THE IMPERFECT MIRROR: NON-INTRUSIVE MODEL AUDITING VIA INFORMATION-BOTTLENECK DETERIORATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing non-intrusive model auditing methods mainly rely on fingerprinting, which identifies stolen models through behavioral similarity preserved from the victim model. However, this similarity is fragile under practical stealing and evasion, such as cross-architecture extraction and adversarial training. Given these limitations, we propose a novel non-intrusive framework that shifts auditing from preserved similarity to structured information deterioration. Our key insight is that independently trained models tend to exhibit a characteristic information-bottleneck balance, reflecting how they compress input information while preserving label-relevant information. In contrast, model stealing can disrupt this learned balance, leading the stolen model to retain more input information while preserving less label-relevant information than the victim, which provides a principled auditing signal. We formalize this insight into a two-stage auditing procedure that first screens for this directional deterioration and then statistically assesses whether the suspect's information-bottleneck behavior is consistent with that of independently trained models, yielding a quantitative auditing score. Experiments demonstrate effective auditing with low false-positive rates and improved robustness over the evaluated baselines under cross-architecture stealing and adversarial-training evasion.

1 INTRODUCTION

Training a deep learning model requires substantial investments in data collection, computational resources, and optimization. However, an adversary can reuse a trained model's capabilities without repeating the original training process, either through parameter-based reuse such as fine-tuning and pruning (Maini et al., 2021), or through black-box extraction and distillation (Tramèr et al., 2016; Orekondy et al., 2019; Hinton et al., 2015). These threats motivate *model auditing*, which provides evidence of whether a suspect model was derived from a protected victim model.

Existing model-auditing approaches mainly rely on watermarking or fingerprinting. Watermarking methods embed ownership signals into the protected model, but require modifying the model and may introduce utility trade-offs or security risks (Uchida et al., 2017; Adi et al., 2018; Jia et al., 2021). Fingerprinting avoids such modification by identifying naturally occurring model characteristics, including decision-boundary responses (Cao et al., 2021; Xu et al., 2024), training-data-specific boundary patterns (Maini et al., 2021), and similarities in model outputs or internal representations (Chen et al., 2022). However, these methods still rely on behavioral similarity being preserved from the victim to the stolen model, which can become unreliable under practical stealing and evasion.

In this paper, we propose a non-intrusive framework for model auditing that shifts the focus from what model stealing preserves to what it changes. Our motivation is that model stealing can be imperfect: a stolen model may retain useful predictive functionality without preserving the victim's learned information processing. Specifically, we examine whether stealing induces structured changes in how the model compresses input information and retains label-relevant information on the victim's private training data, as characterized by the information-bottleneck principle (Tishby et al., 2000; Shwartz-Ziv & Tishby, 2017). The left part of Figure 1 illustrates our motivation. Turning this perspective into a practical auditing framework requires addressing two connected challenges: identifying a theoretically

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

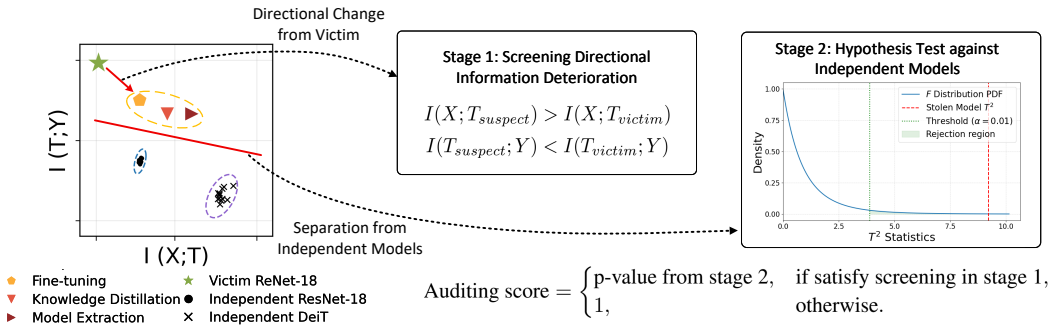


Figure 1: **Left**: The information plane reveals two characteristic patterns of model stealing. First, stolen models produced by different stealing attacks consistently shift toward higher $I(X;T)$ and lower $I(T;Y)$, indicating weaker input compression and reduced label-information retention. Second, these models are separated from the natural variation of independently trained models. **Middle**: Stage 1 screens for this directional information deterioration relative to the victim. **Right**: Stage 2 compares suspects against independent-model references using a predictive Hotelling's T^2 test and converts the departure into a p-value. The final auditing score uses this p-value only when Stage 1 is satisfied; thus, a small score requires both directional deterioration from the victim and separation from independent-model variation.

meaningful pattern of information change associated with model stealing, and distinguishing this pattern from the natural variation among independent models.

To address these challenges, our method consists of two stages: *directional deterioration screening* and *testing against independent models*. First, we establish sufficient conditions under which a stolen model retains more input information while preserving less label-relevant information than the victim. We then compare the suspect with the victim using these two information quantities to screen for this directional deterioration. Second, because independently trained models may also exhibit the same direction of change, we construct independently trained reference models and evaluate their information quantities. For suspects that pass the deterioration screen, we use Hotelling's T^2 test to determine whether their information pattern is atypical of independently trained models. Combining these two steps yields a quantitative auditing score, where stronger evidence requires both directional deterioration relative to the victim and departure from independent models.

Extensive experiments are conducted to evaluate our auditing framework across different dataset-architecture settings under multiple stealing strategies, including fine-tuning, pruning, knowledge distillation, and model extraction. The results demonstrate effective auditing with low false-positive rates. More importantly, our method remains robust under cross-architecture stealing and adversarial-training evasion, two settings that substantially weaken existing fingerprinting methods.

2 Related Work

Watermarking-based Auditing. Watermarking-based methods verify model ownership by injecting specific watermarks into the protected model and detecting their presence in a suspect model. Early approaches embed watermarks in model parameters (Uchida et al., 2017), encode them in hidden-layer activation distributions (Darvish Rouhani et al., 2019), or introduce backdoors for black-box verification (Adi et al., 2018; Zhang et al., 2018). Subsequent methods introduce passport layers to encode ownership signals (Fan et al., 2019) or entangle the watermark with task-relevant representations (Jia et al., 2021) to improve watermark retention under evasion attacks. More recently, EaaW (Shao et al., 2025) encodes watermarks in feature-attribution explanations, mitigating the risks of backdoor-induced misclassification. However, despite their effectiveness in auditing, these methods remain intrusive, inevitably introducing utility trade-offs or security risks.

Fingerprinting-based Auditing. Fingerprinting-based methods extract intrinsic characteristics from a trained model as fingerprints and examine whether similar patterns are preserved in a suspect model. A prominent line of work focuses on decision-boundary characteristics. IPGuard (Cao et al., 2021) constructs adversarial inputs near the victim's decision boundary and measures prediction

agreement, while ADV-TRA (Xu et al., 2024) extends such probes to adversarial trajectories spanning multiple classes. Dataset Inference (Maini et al., 2021) instead uses the decision-boundary distance of training samples, and DeepJudge (Chen et al., 2022) measures similarity at multiple levels, including output behavior, neuron- and layer-level activations. Recent work further extends fingerprinting to language models through prompt-injection behaviors (Hu et al., 2026).

Despite differences in fingerprint construction, these methods rely on the preservation of such patterns in stolen models, which can become unreliable under practical stealing and evasion. Model extraction and distillation can transfer useful functionality without preserving the internal representations or boundary responses used as fingerprints (Peng et al., 2022). Adversarial training can likewise reshape decision boundaries, as Szyller et al. (2023) report false negatives in Dataset Inference and Xu et al. (2024) observe substantial reductions in IPGuard's fingerprint detection rate. Moreover, adaptive attacks can explicitly suppress such fingerprints without eliminating the stolen model's utility (Yao et al., 2023). These limitations motivate our shift from preserved similarity toward structured change.

3 Problem Statement

3.1 Threat Model

We consider a model auditing scenario in which a *Model Owner* seeks to assess evidence that a *Suspect Model* was derived from its protected model. The owner holds a protected classification model, referred to as the *victim model* $f_v(\cdot)$, trained on a private labeled dataset $D_v = \{(x_i, y_i)\}_{i=1}^n$, which remains unavailable to the adversary during stealing and evasion. The owner suspects that a *suspect model* $f_s(\cdot)$ has used $f_v(\cdot)$ to develop its functionality for the same classification task. We consider black-box auditing, where the owner can query f_s with D_v to obtain predicted probability vectors but cannot inspect its parameters, intermediate activations, or training configurations. Formally, the model auditing objective is defined as the function

$$\mathcal{F}(f_v, A(f_s), D_v) \in \mathbb{R}, \tag{1}$$

where $A(f_s)$ represents f_s's predicted probability vectors. It returns a quantitative score supporting the owner's assessment.

We define a *stolen model* as a model developed by reusing the victim model's trained parameters or using its predictions as supervision. With white-box access to f_v, stealing may involve fine-tuning or pruning, potentially followed by further fine-tuning. With only black-box access, model extraction or distillation can train f_s using victim predictions as supervision, potentially with an architecture different from that of the victim. Adversarial training may additionally be applied to the stolen model to evade auditing. An *independent model*, in contrast, is trained without inheriting the victim's parameters or using its predictions as supervision.

3.2 Challenges and Key Intuitions

Reliability. The primary challenge is to obtain evidence when similarity between the victim and a stolen model becomes unreliable. Although more tolerant fingerprints can accommodate larger behavioral changes (Xu et al., 2024), their preservation is not necessary for retaining useful model functionality, particularly under extraction and distillation. We therefore shift auditing from what stealing preserves to what it changes. Our key intuition is that independently trained models tend to develop a balance between compressing input information and preserving label-relevant information (Shwartz-Ziv & Tishby, 2017), whereas model stealing disrupts this learned balance. We investigate such structured information-bottleneck changes as evidence of model derivation.

Attributability. An arbitrary information discrepancy, however, does not necessarily indicate model stealing, since independently trained models can naturally differ from the victim due to architecture, training data, and initialization. The second challenge is therefore to distinguish stealing-associated changes from natural variation. Our key intuition is that model stealing can induce a directional deterioration tied to the victim's learned information processing, whose pattern also differs from that of independently trained models even when they share the stolen model's architecture and training data. Distinguishing these patterns helps assess whether the observed deterioration is attributable to model stealing rather than independent training.

4 OUR METHOD

4.1 PRELIMINARIES: INFORMATION BOTTLENECK

Let X denote an input, Y its ground-truth label, and $T = f(X;\theta)$ the learned representation produced by a neural network with parameters θ. The mutual information $I(X;T)$ quantifies how much information about the input is retained in T, while $I(T;Y)$ measures how much label-relevant information it preserves. The *information bottleneck* (IB) principle (Tishby et al., 2000) seeks a representation that preserves label-relevant information while discarding unnecessary input details, expressed as

$$\mathcal{J}_\beta = I(X;T) - \beta I(T;Y), \qquad \beta > 0, \tag{2}$$

where β controls the relative importance of label-information retention. The *information plane* (Shwartz-Ziv & Tishby, 2017) further visualizes how this balance is established during training by tracing the coordinates $\big(I(X;T), I(T;Y)\big)$. Under this view, independently trained models tend to develop a characteristic information-bottleneck balance on their training data. However, our auditing method neither treats this pattern as a fingerprint nor assumes that a stolen model preserves a similar information-bottleneck balance. Instead, we analyze how model stealing can induce a directional deterioration in this balance on the victim's training data. In our black-box setting, T is instantiated from the model's prediction vector at a fixed finite resolution, as detailed in Appendix B.1.

4.2 INFORMATION DETERIORATION FOR MODEL AUDITING

Modeling the stolen model. Let $T_v = f_v(X)$ and $T_s = f_s(X)$ denote the learned representations of the victim and a model derived from it, respectively. All information quantities below are evaluated on the same inputs and ground-truth labels from D_v. We statistically describe their relationship as

$$T_s = g\big(T_v, R(X)\big), \tag{3}$$

where $R(X)$ captures input-dependent variation introduced or re-exposed by model stealing, and g describes its effect on the resulting representation. Model stealing may introduce input distinctions that the victim's representation does not retain, while removing information that it previously encoded. The following decomposition separates these information gains and losses.

Proposition 4.1 (Information-transfer decomposition)**.** *For representations T_v and T_s evaluated under the same joint distribution of (X, Y), with all information quantities below finite,*

$$I(X;T_s) - I(X;T_v) = I(X;T_s \mid T_v) - I(X;T_v \mid T_s), \tag{4}$$

$$I(T_s;Y) - I(T_v;Y) = I(T_s;Y \mid T_v) - I(T_v;Y \mid T_s). \tag{5}$$

The proof follows from the mutual information chain rule and is provided in Appendix A.1. Directional deterioration occurs when additional input retention outweighs input-information loss, while label-information loss outweighs any gain. Appendix A.6 further examines these gain-loss terms empirically across stealing methods.

Information-bottleneck deterioration. We then apply Proposition 4.1 to identify sufficient conditions for joint deterioration in input compression and label-information retention on D_v. We first consider label-information retention. When the residual variation provides no additional label information beyond T_v, transfer cannot compensate for label information lost by mixing victim representations with different conditional label distributions, therefore yielding the following result.

Theorem 4.2 (Label-information deterioration)**.** *For the representation-transfer model in equation 3, under Assumptions A.1 and A.3 in Appendix A.2,*

$$I(T_s;Y) < I(T_v;Y). \tag{6}$$

The proof is given in Appendix A.3, which focuses on demonstrating $I(T_s;Y \mid T_v) = 0$ and $I(T_v;Y \mid T_s) > 0$. Thus, in the label-information balance equation 5, the gain term vanishes while the loss term is strictly positive, establishing a loss of label information on D_v.

Reduced label information alone does not imply weaker input compression. We therefore examine representation fragmentation, which introduces distinctions between inputs that share the same victim representation. The structural condition in Assumption A.2 ensures that the resulting input-information gain outweighs information lost elsewhere during transfer.

Theorem 4.3 (Input-compression deterioration)**.** *Suppose T_v and T_s are deterministic, finite-valued representations of X. Under Assumption A.2 in Appendix A.2,*

$$I(X;T_s) > I(X;T_v). \tag{7}$$

The proof is given in Appendix A.4. Under this condition, $I(X;T_s \mid T_v)$ exceeds $I(X;T_v \mid T_s)$ in the input-information balance equation 4. The stolen representation thus retains more input information than the victim representation, indicating weaker input compression.

Together, Theorems 4.2 and 4.3 establish a rightward and downward displacement in the information plane that characterizes the information bottleneck deterioration, enabling us to identify the stolen model through directional deterioration relative to the victim.

4.3 Two-Stage Auditing and Score Calculation

Although model stealing can induce directional deterioration in information-bottleneck balance, independently trained models may exhibit the same direction of change. To address this ambiguity, we propose a two-stage auditing framework.

4.3.1 Stage 1: Screening Directional Deterioration

We query the victim and suspect on the same private training dataset D_v and compute their information-plane coordinates. Formally, for a model f with representation T_f, define

$$\mathbf{m}(f) = \left(\widehat{I}_{D_v}(X;T_f) \quad \widehat{I}_{D_v}(T_f;Y)\right)^\top \in \mathbb{R}^d, \qquad d = 2, \tag{8}$$

where the hats denote empirical information estimates computed using a common procedure, detailed in Appendix B.1. Let $\mathbf{m}_v = \mathbf{m}(f_v)$ and $\mathbf{m}_s = \mathbf{m}(f_s)$ denote the victim's and suspect's information vectors, respectively. We define the directional gate

$$G_s = \mathbf{1}\{m_{s,1} > m_{v,1} \text{ and } m_{s,2} < m_{v,2}\}. \tag{9}$$

A suspect with $G_s = 1$ exhibits weaker input compression and lower label-information retention relative to the victim. It therefore satisfies the deterioration pattern and proceeds to the second stage for statistical comparison with independently trained models.

4.3.2 Stage 2: Hypothesis Test against Independent Models

We construct independent-model references to assess whether the suspect's information-bottleneck behavior can be explained by independent training. When the suspect's architecture is available, we use architecture-matched models; otherwise, separate reference pools can be constructed for multiple candidate architectures. For a reference pool $\mathcal{N} = \{f_-^{(1)}, \ldots, f_-^{(k)}\}$, each model is trained independently of the victim on data excluding D_v. We evaluate every negative model on D_v to obtain its information vector $\mathbf{m}_-^{(j)} = \mathbf{m}(f_-^{(j)})$.

We formulate the comparison with the reference pool as a hypothesis test. Let P_- denote the distribution of information vectors induced by independent models. The null hypothesis is

$$H_0^{\text{ind}}: \quad \mathbf{m}_s, \mathbf{m}_-^{(1)}, \ldots, \mathbf{m}_-^{(k)} \overset{\text{iid}}{\sim} P_-. \tag{10}$$

Under this null, the suspect's information vector follows the same distribution as those of the independently trained reference models. Let $\widehat{\boldsymbol{\mu}}_-$ and $\widehat{\Sigma}_-$ denote the sample mean and covariance estimated from the reference pool. To measure departure from the reference jointly in both information dimensions, we use the predictive form of Hotelling's T^2 statistic (Tracy et al., 1992):

$$T^2_{\text{audit}} = \frac{k}{k+1}(\mathbf{m}_s - \widehat{\boldsymbol{\mu}}_-)^\top \widehat{\Sigma}_-^{-1}(\mathbf{m}_s - \widehat{\boldsymbol{\mu}}_-). \tag{11}$$

This statistic measures the suspect's distance from the reference while accounting for the scales and correlation of the two information quantities. The following theorem provides its null distribution.

Theorem 4.4. *Under H_0^{ind}, suppose $P_- = \mathcal{N}_d(\boldsymbol{\mu}_-, \Sigma_-)$ with $\Sigma_- \succ 0$ and $k > d$. Then*

$$\frac{k-d}{d(k-1)} T^2_{\text{audit}} \sim F_{d,k-d}. \tag{12}$$

The proof is provided in Appendix B.3. We further exhibit the Q-Q plots on held-out independent models in Figures 6, 7, and 8 in Appendix B.3, which show close agreement with the reference F distribution, supporting the empirical calibration of the resulting p-values.

Let $F_{d,k-d}(\cdot)$ denote the cumulative distribution function of the corresponding F distribution. The upper-tail p-value is

$$p_s = 1 - F_{d,k-d}\big((k-d)/[d(k-1)] \cdot T^2_{\text{audit}}\big)\,. \tag{13}$$

A smaller p_s indicates stronger statistical evidence against the independent-training explanation.

For an unknown suspect architecture, let $\mathcal{A}$ denote a prespecified set of candidate architectures and $p_{s,a}$ the value obtained from the reference pool for architecture a. We use $p_s = \max_{a\in\mathcal{A}} p_{s,a}$, as detailed in Appendix B.4. Thus, a small combined value requires the suspect's information-bottleneck behavior to be atypical of every candidate reference considered.

4.3.3 Auditing Score

We combine the outputs of the two stages into a single auditing score. Let $\mathfrak{N}$ denote the collection of reference pools and p_s the resulting second-stage p-value. The auditing function is refined as

$$\mathcal{F}(f_v, A(f_s), D_v; \mathfrak{N}) = \begin{cases} p_s, & G_s = 1, \\ 1, & G_s = 0. \end{cases} \tag{14}$$

By construction, a small score requires the suspect to exhibit both the stealing-associated deterioration relative to the victim and an information pattern atypical of the independent-training references. We therefore interpret smaller auditing scores as stronger evidence supporting the hypothesis that the suspect was derived from the victim. We further demonstrate that combining these two requirements preserves the calibration of a valid second-stage p-value, with details shown in Appendix B.5.

5 Evaluations

5.1 Experimental Settings

We conduct experiments on ResNet, VGG, and DeiT-Tiny on the CIFAR datasets, and Swin-T on ImageNet-100. For each dataset, we divide the samples into two class-balanced halves. One half serves as the owner's private dataset and is used to train the victim model under each dataset-architecture setting. The other half serves as non-private training data to train 50 independent models as negative suspects under each dataset-architecture setting. In addition, we construct separate reference pools of 30 independently trained models with different random initializations and selected training configurations to introduce diversity into the independent-model reference distribution.

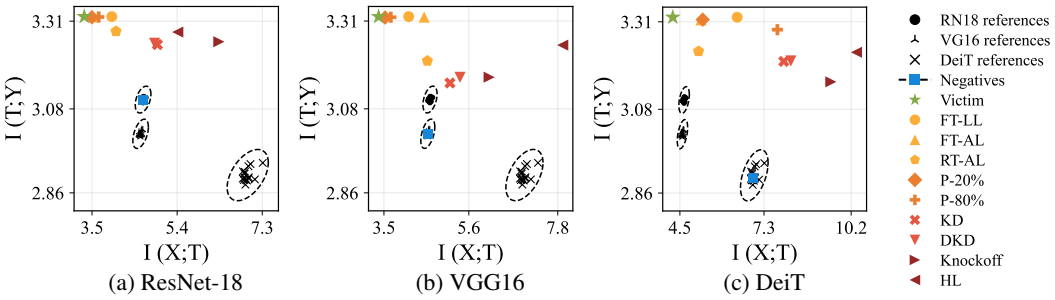
For each stealing configuration, we construct 50 positive suspects derived from the victim through model stealing methods. White-box stealing includes fine-tuning with LL, AL, and RT-AL, as well as pruning with different pruning ratios. Knowledge-transfer attacks include standard knowledge distillation (KD), decoupled knowledge distillation (DKD), Knockoff, and HL (data-free model stealing under hard-label access). We report the detection rate for two groups, defined as the fraction of suspects whose model type is correctly identified. Detailed settings are shown in Appendix C.

5.2 Overall Performance

We evaluate our framework under the unknown-architecture setting, where the auditor does not know the suspect model's architecture. Multiple reference pools with different model architectures are employed in the second stage for testing against independent models. Each reference pool characterizes the independent-model distribution under a different candidate architecture, allowing the auditor to account for architecture uncertainty without assuming prior knowledge of the suspect architecture. Table 1 summarizes the results on CIFAR-10 and ImageNet-100, with more results shown in Appendix D.1. Each entry reports the fraction of 50 suspects correctly classified at $\alpha = 0.01$, with the mean auditing score in parentheses. We can observe that our framework correctly detects all 50 stolen suspects across all evaluated stealing attacks for the four model architectures. Here, a 100% detection rate for both positive and negative suspects indicates that all stolen models are detected and all independently trained models are correctly identified, corresponding to $\mathrm{FN} = 0$ and

Table 1: The detection rate (%) for positive and negative suspects on CIFAR-10 and ImageNet-100

Model Type		RN18-CF10	VG16-CF10	DeiT-CF10	Swin-IN100
Positive Suspects	FT-LL	100 (3.93×10^{-15})	100 (5.83×10^{-15})	100 (1.74×10^{-12})	100 (1.52×10^{-19})
	FT-AL	100 (1.63×10^{-15})	100 (1.51×10^{-14})	100 (1.18×10^{-13})	100 (1.28×10^{-20})
	RT-AL	100 (8.81×10^{-15})	100 (2.43×10^{-10})	100 (4.26×10^{-11})	100 (2.82×10^{-17})
	P-20%	100 (1.54×10^{-15})	100 (1.67×10^{-15})	100 (1.24×10^{-13})	100 (1.20×10^{-20})
	P-60%	100 (2.11×10^{-15})	100 (2.24×10^{-15})	100 (7.59×10^{-11})	100 (3.55×10^{-13})
	KD	100 (1.83×10^{-11})	100 (1.94×10^{-9})	100 (2.24×10^{-9})	100 (1.09×10^{-16})
	DKD	100 (1.26×10^{-11})	100 (1.15×10^{-11})	100 (1.91×10^{-9})	100 (1.16×10^{-17})
	Knockoff	100 (9.50×10^{-12})	100 (9.26×10^{-11})	100 (1.13×10^{-10})	100 (3.51×10^{-11})
	HL	100 (3.29×10^{-13})	100 (4.25×10^{-10})	100 (2.95×10^{-12})	100 (4.60×10^{-12})
Negative Suspects		100 (0.60)	100 (0.53)	100 (0.59)	100 (0.52)



(a) ResNet-18 (b) VGG16 (c) DeiT

Figure 2: Information-plane visualization under unknown suspect architecture on CIFAR-10.

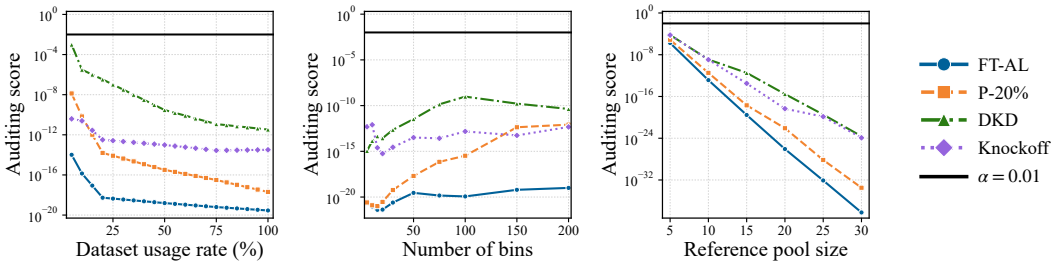
FP $= 0$. The reported auditing scores for stolen suspects range from 1.20×10^{-20} to 2.24×10^{-9}. Because the unknown-architecture score takes the maximum over candidate reference pools, these small values indicate that the stolen suspects are atypical of every candidate independent-model population considered. Meanwhile, all negative suspects are also correctly identified, with large scores between 0.52 and 0.60. These results demonstrate effective detection of the evaluated stolen models without incurring false positives, despite the absence of architecture knowledge.

Moreover, Figure 2 visualizes different models on the information plane on CIFAR-10, with more results shown in Appendix D.1. We can observe that positive suspects lie to the lower right of their corresponding victim, exhibiting higher $(I(X;T))$ and lower $(I(T;Y))$, consistent with the proposed information-bottleneck deterioration pattern. Importantly, the positive groups also remain well separated from the independent-model reference regions of all three architectures, whereas negative suspects align with their corresponding references. These observations explain the strong auditing performance under unknown-architecture and support our motivation for using structured information deterioration as auditing evidence.

We further examine the sensitivity of our auditing score on the stolen model to the number of query data, bins used for information estimation, and the independent models in the reference pool. Here, we take the DeiT trained on CIFAR-100 as an example. As shown in Figure 3, increasing the fraction of D_v used for auditing generally leads to smaller scores, indicating stronger auditing evidence with more query samples. Varying the number of bins changes the numerical values of the scores, but all evaluated stealing methods remain consistently below the significance level $\alpha = 0.01$ across a wide range of discretization resolutions. Increasing the independent model size also substantially reduces the auditing scores, as more independent models provide a better-estimated statistical reference. More results under different settings are shown in Appendix D.1. Overall, the auditing conclusions remain stable across these configurations.

5.3 Comparison Results

We compare our method with four representative non-intrusive fingerprinting baselines, including IPGuard, DeepJudge, ADV-TRA, and Dataset Inference (DI) under different practical stealing methods. Detailed implementation settings of baselines are provided in Appendix C.7. Moreover,



(a) Dataset usage rate (b) Number of bins (c) Reference pool size

Figure 3: Effects of the number of query data, bins used for information estimation, and the independent models on the auditing score on CIFAR-100 with DeiT.

Table 2: Cross-architecture detection rate (%) under extraction and distillation. Each entry reports **Knockoff / HL** and **KD / DKD**, respectively

	Extraction				Distillation	
Victim	**RN18-CF10**		**DeiT-CF100**		**DeiT-CF10**	**VG19-CF100**
Suspect	**VGG16**	**DeiT**	**RN18**	**VGG16**	**DeiT**	**VG16**
IPGuard	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DeepJudge	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DI	74 / 76	70 / 72	69 / 64	67 / 68	83 / 80	78 / 78
ADV-TRA	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
Ours	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**

Table 3: Detection rate (%) under adversarial evasion on RN18-CF10 with perturbation budgets $\epsilon = 4/255$. Each entry reports the rate under post-stealing and stealing-time adversarial training

	FT-AL	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DeepJudge	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DI	43 / 47	37 / 41	9 / 3	11 / 3	3 / 0	2 / 0
ADV-TRA	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
Ours	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**
Clean Acc. Drop	7.18/4.46	6.37/2.81	7.12/5.23	7.04/5.30	8.38/6.16	8.21/6.44

Table 8 in Appendix C.7 exhibits their good auditing performance on scenarios without cross-architecture and adversarial evasion.

Cross-Architecture Model Stealing. We evaluate cross-architecture stealing under both model extraction (Knockoff and HL) and knowledge distillation (KD and DKD). As shown in Table 2, our method achieves 100% detection rates across all evaluated configurations. In contrast, IPGuard, DeepJudge, and ADV-TRA consistently yield 0% detection rates, indicating that their auditing signals are severely disrupted when functionality is transferred across architectures. Although DI is more robust than these baselines, its detection rate still drops to 64-83% across the evaluated settings. These results support our motivation that fingerprinting methods relying on preserved similarity between the victim and stolen models can be fragile under cross-architecture stealing. In contrast, our method exploits directional information-bottleneck deterioration, which remains stable across the evaluated cross-architecture transfers. Additional results on other settings are provided in Appendix D.2.

Adversarial Evasion. To study auditing robustness against evasion, we consider two adversarial-training strategies: *post-stealing adversarial training*, which applies adversarial training after a stolen model has been obtained, and *stealing-time adversarial training*, which incorporates an adversarial-training objective during the stealing process. Detailed settings are provided in Appendix C.

Table 3 reports the detection rates on RN18-CF10 under PGD-10 adversarial training with a perturbation budget of $\epsilon = 4/255$, where each entry corresponds to post-stealing / stealing-time adversarial training. Additional results under different settings and perturbation budgets are provided in Appendix D.3. Across all stealing methods, post-stealing adversarial training causes larger clean accuracy degradation, with drops of 6.37-8.38 percentage points, compared with 2.81-6.44 points

under stealing-time adversarial training. IPGuard, DeepJudge, and ADV-TRA yield 0% detection rates under both strategies, while DI is also substantially degraded, with detection rates of only 2-43% under post-stealing adversarial training and 0-47% under stealing-time adversarial training. For DI, post-stealing adversarial training provides slightly stronger evasion for fine-tuning and pruning, whereas stealing-time adversarial training is more effective for the evaluated distillation and extraction attacks. In contrast, our method consistently achieves 100% detection rates across all stealing methods under both adversarial-training strategies. These results demonstrate that our auditing signal remains robust to adversarial evasion regardless of whether adversarial training is applied after or during model stealing.

We further evaluate the more challenging setting that combines cross-architecture stealing with adversarial evasion. As shown in Tables 17, 18, and 19 in Appendix D.4, our method maintains effective auditing under this combined setting.

6 Discussion

Adaptive Information-Bottleneck-Aware Stealing. A natural adaptive attack is to explicitly optimize the stolen model to reproduce the victim's information-bottleneck behavior and thereby evade our auditing signal. While conceptually possible, such an attack is challenging because the relevant IB balance is measured on the victim's private dataset D_v, which is unavailable to the attacker. This is particularly difficult under black-box stealing, where the attacker can query the victim to reproduce its functionality but cannot directly observe or perfectly transfer the fragile information-processing balance learned on D_v. Nevertheless, developing IB-aware stealing strategies constitutes important future work.

Extension to LLMs and Generative Models. Our current framework focuses on classification models, while extending it to LLMs and generative models raises both formulation and estimation challenges. Without explicit labels, $I(T;Y)$ may be unavailable; nevertheless, our input-compression result for $I(X;T)$ does not depend on labels or a specific representation form, suggesting that a label-free auditing framework based on $I(X;T)$ and independent-model references may still be possible. The fundamental challenge is mutual-information estimation: generated text and images are high-dimensional structured outputs for which our current binning-based estimator is not directly applicable, and naive discretizations may eliminate meaningful compression differences. Developing semantically meaningful MI estimators is necessary to extend our framework beyond classification.

Scope of Model Stealing. In practice, model stealing spans a continuum in the extent of model modification. At one extreme, an exact copy or a nearly identical model produced by negligible modifications may exhibit little information-bottleneck deterioration. In such cases, however, the strong victim-suspect similarity that limits our approach is precisely the signal exploited by conventional fingerprinting methods. At the other extreme, model provenance may involve multiple stages, where a model is extracted from an intermediate model that was itself derived from the original victim. Such a downstream stolen model may never directly access the original victim, making whether it should still be regarded as stolen inherently dependent on the operational definition of model stealing. To examine whether auditing evidence can nevertheless persist, we perform sequential Knockoff extraction. As shown in Table 20 in Appendix D.5, our method detects all evaluated two-step stolen models, both when the two extraction stages use the same dataset and when different datasets are used. These observations suggest that information-deterioration auditing and similarity-based fingerprinting can provide complementary evidence across different regions of the model-derivation spectrum, while the ultimate interpretation of such evidence depends on how model stealing is practically defined.

7 Conclusion

We presented a non-intrusive model auditing framework based on information-bottleneck deterioration. We develop a two-stage auditing framework that combines directional deterioration screening with statistical comparison against independently trained models. Experiments demonstrate effective auditing with low false-positive rates and improved robustness under cross-architecture stealing and adversarial-training evasion.

AI USE STATEMENT

In this work, generative AI tools were used only for code refinement and debugging and for improving the clarity and presentation of the manuscript. All research ideas, problem formulations, methodological designs, theoretical analyses, experimental designs, result interpretations, and scientific conclusions were independently developed by the authors without relying on generative AI. All AI-assisted code and writing were carefully reviewed, verified, and revised by the authors. We take full responsibility for the final content of this work, including all text, claims, results, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide an anonymized code repository at `https://anonymous.4open.science/r/ICLR_2027_Model_Auditing/`, which contains the necessary implementation code, experiment scripts, and configurations for reproducing the main results in this paper. The appendix further provides detailed descriptions of the datasets and data partitions, victim and independent model training, model-stealing configurations, adversarial-training settings, information estimation procedure, reference-model construction, and evaluation protocols.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium (USENIX Security 18)*, pp. 1615–1631, 2018.

Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. IPGuard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pp. 14–25, 2021.

Jialuo Chen, Jingyi Wang, Tinglan Peng, Youcheng Sun, Peng Cheng, Shouling Ji, Xingjun Ma, Bo Li, and Dawn Song. Copy, right? a testing framework for copyright protection of deep learning models. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 824–841, 2022.

Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *Proceedings of the twenty-fourth international conference on architectural support for programming languages and operating systems*, pp. 485–497, 2019.

Lixin Fan, Kam Woh Ng, and Chee Seng Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. *Advances in neural information processing systems*, 32, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Yuepeng Hu, Zhengyuan Jiang, Mengyuan Li, Osama Ahmed, Zhicong Huang, Cheng Hong, and Neil Zhenqiang Gong. Fingerprinting llms via prompt injection. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11795–11810, 2026.

Hengrui Jia, Christopher A. Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 1937–1954, 2021.

Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *International Conference on Learning Representations*, 2021.

National Institute of Standards and Technology. Hotelling Control Chart. Dataplot Reference Manual, 2023. Phase II with individual observations. Last updated December 4, 2023.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4954–4963, 2019.

Zirui Peng, Shaofeng Li, Guoxing Chen, Cheng Zhang, Haojin Zhu, and Minhui Xue. Fingerprinting deep neural networks globally via universal adversarial perturbations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13420–13429, 2022.

Sunandini Sanyal, Sravanti Addepalli, and R. Venkatesh Babu. Towards data-free model stealing in a hard label setting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15284–15293, June 2022.

Shuo Shao, Yiming Li, Hongwei Yao, Yiling He, Zhan Qin, and Kui Ren. Explanation as a watermark: Towards harmless and multi-bit model ownership verification via watermarking feature attribution. In *Network and Distributed System Security Symposium (NDSS)*, 2025.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

Sebastian Szyller, Rui Zhang, Jian Liu, and N. Asokan. On the robustness of dataset inference. *Transactions on Machine Learning Research*, 2023.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision – ECCV 2020*, volume 12356 of *Lecture Notes in Computer Science*, pp. 776–794. Springer, 2020. doi: 10.1007/978-3-030-58621-8_45.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. *arXiv preprint arXiv:physics/0004057*, 2000.

Nola D. Tracy, John C. Young, and Robert L. Mason. Multivariate control charts for individual observations. *Journal of Quality Technology*, 24(2):88–95, 1992.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction APIs. In *USENIX Security Symposium (USENIX Security 16)*, pp. 601–618, 2016.

Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, pp. 269–277, 2017.

Ronald L. Wasserstein and Nicole A. Lazar. The ASA statement on p-values: Context, process, and purpose. *The American Statistician*, 70(2):129–133, 2016.

Tianlong Xu, Chen Wang, Gaoyang Liu, Yang Yang, Kai Peng, and Wei Liu. United we stand, divided we fall: Fingerprinting deep neural networks via adversarial trajectories. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Hongwei Yao, Zheng Li, Kunzhe Huang, Jian Lou, Zhan Qin, and Kui Ren. Removalnet: Dnn fingerprint removal attacks. *Transactions on Dependable and Secure Computing*, 21(4):2645–2658, 2023.

Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the Asia conference on computer and communications security*, pp. 159–172, 2018.

Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11953–11962, June 2022.

Appendix Contents

A	**Theoretical Analysis of Information Deterioration**	**13**
	Proof of Proposition 4.1	13
	Formal Setting and Assumptions	13
	Proof of Theorem 4.2	14
	Proof of Theorem 4.3	14
	Compatibility of the Assumptions	15
	Scope and Empirical Checks	17
B	**Statistical Calibration and Computation of Two-Stage Auditing**	**17**
	Information Measurements and Reference Construction	17
	Reference Population and Calibration Conditions	18
	Proof of Theorem 4.4	19
	References for Unknown Suspect Architectures	20
	Calibration of the Two-Stage Auditing Score	21
	Protocol Boundaries and Reporting	23
C	**Experimental Details**	**23**
	Datasets and Data Partition	23
	Victim and Independent Model Training	23
	Suspect-Model Construction	24
	Adversarial-Training Evasion	26
	Auditing Configuration	26
	Evaluation Metrics	27
	Baseline Methods	27
D	**Experimental Results**	**28**
	Overall Performance	28
	Cross Architecture	30
	Adversarial Evasion	31
	Cross Architecture with Adversarial Evasion	34
	Sequential Knockoff Extraction	34

A Theoretical Analysis of Information Deterioration

A.1 Proof of Proposition 4.1

The identities require only a common joint distribution and finiteness of the information quantities involved, not the structural assumptions introduced below.

Proof. Expanding the mutual information between X and (T_v, T_s) in two orders gives

$$\begin{aligned} I(X; T_v, T_s) &= I(X; T_v) + I(X; T_s \mid T_v) \\ &= I(X; T_s) + I(X; T_v \mid T_s). \end{aligned}$$

Rearranging yields

$$I(X; T_s) - I(X; T_v) = I(X; T_s \mid T_v) - I(X; T_v \mid T_s),$$

which establishes equation 4. Similarly,

$$\begin{aligned} I(Y; T_v, T_s) &= I(Y; T_v) + I(Y; T_s \mid T_v) \\ &= I(Y; T_s) + I(Y; T_v \mid T_s). \end{aligned}$$

Rearranging and using symmetry gives

$$I(T_s; Y) - I(T_v; Y) = I(T_s; Y \mid T_v) - I(T_v; Y \mid T_s),$$

which establishes equation 5. □

These identities apply to independently trained models as well as models derived from the victim. They express information changes as balances of gains and losses, without establishing their signs or model provenance.

A.2 Formal Setting and Assumptions

Proposition 4.1 is a chain-rule identity for jointly distributed representations with finite information quantities. The structural assumptions and conditional deterioration results use the following more specific setting.

Fix the victim's private dataset D_v and let $(X, Y) \sim \widehat{P}_v$, with Y denoting the ground-truth label. Both models are fixed and evaluated deterministically. For the conditional deterioration results, we use deterministic, finite-valued representations. When finite resolution is obtained through a map Q, it is predetermined and shared across models, so that $T_v = Q(f_v(X))$ and $T_s = Q(f_s(X))$. Here T_v and T_s denote these finite-resolution versions of the learned representations. All probabilities below refer to the same empirical distribution, and logarithms use base two. Consequently,

$$I(X; T_v) = H(T_v), \qquad I(X; T_s) = H(T_s), \tag{15}$$

because $H(T_v \mid X) = H(T_s \mid X) = 0$. In the transfer model $T_s = g(T_v, R(X))$, both g and R are fixed functions; $R(X)$ is input-dependent variation rather than query-time noise.

Let $\mathcal{V}$ and $\mathcal{S}$ be the supports of T_v and T_s, restricted to states with positive probability. Define the induced transfer kernel

$$K(b \mid a) = \Pr(T_s = b \mid T_v = a), \qquad a \in \mathcal{V},\ b \in \mathcal{S}. \tag{16}$$

This conditional distribution is induced by sampling inputs that share a victim representation; it does not impose $X \to T_v \to T_s$.

Assumption A.1 (No additional label information)**.** The residual variation supplies no label information beyond the victim representation:

$$Y \perp R(X) \mid T_v. \tag{17}$$

Together with the transfer model, this gives $Y \perp T_s \mid T_v$. It allows additional distinctions between inputs sharing T_v, provided that the residual supplies no additional label information. The condition holds automatically when $H(Y \mid T_v) = 0$, although perfect label recovery is not required.

Assumption A.2 (Fragmentation-dominated transfer)**.** For each $b \in \mathcal{S}$, define

$$c_b = \sum_{a \in \mathcal{V}} K(b \mid a). \tag{18}$$

Then $c_b \leq 1$ for every $b \in \mathcal{S}$, with strict inequality for at least one such state.

Each row of K sums to one, whereas c_b sums conditional transition probabilities over source states and is not the marginal probability $P(T_s = b)$. The condition limits many-to-one concentration rather than assuming an information ordering. For example, if each source state in an isolated block spreads uniformly over m destination states and each destination receives contributions from at most $r < m$ sources, then $c_b \leq r/m < 1$ within that block. Disjoint unchanged states may have $c_b = 1$. Merely increasing the number of representation states would not control their probability concentration.

Assumption A.3 (Nontrivial label mixing)**.** There exist $a_1, a_2 \in \mathcal{V}$ and $b \in \mathcal{S}$ such that

$$\begin{aligned} &K(b \mid a_1)K(b \mid a_2) > 0, \\ &P(Y \mid T_v = a_1) \neq P(Y \mid T_v = a_2). \end{aligned} \tag{19}$$

Thus, at least one stolen representation state combines victim states with different conditional label distributions. Under Assumption A.1, this mixing entails a strict loss of label information, as established below.

A.3 Proof of Theorem 4.2

Under Assumptions A.1 and A.3, we establish the exact positive gap

$$\delta_Y := I(T_v; Y) - I(T_s; Y) = I(T_v; Y \mid T_s) > 0. \tag{20}$$

Proof. Since $T_s = g(T_v, R(X))$, Assumption A.1 implies $I(T_s; Y \mid T_v) = 0$. The label-information balance equation 5 therefore gives

$$I(T_v; Y) - I(T_s; Y) = I(T_v; Y \mid T_s).$$

The same conditional independence gives $P(Y \mid T_v = a, T_s = b) = P(Y \mid T_v = a)$ on the joint support. Consequently,

$$\begin{aligned} \delta_Y = &\sum_{a,b} P(T_v = a, T_s = b) \\ &\cdot D_{\mathrm{KL}}(P(Y \mid T_v = a) \,\|\, P(Y \mid T_s = b)) \,. \end{aligned} \tag{21}$$

For a state b identified by Assumption A.3, $P(Y \mid T_s = b)$ is a mixture containing at least two distinct distributions $P(Y \mid T_v = a)$ with positive weights. These distributions cannot all equal their mixture. At least one positively weighted divergence is therefore strictly positive, establishing $\delta_Y > 0$. □

Relaxing exact label noninnovation. Without Assumption A.1, the chain-rule identity still holds. If $H(Y \mid T_v) \leq \eta$, then $I(T_s; Y \mid T_v) \leq \eta$, and hence

$$\begin{aligned} I(T_v; Y) - I(T_s; Y) &= I(T_v; Y \mid T_s) - I(T_s; Y \mid T_v) \\ &\geq I(T_v; Y \mid T_s) - \eta. \end{aligned} \tag{22}$$

This bound guarantees strict deterioration when $I(T_v; Y \mid T_s) > \eta$; a small $H(Y \mid T_v)$ alone does not guarantee the sign. The input-compression result is unaffected because it does not use label noninnovation.

A.4 Proof of Theorem 4.3

We establish the quantitative lower bound

$$\begin{aligned} &I(X; T_s) - I(X; T_v) \geq \delta_X > 0, \\ &\delta_X := -\sum_{b \in \mathcal{S}} P(T_s = b) \log c_b. \end{aligned} \tag{23}$$

Unlike δ_Y, δ_X is a lower bound, not generally the exact gap.

Proof. Write $p_v(a) = P(T_v = a)$ and $p_s(b) = P(T_s = b)$. For each $b \in \mathcal{S}$, $p_s(b) > 0$ and $c_b > 0$. Define two probability distributions on $\mathcal{V}$:

$$\pi_b(a) = \frac{p_v(a)K(b \mid a)}{p_s(b)}, \qquad r_b(a) = \frac{K(b \mid a)}{c_b}. \tag{24}$$

Expanding the divergence gives

$$\begin{aligned}\sum_b p_s(b) D_{\mathrm{KL}}(\pi_b \| r_b) &= \sum_{a,b} p_v(a) K(b \mid a) \log \frac{p_v(a) c_b}{p_s(b)} \\ &= H(T_s) - H(T_v) + \sum_b p_s(b) \log c_b.\end{aligned}$$

Zero-probability contributions are interpreted as zero. It follows that

$$\begin{aligned}H(T_s) - H(T_v) &= \sum_b p_s(b) D_{\mathrm{KL}}(\pi_b \| r_b) + \delta_X \\ &\geq \delta_X > 0.\end{aligned} \tag{25}$$

The divergence terms are nonnegative. Moreover, Assumption A.2 makes every term $-p_s(b) \log c_b$ nonnegative and at least one strictly positive. Equation (15) then establishes Equation (23). Using the input-information balance equation 4, we also obtain

$$\begin{aligned}&I(X; T_s \mid T_v) - I(X; T_v \mid T_s) \\ &\quad = I(X; T_s) - I(X; T_v) \geq \delta_X > 0.\end{aligned}$$

Thus, the structural condition implies that additional input retention exceeds input-information loss, rather than assuming that ordering. □

The divergence identity remains valid without Assumption A.2. In particular, $\sum_b p_s(b) \log c_b < 0$ is a weaker sufficient condition for a positive gap, even when some $c_b > 1$. The theorem uses the simpler columnwise condition and makes no necessity claim.

A.5 Compatibility of the Assumptions

The induced kernel $K(b \mid a)$ must not be interpreted as imposing $X \to T_v \to T_s$. That Markov chain would imply $I(X; T_s) \leq I(X; T_v)$ by the data-processing inequality. Instead, $T_s = g(T_v, R(X))$ allows additional input information through $R(X)$. The condition $Y \perp R(X) \mid T_v$ excludes additional label information, not additional input information. The two deterioration results therefore use compatible, rather than contradictory, conditional-independence requirements.

To demonstrate that all three assumptions can hold simultaneously, let Y and U be independent fair bits and let $E \sim \mathrm{Bernoulli}(\epsilon)$ be independent of both, with $0 < \epsilon < 1/2$. Treat (Y, E, U) as input attributes and define

$$T_v = Y, \qquad T_s = (Y \oplus E, U), \qquad R(X) = (E, U). \tag{26}$$

All quantities are deterministic for a fixed input; E is not resampled on a query. In column order $(0,0), (0,1), (1,0), (1,1)$, the induced kernel is

$$K = \frac{1}{2} \begin{pmatrix} 1-\epsilon & 1-\epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & 1-\epsilon & 1-\epsilon \end{pmatrix}. \tag{27}$$

Every column sum equals $1/2$, and every destination mixes distinct label distributions. Since $T_v = Y$, label noninnovation also holds. Direct calculation gives

$$\begin{aligned} I(X; T_v) &= 1, & I(X; T_s) &= 2, \\ I(T_v; Y) &= 1, & I(T_s; Y) &= 1 - h_2(\epsilon), \end{aligned}$$

where h_2 is binary entropy. For $\epsilon = 0.01$, an exact empirical realization uses 400 distinct inputs: for each (y, u), take 99 with $e = 0$ and one with $e = 1$, using identifiers ignored by both representations to distinguish repeated attribute combinations. This example demonstrates compatibility of the assumptions, not an empirical result for a model-stealing procedure.

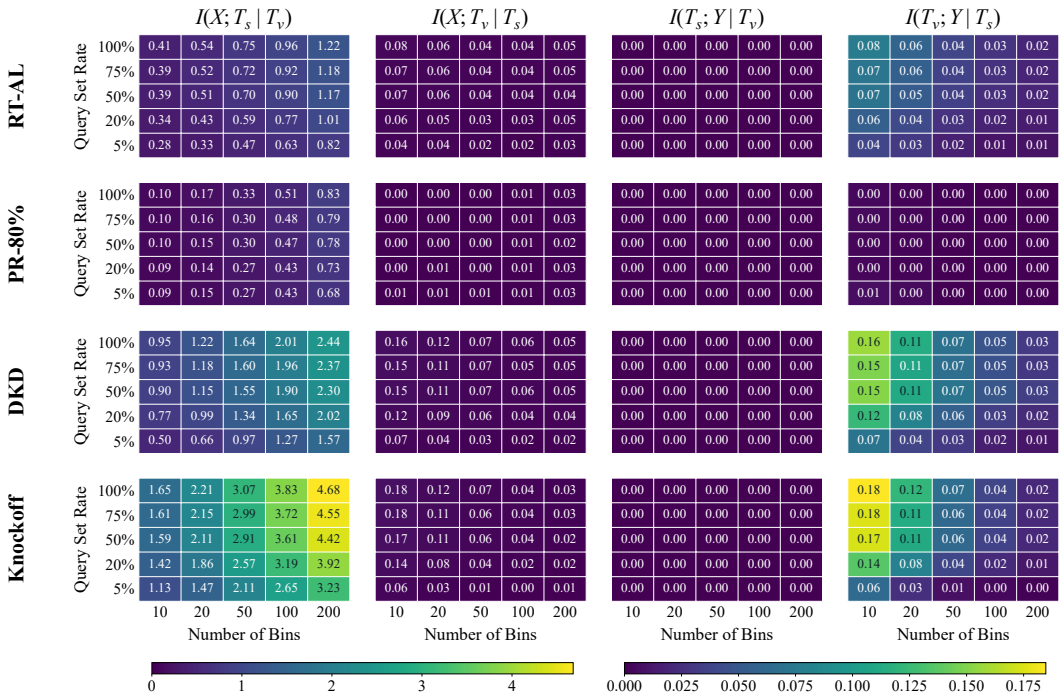


Figure 4: Conditional information gains and losses on CIFAR-10 with ResNet-18.

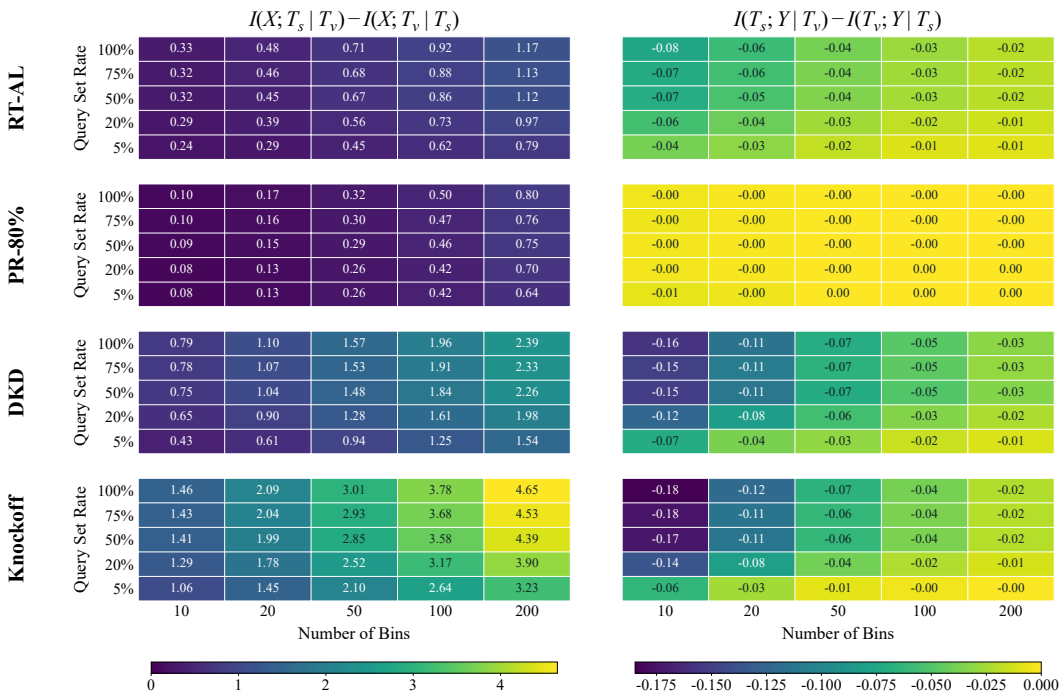


Figure 5: Net differences between conditional information gains and losses on CIFAR-10 with ResNet-18.

A.6 Scope and empirical checks

The assumptions constrain representation transfer on D_v. They are not implied merely by naming a stealing procedure or withholding the victim's training data. Likewise, the functional description $T_s = g(T_v, R(X))$ alone is not evidence of model derivation.

The conditional deterioration results concern the specified finite-resolution representations. The resolution must be fixed in advance and applied consistently across models. If it distinguishes every one of N uniformly sampled private inputs for both models, then $I(X;T_v) = I(X;T_s) = \log N$, eliminating the strict input-information difference. An excessively coarse resolution can instead remove fragmentation or label distinctions. The results should therefore not be interpreted as an estimator-independent ordering for raw continuous representations.

We further examine whether stolen models empirically exhibit the gain-loss structure underlying Proposition 4.1 and Theorems 4.2 and 4.3. These results are intended as diagnostics of the information-transfer mechanism rather than as direct verification of Assumptions A.1-A.3, which provide sufficient but not necessary conditions for directional deterioration.

Recall that Proposition 4.1 decomposes the input-information change as

$$I(X;T_s) - I(X;T_v) = I(X;T_s \mid T_v) - I(X;T_v \mid T_s),$$

and the label-information change as

$$I(T_s;Y) - I(T_v;Y) = I(T_s;Y \mid T_v) - I(T_v;Y \mid T_s).$$

We directly measure these four conditional-information terms using the same finite-resolution prediction representations as in the auditing procedure. Figure 4 reports the results for representative stealing methods on CIFAR-10 with ResNet-18 across different query-set rates and discretization resolutions.

On the input side, $I(X;T_s \mid T_v)$ consistently exceeds $I(X;T_v \mid T_s)$ across the reported stealing methods and measurement settings. Thus, the additional input distinctions retained by the stolen representation dominate the input information lost relative to the victim, producing the positive input-information change predicted by Theorem 4.3. On the label side, $I(T_s;Y \mid T_v)$ is negligible at the reported precision, while $I(T_v;Y \mid T_s)$ is positive for RT-AL, DKD, and Knockoff. This indicates that the stolen representation contributes little additional label information beyond the victim while losing part of the label information retained by the victim, consistent with the mechanism underlying Theorem 4.2. PR-80% exhibits the same input-side pattern but only a negligible label-information gap at the evaluated resolutions.

Figure 5 summarizes the corresponding net differences. Although their magnitudes vary with query-set size and binning resolution, the positive input-side gap is stable across all reported settings, while the label-side gap remains negative for RT-AL, DKD, and Knockoff and is close to zero for heavy pruning. These observations provide empirical support for the gain-loss mechanism underlying information-bottleneck deterioration, beyond observing only the final displacement in the information plane.

B Statistical Calibration and Computation of Two-Stage Auditing

B.1 Information Measurements and Reference Construction

Binning-based information estimation. We use binning to estimate the two information quantities from model predictions. Let $D_v = \{(x_i, y_i)\}_{i=1}^n$ denote the victim's private training dataset. For each model f, we discretize its learned representations using a binning map Q, which assigns each prediction vector to a discrete bin:

$$T_f = Q\big(f(X)\big), \qquad (X,Y) \sim \widehat{P}_v. \tag{28}$$

The victim, suspect, and negative models are evaluated deterministically on the same dataset using the same binning rule and bin boundaries. Thus, T_f denotes the binned representation obtained from model predictions, consistent with the finite-valued setting in Appendix A.2.

For each bin t and ground-truth label y, define the sample counts

$$n_f(t) = \sum_{i=1}^{n} \mathbf{1}\{Q(f(x_i)) = t\}\,,$$
$$n_f(t,y) = \sum_{i=1}^{n} \mathbf{1}\{Q(f(x_i)) = t,\ y_i = y\}\,. \tag{29}$$

The empirical marginal and joint probabilities are

$$\widehat{p}_f(t) = \frac{n_f(t)}{n}, \qquad \widehat{p}_f(t,y) = \frac{n_f(t,y)}{n}, \tag{30}$$

with $\widehat{p}(y) = n^{-1}\sum_{i=1}^{n} \mathbf{1}\{y_i = y\}$.

Because the binned representation T_f is a deterministic function of X, its conditional entropy given X is zero. The input-information estimate therefore equals the entropy of the bin assignments:

$$\widehat{I}_{D_v}(X;T_f) = -\sum_{t} \widehat{p}_f(t)\log_2 \widehat{p}_f(t). \tag{31}$$

The label-information estimate is computed from the joint distribution of bin assignments and ground-truth labels:

$$\widehat{I}_{D_v}(T_f;Y) = \sum_{t,y} \widehat{p}_f(t,y)\log_2 \frac{\widehat{p}_f(t,y)}{\widehat{p}_f(t)\widehat{p}(y)}. \tag{32}$$

Terms with zero empirical probability are omitted. Both quantities measure information at the chosen binning resolution, which is held fixed across all model comparisons.

Reference moments. For the architecture-matched negative-model pool,

$$\mathcal{N}_{a_s} = \{f_-^{(1)}, \ldots, f_-^{(k)}\}, \qquad \mathbf{m}_-^{(j)} = \mathbf{m}(f_-^{(j)}). \tag{33}$$

Each negative model contributes one information vector defined by equation 8, computed using the same binning-based procedure. The sample mean and covariance used in equation 11 are

$$\widehat{\boldsymbol{\mu}}_- = \frac{1}{k}\sum_{j=1}^{k} \mathbf{m}_-^{(j)},$$
$$\widehat{\Sigma}_- = \frac{1}{k-1}\sum_{j=1}^{k} \big(\mathbf{m}_-^{(j)} - \widehat{\boldsymbol{\mu}}_-\big)\big(\mathbf{m}_-^{(j)} - \widehat{\boldsymbol{\mu}}_-\big)^{\top}. \tag{34}$$

The suspect is excluded from both estimates. The reference pool is not filtered by the directional gate, since such filtering would change the reference population and require a different calibration. The private dataset D_v is used to evaluate the negative models, not to train them.

B.2 Reference Population and Calibration Conditions

The unit of replication is a trained model, not an example, a repeated query, or a bootstrap resample from one fixed model. Condition on the victim, D_v, the common measurement procedure, the known suspect architecture a_s, and the prespecified independent-training protocol. Any common fixed auxiliary training data are also included in this conditioning. The negative models are generated using independent training randomness, without inheriting the victim's parameters or using its predictions as supervision, and their training data exclude D_v.

Let P_- be the resulting distribution of information vectors. The reference null is

$$H_0^{\mathrm{ind}}: \quad \mathbf{m}_s, \mathbf{m}_-^{(1)}, \ldots, \mathbf{m}_-^{(k)} \overset{\mathrm{iid}}{\sim} P_-. \tag{35}$$

Thus, a null suspect follows the same model-generation law and is independent of the reference pool. Architecture matching alone does not establish this condition. Differences in training data, optimization, regularization, training duration, robustness training, or prediction post-processing can change P_-.

For analytic calibration, we additionally adopt the working model

$$P_- = \mathcal{N}_d(\boldsymbol{\mu}_-, \Sigma_-), \qquad \Sigma_- \succ 0, \quad k > d. \tag{36}$$

The calibration in Theorem 4.4 is exact under this Gaussian reference model. Actual finite-resolution empirical information vectors are bounded and determined by empirical counts, so they are not literally nondegenerate Gaussian variables. The analytic calibration is therefore a model-based approximation in the auditing application, whose tail behavior must be assessed using held-out independent models. Neither Gaussianity nor reference-population matching follows from the private-data setting. The reference protocol and measurement settings must not be selected to minimize the suspect's p-value.

B.3 Proof of Theorem 4.4

We prove Theorem 4.4, which gives the corresponding analytic reference distribution (Tracy et al., 1992; National Institute of Standards and Technology, 2023).

Proof. Write $\overline{\mathbf{m}} = \widehat{\boldsymbol{\mu}}_-$ and $S = \widehat{\Sigma}_-$. The independent suspect and reference mean satisfy

$$\mathbf{m}_s - \overline{\mathbf{m}} \sim \mathcal{N}_d\left(0, \left(1 + \frac{1}{k}\right)\Sigma_-\right). \tag{37}$$

Under Gaussian sampling, the reference mean and covariance are independent; the suspect is also independent of the entire reference pool. Consequently,

$$\mathbf{z} = \sqrt{\frac{k}{k+1}}\,\Sigma_-^{-1/2}(\mathbf{m}_s - \overline{\mathbf{m}}) \sim \mathcal{N}_d(0, I_d) \tag{38}$$

is independent of

$$W = (k-1)\Sigma_-^{-1/2} S \Sigma_-^{-1/2} \sim \mathrm{Wishart}_d(I_d, k-1). \tag{39}$$

Substitution into the Hotelling T^2 formula gives

$$T^2_{\mathrm{audit}} = (k-1)\mathbf{z}^\top W^{-1}\mathbf{z}. \tag{40}$$

The Gaussian-Wishart quadratic-form identity yields

$$\frac{k-d}{d}\mathbf{z}^\top W^{-1}\mathbf{z} \sim F_{d,k-d}, \tag{41}$$

which establishes equation 12.

For completeness, let $\nu = k-1$ and write $\mathbf{z} = \|\mathbf{z}\|u$. The squared norm satisfies $\|\mathbf{z}\|^2 \sim \chi^2_d$ and is independent of both u and W. For any fixed unit vector u, the Wishart Schur-complement identity gives $(u^\top W^{-1}u)^{-1} \sim \chi^2_{\nu-d+1}$. Orthogonal invariance makes this distribution independent of the sampled direction. Hence $\mathbf{z}^\top W^{-1}\mathbf{z}$ is a ratio of independent χ^2_d and χ^2_{k-d} variables, giving the stated F law. □

Computing the reference p-value. Let $F_{r,q}(\cdot)$ denote the cumulative distribution function of an F variable with degrees of freedom (r, q). The second-stage value is

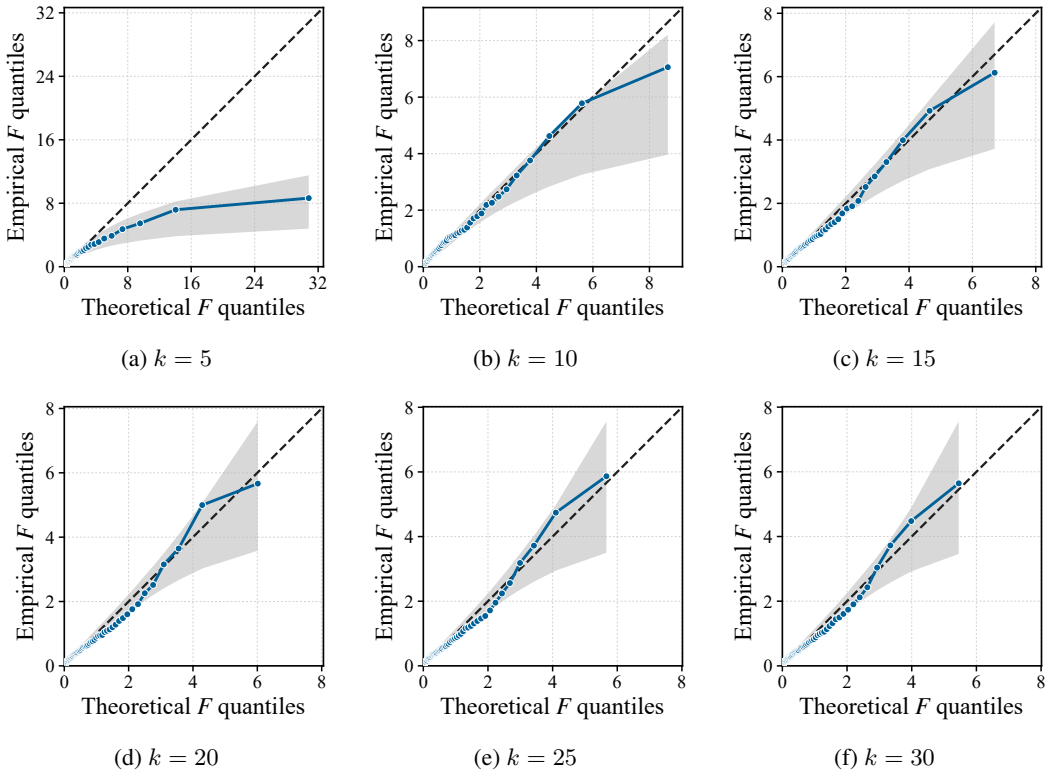
$$p_s = 1 - F_{d,k-d}\left(\frac{k-d}{d(k-1)}T^2_{\mathrm{audit}}\right). \tag{42}$$

In this application, $d = 2$, so the reference argument is $(k-2)T^2_{\mathrm{audit}}/[2(k-1)]$ and the degrees of freedom are $(2, k-2)$. The requirement $k > 2$ is algebraic; it does not establish that three negative models suffice for useful calibration or statistical power. Numerically, the upper tail should be evaluated using the F survival function, or its logarithm for very small values, rather than subtracting a cumulative probability close to one.

Limits of increasing the reference-pool size. For a general P_- with finite second moments and nonsingular covariance, increasing k consistently estimates the reference moments, but the limiting statistic for an independent singleton suspect is

$$T^2_{\mathrm{audit}} \longrightarrow (\mathbf{m}_s - \boldsymbol{\mu}_-)^\top \Sigma_-^{-1}(\mathbf{m}_s - \boldsymbol{\mu}_-). \tag{43}$$

The suspect's variation is not averaged away. Consequently, a central limit theorem for the reference mean does not by itself justify a chi-squared reference for this quantity. Under the Gaussian model, the finite-k law converges to χ^2_d, but finite second moments alone are insufficient.



(a) $k = 5$ (b) $k = 10$ (c) $k = 15$

(d) $k = 20$ (e) $k = 25$ (f) $k = 30$

Figure 6: Exact-F Q–Q plots for ResNet-18 in CIFAR-10 with different reference set sizes k.

Empirical calibration of the reference distribution. We further examine the adequacy of the theoretical F reference distribution using Q-Q plots constructed from held-out independently trained models. Figures 6, 7, and 8 compare the empirical quantiles of the transformed Hotelling statistic with the corresponding theoretical F quantiles under different reference-pool sizes on different models trained on CIFAR-10. Across all evaluated pool sizes, the empirical quantiles follow the theoretical reference reasonably closely. The agreement further improves as the number of reference models increases, with the points becoming progressively more aligned with the reference diagonal. This is expected because a larger reference pool provides more stable estimates of the mean and covariance of the independent-model population. Importantly, even with relatively small reference pools, the Q-Q plots show only modest deviations from the theoretical reference rather than a systematic departure. These results support the practical adequacy of the F-based calibration over the range of reference-pool sizes considered in our experiments.

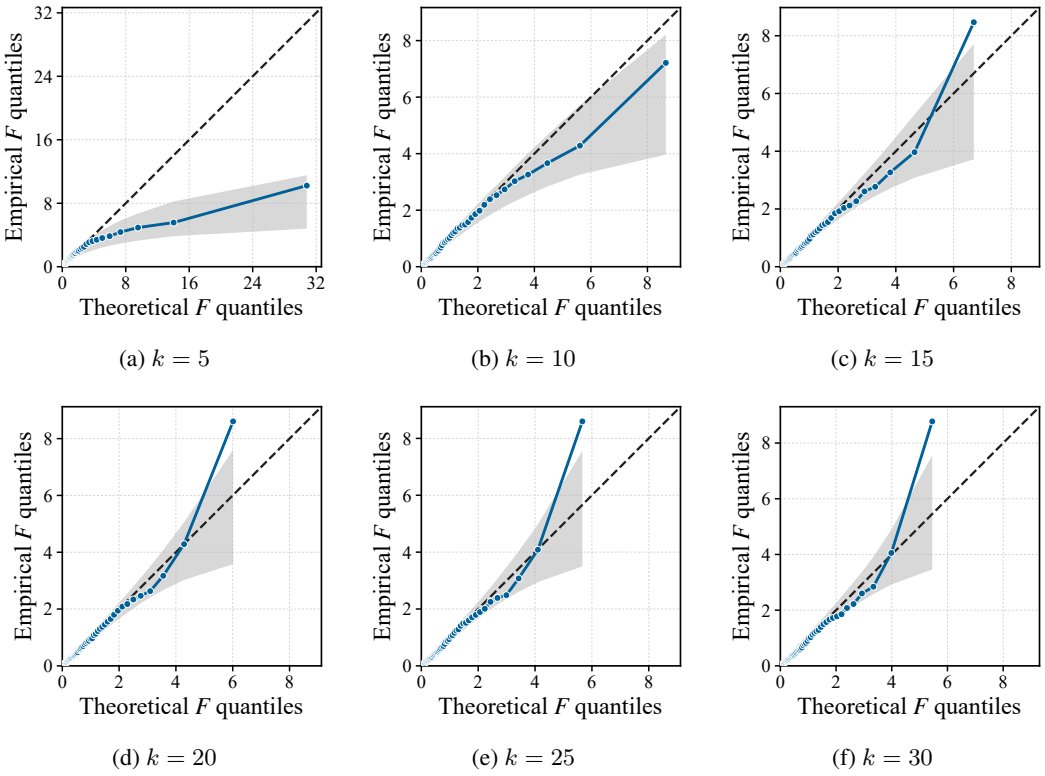
B.4 References for Unknown Suspect Architectures

When the suspect's architecture is unavailable, let $\mathcal{A}$ be a prespecified set of candidate architectures. For each $a \in \mathcal{A}$, construct a separate negative-model pool and apply the test in Section 4.3.2, obtaining an architecture-specific value $p_{s,a}$. The reference moments and predictive calibration are computed separately for each pool rather than from a pooled cross-architecture distribution.

The composite independent-training explanation is that the suspect follows at least one of the candidate reference populations. We define the second-stage value as

$$p_s = \max_{a \in \mathcal{A}} p_{s,a}. \tag{44}$$

This aggregation requires the suspect to be atypical of every candidate reference before the combined value can be small.



(a) $k = 5$ (b) $k = 10$ (c) $k = 15$

(d) $k = 20$ (e) $k = 25$ (f) $k = 30$

Figure 7: Exact-F Q–Q plots for VGG16 in CIFAR-10 with different reference set sizes k.

To establish its calibration, suppose the suspect follows a candidate reference population indexed by $a_0 \in \mathcal{A}$, and p_{s,a_0} is valid under that reference null. For every $0 < t < 1$,

$$\begin{aligned} \Pr(p_s \le t) &= \Pr\left(\bigcap_{a\in\mathcal{A}} \{p_{s,a} \le t\}\right) \\ &\le \Pr(p_{s,a_0} \le t) \\ &\le t. \end{aligned} \tag{45}$$

Thus, the combined value is conservative under the composite null. No independence between the architecture-specific values is required. The gated-score calibration in Appendix B.5 then applies unchanged.

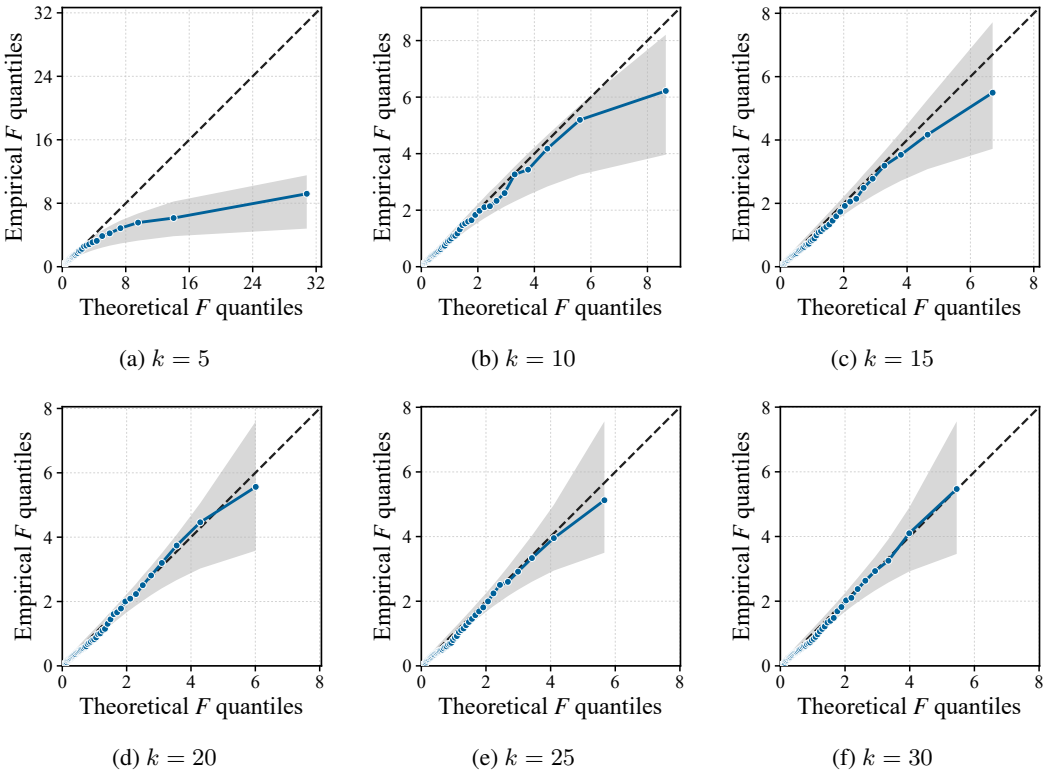
This extension requires the suspect's independent-training population to be represented among the candidate references and the corresponding reference-specific calibration to be valid. Including its architecture alone does not ensure matching training distributions or protocols. The aggregation does not guarantee that every derived model receives a small value.

B.5 Calibration of the two-stage auditing score

Fix the victim, its private dataset D_v, the information-measurement procedure, and the reference-construction protocols. The probability statements below average over the null suspect and reference-pool generation under this fixed evaluation setting. The following theorem establishes that the two requirements in the auditing score calculation preserve the calibration of a valid second-stage p-value.

Theorem B.1. *Suppose p_s is valid under the reference null H_0^{ind}, meaning that $\Pr_{H_0^{\mathrm{ind}}}(p_s \le t) \le t$ for every $t \in (0, 1)$. Then the auditing score in equation 14 satisfies*

$$\Pr_{H_0^{\mathrm{ind}}}(\mathcal{F} \le t) \le \Pr_{H_0^{\mathrm{ind}}}(p_s \le t) \le t, \tag{46}$$



(a) $k = 5$ (b) $k = 10$ (c) $k = 15$

(d) $k = 20$ (e) $k = 25$ (f) $k = 30$

Figure 8: Exact-F Q–Q plots for DeiT in CIFAR-10 with different reference set sizes k.

Proof. By assumption, the second-stage value satisfies

$$\Pr_{H_0^{\mathrm{ind}}}(p_s \le t) \le t, \qquad 0 < t < 1. \tag{47}$$

For any $t \in (0, 1)$, the branch with $G_s = 0$ returns a score of 1 and therefore cannot contribute to the event $\{\mathcal{F} \le t\}$. Consequently,

$$\{\mathcal{F} \le t\} = \{G_s = 1\} \cap \{p_s \le t\} \subseteq \{p_s \le t\}. \tag{48}$$

Taking probabilities gives

$$\begin{aligned} \Pr_{H_0^{\mathrm{ind}}}(\mathcal{F} \le t) &= \Pr_{H_0^{\mathrm{ind}}}(G_s = 1,\ p_s \le t) \\ &\le \Pr_{H_0^{\mathrm{ind}}}(p_s \le t) \\ &\le t, \end{aligned}$$

where the final inequality follows from equation 47. The argument uses only event inclusion and does not require independence between the gate and the second-stage value. □

Connection to reference calibration. For a single reference pool, Theorem 4.4 establishes the required validity under its Gaussian reference model. For multiple candidate architectures, the same conclusion applies under the composite null that the suspect follows at least one candidate reference population. As shown in Appendix B.4, the combined value $p_s = \max_{a \in \mathcal{A}} p_{s,a}$ is valid whenever the value for a matching reference population is valid. The gated-score argument therefore applies to both reference constructions.

Interpretation and scope. The result bounds the probability of obtaining a small auditing score under the specified reference null. It does not assert validity conditional on passing the gate or on every particular realized reference pool. It is a per-audit calibration statement, not simultaneous

control over selectively reported audits. When the second-stage calibration is approximate, the guarantee depends on the actual validity of the resulting p-values; the gate alone does not correct reference miscalibration. The score is not a posterior probability of model theft, and a large value does not establish independent training (Wasserstein & Lazar, 2016).

B.6 Protocol Boundaries and Reporting

Reference coverage. The procedure requires separate task data for training the negative models, together with either knowledge of the suspect's architecture or a prespecified set of candidate architectures. Reports should specify these data and their relationship to the suspect's possible training data, the reference-pool size, optimization and robustness-training protocols, the information-measurement settings, and the calibration variant used. Architecture matching addresses one source of variation but does not establish coverage of arbitrary independent-training procedures. Held-out independent controls should reflect the claimed scope of the evaluation.

Interpretation of the score. Failure of the gate means that the strict deterioration pattern is absent at the chosen resolution, not that the suspect was independently trained. An exact copy with the same finite-resolution representations as the victim fails the strict inequalities; near-identical models or changes below the resolution can do so as well. When the gate passes, a small score indicates incompatibility with the specified independent-model population, which may also arise from reference mismatch.

Covariance failures. The analytic procedure requires a nonsingular reference covariance. If this condition fails when the second stage is required, the analytic score should be reported as unavailable, or a separately calibrated alternative should be used. A singular covariance must not be silently regularized while retaining the original F reference, and an unavailable score must not be conflated with the score 1 assigned by the directional gate.

C Experimental Details

C.1 Datasets and Data Partition

We evaluate our auditing framework on CIFAR-10, CIFAR-100, and ImageNet-100. For ImageNet-100, we adopt the 100-class subset of ImageNet-1k introduced by Tian et al. (2020) and apply the same class-balanced partitioning procedure.

The first half forms the owner's private dataset D_v. It is used to train the victim model and subsequently serves as the private query source for computing the information quantities during auditing. The second half is treated as non-private data and is available to train independent models and stolen suspect models. Independent models never use samples from D_v during training.

C.2 Victim and Independent Model Training

Victim models. We consider ResNet-18, VGG16, and DeiT-Tiny on CIFAR-10 and CIFAR-100, and Swin-T on ImageNet-100, resulting in seven dataset-architecture settings. For cross-architecture distillation, we additionally train ResNet-34 and VGG19 as victim models to cover a broader range of victim-student architecture pairs. For each setting, we train one victim model from scratch exclusively on the corresponding private dataset D_v.

For the CIFAR datasets, the CNN models use random cropping with 4-pixel padding and horizontal flipping with probability 0.5. They are trained with SGD using an initial learning rate of 0.1, momentum 0.9, Nesterov acceleration, batch size 128, and cosine decay to 10^{-6}. The weight decay is 5×10^{-4}, except for ResNet-18 on CIFAR-100, where it is 10^{-3}. ResNet-18 and VGG16 are trained for 160 epochs on CIFAR-10 and for 360 and 400 epochs on CIFAR-100, respectively. For both datasets, we use distilled DeiT-Tiny with 4×4 patches and hard distillation from a ResNet-18 teacher, weighting the ground-truth and distillation losses equally. Its augmentation pipeline consists of random resized cropping with scale $[0.85, 1.0]$ and aspect ratio $[0.9, 1.1]$, horizontal flipping with probability 0.5, RandAugment with two operations, and random erasing. On CIFAR-10, we use

Table 4: Victim-model configurations and predictive performance

Dataset	Architecture	Training Accuracy	Evaluation Accuracy
CIFAR-10	ResNet-18	100.00	92.99
CIFAR-10	ResNet-34	99.99	93.56
CIFAR-10	VGG16	99.98	91.65
CIFAR-10	VGG19	99.98	91.46
CIFAR-10	DeiT-Tiny	99.99	84.15
CIFAR-100	ResNet-18	100.00	72.24
CIFAR-100	ResNet-34	100.00	73.07
CIFAR-100	VGG16	99.98	67.69
CIFAR-100	VGG19	99.98	67.60
CIFAR-100	DeiT-Tiny	99.99	60.80
ImageNet-100	Swin-T	96.51	76.88

RandAugment magnitude 7, random-erasing probability 0.15, and label smoothing 0.05; on CIFAR-100, these values are 9, 0.20, and 0.15, respectively. The DeiT models are trained with AdamW at a learning rate of 7×10^{-4}, batch size 128, and linear warm-up followed by cosine decay to 10^{-6}. We train for 200 epochs with weight decay 0.03 on CIFAR-10 and for 260 epochs with weight decay 0.05 on CIFAR-100.

For ImageNet-100, we use the Swin-T model with patch size 4 and window size 7. Training images are processed at a resolution of 224×224 using random resized cropping with scale $[0.08, 1.0]$ and aspect ratio $[3/4, 4/3]$, horizontal flipping with probability 0.5, RandAugment with two operations and magnitude 9, and random erasing with probability 0.25. The model is trained from scratch for 200 epochs using AdamW with an initial learning rate of 2×10^{-4}, weight decay 0.05, and batch size 128. The learning rate is linearly warmed up from 2×10^{-6} for the first 20 epochs and then cosine-decayed to 10^{-6}. Weight decay is excluded from normalization parameters, biases, and relative-position bias tables. We additionally use label smoothing of 0.1, a drop-path rate of 0.1, gradient clipping at norm 1.0, and bfloat16 mixed-precision training. At evaluation time, images are resized to 256 pixels along the shorter side and center-cropped to 224×224. The predictive performance of all victim models is reported in Table 4.

Independent models. For each dataset-architecture setting, we independently train 50 models using only the corresponding non-private subset. These models neither inherit parameters from the victim nor use victim predictions as supervision and are treated as negative suspects. In addition, we construct separate reference pools of 30 independently trained models with different random initializations and selected training configurations to introduce diversity into the independent-model reference distribution.

C.3 Suspect-Model Construction

For each stealing configuration, we construct 50 stolen models as positive suspects. Together with the fixed 50 independently trained negative suspects, each configuration therefore contains 100 suspect models and yields a balanced auditing evaluation.

We consider both white-box and black-box model-stealing settings. Table 5 summarizes the main attack families, victim access, training data, and whether cross-architecture stealing is considered.

Fine-tuning. We evaluate three fine-tuning variants: LL, AL, and RT-AL. All variants initialize the suspect model from the trained victim and use only the non-private data for subsequent training.

For LL, we update only the final classification head while freezing the backbone. For AL, we update all parameters initialized from the victim model. For RT-AL, we re-initialize the classification head and then update the entire network. We train LL, AL, and RT-AL for 10, 25, and 50 epochs, respectively, across all dataset-architecture settings. Depending on the dataset and architecture, SGD is used for CNNs and AdamW is used for DeiT and Swin-T, with learning rates between 10^{-4} and 10^{-2}, batch size 128, and standard weight decay.

Table 5: Overview of model-stealing configurations

Attack	Victim Access	Training Data	Architecture
LL Fine-tuning	Parameters	Non-private	Same
AL Fine-tuning	Parameters	Non-private	Same
RT-AL Fine-tuning	Parameters	Non-private	Same
Pruning	Parameters	Non-private	Same
KD	Predictions	Non-private	Same / Cross
DKD	Predictions	Non-private	Same / Cross
Knockoff	Queries	Non-private query data	Same / Cross
Data-free stealing	Hard labels	Proxy / generated data	Same / Cross

Table 6: Cross-architecture stealing configurations

Attack	Victim architecture	Suspect architecture
KD DKD Knockoff Data-free	ResNet-18/34 VGG16/19 DeiT-Tiny	ResNet-18 VGG16 DeiT-Tiny

Pruning. We construct pruned suspects from the victim models using pruning ratios of 20% to 80%. We apply global unstructured pruning to all convolutional and linear layers, excluding the DeiT patch-embedding projection. Each pruned model is evaluated after accuracy-restoring AL fine-tuning on the non-private data for 50 epochs. The pruning criterion is the absolute magnitude of individual weights, ranked globally across all eligible layers.

We evaluate standard knowledge distillation (KD) and decoupled knowledge distillation (DKD).

KD. For standard KD, the victim serves as the teacher and the suspect as the student. The student is trained on the non-private data using

$$\mathcal{L}_{\mathrm{KD}} = (1-\lambda)\,\mathcal{L}_{\mathrm{CE}}(z_s, y) + \lambda\, T^2\, \mathrm{KL}(\sigma(z_t/T) \,\|\, \sigma(z_s/T))\,,$$

where $T = 4$ and $\lambda = 0.9$. Students are trained from scratch on non-private data using the same architecture-specific training protocol as the independently trained models (SGD for CNNs and AdamW for DeiT), with batch size 256. We train for 200 epochs on CIFAR-10 and 260-400 epochs on CIFAR-100, depending on the student architecture.

DKD. Following Zhao et al. (2022), we decouple logit distillation into target-class knowledge distillation (TCKD) and non-target-class knowledge distillation (NCKD):

$$\mathcal{L}_{\mathrm{DKD}} = \alpha\tau^2 D_{\mathrm{KL}}(\mathbf{q}^{\mathrm{T}} \| \mathbf{q}^{\mathrm{S}}) + \beta\tau^2 D_{\mathrm{KL}}(\widetilde{\mathbf{p}}^{\mathrm{T}} \| \widetilde{\mathbf{p}}^{\mathrm{S}}).$$

Here, $\mathbf{q}$ represents the target-versus-rest distribution, while $\widetilde{\mathbf{p}}$ is the distribution over non-target classes. We use $\alpha = 1$, $\beta = 8$, and $\tau = 4$, with a 20-epoch linear warm-up for the DKD term and unit weight for the supervised cross-entropy loss.

Knockoff. We implement the random-query variant of Knockoff Nets (Orekondy et al., 2019), setting the query budget equal to the number of training samples of the victim model. The attacker submits clean images drawn from a disjoint in-distribution auxiliary set, and obtains the victim's full probability vectors as supervision. The extracted model is trained from scratch by minimizing the KL divergence to these predictions, using the corresponding architecture-specific training configuration for 160-400 epochs.

Data-free model stealing. We implement the DFMS-HL framework (Sanyal et al., 2022) following their published official implementation settings, with necessary adaptations to our proxy-data and model configurations. In this case, the attacker has no access to the victim's private data and observes only its top-1 predictions. Using unrelated public images as a weak prior, the generator and extracted model are trained alternately with a total query budget of approximately 8.1 million.

Table 7: Model extraction configurations

Attack	Query input → victim response	Queries	Epochs
Knockoff	Non-private data → probability vectors	25K	160-400
Data-free	Proxy/generated data → hard labels	≈ 8.1M	800 (alternating)

Additional extraction attack-specific settings are summarized in Table 7. When cross-architecture stealing is evaluated, the victim and suspect model architecture combinations follow Table 6. For cross-architecture stealing, the victim and suspect architectures are selected from the corresponding sets with different architectures, i.e., $a_{\text{victim}} \neq a_{\text{suspect}}$.

C.4 Adversarial-Training Evasion

We consider two adversarial-training strategies designed to alter the stolen model after or during model stealing.

Post-stealing adversarial training. We first obtain a stolen model using the corresponding stealing procedure and subsequently apply adversarial training using the attacker's available non-private data. We use ℓ_∞ PGD adversarial training with perturbation radius $\epsilon \in \{2/255, 4/255, 8/255\}$, step size $\epsilon/4$, and 10 attack steps. The model is adversarially trained for 30 additional epochs.

Stealing-time adversarial training. We additionally consider an adaptive attacker that incorporates an adversarial-training objective directly into the stealing procedure. Let $q_i = p_V(x_i)$ denote the probability vector returned by the victim for a clean query x_i, and let $p_\theta(x_i)$ denote the corresponding prediction of the stolen model. The training objective is

$$\mathcal{L}_{\text{steal+adv}} = \mathcal{L}_{\text{steal}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}},$$

where $\mathcal{L}_{\text{steal}}$ encourages the stolen model to reproduce the victim's predictions on the original query inputs, while $\mathcal{L}_{\text{adv}}$ encourages it to preserve the same predictions under adversarial perturbations. For example, in our soft-label Knockoff setting, the standard stealing loss is

$$\mathcal{L}_{\text{steal}} = \frac{1}{m} \sum_{i=1}^{m} D_{\text{KL}}(q_i \,\|\, p_\theta(x_i)) \,,$$

where m is the minibatch size. For each query, we generate an adversarial example against the current stolen model,

$$x_i^{\text{adv}} \approx \underset{\|x' - x_i\|_\infty \le \epsilon}{\arg\max} \; \text{CE}(p_\theta(x'), \hat{y}_i) \,, \qquad \hat{y}_i = \arg\max_c q_{i,c},$$

and define the adversarial term as

$$\mathcal{L}_{\text{adv}} = \frac{1}{m} \sum_{i=1}^{m} D_{\text{KL}}\big(q_i \,\|\, p_\theta(x_i^{\text{adv}})\big) \,.$$

Thus, the same victim prediction q_i supervises both the clean and adversarial inputs. No additional victim query is issued for x_i^{adv}, so adversarial training does not increase the stealing query budget. We set $\lambda_{\text{adv}} = 1$ and generate x_i^{adv} using ℓ_∞ PGD with the same perturbation parameters as above.

To account for training-regime uncertainty, we construct multiple independent-model reference pools covering both standard and adversarial-training configurations. We compute the auditing score against each reference pool separately and take the maximum p-value, such that a small final score requires the suspect to be atypical of all considered independent-model populations.

C.5 Auditing Configuration

Query set. Unless otherwise specified, we use the full private training set D_v as the auditing query set. The same sampled inputs are used to query the victim, suspect, and all reference models within an auditing instance. The effect of query-set size is evaluated separately using ratios

$$\rho \in \{5\%, 10\%, 20\%, 50\%, 75\%, 100\%\}$$

Information estimation. We estimate $I(X;T)$ and $I(T;Y)$ using the binning-based estimator. Unless otherwise specified, each output dimension is discretized using 50 bins. The same bin boundaries are applied across the victim, suspect, and reference models within each comparison. We additionally evaluate bin counts

$$B \in \{5, 10, 15, 20, 30, 50, 75, 100, 150, 200\}$$

to examine sensitivity to the discretization resolution.

Reference models. For each dataset-architecture setting, up to 30 independently trained models are available as references. Unless otherwise specified, the main experiments use $k = 15$ reference models. We evaluate the effect of the reference-set size using

$$k \in \{5, 10, 15, 20, 25, 30\}.$$

For known-architecture experiments, references are drawn from the pool corresponding to the suspect architecture. For unknown-architecture evaluation, which serves as our primary overall auditing setting, we construct dataset-specific candidate architecture sets. For the CIFAR experiments, we consider

$$\mathcal{N}_{\mathrm{CIFAR}} = \{\text{ResNet-18}, \text{VGG16}, \text{DeiT-Tiny}\},$$

while for ImageNet-100, we consider

$$\mathcal{N}_{\mathrm{IN100}} = \{\text{DeiT}, \text{Swin-T}\}.$$

We construct a separate independent-model reference pool for each candidate architecture and combine the resulting second-stage values.

Auditing score and decision threshold. The quantitative auditing score is used as the primary output. For reporting classification-based evaluation metrics, we use a significance level of

$$\alpha = 0.01.$$

A suspect satisfying the corresponding auditing criterion is counted as detected for the purpose of detection rate. The significance threshold is used only for evaluation and does not change the interpretation of the auditing score as the primary quantitative output.

C.6 Evaluation Metrics

For each stealing configuration, we evaluate 50 positive suspects and 50 negative suspects. We report the detection rate separately for the two groups, defined as the fraction of suspects whose model type is correctly identified. For positive suspects, the detection rate is

$$\mathrm{DR}_{\mathrm{pos}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}},$$

while for negative suspects, it is

$$\mathrm{DR}_{\mathrm{neg}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}.$$

Therefore, a 100% detection rate for both positive and negative suspects indicates that all stolen models are detected and all independently trained models are correctly identified, corresponding to $\mathrm{FN} = 0$ and $\mathrm{FP} = 0$.

C.7 Baseline Methods

We compare our method with four representative non-intrusive model fingerprinting methods. All baselines are evaluated under the same victim and suspect models used in our experiments.

IPGuard. IPGuard fingerprints a victim model through samples located near its classification boundary. These boundary samples are queried on both the victim and suspect models, and their prediction consistency is used to determine whether the suspect model is derived from the victim. Therefore, its auditing signal primarily relies on the preservation of victim-specific decision-boundary behavior in the suspect model.

Table 8: Baseline overall auditing performance on CIFAR-10. Each entry denotes detection rates (%) of IPGuard, DeepJudge, DI, and ADV-TRA, respectively

Model Type		RN18-CF10	VG16-CF10	DeiT-CF10
Positive Suspects	FT-LL	100 / 100 / 100 / 100	100 / 100 / 100 / 100	100 / 100 / 100 / 100
	FT-AL	100 / 100 / 100 / 100	100 / 100 / 100 / 100	100 / 100 / 100 / 100
	RT-AL	92 / 93 / 100 / 100	90 / 89 / 100 / 100	91 / 90 / 100 / 100
	P-20%	100 / 100 / 100 / 100	100 / 100 / 100 / 100	100 / 100 / 100 / 100
	P-80%	100 / 100 / 100 / 100	100 / 100 / 100 / 100	100 / 100 / 100 / 100
	KD	82 / 85 / 86 / 83	83 / 86 / 84 / 87	80 / 89 / 81 / 81
	DKD	83 / 84 / 82 / 81	82 / 81 / 80 / 79	81 / 81 / 82 / 84
	Knockoff	0 / 23 / 89 / 0	0 / 20 / 75 / 0	0 / 24 / 72 / 0
	HL	0 / 17 / 85 / 0	0 / 21 / 77 / 0	0 / 19 / 71 / 0
Negative Suspects		97 / 95 / 99 / 98	96 / 96 / 98 / 99	95 / 95 / 97 / 96

Table 9: The overall detection rate (%) for both positive and negative suspects on CIFAR-100

Model Type		RN18-CF100	VG16-CF100	DeiT-CF100
Positive Suspects	FT-LL	100 (5.89×10^{-21})	100 (5.93×10^{-21})	100 (1.33×10^{-16})
	FT-AL	100 (1.37×10^{-20})	100 (3.50×10^{-18})	100 (2.87×10^{-20})
	RT-AL	100 (9.58×10^{-18})	100 (2.20×10^{-16})	100 (4.59×10^{-18})
	P-20%	100 (1.49×10^{-20})	100 (1.50×10^{-19})	100 (1.92×10^{-18})
	P-80%	100 (1.98×10^{-19})	100 (6.80×10^{-19})	100 (4.74×10^{-11})
	KD	100 (1.34×10^{-13})	100 (1.15×10^{-14})	100 (9.78×10^{-13})
	DKD	100 (1.04×10^{-13})	100 (1.14×10^{-14})	100 (3.43×10^{-12})
	Knockoff	100 (8.28×10^{-11})	100 (1.61×10^{-11})	100 (3.21×10^{-14})
	HL	100 (1.73×10^{-13})	100 (2.75×10^{-14})	100 (6.39×10^{-16})
Negative Suspects		100 (0.57)	100 (0.57)	100 (0.55)

DeepJudge. DeepJudge is a testing-based framework that measures behavioral similarity between the victim and suspect models using specially generated test cases. Under the black-box setting considered in our work, DeepJudge compares their output behaviors through black-box testing metrics, including output-distribution and robustness-related discrepancies. A suspect model exhibiting sufficiently similar behaviors to the victim is regarded as a potential stolen copy.

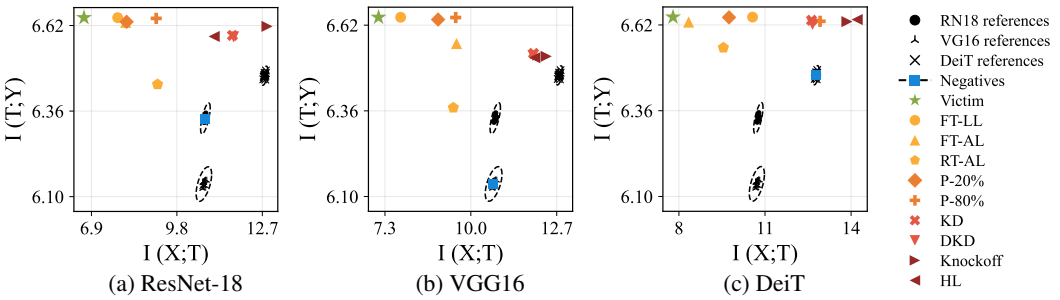
ADV-TRA. ADV-TRA constructs model fingerprints using adversarial trajectories rather than individual boundary samples. Specifically, it generates sequences of progressively perturbed adversarial examples around the victim's decision surface and evaluates whether the suspect model exhibits consistent behaviors along these trajectories. The trajectory design improves tolerance to moderate decision-boundary changes while still relying on behavioral patterns inherited from the victim model.

Dataset Inference (DI). Dataset Inference identifies model derivation by exploiting the victim's private training data. Its key observation is that a stolen model may inherit knowledge associated with the victim's training samples. Accordingly, DI estimates the distance of private training samples to the suspect model's decision boundary and statistically compares their boundary characteristics against non-training samples. We adopt DI as a representative data-dependent non-intrusive baseline in our experiments.

D Experimental Results

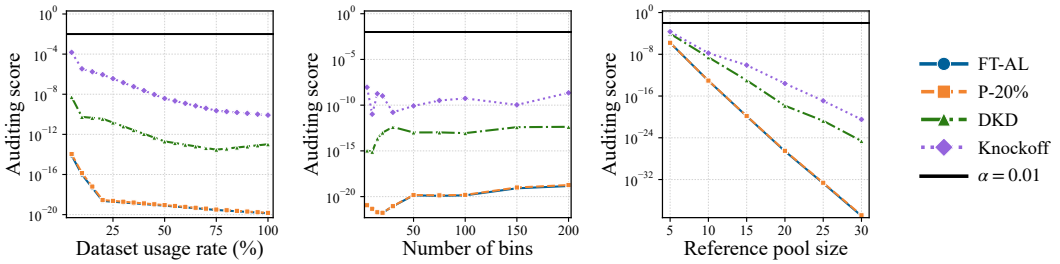
D.1 Overall Performance

Here we exhibit the overall auditing performance under different datasets and model structures. Multiple reference pools with different model architectures are employed in the second stage for



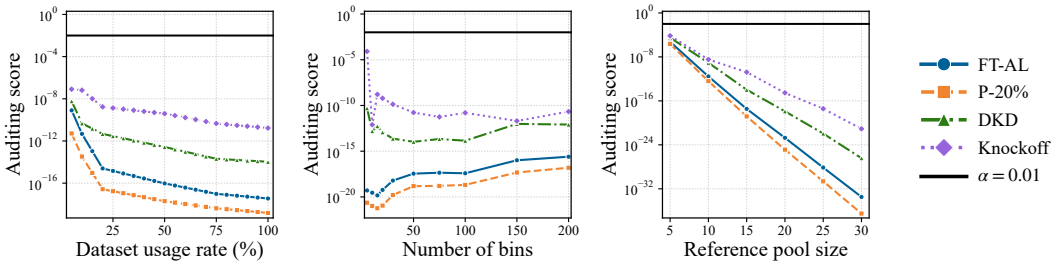
(a) ResNet-18 (b) VGG16 (c) DeiT

Figure 9: Information-plane visualization under unknown suspect architecture on CIFAR-100.



(a) Dataset usage rate (b) Number of bins (c) Reference pool size

Figure 10: Effects of the number of query data, bins used for information estimation, and the independent models on the auditing score on CIFAR-100 with ResNet-18.



(a) Dataset usage rate (b) Number of bins (c) Reference pool size

Figure 11: Effects of the number of query data, bins used for information estimation, and the independent models on the auditing score on CIFAR-100 with VGG16.

testing against independent models. Each reference pool characterizes the independent-model distribution under a different candidate architecture, allowing the auditor to account for architecture uncertainty without assuming prior knowledge of the suspect architecture. Table 9 exhibits the detection rate on the CIFAR-100 dataset. We can observe that our framework correctly detects all 50 stolen suspects across all evaluated stealing attacks for the three model architectures.

Moreover, different models on the information plane under different datasets and model structures are also provided in Figure 9. We can also witness the information-bottleneck deterioration pattern of all positive suspects. Besides, these stolen models remain separated from the independent-model reference regions of all three architectures, whereas negative suspects align with their corresponding references.

Finally, regarding the sensitivity analysis, comprehensive results are shown in Figures 10 and 11. We can observe that increasing the fraction of D_v used for auditing consistently leads to smaller auditing scores. Besides, all evaluated stealing methods remain consistently below the significance level $\alpha = 0.01$ under different numbers of bins. Increasing the independent model size also helps decrease auditing scores on the stolen model.

Table 10: Detection results for different model auditing baselines under model distillation. Each entry reports the detection accuracy for **KD / DKD**, respectively. All results are reported as percentages

	Distillation		
	RN34-CF10	**VG19-CF10**	**DeiT-CF10**
	RN18	**VG16**	**DeiT**
IPGuard	0 / 0	0 / 0	0 / 0
DeepJudge	0 / 0	0 / 0	0 / 0
DI	82 / 84	81 / 81	83 / 80
ADV-TRA	0 / 0	0 / 0	0 / 0
Ours	**100 / 100**	**100 / 100**	**100 / 100**
	RN34-CF100	**VG19-CF100**	**DeiT-CF100**
	RN18	**VG16**	**DeiT**
IPGuard	0 / 0	0 / 0	0 / 0
DeepJudge	0 / 0	0 / 0	0 / 0
DI	79 / 74	78 / 80	75 / 79
ADV-TRA	0 / 0	0 / 0	0 / 0
Ours	**100 / 100**	**100 / 100**	**100 / 100**

Table 11: Cross-architecture detection results for different model auditing baselines. Each entry reports the detection accuracy for **Knockoff / DFMS-HL**, respectively. All results are reported as percentages

	RN18-CF10		**VG16-CF10**		**DeiT-CF10**	
	VG16	**DeiT**	**RN18**	**DeiT**	**RN18**	**VG16**
IPGuard	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DeepJudge	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DI	74 / 76	70 / 72	75 / 77	71 / 69	73 / 72	71 / 71
ADV-TRA	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
Ours	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**
	RN18-CF100		**VG16-CF100**		**DeiT-CF100**	
	VG16	**DeiT**	**RN18**	**DeiT**	**RN18**	**VG16**
IPGuard	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DeepJudge	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
DI	67 / 69	64 / 66	68 / 70	65 / 63	66 / 65	64 / 63
ADV-TRA	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0	0 / 0
Ours	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**	**100 / 100**

D.2 Cross Architecture

We further evaluate cross-architecture stealing under both model extraction (Knockoff and HL) and knowledge distillation (KD and DKD). Tables 10 and 11 demonstrate that our method achieves 100% detection rates across all evaluated configurations. In contrast, IPGuard, DeepJudge, and ADV-TRA consistently yield 0% detection rates, while DI also exhibits a decreased detection rate. These results support our motivation that fingerprinting methods relying on preserved similarity between the victim and stolen models can be fragile under cross-architecture stealing.

Table 12: Detection rate (%) under post-stealing adversarial training on RN18-CF10 across different perturbation budgets

$\epsilon = 2/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	52	41	16	21	13	10
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	3.42	3.12	3.39	3.35	3.95	3.88
$\epsilon = 4/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	43	37	9	11	3	2
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	7.18	6.37	7.12	7.04	8.38	8.21
$\epsilon = 8/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	3	2	0	0	0	0
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	12.24	11.35	12.18	12.05	13.42	13.25

D.3 Adversarial Evasion

Tables 12, 13, 14, 15, and 16 report the detection rates on different models and datasets under PGD-10 adversarial training with different perturbation budgets under post-stealing adversarial evasion. IPGuard, DeepJudge, and ADV-TRA yield 0% detection rates under both strategies, while DI is also substantially degraded. In contrast, our method consistently achieves 100% detection rates, demonstrating that our auditing signal remains robust to adversarial evasion.

Table 13: Detection rate (%) under post-stealing adversarial training on VG16-CF10 across different perturbation budgets

$\epsilon = 2/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	48	39	18	22	11	8
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	3.45	3.18	3.38	3.32	3.91	3.85
$\epsilon = 4/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	40	34	10	12	4	4
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	7.35	7.05	7.28	7.22	7.88	7.82
$\epsilon = 8/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	4	2	1	0	0	0
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	12.38	11.42	12.25	12.12	13.35	13.18

Table 14: Detection rate (%) under post-stealing adversarial training on DeiT-CF10 across different perturbation budgets

$\epsilon = 2/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	41	32	11	15	8	6
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	3.72	3.45	3.68	3.62	3.98	3.94
$\epsilon = 4/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	29	23	5	7	2	1
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	7.65	7.32	7.58	7.51	7.98	8.02
$\epsilon = 8/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	1	1	0	0	0	0
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	12.85	13.92	12.74	12.65	13.48	14.36

Table 15: Detection rate (%) under post-stealing adversarial training on RN18-CF100 across different perturbation budgets

$\epsilon = 2/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	38	29	12	15	7	8
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	7.45	7.12	7.38	7.32	7.75	8.24
$\epsilon = 4/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	24	18	5	6	1	1
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	11.48	12.15	11.42	11.36	12.82	12.65
$\epsilon = 8/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	1	0	0	0	0	0
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	15.25	16.68	15.18	15.12	16.42	16.35

Table 16: Detection rate (%) under post-stealing adversarial training on VG16-CF100 across different perturbation budgets

$\epsilon = 2/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	34	25	10	13	5	4
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	7.62	8.25	7.54	7.48	8.42	9.36
$\epsilon = 4/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	21	15	4	5	1	0
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	12.75	13.78	12.65	12.54	12.92	13.28
$\epsilon = 8/255$	**FT-AL**	**P-20%**	**KD**	**DKD**	**Knockoff**	**HL**
IPGuard	0	0	0	0	0	0
DeepJudge	0	0	0	0	0	0
DI	0	0	0	0	0	0
ADV-TRA	0	0	0	0	0	0
Ours	**100**	**100**	**100**	**100**	**100**	**100**
Clean Acc. Drop	15.58	15.87	15.06	15.18	16.48	17.29

Table 17: Detection rate (%) under cross-architecture **Knockoff** and **DKD** after adversarial evasion with perturbation budget $\epsilon = 2/255$

	Knockoff				**DKD**	
Victim	**RN18-CF10**		**DeiT-CF10**		**RN34-CF10**	**VG19-CF10**
Suspect	**VG16**	**DeiT**	**RN18**	**VG16**	**RN18**	**VG16**
	100	100	100	100	100	100
p-value	2.68×10^{-26}	3.95×10^{-27}	7.54×10^{-26}	7.65×10^{-26}	2.11×10^{-26}	3.49×10^{-26}
Clean Acc. Drop	3.18	4.08	3.23	4.34	3.33	3.75

Table 18: Detection rate (%) under cross-architecture **Knockoff** and **DKD** after adversarial evasion with perturbation budget $\epsilon = 4/255$

	Knockoff				**DKD**	
Victim	**RN18-CF10**		**DeiT-CF10**		**RN34-CF10**	**VG19-CF10**
Suspect	**VG16**	**DeiT**	**RN18**	**VG16**	**RN18**	**VG16**
	100	100	100	100	100	100
p-value	3.85×10^{-37}	8.12×10^{-38}	5.46×10^{-37}	7.29×10^{-37}	1.65×10^{-36}	2.94×10^{-36}
Clean Acc. Drop	7.33	8.08	7.44	8.82	7.70	7.62

Table 19: Detection rate (%) under cross-architecture **Knockoff** and **DKD** after adversarial evasion with perturbation budget $\epsilon = 8/255$

	Knockoff				**DKD**	
Victim	**RN18-CF10**		**DeiT-CF10**		**RN34-CF10**	**VG19-CF10**
Suspect	**VG16**	**DeiT**	**RN18**	**VG16**	**RN18**	**VG16**
	100	100	100	100	100	100
p-value	4.25×10^{-48}	7.80×10^{-49}	6.15×10^{-48}	8.42×10^{-48}	1.95×10^{-47}	3.12×10^{-47}
Clean Acc. Drop	12.05	11.48	12.30	13.15	12.89	13.71

Table 20: Detection rate (%) for sequential Knockoff extraction. Each entry reports the detection rate followed by the corresponding p-value in parentheses

Victim	**RN18-CF10**	
	Same Dataset	**Different Dataset**
VG16 → DeiT	100 (5.76×10^{-06})	100 (3.16×10^{-15})
DeiT → VG16	100 (4.57×10^{-12})	100 (7.12×10^{-23})

D.4 Cross Architecture with Adversarial Evasion

Tables 17, 18, and 19 exhibit auditing performance under settings that combine cross-architecture stealing with adversarial evasion. We can observe that our method maintains effective auditing under this combined setting.

D.5 Sequential Knockoff Extraction

To examine whether auditing evidence can nevertheless persist under multiple extraction attacks, we perform sequential Knockoff extraction. As shown in Table 20, our method can successfully detect all evaluated stolen models.