fingerprints and verify the ownership of a model, the fingerprint with a later timestamp will be deemed invalid. The model developer and the verifier are assumed to have **(1)** white-box access to their source model and **(2)** black-box access to the suspicious model.

3) Assumptions of the Model Reuser: Model reusers aim to avoid having their authorized reuse detected by the verifier. To achieve this, they can first modify the victim model via various techniques, such as fine-tuning, pruning, transfer learning, and model extraction, before deployment.

B. The Formulation of Existing Fingerprinting

In this section, we outline the formulations of existing fingerprinting methods to aid in the analysis and design process in the subsequent sections of this paper and follow-up research. In this paper, we focus on black-box methods since they are more practical in the real world. In general, existing black-box model fingerprinting methods can be categorized into two types: adversarial example-based (AE-based) fingerprinting methods and testing-based fingerprinting methods. We also include a broader discussion about other fingerprinting methods in Section V-A.

1) AE-Based Fingerprinting Methods: AE-based fingerprinting methods [14], [19], [20] assume that the independent model has a unique decision boundary. Based on this assumption, they exploit adversarial examples (AE) [21] to characterize the properties of the decision boundary of a model. AEs are visually similar to the benign samples but misclassified by the model [21], [22]. Due to such a property, AEs tend to be near the decision boundary. Subsequently, AE-based fingerprinting methods validate whether the AEs are misclassified by the source model and the suspicious model. If so, the suspicious model can be treated as a reused version of the source model. The definition of the ownership verification in AE-based methods can be formulated as Definition 1.

Definition 1 (Ownership Verification of AE-based Fingerprinting Methods): Let M_o be the source model and M_s be the suspicious model, and $g(\boldsymbol{x})$ is the function that always outputs the ground-truth label of any input data $\boldsymbol{x}$. If for any $\boldsymbol{x} \in \mathcal{X}_T$ ($\mathcal{X}_T$ denotes the set of testing samples), we have

$$M_o(\boldsymbol{x}) = M_s(\boldsymbol{x}) \neq g(\boldsymbol{x}). \tag{1}$$

The suspicious model M_s can be asserted as a reused version of the source model M_o.

2) Testing-Based Fingerprinting Methods: Testing-based fingerprinting methods [15], [16], [23] aim to compare the suspicious model with the source model on a specific mapping function $f(\cdot)$. If the outputs are similar, the suspicious model can be regarded as being reused from the source model. As such, the core of testing-based fingerprinting methods is how to design the mapping function $f(\cdot)$. The definition of ownership verification in testing-based fingerprinting methods can be formulated as Definition 2.

Definition 2 (Ownership Verification of Testing-based Fingerprinting Methods): Let M_o be the source model and M_s be the suspicious model. If for a specific mapping function $f(\cdot)$ and any $\boldsymbol{x} \in \mathcal{X}_T$ ($\mathcal{X}_T$ is the set of testing samples), we have

$$\frac{1}{|\mathcal{X}_T|} \sum_{\boldsymbol{x} \in \mathcal{X}_T} \texttt{dist}(f[M_o(\boldsymbol{x})], f[M_s(\boldsymbol{x})]) \leq \tau, \tag{2}$$

where τ is a small positive threshold and $\texttt{dist}(\cdot, \cdot)$ is a distance function. Then, the suspicious model M_s can be asserted as reused from the source model M_o.

C. False Claim Attack Against Model Fingerprinting

Existing model fingerprinting methods primarily assume that the model reuser is the adversary while paying little attention to the false claim attack [18][1] where **the model developer is the adversary**. The formal definition of the false claim attack is as follows.

Definition 3: A false claim attack refers to a malicious attempt by a malicious model developer to falsely assert the ownership of an independent model M_I by registering some fraudulent testing samples $\bar{\boldsymbol{x}}$ that can pass the ownership verification of Definition 1 or Definition 2.

As depicted in Fig. 2, there are also three different parties involved in the threat model of false claim attacks, including the malicious developer, the verifier, and an independent developer.

In false claim attacks, the malicious developer is the adversary who aims to craft and register a *transferable* fingerprint to falsely claim the ownership of the independent developer's model M_I. The malicious developer is assumed to have adequate computational resources and datasets to train a high-performance model and carefully craft transferable model fingerprints. The primary goal of the malicious developer is that the registered model fingerprints can be verified in as many other models as possible. By generating the transferable fingerprint, the malicious developer can (falsely) claim the ownership of any third-party models (that are registered later than that of the malicious developer).

1) Process of False Claim Attacks: The process of false claim attacks can be divided into 3 steps, including fingerprint generation, fingerprint registration, and false ownership verification.

1) **Fingerprint Generation**: In this step, the malicious model developer trains their source model M_o and attempts to generate a *transferable* fingerprint of M_o.
2) **Fingerprint Registration**: After generating the fingerprint, the malicious model developer registers the *transferable* fingerprint and the model with a timestamp to a trustworthy third-party verifier.
3) **False Ownership Verification**: The malicious model developer could try to use the transferable fingerprint to falsely claim the ownership of another independently

[1]The concept and definition of false claim attacks were initially introduced in [18], [24] and primarily targeted at attacking model watermarking methods.

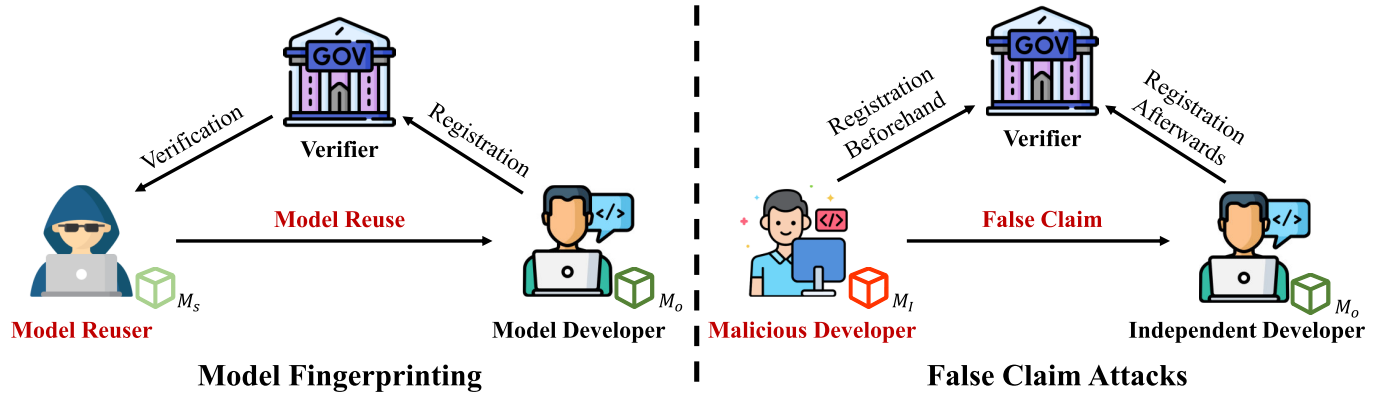


Fig. 2. The threat models and detailed processes of model fingerprinting and false claim attacks. In model fingerprinting, the model developer generates and registers the model and the fingerprint with a third-party verifier. Once the model is reused by a model reuser, the verifier can determine the model ownership by comparing the fingerprints. Instead, in false claim attacks, the malicious developer attempts to register a transferable fingerprint to falsely claim other independent developers' models.

TABLE I

FALSE CLAIM ATTACK AGAINST THREE TESTING-BASED METHODS. IT IS OBSERVED THAT THE DISTANCES BETWEEN INDEPENDENT MODELS AND THE SOURCE MODEL (IND. MODEL DIST.) AFTER THE ATTACK ARE APPROXIMATELY EQUAL TO OR LESS THAN THE AVERAGE DISTANCE BETWEEN THE REUSED MODELS AND THE SOURCE MODEL (REUSED MODEL DIST.), DEMONSTRATING THE VULNERABILITY OF EXISTING METHODS AGAINST FALSE CLAIM ATTACKS

	Method→	ModelDiff		Zest		SAC	
	Dataset→	SDogs120	Flowers102	SDogs120	Flowers102	SDogs120	Flowers102
Ind. Model Dist.	Before Attack	0.131	0.114	0.177	0.161	0.080	0.094
	After Attack	0.093	0.083	0.114	0.098	0.078	0.092
Reused Model Dist.	Average	0.108	0.092	0.095	0.072	0.079	0.081

trained model M_I. Since the fingerprint is registered beforehand, the ownership verification won't be rejected due to the timestamp. Subsequently, the benign developer may be accused of infringement.

Since registering the fingerprint with a timestamp can prevent any false claim after registration, its success hinges on generating a transferable fingerprint. For AE-based methods, [18] has successfully implemented the false claim attacks by constructing transferable AEs that force diverse models to consistently misclassify them into specific wrong classes. As such, we hereby mainly focus on designing false claim attacks against cutting-edge testing-based methods. While our attack shares the fundamental insight of exploiting transferability, it operates in an opposite manner for testing-based fingerprinting. Instead of inducing misclassification, our primary insight is to craft transferable "inverse-AEs" $\bar{\boldsymbol{x}}$ as the malicious fingerprinting samples. $\bar{\boldsymbol{x}}$ are abnormally easy samples that force independent models to consistently classify them correctly with exceptionally high confidence, leading to:

$$M_o(\bar{\boldsymbol{x}}) \approx M_I(\bar{\boldsymbol{x}}) \Rightarrow \texttt{dist}(f[M_o(\bar{\boldsymbol{x}})], f[M_I(\bar{\boldsymbol{x}})]) \approx 0 \leq \tau. \quad (3)$$

To execute this strategy, motivated by the fast gradient sign method (FGSM) [25] for AE generation, we propose to leverage Eq. (4) to generate malicious fingerprinting samples, as follows:

$$\bar{\boldsymbol{x}} = \boldsymbol{x} - \gamma \cdot \texttt{sign}(\nabla J(M_o, \boldsymbol{x}, \boldsymbol{y})), \quad (4)$$

where $\texttt{sign}(\cdot)$ denotes the sign function, $J(\cdot)$ represents the loss function associated with the original task of M_o, and γ signifies the magnitude of the perturbation. More powerful transferable adversarial attacks can be exploited here, but we aim to show that using simple FGSM can also falsely claim to have ownership of some independent models.

2) Results: We exploit 3 representative testing-based methods, *i.e.*, ModelDiff [15], Zest [16], and SAC [23], to validate the effectiveness of our false claim attacks. As shown in Table I, SAC is poor at identifying models of the same tasks, even without attacks. Moreover, after attacks, the distances between the source model M_o and the independent model M_I of all three methods are approximately equal to or less than the average distances between reused models and the source model. It indicates that M_I will be asserted as reused from M_o, which is a false alarm. The results demonstrate that existing model fingerprinting methods are vulnerable to false claim attacks.

III. METHODOLOGY

A. Design Objectives

The objectives of model fingerprinting methods can be summarized in three-fold, including effectiveness, conferrability, and resistance to false claim attacks.

- **Effectiveness:** Effectiveness indicates that the model developer can successfully verify the ownership of the source model using the fingerprinting method.

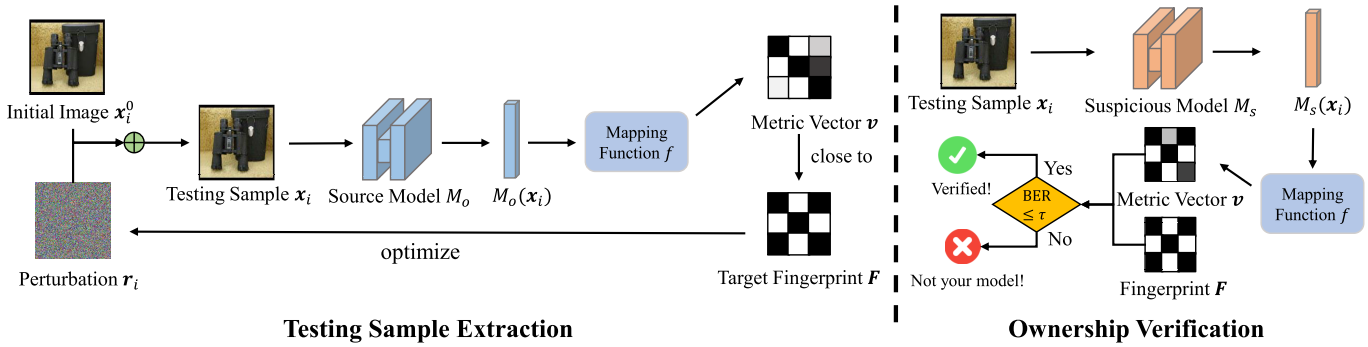


Fig. 3. The pipeline of FIT-Print. In testing sample extraction, FIT-Print optimizes the perturbations to turn the fingerprint vector close to the target fingerprint. In the ownership verification stage, FIT-Print extracts the fingerprint from the suspicious model and compares it with the original fingerprint.

- **Conferrability:** Conferrability requires the model fingerprint to be conferrable to models reused from the source model. In other words, the fingerprints of the source and reused models should be highly similar.
- **Resistance to False Claim Attacks:** This requires that independently trained models have distinctly different fingerprints. Additionally, it prevents malicious developers from constructing a transferable fingerprint that can be extracted from independent models.

B. The Insight of Our FIT-Print

As discussed in Section II-C, existing model fingerprinting methods are vulnerable to false claim attacks. We argue that the vulnerability stems primarily from the "untargeted" characteristic of the fingerprinting methods. The untargeted characteristic leads to a large fingerprint space that can accommodate transferable adversarial fingerprints. In this paper, we propose FIT-Print, a targeted model fingerprinting framework to mitigate false claim attacks. Our main insight is that although it is tough to find the space that can only transfer among reused models, we can turn the fingerprint into a target one to restrict the fingerprinting space and reduce the adversarial transferability of fingerprints.

Given a mapping function $f(\cdot)$ and a target fingerprint $\boldsymbol{F}$, our goal is to make the fingerprint vector $\boldsymbol{v} = f(M_s(\boldsymbol{x}))$ close to $\boldsymbol{F}$. Accordingly, the definition of our FIT-Print can be defined as follows.

Definition 4: Let M_s be the suspicious model. If for a specific mapping function $f(\cdot)$ and testing sample $\boldsymbol{x} \in \mathcal{X}_T$ ($\mathcal{X}_T$ is the set of testing samples), we have

$$\frac{1}{|\mathcal{X}_T|} \sum_{\boldsymbol{x} \in \mathcal{X}_T} \texttt{dist}(f[M_s(\boldsymbol{x})], \boldsymbol{F}) \leq \tau, \tag{5}$$

where τ is a small threshold and $\texttt{dist}(\cdot,\cdot)$ is a distance function, the suspicious model M_s can be asserted as reused from the owner of the fingerprint $\boldsymbol{F}$.

In FIT-Print, we assume that the target fingerprint $\boldsymbol{F} \in \{-1, 1\}^k$ is a binary vector consisting of -1 or 1, and we can get the output logits of $M_s(\boldsymbol{x})$. The discussion on the label-only scenario, where we can only get the Top-1 label, can be found in Section IV-F. We assume that the target fingerprint needs to be registered with a third-party institution. Specifically, in the registration stage, a model developer utilizing FIT-Print registers three items to the trustworthy verifier with a timestamp: the source model (M_o), the optimized testing samples ($\mathcal{X}_T$), and the pre-defined target fingerprint ($\boldsymbol{F}$). Unlike conventional untargeted methods that passively register arbitrary output vectors as fingerprints, FIT-Print actively registers a semantic, specific target fingerprint (*e.g.*, a logo) that the optimized samples are mathematically constrained to produce.

Generally, as shown in Fig. 3, FIT-Print can be divided into two stages: testing sample extraction and ownership verification.

C. Testing Sample Extraction

In the testing sample extraction stage, we aim to find the optimal testing sample set $\mathcal{X}_T$ to make any reused models satisfy Eq. (5) in Definition 4. Therefore, in FIT-Print, we first initialize the testing samples $\mathcal{X}_T$ and the corresponding perturbations $\mathcal{R}$. We denote the i-th element in $\mathcal{X}_T$ and $\mathcal{R}$ as $\boldsymbol{x}_i$ and $\boldsymbol{r}_i$ respectively. The element $\boldsymbol{x}_i$ is set to an initial value $\boldsymbol{x}_i^0$, and we can initialize the testing samples to any images. The testing samples in $\mathcal{X}_T$ can be constructed by adding the perturbations to the initial values, *i.e.*, $\boldsymbol{x}_i = \boldsymbol{x}_i^0 + \boldsymbol{r}_i$. After that, we need to optimize the perturbations $\mathcal{R}$ to make the fingerprint vector $\boldsymbol{v}$ close to the target fingerprint $\boldsymbol{F}$. We can define the testing sample extraction as an optimization problem, which can be formalized as follows.

$$\min_{\mathcal{R}=\{\boldsymbol{r}_1,\ldots,\boldsymbol{r}_{|\mathcal{R}|}\}} \frac{1}{|\mathcal{X}_T|} \sum_{i=1}^{|\mathcal{X}_T|} [\mathcal{L}(f(M_o(\boldsymbol{x}_i^0 + \boldsymbol{r}_i), \boldsymbol{F}) + \lambda \cdot \|\boldsymbol{r}_i\|_2], \tag{6}$$

where $\|\cdot\|_2$ calculates the ℓ_2-norm. The first term in Eq. (6) quantifies the dissimilarity between the output fingerprint vector $\boldsymbol{v}$ and the target fingerprint $\boldsymbol{F}$. The second term regularizes the extent of the perturbations $\mathcal{R}$. We utilize the hinge-like loss [26] as $\mathcal{L}(\cdot)$, as follows.

$$\mathcal{L}(\boldsymbol{v}, \boldsymbol{F}) = \sum_{i=1}^{k} \max(0, \varepsilon - \boldsymbol{v}_i \cdot \boldsymbol{F}_i). \tag{7}$$

In Eq. (7), $\boldsymbol{v}$ is the fingerprint vector, where $\boldsymbol{v}_i = f[M_s(\boldsymbol{x}_i)]$, and ε is the control parameter. $\boldsymbol{F}_i$ is the i-th element in $\boldsymbol{F}$. Optimizing Eq. (7) can make the signs of the corresponding elements in $\boldsymbol{v}$ and $\boldsymbol{F}$ the same. Moreover, inspired by the

insight of [19], we craft some augmented models by applying model reuse techniques (*e.g.*, fine-tuning, pruning, or transfer learning) and exploit them to extract the fingerprint to improve the conferrability of FIT-Print. The set of augmented models is denoted as $\mathcal{M}$. The loss function with augmented models can be defined as Eq. (8).

$$\min_{\mathcal{R}=\{\boldsymbol{r}_1,\ldots,\boldsymbol{r}_{|\mathcal{R}|}\}} \frac{1}{|\mathcal{M}|\cdot|\mathcal{X}_T|}\sum_{M\in\mathcal{M}}\sum_{i=1}^{|\mathcal{X}_T|}[\mathcal{L}(\boldsymbol{v},\boldsymbol{F})+\lambda\cdot\|\boldsymbol{r}_i\|_2]. \quad (8)$$

By optimizing Eq. (8), we can get the optimal testing samples that are conferrable to reused models, and the model developer can afterward utilize them to verify the ownership.

D. Ownership Verification

In the ownership verification stage, given a suspicious model M_s, FIT-Print examines whether the suspicious model M_s is reused from the source model M_o by justifying whether M_s satisfies Eq. (5). Specifically, we first calculate the fingerprint vector $\tilde{\boldsymbol{v}}$ of the suspicious model M_s using the extracted testing samples in $\mathcal{X}_T$. Each element $\tilde{\boldsymbol{v}}_i = f(M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i))$. Since optimizing Eq. (8) makes the signs of the fingerprint vector $\tilde{\boldsymbol{v}}$ represent the fingerprint $\tilde{\boldsymbol{F}}$ of the model, we need to transform $\tilde{\boldsymbol{v}}$ into a binary vector by applying the sign function $\texttt{sign}(\cdot)$ to get $\tilde{\boldsymbol{F}}$, as Eq. (9).

$$\tilde{\boldsymbol{F}}_i = \texttt{sign}(\tilde{\boldsymbol{v}}_i) = \begin{cases} 1, & \tilde{\boldsymbol{v}}_i \geq 0 \\ -1, & \tilde{\boldsymbol{v}}_i < 0 \end{cases}. \quad (9)$$

Subsequently, we leverage the bit error rate (BER) as the distance function $\texttt{dist}(\cdot)$ in Eq. (5) and the BER is the distance between the extracted and the target fingerprints, as follows.

$$\texttt{BER} = \frac{1}{k}\sum_{i=1}^{k}\mathbb{I}\{\tilde{\boldsymbol{F}}_i \neq \boldsymbol{F}_i\}, \quad (10)$$

where k is the length of the fingerprint and $\mathbb{I}\{\cdot\}$ is the indicator function defined as Eq. (11).

$$\mathbb{I}\{A\} = \begin{cases} 0, A = \text{False} \\ 1, A = \text{True} \end{cases}. \quad (11)$$

As Definition 4, if the BER is lower than the threshold τ, the suspicious model M_s can be asserted as a reused model. For choosing the threshold τ to reduce false alarms and resist false claim attacks, we have Proposition 1.

Proposition 1: Given the security parameter κ and the fingerprint $\boldsymbol{F}\in\{-1,1\}^k$, if τ satisfies that

$$\sum_{d=0}^{\lfloor\tau k\rfloor}\binom{k}{d}\left(\frac{1}{2}\right)^k \leq \kappa, \quad (12)$$

where $\binom{k}{d} = k!/[d!(k-d)!]$, the probability of a false alarm (*i.e.*, the BER is less than τ with random testing samples) is bounded by κ.

The proof of Proposition 1 can be found in the appendix (see Supplementary Material). We also conduct an empirical evaluation on the resistance of FIT-Print against adaptive false claim attacks in Section IV-D.

E. Designing the Mapping Function in FIT-Print

In Section III-C-III-D, we introduced the main pipeline and paradigm of our FIT-Print. The key to implementing FIT-Print is to design the mapping function $f(\cdot)$. We hereby illustrate how to leverage the paradigm of FIT-Print and design two targeted model fingerprinting methods, including FIT-ModelDiff and FIT-LIME, as the representatives of bit-wise and list-wise methods, respectively.

1) FIT-ModelDiff: FIT-ModelDiff is a bit-wise fingerprinting method that extracts the fingerprint bit by bit. The main insight of FIT-ModelDiff is to compare the distance between the output logits of perturbed samples $\boldsymbol{x}_i^0+\boldsymbol{r}_i$ and benign samples $\boldsymbol{x}_i^0$. The vector of the distances is called the decision distance vector (DDV). Given the suspicious model M_s, DDV can be calculated as follows:

$$\begin{aligned}\texttt{DDV}_i &= \texttt{cos\_sim}(M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i), M_s(\boldsymbol{x}_i^0)) \\ &= \frac{M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i)\cdot M_s(\boldsymbol{x}_i^0)}{\|M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i)\|\cdot\|M_s(\boldsymbol{x}_i^0)\|}.\end{aligned} \quad (13)$$

$\texttt{DDV}_i$ represents the i-th element in the DDV and $\texttt{cos\_sim}(\cdot,\cdot)$ is the cosine similarity function. Since the output logits after softmax are always positive, the range of the DDV is $[0,1]$. As proposed in Section III-C, we aim to make the sign of the fingerprint vector $\boldsymbol{v}$ the same as the target fingerprint $\boldsymbol{F}$. Therefore, to achieve this goal, we need to subtract a factor from DDV to make the range of $\boldsymbol{v}$ including both positive and negative values, as Eq. (14).

$$\begin{aligned}\boldsymbol{v}_i &= f(M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i), M_s(\boldsymbol{x}_i^0)) = \texttt{DDV}_i - \texttt{cos}(\alpha) \\ &= \frac{M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i)\cdot M_s(\boldsymbol{x}_i^0)}{\|M_s(\boldsymbol{x}_i^0+\boldsymbol{r}_i)\|\cdot\|M_s(\boldsymbol{x}_i^0)\|} - \texttt{cos}(\alpha),\end{aligned} \quad (14)$$

where $\texttt{cos}(\cdot)$ is the cosine function and α is the bias parameter. The final fingerprint vector $\boldsymbol{v}$ can be used for testing sample extraction or ownership verification.

2) FIT-LIME: FIT-LIME is a list-wise method that extracts the fingerprint as a whole list. FIT-LIME implements the mapping function $f(\cdot)$ via a popular feature attribution algorithm, local interpretable model-agnostic explanation (LIME) [27]. LIME outputs a real-value importance score for each feature in the input sample $\boldsymbol{x}$. We enhance the LIME algorithm and develop FIT-LIME to better cater to the needs of ownership verification. The details of FIT-LIME are elaborated as follows.

The first step of FIT-LIME is to generate c samples that are neighboring to the input image $\boldsymbol{x}$. We also gather the adjacent pixels in the image into a superpixel. We uniformly segment the input space into k superpixels, where k is the length of the targeted fingerprint. Assuming that $k=\mu\times\nu$, the image can be divided into μ rows and ν columns. Each superpixel represents a rectangular region that has $\lceil w/\mu\rceil\times\lceil h/\nu\rceil$ pixels in the image. Then, we randomly generate c masks where each mask is a k-dimension binary vector, constituting $\boldsymbol{A}\in\{0,1\}^{c\times k}$. Each element in each row of the matrix $\boldsymbol{A}$ corresponds to a superpixel in the image $\boldsymbol{x}$. After that, we exploit the binary matrix to mask the image $\boldsymbol{x}$ and generate the masked examples $\mathcal{X}^m$. If the element in the i-th row of the mask $\boldsymbol{A}$ is 1, the corresponding superpixel preserves its original value.

Otherwise, the superpixel is aligned with 0. Each row of the mask can generate a masked sample, and the c masked samples constitute the masked sample set $\mathcal{X}^m$.

The second step is to evaluate the output of the masked samples $\mathcal{X}^m$ on the suspicious model M_s. Different from primitive LIME, we utilize the entropy of the outputs so that it no longer depends on the label of $\boldsymbol{x}$. The intuition is that if the important features are masked, the prediction entropy will significantly increase. Following this insight, we calculate the following equation in this step.

$$\boldsymbol{p}_i = H[M_s(\mathcal{X}_i^m)], \tag{15}$$

In Eq. (15), $H(\cdot)$ calculates the entropy, $\boldsymbol{p}_i$ is the i-th element in $\boldsymbol{p}$, and $\mathcal{X}_i^m$) is the i-th masked sample.

After that, the final step is to fit a linear surrogate model to approximate the relationship between the binary masks $\boldsymbol{A}$ and the corresponding prediction entropies $\boldsymbol{p}$. Specifically, following [28], we optimize an ordinary least squares objective to find the feature importance vector v that minimizes the approximation error:

$$\min_{\boldsymbol{v}} \|\boldsymbol{A}\boldsymbol{v} - \boldsymbol{p}\|_2^2. \tag{16}$$

Eq. (16) can be solved using the well-known Normal Equation. This yields the importance score vector $\boldsymbol{v}$ as formulated in Eq. (17):

$$\boldsymbol{v} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{p}. \tag{17}$$

This importance score vector $\boldsymbol{v}$ is then utilized as the fingerprint vector in both the testing sample extraction and ownership verification stages.

IV. Experiments

In this section, we evaluate the effectiveness, conferrability, and resistance to the false claim attack of our FIT-Print methods. We also include ablation studies on the hyperparameters, different targets, initializations, and different numbers of augmented models in FIT-Print.

A. Experimental Settings

1) Models and Datasets: Following prior works [15], [16], we utilize two widely-used convolutional neural network (CNN) architectures, MobileNetV2 [29] (mbnetv2 for short) and ResNet18 [30], in our experiments. We train MobileNetV2 and ResNet18 using two different datasets, Oxford Flowers 102 (Flowers102) [31] and Stanford Dogs 120 (SDogs120) [32], in total 4 source models. Flowers102 contains RGB images of 102 different flowers, while SDogs120 has images of 120 different breeds of dogs. We primarily focus on image classification models in our experiments. In particular, we provide a case study about implementing FIT-Print to text generation models in Section V-B. We utilize the data from ImageNet [33] as the default initial images of the testing samples $\mathcal{X}_T$. We also utilize the models pre-trained on ImageNet [33] as the models trained by other parties, *i.e.*, negative models.

2) Model Reuse Techniques: We evaluate FIT-Print against the following five categories of model reuse techniques, including copying, fine-tuning, pruning, model extraction, and transfer learning. We further consider different implementations of these model reuse techniques in various settings and scenarios. For each source model, we train and craft three fine-tuning models, three pruning models, two extraction models, and three transfer learning models. These 12 models constitute the set of reused models. When experimenting on one source model, the other 36 models that are reused from other source models are treated as independent models.

3) Baseline Methods: We consider both AE-based and testing-based fingerprinting methods. For the former, we implement two typical methods, IPGuard [14] and MetaV [34]. While for the latter, we take three different methods, ModelDiff [15], Zest [16], and SAC [23] as the baseline methods. We also include a state-of-the-art (SOTA) white-box model fingerprinting method, *i.e.*, ModelGiF [35], for reference.

4) Target Fingerprint: As default, we select a logo of a file and a pen as the targeted fingerprint $\boldsymbol{F}$. We set the default length k of the fingerprint $\boldsymbol{F}$ to be 256 and thus $\boldsymbol{F}$ is resized to 16×16. We set the security parameter $\kappa = 10^{-9}$. According to Eq. (12), the threshold τ is 0.316 in our experiments.

5) Optimization Details: We utilize the stochastic gradient descent (SGD) with momentum as the optimizer. We set the initial learning rate to 1.2×10^{-2}, the momentum to 0.9, and the weight decay to 5×10^{-4}. We apply a cosine annealing schedule [36] to reduce the learning rate gradually to a minimum of 4×10^{-3}. Following [28], we set the control parameter ε in the hinge-like loss in Eq. (7) to 0.01. We optimize the perturbations on the testing samples for 300 epochs. In Eq. (14) of FIT-ModelDiff, we set the bias parameter α to be $\pi/8$.

6) Computational Resources: All our experiments are implemented with at most 8 RTX 3090 GPUs.

B. Evaluation on Effectiveness and Conferrability

Table II illustrates the percentage of successfully identified reused models (ownership verification rate). Both FIT-ModelDiff and FIT-LIME can recognize the reused models under five reuse techniques with 100% ownership verification rates, which outperform existing fingerprinting methods and perform on par with the SOTA white-box method, ModelGiF. Also, FIT-ModelDiff and FIT-LIME achieve 0.0% ownership verification rates on the independent models, indicating that our methods do not lead to false alarms. Fig. 4 illustrates the BERs of the reused models, which are all less than the threshold τ with a maximum of 0.227. The results validate the effectiveness and conferrability of FIT-Print.

C. Ablation Study

1) Effects of the Length of the Fingerprint: In this experiment, we investigate the impact of varying lengths of the fingerprint $\boldsymbol{F}$. In addition to the default length of $256 = 16 \times 16$, we set the length to be 12×12, 20×20, and 24×24. The results illustrated in Fig. 5 indicate that both FIT-ModelDiff and FIT-LIME can recognize the reused

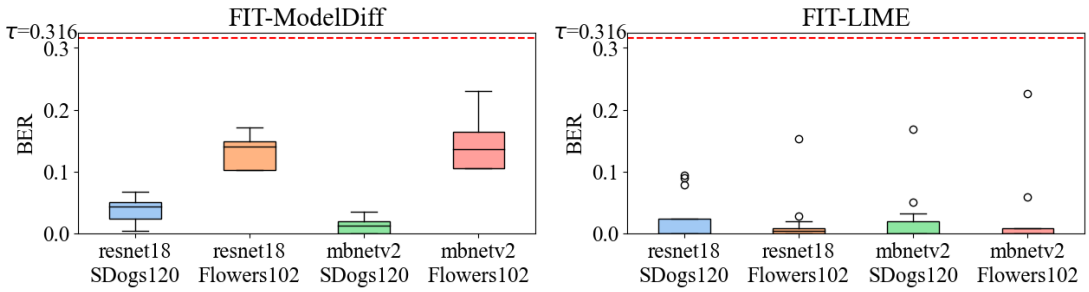


Fig. 4. The BERs of different source models and their reused models with FIT-ModelDiff and FIT-LIME. The BERs are all less than the threshold τ marked with a red dashed line, indicating that FIT-ModelDiff and FIT-LIME can successfully recognize the reused models.

TABLE II

SUCCESSFUL OWNERSHIP VERIFICATION RATES OF DIFFERENT MODEL FINGERPRINTING METHODS. "#MODELS" DENOTES THE NUMBER OF REUSED MODELS, AND THE "N/A" INDICATES THAT THE METHOD CANNOT BE APPLIED TO DETECT THIS TYPE OF MODEL REUSE TECHNIQUE. IN PARTICULAR, WE MARK FAILED CASES (*i.e.*, < 80% OR "N/A") IN RED. MOREOVER, THE BERS OF FIT-PRINT ARE ALL 0.0%

Reuse Task↓	#Models↓	AE-based		Testing-based			White-box	FIT-Print	
		IPGuard	MetaV	ModelDiff	Zest	SAC	ModelGiF	FIT-ModelDiff	FIT-LIME
Copying	4	100%	100%	100%	100%	100%	100%	100%	100%
Fine-tuning	12	100%	100%	100%	100%	100%	100%	100%	100%
Pruning	12	100%	100%	100%	91.67%	100%	100%	100%	100%
Extraction	8	50%	87.5%	50%	25%	100%	100%	100%	100%
Transfer	12	N/A	N/A	100%	N/A	0%	100%	100%	100%
Independent	144	30.6%	4.8%	4.0%	7.6%	39.6%	0.0%	0.0%	0.0%

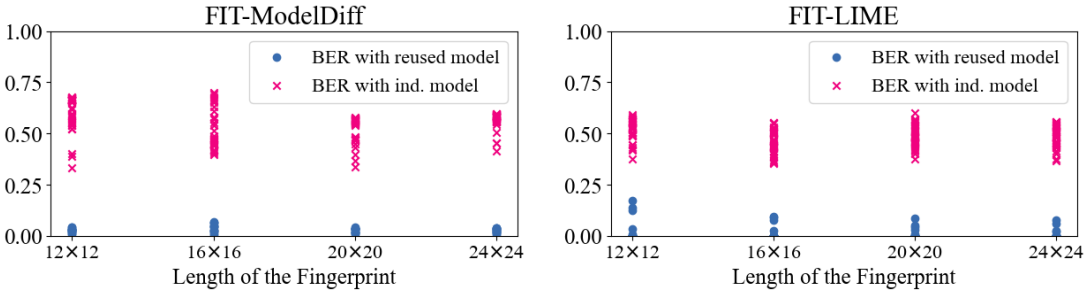


Fig. 5. The BERs of the reused models and independent models with different lengths of fingerprint. As the length increases, the BERs with reused and independent models become more concentrated.

TABLE III

THE AVERAGE DISTANCES OF REUSED MODELS (AVG. REUSED MODEL DIST.) AND INDEPENDENT MODELS (AVG. IND. MODEL DIST.) AND THE ℓ_2-NORM OF THE PERTURBATIONS $\mathcal{R}$ (ℓ_2-NORM OF PERT.) WITH DIFFERENT λ

Method→	FIT-ModelDiff					FIT-LIME				
Metric ↓ λ →	0.0	1.0	5.0	10.0	100.0	0.0	0.5	1.0	2.0	5.0
Avg. Reused Model Dist.	0.024	0.038	0.030	0.032	0.029	0.029	0.029	0.034	0.036	0.047
Avg. Ind. Model Dist.	0.568	0.570	0.561	0.566	0.577	0.505	0.505	0.510	0.506	0.512
ℓ_2-norm of Pert.	0.007	0.007	0.007	0.007	0.006	0.020	0.019	0.018	0.017	0.014

models and the independent models with different lengths of fingerprints. Moreover, as the length of $\boldsymbol{F}$ increases, the BERs of both reused and independent models are more concentrated, signifying that a larger fingerprint length can reduce the probability of outliers and have better security.

2) Effects of the ℓ_2-Norm Coefficient: λ is the coefficient of the scale of the perturbations in the loss function Eq. (8). In this experiment, we study the effect of λ on FIT-Print and adopt FIT-ModelDiff and FIT-LIME with five different λ. From Table III, since the scale of the perturbations in FIT-ModelDiff is quite small, varying λ does not significantly affect the perturbations as well as the distances with reused models. While in FIT-LIME, a larger λ can lead to a smaller perturbation. The ℓ_2-norm of the perturbations reduces from 0.020 to 0.014. In the meantime, the average distances with reused models become larger. Our experiments also suggest that the effect of λ on the distances with independent models is not significant. The visualization of the perturbed testing

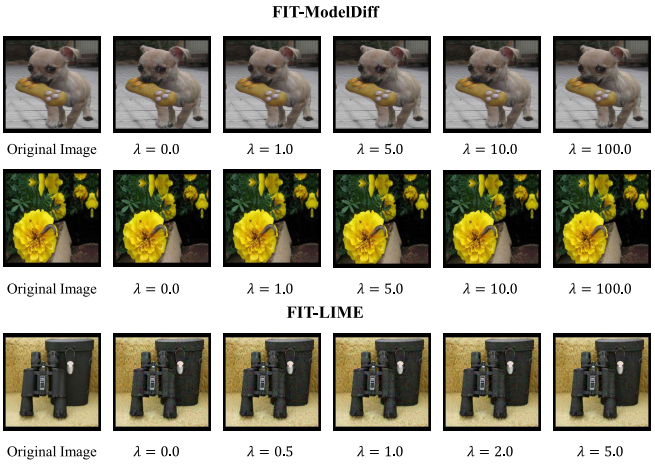
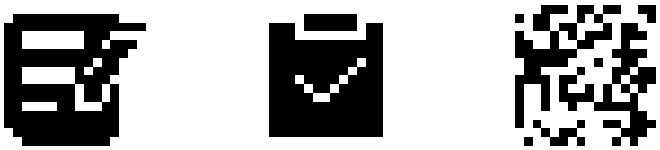


Fig. 6. The visualization of the original images and the perturbed testing samples with different λ.



(a) Target-file (b) Target-tick (c) Target-noise

Fig. 7. The visualization of the targeted fingerprints. We utilize three different target fingerprints in our experiments. The "Target-file" fingerprint is used in our main experiments.

samples with different λ can be found in Fig. 6 and it also confirms our above conclusion.

3) Effects of Different Target Fingerprints: We hereby explore how the properties of the target fingerprint (*e.g.*, spatial pattern and complexity) affect the collision probability (*i.e.*, false alarms on independent models) and overall security. Specifically, we choose three distinct target fingerprints representing different levels of complexity: **(1)** a "tick" image (simple spatial pattern, moderate complexity), **(2)** an image of a "file" and a "pen" (high complexity), and **(3)** a random noise image (no spatial pattern, maximum entropy). The visualization of the three fingerprints (resized to 16×16) is shown in Fig. 7.

The experimental results are shown in Fig. 8. From Fig. 8, we can find that regardless of the signature's pattern and complexity, all the BERs of reused models are lower, and the BERs of independent models are larger than the threshold. This demonstrates that our FIT-Print is effective with different target fingerprints.

4) Effects of Different Numbers of Augmented Models: In the testing sample extraction stage, FIT-Print utilizes the reused models as augmented models to enhance the conferrability of the fingerprint. In this section, we study leveraging different numbers of augmented models to extract the testing samples and test whether FIT-Print maintains a satisfactory conferrability. Fig. 9 depicts the BERs of the reused models with different numbers of augmented models. We set the number to 5, 6, 7, 8. From Fig. 9, we can find that as the number of augmented models increases, the BERs on reused models become smaller and more concentrated, which means using more reused models as augmented models can enhance the conferability of FIT-Print. In addition, while using only 5 reused models as augmented models, all the BERs are smaller than the threshold τ, which signifies the conferrability of FIT-Print.

TABLE IV

THE AVERAGE BIT ERROR RATES OF DIFFERENT α. THE RESULTS DEMONSTRATE THAT LARGER $\cos\alpha$ LEADS TO LOWER BERS, BUT OUR METHOD IS GENERALLY ROBUST TO THE CHOICE OF $\cos\alpha$

α	$\frac{1}{16}\pi$	$\frac{1}{8}\pi$	$\frac{1}{4}\pi$	$\frac{1}{3}\pi$
$\cos\alpha$	0.9808	0.9239	0.7071	0.5000
Avg. BER (↓)	0.015	0.033	0.118	0.217

5) Effects of the Bias Parameter α: In this section, we study the influence of the bias parameter α (as defined in Eq. (14)) on the performance of FIT-ModelDiff. We set α to $\frac{1}{16}\pi$, $\frac{1}{8}\pi$, $\frac{1}{4}\pi$, and $\frac{1}{3}\pi$. The results in Table IV show that larger $\cos\alpha$ leads to lower BERs, and thus may cause better robustness. However, our method is generally robust to the choice of the bias parameter α.

D. The Resistance to Adaptive False Claim Attack

We hereby evaluate our FIT-Print against the adaptive false claim attack, where the adversary utilizes Eq. (8) to optimize the testing samples yet intentionally crafts some independent models as augmented models to enhance the transferability of the adversary's false fingerprint. We utilize the models trained on ImageNet and their corresponding reused models as the augmented models. Fig. 10 demonstrates that adding independent augmented models does not significantly enhance the transferability of the fingerprint in FIT-Print because the BERs on the independent models are nearly unchanged. However, as shown in Section II-C, existing fingerprinting methods are vulnerable to false claim attacks due to their untargeted nature. In contrast, our targeted FIT-Print is significantly more resistant to false claims since targeted transferable fingerprints are much more difficult to craft (as analyzed in Section III-B). Empirically, such a phenomenon is also validated in the area of adversarial attacks [37], [38] (*i.e.*, targeted adversarial examples have lower transferability than untargeted ones).

E. The Resistance to Adaptive Fingerprint Removal Attacks

In real-world scenarios, the model reuser usually knows which model fingerprinting method is leveraged by the model developer and can accordingly design an adaptive attack against the utilized model fingerprinting method. Generally speaking, there are two different ways to attack a model fingerprinting method [39]: **(1)** fine-tuning the model (*i.e.*, model-based attacks) or **(2)** perturbing or preprocessing the input data (*i.e.*, input-based attacks) to obfuscate the fingerprint of the model.

In model-based adaptive attacks, the model reuser can fine-tune the model, attempting to remove the original fingerprint inside it. Based on the knowledge of the model reuser with the

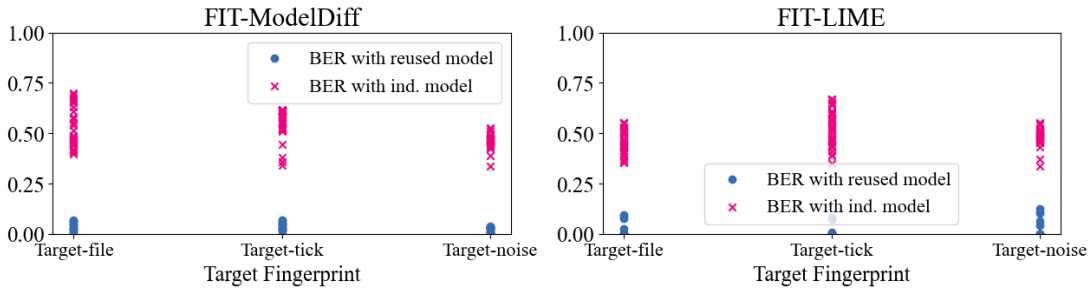


Fig. 8. The BERs of the reused models and independent models with different target fingerprints. Regardless of the target fingerprints, our FIT-ModelDiff and FIT-LIME have the capability to distinguish the reused models and the independent models.

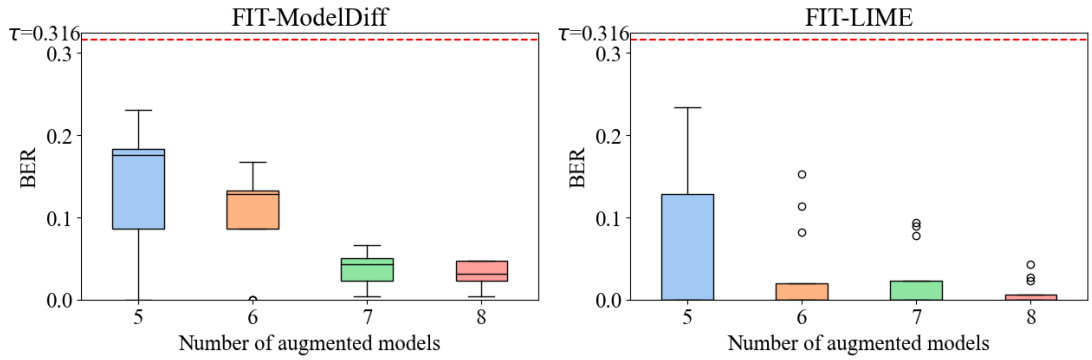


Fig. 9. The BERs of the reused models with different numbers of augmented models. As shown in this figure, as we expected, using more models for augmentations can have lower BERs.

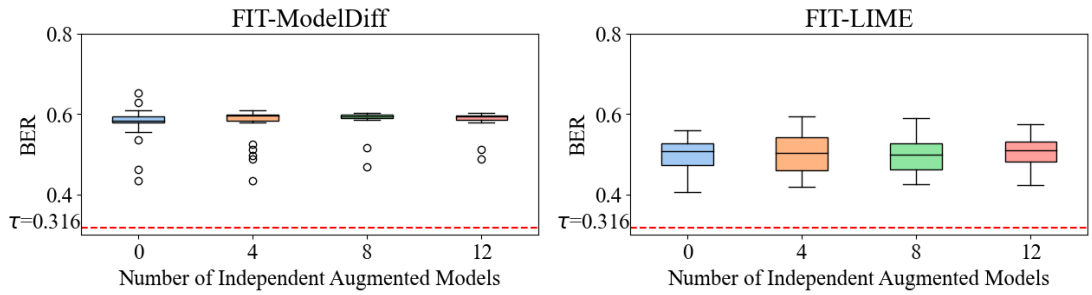


Fig. 10. The BERs of the independent models while conducting adaptive false claim attacks using different numbers of independent models as augmented models. The BERs are all larger than the threshold τ.

fingerprint of the model developer, we consider two different model-based adaptive attacks.

- *Overwriting Attack:* In overwriting attacks, we assume that the model reuser has no knowledge of the testing samples $\mathcal{X}_T$ and the target fingerprint $\boldsymbol{F}$ utilized by the model developer. Thereby, the model reuser may independently generate their own $\hat{\mathcal{X}}_T$ and $\hat{\boldsymbol{F}}$, and then fine-tunes the model to make the outputs of $\hat{\mathcal{X}}_T$ close to the target fingerprint $\hat{\boldsymbol{F}}$. The loss function can be defined as follows.

$$\min_{M_o} \frac{1}{|\hat{\mathcal{X}}_T|} \sum_{\hat{\boldsymbol{x}} \in \hat{\mathcal{X}}_T} \mathcal{L}(f(M_o(\hat{\boldsymbol{x}}), \hat{\boldsymbol{F}}). \tag{18}$$

- *Unlearning Attack:* In unlearning attacks, we assume that the model reuser knows the target fingerprint of the model developer, since it may be registered in a third-party institution and publicly accessible. However, the model reuser still has no knowledge of the testing samples. As such, the model reuser can construct some independent testing samples to unlearn the target fingerprint $\boldsymbol{F}$ from the model. The loss function can be defined as follows.

$$\max_{M_o} \frac{1}{|\hat{\mathcal{X}}_T|} \sum_{\hat{\boldsymbol{x}} \in \hat{\mathcal{X}}_T} \mathcal{L}(f(M_o(\hat{\boldsymbol{x}}), \boldsymbol{F}). \tag{19}$$

The results of the two adaptive attacks are demonstrated in Table V. The results indicate that neither attack can successfully remove the fingerprint from the model. Due to more knowledge about the fingerprint $\boldsymbol{F}$, the unlearning attack is slightly more effective than the overwriting attacks, but still not able to bypass the ownership verification, with a BER of 0.149. The experimental results show that the model reuser cannot destroy the fingerprint inside the model when having

TABLE V

THE BERS BEFORE AND AFTER THE ADAPTIVE OVERWRITING ATTACK AND UNLEARNING ATTACK. THE BERS AFTER ATTACKS ARE STILL LOW ENOUGH TO BE IDENTIFIED AS A REUSED MODEL. THUS, FIT-PRINT IS ABLE TO RESIST THE OVERWRITING ATTACK AND UNLEARNING ATTACK

Method	Before Attack	After Overwriting	After Unlearning
FIT-ModelDiff	0.047	0.051	0.149
FIT-LIME	0.000	0.016	0.016

TABLE VI

THE PERFORMANCE OF FIT-MODELDIFF AND FIT-LIME IN THE LABEL-ONLY SCENARIO. THE RESULTS SHOW THAT FIT-PRINT CAN STILL DISTINGUISH THE REUSED MODELS WITH ONLY TOP-1 LABELS

Metric↓ Method→	FIT-ModelDiff	FIT-LIME
Avg. BER	0.227	0.135
Ownership Verification Rate	1.000	1.000
Avg. BER of Ind. Models	0.369	0.399
False Positive Rate	0.000	0.000

no knowledge of the testing samples. Our FIT-Print can resist both the overwriting attack and the unlearning attack.

F. FIT-Print in the Label-Only Scenario

In this section, we investigate the effectiveness of FIT-Print in the label-only scenario. In such a scenario, the verifier can only obtain the Top-1 label instead of the logits as output. Unfortunately, detecting transfer learning models, which is one of our considered important model reuse settings, is still an open problem in model ownership verification [10]. Transfer learning can change the task of the model and the output classes. It is hard to determine whether a model is transferred from another with only top-1 labels. As such, we do not consider transfer learning models in the following discussion.

FIT-Print can easily be extended to distinguish the reused models (except transfer learning models) with the top-1 labels. In the label-only scenario, we can construct a binary vector $\boldsymbol{b}$ to replace the original logits. Assuming that the predicted top-1 class is a, the a-th element in $\boldsymbol{b}$ is set to 1, and the other elements are 0. The other processes remain unchanged.

To verify the effectiveness, we conduct additional experiments. Table VI shows the average bit error rate (Avg. BER) on the reused models and the independent models (Ind. Models). We also present the ownership verification rates on the reused models and the false positive rates on the independent models. The results show that our methods are still highly effective under the label-only setting, although the average BERs of the reused models decrease in this scenario.

V. DISCUSSION

A. Related Work

As the deployment of deep learning expands, ensuring the security, privacy, and trustworthiness of AI models and systems has drawn significant attention [40], [41]. Within the broader context of trustworthy AI, in this section, we provide a comprehensive discussion on model fingerprinting. In general, existing model fingerprinting can be categorized into white-box and black-box methods [10].

1) White-Box Model Fingerprinting Methods: In the white-box scenario, since the model developer can get access to the parameters of the suspicious model, a direct way to compare the models is to compare the weights (or their hash values) of the models. Some existing white-box model fingerprinting methods leveraged the path of model training [42], the random projection of model weights [43], or the learnable hash of the model weights [44]. Some recent works also explored utilizing the deep representations (*e.g.*, gradients [35]) or the intermediate results [45] of the testing samples as the fingerprint. Recently, these concepts have also been extended to large language models (LLMs) [46] by extracting unique fingerprints from static transformer weights [47], dynamic forward-pass representations [48], or backward-pass gradients [49]. However, similar to traditional white-box watermarking, the strict requirement of full parameter access severely restricts their practical deployment in real-world scenarios, especially for closed-source commercial models.

2) Black-Box Model Fingerprinting Methods: In the black-box scenario, the model developer is assumed to have only API access to the suspicious model. Existing black-box model fingerprinting methods can be classified into adversarial-example-based (AE-based) methods [14], [19], [50] and testing-based methods [15], [45], [51] and we have presented their formulations in Section II-B. AE-based methods craft adversarial examples to identify the decision boundary of different models [14]. Lukas et al. [19] proposed to craft some reused models as augmented models to improve the conferrability of the fingerprint. On the contrary, testing-based methods compare the model behavior on the testing samples at the specific mapping function. ModelDiff [15] and SAC [23] utilized the distances between the output logits of different input samples, while Zest [16] took the feature attribution map output by LIME [27] as the mapping function. In addition, Chen et al. [45] proposed a series of mapping functions to calculate model similarity. Compared to AE-based methods, testing-based methods have the capability to compare models across different tasks and output formats, thereby attracting greater attention in practice. In the context of modern LLMs, black-box approaches are further categorized into untargeted and targeted paradigms [46]. While untargeted methods analyze the statistical or semantic features of general generation responses [52], [53], targeted LLM fingerprinting utilizes adversarially optimized prompts to force the model to output specific, pre-defined textual signatures [54]. These emerging LLM studies share similar underlying philosophies with our FIT-Print paradigm in pursuing more reliable and resilient copyright auditing.

B. Extending FIT-Print to Other Models and Datasets

In our main experiments, we focus on image classification models. It is also technically feasible to extend our FIT-Print to models of any task. In this section, we discuss how FIT-Print can generalize to different types of models and data.

1) The Extension to Other Models: We argue that our FIT-Print can also generalize to models with different architectures

TABLE VII
THE BIT ERROR RATES (BER) OF APPLYING FIT-MODELDIFF AND FIT-LIME TO ADVANCED IMAGE CLASSIFICATION MODELS

Model	ViT	SwinViT	EfficientNet
FIT-ModelDiff	0.117	0.125	0.164
FIT-LIME	0.008	0.000	0.012

TABLE VIII
THE BERS OF APPLYING FIT-MODELDIFF AND FIT-LIME TO TEXT GENERATION MODELS (LOWER IS BETTER)

Method→	FIT-ModelDiff		FIT-LIME	
Dataset↓ Model→	GPT-2	BERT	GPT-2	BERT
ptb-text-only	0.188	0.188	0.031	0.000
lambada	0.188	0.125	0.062	0.000

and tasks. For models with different architectures, since we do not make any assumptions about the architecture of the models and we also do not need to alter or fine-tune the model, our method can fundamentally generalize to models with other architectures (*e.g.*, transformers [55]) as well. For models with different tasks, the major difference between models with different tasks is the output format. For instance, the image generation model outputs a tensor consisting of a sequence of logits. FIT-ModelDiff calculates the cosine similarity between the outputs, and FIT-LIME calculates the average entropy of the output. Arguably, these calculation methods can be applied to any output format (*e.g.*, 1-D vectors, 2-D matrices, or tensors). As such, our methods are naturally feasible for models with different tasks in practice.

To further demonstrate the extensibility, we conduct additional empirical evaluations on several advanced and widely adopted image classification architectures, specifically Vision Transformer (ViT) [56], Swin Transformer (SwinViT) [57], and EfficientNet [58]. The experimental results are in Table VII. As illustrated, both FIT-ModelDiff and FIT-LIME maintain exceptional performance with BERs < 0.17 across these diverse model architectures. These results compellingly indicate that FIT-Print can successfully extract targeted fingerprints and verify ownership regardless of the underlying model structure.

2) The Extension to Other (Types of) Datasets: Our FIT-Print can also generalize to other types of datasets. Our primitive FIT-Print aims to optimize a perturbation $\boldsymbol{r}$ on the input $\boldsymbol{x}$ to make the mapping vector close to the targeted fingerprint. The main part of the loss function is as Eq. (20).

$$\min \mathcal{L}(f(M_o(\boldsymbol{x}+\boldsymbol{r}), \boldsymbol{F}). \tag{20}$$

However, for the discrete data (*e.g.*, text data), it is not feasible to directly add the perturbation to it. Thus, a rewriting function $g(\boldsymbol{x})$ can be introduced to rewrite the characters, words, or sentences. The loss function can be changed to Eq. (21).

$$\min \mathcal{L}(f(M_o(g(\boldsymbol{x})), \boldsymbol{F}). \tag{21}$$

The main challenge lies in how to design an effective optimization method to find a rewriting function $g(\boldsymbol{x})$ which minimizes the above loss function. There are already some existing works [59], [60] to fulfill this task. Accordingly, FIT-Print can be adapted to other data formats (*e.g.*, text or tabular).

3) Case Study on Text Generation Model: We conduct a case study on implementing FIT-Print on text generation models. Text generation models [61], [62] have become the most famous models in recent years and have been widely applied in various domains. Specifically, the text generation model predicts the next token in a sequence of tokens, *i.e.*, the output of the text generation models is a sequence of logits. Given an input sequence $\boldsymbol{s} = \{s_1, s_2, \ldots, s_q\}$, where q is the number of tokens in the sequence, and a vocabulary $\mathcal{V}$, the text generation model outputs a sequence $\boldsymbol{o} \in \mathbb{R}^{q \times |\mathcal{V}|}$. The i-th element in $\boldsymbol{o}$ is the probability logit of the tokens in the vocabulary.

To implement FIT-Print on text generation models, we need to optimize Eq. (21) to generate the testing samples. Arguably, our FIT-ModelDiff and FIT-LIME can easily generalize to protect text generation models. Specifically, the mapping functions used in FIT-ModelDiff and FIT-LIME (*i.e.*, cosine similarity and average entropy) can be directly applied to text generation models. This is because text generation models differ from classification models only in the output dimension, and these two functions are inherently able to calculate data with different dimensions. The main challenge is to optimize the discrete text data to minimize Eq. (21). To achieve this goal, we can exploit existing text optimization methods [59], [60]. Specifically, we implement the optimization method proposed in [60]. It optimizes the embeddings of the text sequences and then finds the nearest token in the embedding space to replace the original token.

We further conduct experiments to verify the effectiveness of our FIT-ModelDiff and FIT-LIME on text generation models. We use two popular text generation models (*i.e.*, GPT-2 [63] and BERT [64]) and two datasets (*i.e.*, ptb-text-only [65] and lambada [66]) for our case study. Table VIII shows the bit error rates (BERs) of applying our methods to text generation models. The BERs are all lower than the threshold $\tau = 0.227$, indicating that FIT-Print is also applicable to protect the IPR on other data formats.

C. The Overhead of FIT-ModelDiff and FIT-LIME

Compared with existing model fingerprinting methods, FIT-Print needs to optimize the testing samples and thus has an extra overhead. We hereby present a detailed analysis of the time and space complexity of the two fingerprinting methods, FIT-ModelDiff and FIT-LIME. FIT-ModelDiff and FIT-LIME are the representatives of bit-wise and list-wise methods, respectively. As such, they have different trade-offs in time and space overhead.

1) Overhead During the Fingerprint Verification Stage:

- For FIT-ModelDiff, the space complexity is $O(1)$ and the time complexity is $O(k)$ where k is the length of the targeted fingerprint. FIT-ModelDiff is a bit-wise fingerprinting method that extracts the fingerprint bit by bit. It only reads two samples to calculate one bit in the fingerprint in each step. The fingerprint can be extracted

TABLE IX
THE TIME OVERHEAD (SECONDS) OF EACH OPTIMIZATION ITERATION APPLYING FIT-MODELDIFF AND FIT-LIME WITH DIFFERENT MODELS. "#PARAMS" INDICATES THE NUMBER OF PARAMETERS

Model	MobileNetV2	EfficientNet	ResNet18	SwinViT	ViT
#Params (M)	3.50	5.29	11.69	28.29	86.57
FIT-ModelDiff	4.30	5.06	4.07	5.25	6.57
FIT-LIME	0.84	1.24	0.74	1.34	2.13

in k steps. Therefore, the space complexity is $O(1)$, and the time complexity is $O(k)$.

- For FIT-LIME, the space complexity is $O(k)$ and the time complexity is $O(k/\beta)$ where β is the batch size. FIT-LIME is a list-wise method that extracts the fingerprint as a whole list. FIT-LIME needs to read all the samples at the same time to extract the fingerprint. On the other hand, FIT-LIME can calculate the outputs of an entire batch at the same time. As such, the space complexity is $O(k)$, and the time complexity is $O(k/\beta)$.

2) Overhead During the Testing Samples Extraction Stage: In each iteration of optimizing the testing samples, we need to perform one forward propagation and one backward propagation of the fingerprint verification method. Assuming that we utilize ξ augmented models during optimization, the time complexities of each iteration of testing sample extraction in FIT-ModelDiff and FIT-LIME are $O(\xi \cdot k)$ and $O(\xi \cdot k/\beta)$. k is the length of the targeted fingerprint and β is the batch size. We evaluate the time overhead during the testing samples extraction stage for different methods, as presented in Table IX. Overall, our proposed FIT-ModelDiff and FIT-LIME demonstrate high efficiency. Even with the large ViT model, the time cost is still low ($<$ 7s per iteration and $<$ 0.6h for the entire 300 iterations). Furthermore, the testing sample extraction process is a one-time operation for the protected model and does not require repeated execution. As such, our FIT-Print is efficient in optimization, and the overall overhead is acceptable.

D. Potential Limitations and Future Directions

As the first attempt at the target fingerprinting paradigm to defeat false claim attacks, we must acknowledge that our method still has the following potential limitations.

First, our FIT-Print depends on a trustworthy third-party institution to register the fingerprint with a timestamp, which is not currently established. However, we argue that this will certainly be realized in the foreseeable future. For example, the existing intellectual property office (IPO) or artificial intelligence regulator (AIR) can be responsible for this duty [67]. First, it is common for developers to register their intellectual property, including valuable models, with the IPO for copyright protection. Second, many countries and regions are in the process of establishing or have established the AIR (*e.g.*, as exemplified in the EU Artificial Intelligence Act) to ensure security and transparency before deploying the AI models. As such, it is also feasible for the AIR to manage model registrations and audit potential infringement.

Second, our FIT-Print fails to provide formal proof of the resistance to false claim attacks due to the complexity of DNNs. We will investigate how to achieve a certified robust model fingerprinting method against false claim attacks with a rigorous conceptual guarantee in our future work.

Thirdly, as discussed in Section V-B, for deterministic models that do not involve randomness (*e.g.*, CNN and LLM), our FIT-Print can generalize to different types of models and datasets. However, for non-deterministic models that involve randomness (*e.g.*, diffusion models [68]), we have to admit that we do not know whether our methods and existing fingerprinting methods can be adapted to them since they have a completely different inference paradigm. We will further explore and discuss this intriguing problem in our future work.

VI. CONCLUSION

In this paper, we revisited existing model fingerprinting methods and revealed their vulnerability to false claim attacks by designing an attack that crafts transferable, "easy" samples. We demonstrated that this vulnerability primarily stems from the untargeted nature of current approaches, which compare the outputs of arbitrary samples across models rather than evaluating their similarity to specific signatures. To address this issue, we proposed FIT-Print, a targeted, false-claim-resistant model fingerprinting paradigm. By optimizing testing samples, FIT-Print effectively transforms the model fingerprint into a specific, verifiable signature. Based on this framework, we developed two corresponding methods: the bit-wise FIT-ModelDiff and the list-wise FIT-LIME. Extensive empirical experiments verified the effectiveness, conferrability, and robust resistance of FIT-Print against false claim attacks. Ultimately, we hope that FIT-Print offers a new perspective on model fingerprinting, facilitating more secure and trustworthy model sharing and trading.

ACKNOWLEDGMENT

The authors sincerely thank Prof. Kui Ren from Zhejiang University for the helpful comments and suggestions on an early draft of this article, and Dr. Najeeb Jebreel from Universitat Rovira i Virgili for his assistance with language editing and proofreading.

REFERENCES

[1] Z. Wang, C. Chen, L. Lyu, D. N. Metaxas, and S. Ma, "DIAGNOSIS: Detecting unauthorized data usages in text-to-image diffusion models," in *Proc. Int. Conf. Learn. Represent.*, 2024, pp. 1–21.

[2] X. Zhuang, L. Zhang, C. Tang, and Y. Li, "DeepReg: A trustworthy and privacy-friendly ownership regulatory framework for deep learning models," *IEEE Trans. Inf. Forensics Security*, vol. 20, pp. 854–870, 2025.

[3] H. Zhang, S. Shao, S. Li, Z. Zhong, Y. Liu, and Z. Qin, "SmartGuard: Leveraging large language models for network attack detection through audit log analysis and summarization," 2025, *arXiv:2506.16981*.

[4] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*, 2018, pp. 273–294.

[5] F. Zhuang et al., "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021.

[6] X. Luan, X. Zhang, J. Wang, and M. Sun, "Protecting deep learning model copyrights with adversarial example-free reuse detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 10, pp. 19187–19199, Oct. 2025.

[7] C. Wei et al., "PointNCBW: Toward dataset ownership verification for point clouds via negative clean-label backdoor watermark," *IEEE Trans. Inf. Forensics Security*, vol. 20, pp. 191–206, 2025.

[8] K. Gao et al., "Toward dataset copyright evasion attack against personalized text-to-image diffusion models," *IEEE Trans. Inf. Forensics Security*, vol. 21, pp. 725–740, 2026.

[9] J. Zhang, D. Chen, J. Liao, W. Zhang, G. Hua, and N. Yu, "Passport-aware normalization for deep model protection," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2020, pp. 22619–22628.

[10] Y. Sun et al., "Deep intellectual property protection: A survey," 2023, *arXiv:2304.14613*.

[11] Y. Li et al., "MOVE: Effective and harmless ownership verification via embedded external features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 6, pp. 4734–4751, Jun. 2025.

[12] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *Proc. USENIX Secur. Symp.*, 2018, pp. 1615–1631.

[13] S. Shao et al., "FedTracker: Furnishing ownership verification and traceability for federated learning model," *IEEE Trans. Dependable Secure Comput.*, vol. 22, no. 1, pp. 114–131, Jan. 2025.

[14] X. Cao, J. Jia, and N. Z. Gong, "IPGuard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, May 2021, pp. 14–25.

[15] Y. Li, Z. Zhang, B. Liu, Z. Yang, and Y. Liu, "ModelDiff: Testing-based DNN similarity comparison for model reuse detection," in *Proc. 30th ACM SIGSOFT Int. Symp. Softw. Test. Anal.*, Jul. 2021, pp. 139–151.

[16] H. Jia, H. Chen, J. Guan, A. S. Shamsabadi, and N. Papernot, "A zest of LIME: Towards architecture-independent model distances," in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–17.

[17] K. Yang, R. Wang, and L. Wang, "MetaFinger: Fingerprinting the deep neural networks with meta-training," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 776–782.

[18] J. Liu, R. Zhang, S. Szyller, K. Ren, and N. Asokan, "False claims against model ownership resolution," in *Proc. USENIX Secur. Symp.*, 2024, pp. 6885–6902.

[19] N. Lukas, Y. Zhang, and F. Kerschbaum, "Deep Neural Network Fingerprinting by Conferrable Adversarial Examples," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–18.

[20] M. Pautov, N. Bogdanov, S. Pyatkin, O. Rogov, and I. Oseledets, "Probabilistically robust watermarking of neural networks," 2024, *arXiv:2401.08261*.

[21] K. Ren, T. Zheng, Z. Qin, and X. Liu, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346–360, Mar. 2020.

[22] Z. Wang, H. Guo, Z. Zhang, W. Liu, Z. Qin, and K. Ren, "Feature importance-aware transferable adversarial attacks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 7619–7628.

[23] J. Guan, R. He, and J. Liang, "Are you stealing my model? Sample correlation for fingerprinting deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 36571–36584.

[24] S. Shao et al., "DATABench: Evaluating dataset auditing in deep learning from an adversarial perspective," 2025, *arXiv:2507.05622*.

[25] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–11.

[26] L. Fan, K. W. Ng, and C. S. Chan, "Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2019, pp. 4714–4723.

[27] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1135–1144.

[28] S. Shao, Y. Li, H. Yao, Y. He, Z. Qin, and K. Ren, "Explanation as a watermark: Towards harmless and multi-bit model ownership verification via watermarking feature attribution," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2025, pp. 1–18.

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[31] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proc. 6th Indian Conf. Comput. Vis., Graph. Image Process.*, Dec. 2008, pp. 722–729.

[32] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, "Novel dataset for fine-grained image categorization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshop*, Jun. 2011, pp. 1–2.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[34] X. Pan, Y. Yan, M. Zhang, and M. Yang, "MetaV: A meta-verifier approach to task-agnostic model fingerprinting," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 1327–1336.

[35] J. Song, Z. Xu, S. Wu, G. Chen, and M. Song, "ModelGiF: Gradient fields for model functional distance," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 6102–6112.

[36] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–16.

[37] K. Wang, J. Shi, and W. Wang, "LFAA: Crafting transferable targeted adversarial examples with low-frequency perturbations," 2023, *arXiv:2310.20175*.

[38] Z. Wang et al., "Towards transferable targeted adversarial examples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 20534–20543.

[39] H. Yao, Z. Li, K. Huang, J. Lou, Z. Qin, and K. Ren, "RemovalNet: DNN fingerprint removal attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 4, pp. 2645–2658, Jul. 2024.

[40] S. Jiang, H. Yang, Q. Xie, C. Ma, S. Wang, and G. Xing, "Lancelot: Towards efficient and privacy-preserving Byzantine-robust federated learning within fully homomorphic encryption," 2024, *arXiv:2408.06197*.

[41] Q. Xie et al., "HARMONY: A privacy-preserving and sensor-agnostic tele-monitoring system," in *Proc. 33rd Int. Joint Conf. Artif. Intell.*, Aug. 2024, pp. 9945–9953.

[42] H. Jia et al., "Proof-of-learning: Definitions and practice," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 1039–1056.

[43] Y. Zheng, S. Wang, and C.-H. Chang, "A DNN fingerprint for non-repudiable model ownership identification and piracy detection," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 2977–2989, 2022.

[44] C. Xiong, G. Feng, X. Li, X. Zhang, and C. Qin, "Neural network model protection with piracy identification and tampering localization capability," in *Proc. 30th ACM Int. Conf. Multimedia*, Oct. 2022, pp. 2881–2889.

[45] J. Chen et al., "Copy, right? A testing framework for copyright protection of deep learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 824–841.

[46] S. Shao et al., "SoK: Large language model copyright auditing via fingerprinting," 2025, *arXiv:2508.19843*.

[47] Y. Hu et al., "HuRef: HUman-REadable fingerprint for large language models," in *Proc. Adv. Neural Inf. Process. Syst. 37*, 2024, pp. 126332–126362.

[48] J. Zhang et al., "REEF: Representation encoding fingerprints for large language models," in *Proc. Int. Conf. Learn. Represent.*, 2025, pp. 1–26.

[49] Z. Wu, Y. Zhao, and H. Wang, "TensorGuard: Gradient-based model fingerprinting for LLM similarity detection and family classification," in *Proc. 40th IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Nov. 2025, pp. 445–456.

[50] S. Wang, X. Wang, P.-Y. Chen, P. Zhao, and X. Lin, "Characteristic examples: High-robustness, low-transferability fingerprinting of neural networks," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 575–582.

[51] Y. Chen, C. Shen, C. Wang, and Y. Zhang, "Teacher model fingerprinting attacks against transfer learning," in *Proc. USENIX Secur. Symp.*, 2022, pp. 3593–3610.

[52] D. Pasquini, E. M. Kornaropoulos, and G. Ateniese, "LLMmap: Fingerprinting for large language models," in *Proc. USENIX Secur. Symp.*, 2025, pp. 299–318.

[53] S. Shao, Y. Li, H. Yao, Y. Chen, Y. Yang, and Z. Qin, "Reading between the lines: Towards reliable black-box LLM fingerprinting via zeroth-order gradient estimation," in *Proc. ACM Web Conf.*, Apr. 2026, pp. 2637–2648.

[54] M. Gubri, D. Ulmer, H. Lee, S. Yun, and S. J. Oh, "TRAP: Targeted random adversarial prompt honeypot for black-box identification," in *Proc. Findings Assoc. Comput. Linguistics ACL*, 2024, pp. 11496–11517.

[55] A. Vaswani et al., "Attention is all you need," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6000–6010.

[56] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–21.

[57] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10012–10022.

[58] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[59] C. Guo, A. Sablayrolles, H. Jégou, and D. Kiela, "Gradient-based adversarial attacks against text transformers," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 5747–5757.

[60] J. Geiping, M. Goldblum, T. Goldstein, N. Jain, J. Kirchenbauer, and Y. Wen, "Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery," in *Proc. Adv. Neural Inf. Process. Syst. 36*, 2023, pp. 51008–51025.

[61] K. Pang, T. Qi, C. Wu, M. Bai, M. Jiang, and Y. Huang, "ModelShield: Adaptive and robust watermark against model extraction attack," *IEEE Trans. Inf. Forensics Security*, vol. 20, pp. 1767–1782, 2025.

[62] J. Guan, J. Liang, Y. Wang, and R. He, "Sample correlation for fingerprinting deep face recognition," *Int. J. Comput. Vis.*, vol. 2024, pp. 1–15, Jan. 2024.

[63] A. Radford et al., "Language models are unsupervised multitask learners," *OpenAI Blog*, pp. 1–24, 2019.

[64] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2018, pp. 4171–4186.

[65] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," *Comput. Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[66] D. Paperno et al., "The LAMBADA dataset: Word prediction requiring a broad discourse context," 2016, *arXiv:1606.06031*.

[67] Y. Li et al., "Rethinking data protection in the (generative) artificial intelligence era," 2025, *arXiv:2507.03034*.

[68] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. NIPS*, vol. 33. Vancouver, BC, Canada: Curran Associates, 2020, pp. 6840–6851.