# IrisFP: Adversarial-Example-based Model Fingerprinting with Enhanced Uniqueness and Robustness

Supplementary Material

## 6. Threat Model

We consider a typical model fingerprinting scenario involving two entities: a model owner and an adversary. Specifically, the model owner first produces a protected model $f$ and then exploits adversarial example techniques to generate a set of fingerprints, which are kept confidential, to protect the IP of $f$. On the other hand, the adversary first obtains an unauthorized copy of $f$, for example, via white-box access or black-box extraction, and subsequently modifies it using model modification techniques, such as fine-tuning (FT), pruning (PR), adversarial training (AT), knowledge distillation (KD), etc., to produce a variant $f_p$. The adversary then deploys $f_p$ in a black-box setting, making it accessible to the public through APIs.

Suppose that the model owner identifies a black-box model $f_t$ as a suspicious target. The owner's goal is to determine whether $f_t$ is derived from the protected model $f$. To perform ownership verification, the owner queries $f_t$ using the generated fingerprints and compares the returned outputs with the expected fingerprint outputs. If the number of matches exceeds a predefined threshold, the target model is deemed to be an infringing model.

## 7. Additional Results

Tables 3 and 4 present additional results on the robustness of the protected models using the MobileNet-V2 and ViT-B/16 architectures, respectively. For MobileNet-V2, we evaluate IrisFP against four representative baselines across five datasets—CIFAR-10, CIFAR-100, Fashion-MNIST, MNIST, and Tiny-ImageNet—under six model modification attacks: FT, PR, KD, AT, NFT, and PFT. IrisFP remains consistently robust across datasets, achieving the highest AUCs on CIFAR-100, Fashion-MNIST, and MNIST under all attacks, and leading on Tiny-ImageNet for five out of six attacks (FT, KD, AT, PFT, NFT), with only a slight drop on PR. On CIFAR-10, IrisFP attains the highest AUCs under PR, AT, and PFT, while remaining competitive on the remaining attacks. For the protected model with ViT-B/16 architecture trained on Tiny-ImageNet, IrisFP achieves the highest AUCs across all six attacks, demonstrating strong robustness on transformer-based architectures as well. Together with the results on ResNet-18 in the main text (i.e., Table 2), these findings confirm that IrisFP maintains high robustness against model modification attacks across diverse model architectures—lightweight and high-capacity convolutional models as well as modern vision transformers.

## 8. Impact of Hyperparameters

To ensure a comprehensive evaluation, we consider three protected models in the following experiments: a ResNet-18 trained on CIFAR-100, a MobileNet-V2 trained on Fashion-MNIST, and a ViT-B/16 trained on Tiny-ImageNet.

### 8.1. Impact of combination of $K$ and $T$

We examine how the allocation between the number of fingerprints $K$ and the number of elements $T$ per fingerprint affects the verification performance when the total number of queries $K \times T$ is fixed at 200. As shown in Figure 7, across three protected models (ResNet-18 on CIFAR-100, MobileNet-V2 on Fashion-MNIST, and ViT-B/16 on Tiny-ImageNet), the AUC varies notably across different $(K, T)$ combinations even though the total query budget remains constant. Specifically, the performance improves steadily as $K$ increases from 10 to 40 while $T$ decreases from 20 to 5, and then drops when $K$ further increases to 100 with $T = 2$. For example, for ResNet-18 on CIFAR-100, the AUC rises from 0.842 at $(10, 20)$ to 0.905 at $(20, 10)$ and peaks at 0.916 for $(40, 5)$, before declining to 0.872 at $(100, 2)$. Similar trends are observed for MobileNet-V2 on Fashion-MNIST and ViT-B/16 on Tiny-ImageNet. Overall, configurations with larger $K$ and smaller $T$ consistently outperform those with smaller $K$ and larger $T$ under the same total query budget. The configuration $(K = 40, T = 5)$ achieves the highest AUC across all settings under the fixed total query budget. This is because a larger $K$ allows the verification process to rely on a broader set of composite-sample fingerprints, which enhances the overall statistical reliability of the verification results.

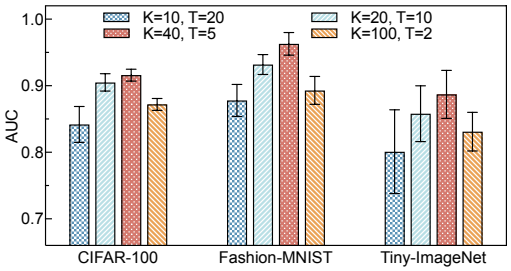


Figure 7. AUCs under different fingerprint numbers (K) and sample counts (T).

Table 3. AUCs under six attacks across datasets and methods on the protected model with MobileNet-V2 architecture.

| Dataset | Method | FT | PR | KD | AT | PFT | NFT |
|---|---|---|---|---|---|---|---|
| CIFAR-10 | IPGuard | 0.951 ± 0.022 | 0.994 ± 0.000 | 0.622 ± 0.048 | 0.481 ± 0.127 | 0.942 ± 0.029 | 0.934 ± 0.061 |
| | UAP | 0.891 ± 0.011 | 0.915 ± 0.003 | 0.526 ± 0.052 | 0.412 ± 0.098 | 0.892 ± 0.031 | 0.880 ± 0.009 |
| | ADV-TRA | **0.992 ± 0.025** | 0.995 ± 0.001 | 0.780 ± 0.068 | 0.214 ± 0.045 | 0.992 ± 0.040 | **0.992 ± 0.063** |
| | AKH | 0.896 ± 0.034 | 0.904 ± 0.010 | **0.885 ± 0.071** | 0.727 ± 0.052 | 0.935 ± 0.073 | 0.921 ± 0.084 |
| | **IrisFP** | 0.981 ± 0.006 | **0.997 ± 0.000** | 0.712 ± 0.055 | **0.978 ± 0.015** | **0.995 ± 0.003** | 0.982 ± 0.005 |
| CIFAR-100 | IPGuard | 0.981 ± 0.002 | 0.983 ± 0.005 | 0.741 ± 0.036 | 0.208 ± 0.087 | 0.982 ± 0.000 | 0.982 ± 0.001 |
| | UAP | 0.979 ± 0.010 | 0.987 ± 0.003 | 0.774 ± 0.079 | 0.723 ± 0.036 | 0.904 ± 0.056 | 0.929 ± 0.014 |
| | ADV-TRA | 0.963 ± 0.012 | 0.981 ± 0.001 | 0.701 ± 0.027 | 0.632 ± 0.018 | 0.885 ± 0.003 | 0.932 ± 0.009 |
| | AKH | 0.958 ± 0.007 | 0.979 ± 0.028 | 0.783 ± 0.068 | 0.656 ± 0.075 | 0.951 ± 0.057 | 0.923 ± 0.005 |
| | **IrisFP** | **0.983 ± 0.002** | **0.991 ± 0.001** | **0.811 ± 0.051** | **0.886 ± 0.055** | **0.983 ± 0.000** | **0.983 ± 0.000** |
| Fashion-MNIST | IPGuard | 0.587 ± 0.027 | 0.945 ± 0.046 | 0.518 ± 0.118 | 0.550 ± 0.090 | 0.555 ± 0.016 | 0.543 ± 0.103 |
| | UAP | 0.897 ± 0.008 | 0.977 ± 0.007 | 0.658 ± 0.076 | 0.735 ± 0.031 | 0.971 ± 0.005 | 0.886 ± 0.013 |
| | ADV-TRA | 0.963 ± 0.032 | 0.964 ± 0.012 | 0.721 ± 0.058 | 0.871 ± 0.036 | 0.898 ± 0.007 | 0.723 ± 0.037 |
| | AKH | 0.879 ± 0.051 | 0.884 ± 0.038 | 0.821 ± 0.079 | 0.832 ± 0.040 | 0.855 ± 0.078 | 0.796 ± 0.016 |
| | **IrisFP** | **0.975 ± 0.012** | **0.991 ± 0.000** | **0.860 ± 0.109** | **0.960 ± 0.017** | **0.979 ± 0.006** | **0.978 ± 0.010** |
| MNIST | IPGuard | 0.690 ± 0.007 | 0.710 ± 0.030 | 0.656 ± 0.020 | 0.560 ± 0.021 | 0.535 ± 0.003 | 0.757 ± 0.018 |
| | UAP | 0.767 ± 0.058 | 0.910 ± 0.000 | 0.668 ± 0.057 | 0.544 ± 0.011 | 0.672 ± 0.060 | 0.693 ± 0.022 |
| | ADV-TRA | 0.821 ± 0.039 | 0.896 ± 0.013 | 0.642 ± 0.033 | 0.557 ± 0.032 | 0.634 ± 0.014 | 0.703 ± 0.021 |
| | AKH | 0.871 ± 0.011 | 0.921 ± 0.015 | 0.569 ± 0.023 | 0.540 ± 0.052 | 0.885 ± 0.021 | 0.836 ± 0.009 |
| | **IrisFP** | **0.983 ± 0.000** | **0.960 ± 0.014** | **0.683 ± 0.041** | **0.577 ± 0.107** | **0.981 ± 0.001** | **0.981 ± 0.003** |
| Tiny-ImageNet | IPGuard | 0.466 ± 0.086 | **0.996 ± 0.005** | 0.786 ± 0.006 | 0.538 ± 0.237 | 0.478 ± 0.074 | 0.475 ± 0.056 |
| | UAP | 0.896 ± 0.014 | 0.985 ± 0.003 | 0.799 ± 0.035 | 0.633 ± 0.097 | 0.872 ± 0.036 | 0.843 ± 0.038 |
| | ADV-TRA | 0.923 ± 0.013 | 0.932 ± 0.011 | 0.805 ± 0.077 | 0.836 ± 0.040 | 0.840 ± 0.087 | 0.848 ± 0.029 |
| | AKH | 0.850 ± 0.037 | 0.901 ± 0.023 | 0.820 ± 0.052 | 0.809 ± 0.014 | 0.843 ± 0.038 | 0.859 ± 0.024 |
| | **IrisFP** | **0.979 ± 0.011** | 0.956 ± 0.043 | **0.883 ± 0.096** | **0.863 ± 0.160** | **0.995 ± 0.001** | **0.963 ± 0.009** |

Table 4. AUCs under six model modification attacks on the protected model with ViT-B/16 architecture.

| Dataset | Method | FT | PR | KD | AT | PFT | NFT |
|---|---|---|---|---|---|---|---|
| Tiny-ImageNet | IPGuard | 0.923 ± 0.006 | 0.692 ± 0.041 | 0.585 ± 0.153 | 0.606 ± 0.239 | 0.864 ± 0.036 | 0.901 ± 0.012 |
| | UAP | 0.979 ± 0.011 | 0.705 ± 0.030 | 0.593 ± 0.086 | 0.762 ± 0.071 | 0.731 ± 0.045 | 0.852 ± 0.022 |
| | ADV-TRA | 0.973 ± 0.022 | 0.765 ± 0.010 | 0.409 ± 0.023 | 0.901 ± 0.038 | 0.801 ± 0.008 | 0.915 ± 0.036 |
| | AKH | 0.938 ± 0.006 | 0.679 ± 0.007 | 0.539 ± 0.017 | 0.853 ± 0.014 | 0.738 ± 0.012 | 0.878 ± 0.010 |
| | **IrisFP** | **0.960 ± 0.007** | **0.829 ± 0.024** | **0.656 ± 0.045** | **0.939 ± 0.060** | **0.766 ± 0.114** | **0.948 ± 0.003** |

## 8.2. The impact of number of queries

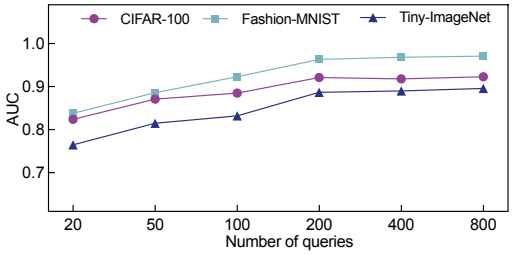


Figure 8. AUCs for different number of queries.

We evaluate how the total number of queries used for verification affects the overall performance. As shown in Figure 8, the AUC increases steadily as the number of queries grows from 20 to 800 across all settings, demonstrating that using more queries improves verification reliability. However, once the number of queries reaches around 200, the AUC values become largely stable, and further increasing the query count yields only marginal gains. For instance, MobileNet-V2 on Fashion-MNIST, the AUC rises sharply from 0.838 at 20 queries to 0.963 at 200, while the improvement beyond 200 (up to 800 queries) is less than 0.8%. This result suggests that a sufficient number of queries is essential for stable verification, but excessive queries provide diminishing returns. In practice, around 200 queries are sufficient to achieve near-optimal verification performance while maintaining computational efficiency.

## 8.3. Impact of $\tau$

We assess the impact of $\tau$ on verification performance. As shown in Figure 9, all three settings exhibit a similar trend: both small and large values of $\tau$ degrade AUC slightly, while a moderate one yields the best performance. This is because a small $\tau$ leads to fingerprints that are overly sensitive to decision boundary shifts, whereas a large one reduces the fingerprint's uniqueness. The optimal $\tau$ value depends on the total number of classes of each task: the larger the total number of classes, the smaller the value of $\tau$ should relatively be, and vice versa. The purpose of this

scaling is to ensure that the fingerprints maintain a certain distance from multiple decision boundaries. To improve the discriminative power, we need to optimize $\tau$ to find an intermediate value that achieves a balance between robustness and uniqueness.

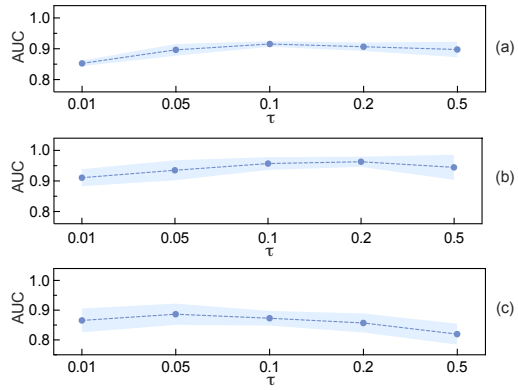


Figure 9. AUCs under different $\tau$ values for ResNet-18 on CIFAR-100 (a), MobileNet-V2 on Fashion-MNIST (b), and ViT-B/16 on Tiny-ImageNet (c).

### 8.4. Impact of $\alpha$

We study the impact of the verification threshold $\alpha$ on fingerprint verification performance across three different settings—ResNet-18 on CIFAR-100, MobileNet-V2 on Fashion-MNIST, and ViT-B/16 on Tiny-ImageNet. The evaluation metric is the overall accuracy defined by

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}, \tag{5}$$

where TP, FP, TN, and FN denote true positive rate, false positive rate, true negative rate, and false negative rate, respectively. As shown in Figure 10, the verification accuracy varies as $\alpha$ changes. For MobileNet-V2 on Fashion-MNIST, the accuracy increases steadily as the threshold moves from $\alpha = 0$ toward the mid-range and reaches a peak of approximately $0.937$ when $\alpha$ is in the range 0.45–0.50, and then gradually declines as $\alpha$ becomes too large. Likewise, ResNet-18 and ViT-B/16 follow a similar trend: accuracy improves rapidly when $\alpha < 0.20$, remains near its optimum for $\alpha$ between $0.45$ and $0.50$, and deteriorates once $\alpha$ exceeds this range.

Overall, these results demonstrate that moderate threshold values (around $0.45$–$0.50$) provide the best verification performance. Extremely low thresholds or extremely high thresholds lead to suboptimal accuracy, highlighting the importance of selecting an appropriate decision threshold for stable and reliable fingerprint verification.

### 8.5. Impact of reference model set size

We study the impact of reference model set size, i.e., the number of models in each of the reference model sets $\mathcal{V}_f$ and $\mathcal{I}_f$, on the verification performance. As shown in Figure 11, enlarging these sets consistently improves AUC across all settings—ResNet-18 on CIFAR-100, MobileNet-V2 on Fashion-MNIST, and ViT-B/16 on Tiny-ImageNet. For instance, for ResNet-18 on CIFAR-100, the AUC increases from 0.792 with one model per set to 0.918 with 12 models per set; for MobileNet-V2 on Fashion-MNIST, it increases from 0.830 to 0.969. Particularly, the most substantial performance gains occur when increasing the set size from one to six, beyond which the performance is getting stable. These results indicate that our method can converge and achieve high stability of the verification performance with only moderately sized reference sets, underscoring its scalability and efficiency.

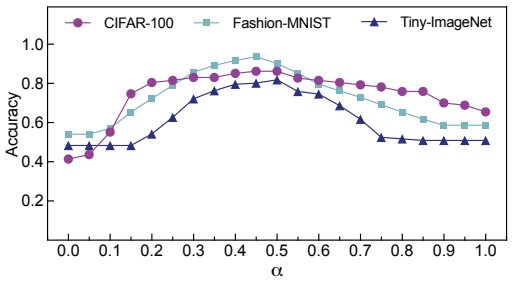


Figure 10. Accuracy for different $\alpha$.

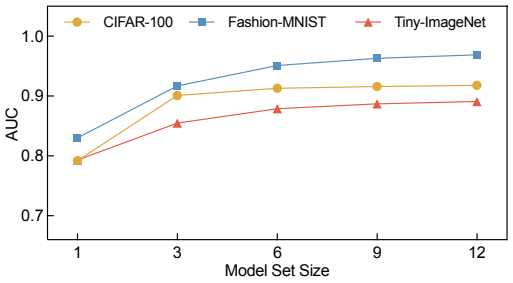


Figure 11. AUCs for different model set sizes.

## 9. Time Overhead

IrisFP's time cost during the fingerprint generation process comes from two parts: (1) training the reference model set, and (2) generating the fingerprints. We compare the time cost of using different model fingerprinting methods. For each method, we generate 200 fingerprints in total. It is noteworthy that IrisFP adopts 40 composite-sample fingerprints, each being composed of 5 samples, ensuring a fair assessment across different methods. Besides, we consider three protected models: a ResNet-18 trained on CIFAR-100, a MobileNet-V2 trained on Fashion-MNIST, and a ViT-B/16 trained on Tiny-ImageNet.

The experimental results are summarized in Table 5. As shown by the table, IrisFP requires 55 $m$ for MobileNet-

V2 on Fashion-MNIST, which is an acceptable cost among different methods. More importantly, even when switching to a substantially larger model ViT, the total cost only increases to 1 $h$ 32 $m$, demonstrating that the computational overhead of IrisFP scales modestly with the model size. This level of computational cost is practical and acceptable in real-world deployments. It is worth noting that for the time of performing ownership verification, we ignore the discussion. This is because querying a target model is nearly real-time, and the methods with the same total number of fingerprints incur same time cost.

Table 5. Time overheads of generating fingerprints across different methods.

| Method | IPGuard | UAP | ADV-TRA | AKH | IrisFP |
|---|---|---|---|---|---|
| **ResNet-18** | $59s$ | $4h40m$ | $32m$ | $11s$ | $1h2m$ |
| **MobileNet-V2** | $41s$ | $3h\ 5m$ | $24m$ | $8s$ | $55m$ |
| **ViT** | $1m21s$ | $7h30m$ | $1h11m$ | $1m6s$ | $1h32m$ |

## 10. Stealthiness

Prior fingerprinting studies rarely report "stealthiness/imperceptibility", as enforcing extremely small visual perturbations often weakens the discriminative strength of the fingerprints. Nevertheless, we add a qualitative evaluation of stealthiness in Fig. 12. As shown in the figure, the generated fingerprints remain visually similar to the original inputs, with only subtle pixel-level differences.

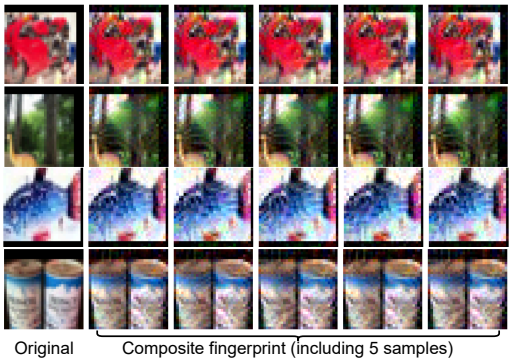


Figure 12. Qualitative evaluation of stealthiness. Each row shows an original sample and its corresponding composite fingerprint consists of five samples.

## 11. Task Transferability

IrisFP targets the standard ownership-verification setting considered in prior work, where the target model is deployed for the same task and shares the same label set as the protected model, even after model-modification attacks. If the task changes and the label space differs, verification can still be conducted using the overlapping subset of labels, since our fingerprint set spans multiple labels. However, the verification performance is expected to degrade as the degree of label overlap decreases.

## 12. Experimental Setting Details

### 12.1. Protected Model

We consider three model architectures for the protected model: ResNet-18, MobileNet-V2, and ViT-B/16. For each of the five benchmark datasets (CIFAR-10, CIFAR-100, Fashion-MNIST, MNIST, and Tiny-ImageNet), we independently train two protected models: one using ResNet-18 and the other using MobileNet-V2. Besides, we also produce a protected model with the ViT-B/16 architecture, a larger and more complex architecture, by finetuning it on Tiny-ImageNet. All ResNet-18 and MobileNet-V2 models follow standard training configurations—SGD with momentum 0.9, weight decay of $5 \times 10^{-4}$, an initial learning rate of 0.1, cosine-annealing scheduling, Xavier initialization, and a batch size of 128 for 600 epochs. For ViT-B/16, which is generally more suitable for higher-resolution data, we initialize the model from pretrained weights and finetune it on Tiny-ImageNet under the same settings, except that we use a smaller initial learning rate of 0.01 and train it for 50 epochs.

### 12.2. Reference Model Sets

Two reference model sets, including the pirated model set and the independently-trained set, serve as the foundation for experimentally assessing the quality of fingerprints during the refinement process. For each protected model, its corresponding pirated model set is constructed by modifying the protected model through three model modification techniques, chosen from six available attack types (FT, PR, KD, AT, PFT, and NFT). These modifications invalidate the original fingerprints, resulting in evading ownership verification. We adopt only three of them to balance computational cost and to emulate realistic conditions where reference models may not encompass all possible removal attempts. In our experiment, we select fine-tuning (FT), knowledge distillation (KD), and adversarial training (AT). Specifically:

- Fine-tuning (FT): Training the protected model for a specified number of epochs.
- Knowledge Distillation (KD): Training a student model—either with the same or a different architecture—under the supervision of the protected (teacher) model by minimizing the KL divergence between their soft outputs, with temperature set to 1.
- Adversarial Training (AT): Crafting adversarial examples via a PGD attack (using Foolbox's "LinfPGD" tool), concatenating them with clean inputs, and then training the

model on the mixed dataset.

For each attack type, we produce three pirated variants instantiated with 3 different random seeds to ensure model, leading to 9 pirated variants in the pirated model set. On the other hand, the easy way for constructing the independent model set is to use public models from an open-source platform, e.g., Hugging Face. In our experiments, the model is trained from scratch with different seeds and hyperparameters from training the protected models, following standard training configurations—SGD with momentum 0.9, weight decay of $5 \times 10^{-4}$, an initial learning rate of 0.1, cosine-annealing scheduling, Xavier initialization, and a batch size of 128 for 600 epochs. Specifically, our independent models are constructed by using three diverse architectures available in the torchvision library—ResNet-18, MobileNet-V2, and DenseNet-121—each type instantiated with 3 different random seeds to create 3 independently-trained models, resulting in a total of 9 models in the independent model set. To guarantee independence, all ResNet-18 and MobileNet-V2 instances are initialized with seeds distinct from that of the protected model.

### 12.3. Testing Model Set

The testing model set consists of pirated and independently-trained models, which are independently trained and have no overlap with the models in the reference model sets.

The pirated models in the testing set are constructed using six representative attacks:

- Fine-tuning (FT): Training the protected model for a specified number of epochs.
- Pruning (PR): Applying unstructured global pruning with sparsity levels ranging from 10% to 90% in increments of 10%, without retraining.
- Prune-then-tune (PFT): Applying pruning at sparsity levels of 30%, 60%, and 90%, followed by fine-tuning.
- Noise-regularized Fine-tuning (NFT): Perturbing each trainable parameter tensor with Gaussian noise scaled by a scaled magnitude of standard deviation of that layer, i.e., $param+ = \alpha \cdot std(param) \cdot \mathcal{N}(0, 1)$, followed by fine-tuning.
- Knowledge Distillation (KD): Training a student model—of the same or a different architecture—using the KL divergence between its outputs and those of the protected model (temperature = 1).
- Adversarial Training (AT): Crafting adversarial examples via a PGD attack (using Foolbox's "LinfPGD" tool), concatenating them with clean inputs, and then training the model on the mixed dataset.

For each attack type, we generate 20 variants with different random seeds, resulting in a total 120 pirated models.

The independently-trained models in the testing set are built from scratch using six architectures: ResNet-18, ResNet-50, MobileNet-V2, MobileNet-V3 Large, EfficientNet-B2, and DenseNet-121. For each architecture, 20 models are trained with distinct seeds, without any access to the protected models or the models in the reference model set, yielding 120 independently-trained models.

### 12.4. Hyperparameters

We use $K = 40$ composite-sample fingerprints, each consisting of $T = 5$ samples for all tasks. The bias parameter $\tau$ is 0.2 for CIFAR-10, Fashion-MNIST, and MNIST, 0.1 for CIFAR-100, and 0.05 for Tiny-ImageNet. We set the regularization coefficients $\lambda_1$ and $\lambda_2$ to 0.05. The training process consists of 800 iterations in Phase I and 200 iterations in Phase II.

### 12.5. Computing Infrastructure

All experiments were run on a Linux server equipped with an NVIDIA A100 GPU and 200 GB of system memory. GPU acceleration was provided by CUDA 12.1, and all models and training pipelines were implemented in PyTorch 2.5.1.

## 13. Related Work

Model fingerprinting has emerged as a promising non-intrusive technique for protecting the intellectual property (IP) of deep neural networks (DNNs), which are highly susceptible to unauthorized use [3, 7, 21, 26, 31, 35, 36, 38]. By exploiting the distinctive behavioral patterns of a protected model, fingerprinting derives verifiable ownership evidence[5, 37].

Researchers have developed various model fingerprinting methods. For example, several fingerprinting methods assess model similarity through internal representations or model parameters instead of adversarial queries. Maho *et al*. [22] propose a greedy scheme that generates fingerprints for a model by analyzing the statistical similarity of intermediate activations from benign inputs using Shannon's information theory. Guan *et al*. [7] identify suspicious models by selecting inputs that yield inconsistent predictions across two sets of reference models and examining their pairwise relationships. Other methods treat the model parameters themselves as intrinsic fingerprints. For instance, Jia *et al*. [10] train linear proxy models on a reference dataset and compare their learned weights via cosine similarity, while Zheng *et al*. [39] project front-layer weights into a random subspace associated with the model owner's identity to achieve non-repudiable ownership verification.

In the recent years, researchers have gained great attention to the adversarial-example-based fingerprinting and have developed many methods for it. This type of fingerprinting leverages adversarial examples to craft input–output pairs that exhibit model-specific behavioral patterns. For instance, Cao *et al*. [3] reveal that a model's decision boundary reflects its unique identity and generates

data points (adversarial samples) near a decision boundary to serve as distinctive fingerprints. Lukas *et al*. [21] optimizes adversarial examples through multi-model training to maximize their transferability to pirated models. However, such transferability often leads to false positives. To address this, Peng *et al*. [23] employ Universal Adversarial Perturbations to modify clean inputs into fingerprint samples, and then use a learned encoder to map model logits into embedding vectors for similarity-based ownership verification. Yang *et al*. [34] train a GAN to synthesize natural fingerprint samples by optimizing inputs that yield divergent predictions between pirated and independent models in decision-difference regions, which are then used as black-box queries for verification. Liu *et al*. [20] leverage a conditional GAN with margin loss to generate samples positioned at controlled distances from the classification boundary, enabling robust and distinctive fingerprinting without relying on surrogate models. Instead of isolated adversarial samples, Xu *et al*. [29] construct adversarial trajectories by iteratively perturbing clean inputs to traverse decision-boundary regions, capturing richer and more comprehensive model-specific behaviors. Godinot *et al*. [6] introduce a decomposition-based analysis framework that separates fingerprinting into query construction, representation, and detection, and show that one effective instantiation constructs fingerprints using adversarial examples originating from naturally misclassified samples.

Despite these advancements, a fundamental challenge remains: designing fingerprints that simultaneously achieve uniqueness—distinguishing the protected model from independently trained ones—and robustness—maintaining validity under model modifications or removal attacks. Different from conventional approaches that typically construct each fingerprint near a single decision boundary, we investigate the region near the intersection of multiple decision boundaries and propose a composite-sample fingerprinting framework that jointly enhances both uniqueness and robustness.