FARE: Forensic Acceptance Region Estimation for Catching Bait-and-Switch Image Generators

Kai Yao
School of Informatics
The University of Edinburgh
kai.yao@ed.ac.uk

Marc Juarez
School of Informatics
The University of Edinburgh
marc.juarez@ed.ac.uk

Abstract

Modern AI image generators are increasingly deployed as opaque APIs, where customers can query the deployed service, but cannot inspect model weights or architecture. This creates a practical challenge: a provider may pass governance certification with one generator and later silently switch to a cheaper and lower-quality one for deployment, compromising public trust or even safety in high-stakes domains. We study *integrity auditing* at deployment time and propose **FARE** (Forensic Acceptance Region Estimation). A certified generator is enrolled by training FARE on images sampled from that generator. After deployment, FARE can determine whether a generated image is consistent with the enrolled generator—using *only* that image. FARE's features are based on image generator-specific artifacts that have been proposed for forensic applications. FARE amplifies these features during training by finding hard samples that tighten the acceptance region and increase sensitivity to subtle changes in the certified generator. Across generator swaps, including substitutions with similar model versions and model variants, FARE is effective at detecting swaps, consistently outperforming existing baselines at strict operating points, and remains effective under the black-box and white-box attacks evaluated in this work.

1 Introduction

Generative AI is not only widely used by the public, but it is also increasingly considered in safety-critical domains, such as defense [1] and healthcare [2]. Regulations are emerging to prevent harmful generators from undermining public trust, causing societal harm, or creating security risks. For example, the EU's AI Act establishes conformity-assessment requirements for high-risk AI systems [3]. This requirement raises a key question: *how can external parties, such as auditors or customers, verify that an output from a deployed service truly came from the certified generator?* Even after passing an audit, a provider may replace the certified generator with a cheaper or flawed alternative. This is especially relevant because commercial image generators are often accessed through remote APIs (e.g., [4, 5]), where external parties can query the service and observe outputs but typically cannot inspect model weights, training data, or other ML pipeline components.

This paper addresses this question as an *integrity auditing* problem using only generator outputs. We formalize it in two phases: *enrollment* and *verification*. During *enrollment*, an auditor certifies a generator under an NDA-like legal agreement and enrolls it into FARE by sampling images. This phase follows governance certification, assumes provider collaboration, and is codified in a *contract* specifying the model checkpoint and inference pipeline settings. Models deviating from the contract, e.g., lower-quality models, are non-certified. During *verification*, the auditor receives only a queried image from the deployed generator and decides whether it is consistent with the certified generator. Here, *output-only* means enrollment and calibration use only images sampled from the certified generator: no auxiliary natural-image negatives, no images from other generators, and no access to

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

generator weights. We focus on the harder *single-image verification* setting over batch verification because deployments often require high-confidence decisions on individual queries, and adversaries could otherwise dilute detection signals by mixing certified and non-certified outputs in a batch.

Why is this setting difficult for an auditor? First, many economically motivated changes are subtle: providers may deploy a closely related checkpoint or lightly modified pipeline whose images look similar to end users. Second, an output-only verifier must rely solely on generator-specific artifacts inferred from the image itself [6]. Third, a malicious provider can adapt its deployment to evade a fixed detector via adversarial manipulations [7].

To address these challenges, we introduce **FARE (Forensic Acceptance Region Estimation)**, a verifier trained using *only* images from the certified generator. FARE learns an acceptance region for certified outputs in a feature space derived from image artifacts, motivated by forensic evidence that generator artifacts can support source attribution [6]. FARE scores image patches locally and uses the most suspicious patch scores to capture spatially sparse artifacts. To meet low-FPR requirements, FARE augments training data with certified-derived "hard" examples that reflect contract violations near the decision boundary. At verification, FARE aggregates the most suspicious patch scores into a single image score for the final decision. Our evaluation shows that FARE outperforms compatible baselines across substitution scenarios and remains effective under the evaluated exact-model and decision-only attacks.

In summary, our contributions are:

(i) Post-certification auditing under a contract. We formalize output-only verification of an image generation deployment after certification, where the audit contract covers the model checkpoint and inference pipeline.

(ii) FARE: an output-only verifier. We present a protocol-compatible method that learns a tight acceptance region from only certified outputs using "forensic" features, local scoring, and certified-derived hard examples.

(iii) Strict-FPR evaluation and batch auditing. We evaluate model swaps, model-version swaps, cost-motivated inference variants, and exact-model and decision-only attacks at strict operating points. FARE performs strongly in the primary single-image setting at 1% FPR; at stricter 0.5% and 0.1% budgets, we quantify single-image degradation and show that small-batch verification largely recovers performance.

2 Problem Statement and Threat Model

We study *post-certification auditing* of black-box image-generation APIs as a security game between a service provider and an auditor. The auditor observes only output images and must decide whether deployment-time outputs remain compliant with what was certified.

2.1 Two-phase Auditing: Enrollment vs. Verification

We distinguish a model checkpoint M from an inference pipeline Π (e.g., number of denoising steps, post-processing policy). A typical deployed generator instance is the pair $G = (M, \Pi)$, which induces an output distribution P_G.

Enrollment phase (t_0). The provider exposes the certified generator $G_{\text{cert}} = (M_{\text{cert}}, \Pi_{\text{cert}})$ for audit. The auditor has black-box query access to G_{cert} and collects N certified outputs $\mathcal{D}_{\text{cert}} = \{x_i\}_{i=1}^{N}, \quad x_i \sim P_{G_{\text{cert}}}$. Using only these outputs, the auditor trains a verifier $V(\cdot)$. We assume enrollment-time access is honest, i.e., all responses are generated by G_{cert}. An enrollment *contract* specifies the set of allowed deployments $\mathcal{C} \subseteq \{(M, \Pi)\}$. In this work, we evaluate a strict singleton contract, $\mathcal{C} = \{G_{\text{cert}}\}$, so any undeclared change to the checkpoint M or the pipeline Π is treated as non-certified. More permissive contracts could include pre-agreed pipeline changes, but we do not evaluate them. Appendix A instantiates the strict contract for the generator-swap benchmarks.

Verification phase (t_1). After deployment, the provider may serve outputs from some deployment $G_{\text{dep}} = (M_{\text{dep}}, \Pi_{\text{dep}})$, inducing distribution $P_{G_{\text{dep}}}$. Given only a claimed API output image x, the verifier decides whether x is consistent with the certified deployment, i.e., whether x is plausibly drawn from $P_{G_{\text{cert}}}$ under the agreed contract.

Auditor's goal. The auditor trains only on samples from G_{cert} and aims to reject images produced by a broad, unseen set of plausible non-certified deployments $G_{\text{dep}} \notin \mathcal{C}$ (e.g., model swaps $M_{\text{dep}} \neq M_{\text{cert}}$, and pipeline changes $\Pi_{\text{dep}} \neq \Pi_{\text{cert}}$). We do not aim for universal rejection over arbitrary images.

2.2 Decision Rule and Compliance Operating Point

Following anomaly detection, we treat "non-certified" as the positive class and use a scalar anomaly score $s(\cdot) \in \mathbb{R}$, where larger values are more suspicious. Given an image x, the verifier accepts if

$$V(x) = \mathbb{I}[s(x) \leq \tau_\alpha], \qquad \tau_\alpha = \text{Quantile}_{1-\alpha}\big(\{s(x_i^{\text{cal}})\}_{i=1}^m\big). \tag{1}$$

Here $\{x_i^{\text{cal}}\}_{i=1}^m$ is an independent held-out certified calibration set and α is the target certified-image FPR. Appendix E.2 specifies the order-statistic threshold and its finite-sample uncertainty. We use 1% as the primary single-image operating point and also study stricter budgets such as 0.5% and 0.1%, which better fit real-world model auditing.

2.3 Threat Model

Goals. The provider is considered malicious only after enrollment. To reduce cost, it may replace G_{cert} with a cheaper, lower-fidelity deployment, e.g., a lower-tier model, aggressive compression, or altered inference settings. Such undeclared changes can alter output behavior and violate the contract. An active adversary seeks *targeted acceptance*: producing images from a non-certified deployment $G_{\text{dep}} \neq G_{\text{cert}}$ that the verifier falsely accepts as certified. This resembles forging forensic features onto an image so it is wrongly attributed to the certified model [8]. FARE checks whether an output is consistent with the enrollment contract; it does not infer whether a deviation is benign or malicious. We do *not* claim cryptographic unforgeability.

Knowledge and Capabilities. After enrollment, the provider controls deployment and may mimic the certified distribution while serving substituted outputs. In deployment, it has black-box, decision-only verifier access under a query budget, with no access to verifier internals or gradients. We evaluate this capability with HopSkipJumpAttack (HSJA) [9]. We also use exact-model white-box PGD as a high-information stress test in which the attacker knows the verifier's parameters and gradients. This additional access exceeds the deployed threat model, but a particular white-box optimizer is not a universal upper bound on other attacks. We do not evaluate surrogate-transfer, generation-pipeline, or cross-detector attacks.

3 Related Work

Model Provenance Techniques. Watermarking and fingerprinting can attribute image provenance to a source model. Watermarking embeds identifiable signatures post-generation (e.g., [10, 11]) or directly during generation (e.g., [12–14]). However, watermarking assumes a cooperative provider. If that provider controls the embedding mechanism or key, it can apply the certified watermark to outputs from another generator; watermark presence alone therefore does not establish that the certified generator was used. Passive image forensics instead exploits intrinsic traces left by generators as byproducts of generation, including residual artifacts [6, 15] and frequency cues [16–18]. Although these features are related to FARE's, prior source-attribution studies typically assume a closed world of possible generators, unlike the open-world setting we evaluate here. Open-world attribution methods such as Girish et al. [19] learn from multiple known sources, unlike our single-source enrollment setting. Recent adversarial evaluations also show that fingerprinting is vulnerable to removal and forgery [8].

One-Class Verification Methods. We formulate open-world attribution to a single source generator as one-class classification. OCC-CLIP [20] studies few-shot one-class origin attribution with prompt-tuned CLIP, but its full prompt-tuning setup builds a target class from source-model images and a non-target class from auxiliary clean/open-domain images, violating our certified-only enrollment protocol. We therefore include a protocol-compatible CLIP baseline, CLIP + OC-SVM, and exclude full prompt-tuned OCC-CLIP from the evaluation. FLIPAD [21] is also incompatible because it requires white-box generator access. Classical one-class models, such as One-Class SVM [22] and Deep SVDD [23], are included as certified-only baselines under the same information constraints. DROCC [24] and DOC3 [25] motivate auxiliary examples for shaping one-class decision boundaries.

Rather than using them as fixed off-the-shelf baselines, we adapt them to our pipeline and ablate each resulting mechanism under identical training, calibration, and scoring conditions. To emphasize generator-specific traces over image content, image forensics uses architectural constraints. The Bayar–Stamm layer [26] makes convolutional filters to act as high-pass residual filters, highlighting noise-like artifacts rather than scene content. A patch-based formulation also connects to multiple instance learning, where image-level decisions are formed from local instances [27, 28].

4 FARE: Forensic Acceptance Region Estimation

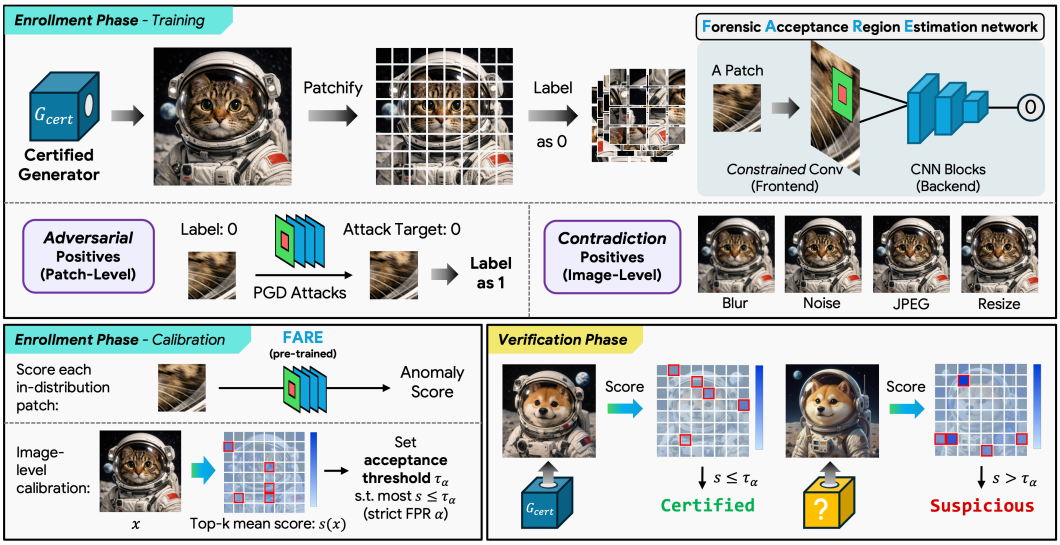

Figure 1: **FARE.** *Enrollment:* using only images from the certified generator, we train a patch verifier in a forensic feature space. Training uses certified patches as negatives and two certified-derived positive sources: (i) near-boundary adversarial positives and (ii) contradiction positives from contract-violating perturbations (then re-tiled into patches). We then calibrate an image-level threshold at target FPR using the same patch tiling and top-k aggregation. *Verification:* tile the queried image into patches, score anomalies, aggregate via top-k mean, and accept/reject via Eq. (1).

4.1 Overview

FARE trains an output-only verifier from images produced by the certified generator, then uses deployed API to audit by checking whether a queried image is consistent with the certified acceptance region (Fig. 1, Algs. C.1–C.2). Our primary goal is reliable *single-image* auditing at the 1% FPR operating point; when an application demands a lower FPR *and* can afford the operational overhead—namely, collecting multiple outputs from the same deployment and incurring additional query and latency costs—batch verification provides an optional tightening. Implementation-level architecture and training details are deferred to Appendix B.

Patch tiling. Given an RGB image $x \in \mathbb{R}^{H\times W\times 3}$, we first partition it into a grid of $d \times d$ square patches to facilitate the extraction of local forensic cues. While patch grids can in general be overlapping, in this work we use a non-overlapping tiling with stride d and assume H and W are divisible by d. For deployments with arbitrary output sizes, the contract should specify a deterministic resize, crop, or padding policy before tiling, and the same preprocessing must be used during enrollment, calibration, and verification.

Anomaly scores. We train a network on patches $f_\theta : \mathbb{R}^{d\times d\times 3} \to \mathbb{R}$ that outputs a logit $a(p) := f_\theta(p)$ where larger $a(p)$ means stronger evidence that p is a non-certified patch. Given $\mathcal{P}(x) = \{p_i\}_{i=1}^{|\mathcal{P}(x)|}$ and anomaly scores $\{a(p_i)\}$, we aggregate with top-k tail averaging. Let $a_{(1)} \ge \cdots \ge a_{(|\mathcal{P}(x)|)}$

denote patch anomaly scores sorted in descending order, the score of an image is then:

$$s(x) \;=\; \frac{1}{k}\sum_{j=1}^{k} a_{(j)}, \qquad 1 \le k \le |\mathcal{P}(x)|. \tag{2}$$

Top-k averaging is a compromise between full-image averaging, which can dilute spatially sparse forensic artifacts, and max pooling, which is overly sensitive to a single noisy patch at low FPR. Appendix D, Table D.2 ablates this choice. Finally, we accept/reject with the calibrated rule in Eq. (1).

4.2 Enrollment (t_0): Learning the Forensic Acceptance Region

FARE network with a constrained convolutional layer. The first layer of f_θ is a Bayar–Stamm constrained convolution [26] applied per RGB channel. For each channel-specific kernel $W^{(c)} \in \mathbb{R}^{m\times m}$ with center index (u_0, v_0), we enforce $W^{(c)}_{u_0,v_0} = -1, \sum_{(u,v)\neq(u_0,v_0)} W^{(c)}_{u,v} = 1$, so the kernel sum is 0 and the layer behaves as a learned prediction-error (high-pass) operator. This biases the representation toward residual forensic traces. After each optimizer step, we project $W^{(c)}$ back onto the constraint set. We implement the backend as standard CNN blocks.

Warm-up with negative samples only. A tight one-class classification boundary can be unstable early in training if hard positives are introduced before the "certified" direction is established. We therefore apply a warm-up stage: for the first T_{warm} iterations, training uses only certified negatives; afterwards we enable adversarial and contradiction positives, which we introduce below.

Learning objective. Let $\mathrm{BCEWithLogits}(z, y)$ denote binary cross-entropy with logits. Training uses images $x \sim P_{G_{\text{cert}}}$ from G_{cert} and their patches $p \in \mathcal{P}(x)$.

(i) Certified negatives: patches from G_{cert}.

$$\mathcal{L}_{\text{cert}} = \mathbb{E}_{x\sim P_{G_{\text{cert}}}} \frac{1}{|\mathcal{P}(x)|} \sum_{p\in\mathcal{P}(x)} \Big[\mathrm{BCEWithLogits}\big(f_\theta(p), 0\big)\Big]. \tag{3}$$

(ii) Near-boundary adversarial positives. To tighten the boundary around certified data and harden it locally, for each certified patch p, we synthesize a hard positive within an ℓ_2 shell: $\mathcal{S} = \big\{\delta \in \mathbb{R}^{d\times d\times 3} \;\big|\; r \le \|\delta\|_2 \le \gamma r\big\}$. Here, $r > 0$ is the inner radius, $\gamma > 1$ controls thickness. We seek the optimal perturbation $\delta^\star(p)$ that minimizes the certified loss:

$$\delta^\star(p) = \arg\min_{\delta\in\mathcal{S}} \mathrm{BCEWithLogits}\Big(f_\theta\big(\mathrm{clip}(p+\delta)\big), 0\Big), \qquad \tilde{p} = \mathrm{clip}\big(p + \delta^\star(p)\big), \tag{4}$$

where $\mathrm{clip}(\cdot)$ clamps to the valid pixel range (e.g., $[0, 1]$). We approximate $\delta^\star(p)$ via projected gradient descent (PGD), applying Euclidean projection $\Pi_{\mathcal{S}}$ onto the shell $\mathcal{S}$ at each iteration; reproducibility-oriented pseudocode is provided in Appendix C, Alg. C.2. This yields a hard positive $\tilde{p}$: we push p along the locally most certified-like direction, then inversely train $\tilde{p}$ as a positive, tightening the decision boundary in the neighborhood of every certified patch:

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{x\sim P_{G_{\text{cert}}}} \frac{1}{|\mathcal{P}(x)|} \sum_{p\in\mathcal{P}(x)} \Big[\mathrm{BCEWithLogits}\big(f_\theta(\tilde{p}), 1\big)\Big]. \tag{5}$$

(iii) Contradiction positives. Let $\mathcal{T}$ be a fixed set of contract-violating transforms. We use Gaussian blur, Gaussian noise, JPEG compression, and crop-and-resizing; exact settings are in Appendix B, Table B.1. Inspired by learning from contradictions [25], we use transformed certified samples as auxiliary boundary-tightening examples. They are not meant to model the non-certified deployments used at test time, nor the cost-motivated model changes in Table 2, such as quantization, pruning, or reduced diffusion steps. Instead, they act as certified-derived boundary-tightening perturbations encoding the strict contract assumption: unapproved changes to the output pipeline Π, including post-processing, should move samples away from the certified acceptance region. If a contract permits JPEG, resizing, or similar post-processing, that operation should be included in Π_{cert} and used consistently for enrollment, calibration, and verification. During training, we apply each $\varphi \in \mathcal{T}$ to form $x^{\text{con}} = \varphi(x)$, and treat re-tiled patches $p^{\text{con}} \in \mathcal{P}(x^{\text{con}})$ as positives:

$$\mathcal{L}_{\text{con}} = \mathbb{E}_{x\sim P_{G_{\text{cert}}}} \frac{1}{|\mathcal{T}|} \sum_{\varphi\in\mathcal{T}} \frac{1}{|\mathcal{P}(x^{\text{con}})|} \sum_{p^{\text{con}}\in\mathcal{P}(x^{\text{con}})} \Big[\mathrm{BCEWithLogits}\big(f_\theta(p^{\text{con}}), 1\big)\Big]. \tag{6}$$

The total learning objective to minimize after warm-up then becomes: $\mathcal{L} = \mathcal{L}_{\text{cert}} + \mu\,\mathcal{L}_{\text{adv}} + \lambda\,\mathcal{L}_{\text{con}}$. Here $\mu, \lambda \geq 0$ trade off adversarial and contradiction terms against the certified negative term.

Calibration at target FPR α. We compute $s(x)$ (Eq. (2)) on held-out images from G_{cert} post-training and set τ_α via Eq. (1). Calibration and verification use the same patch tiling and aggregation.

4.3 Verification (t_1): Detecting Generator Substitutions

Given a queried black-box API output image x, we (i) tile it into patches $\mathcal{P}(x)$, (ii) compute patch anomalies via a pre-trained FARE network, (iii) aggregate them into an image score $s(x)$ via Eq. (2), and (iv) accept/reject via Eq. (1). The full enrollment/training loop and PGD adversarial-patch routine are given as pseudocode in Appendix C, Algs. C.1–C.2.

Intuition: Why Does FARE Work? FARE's design is tightly aligned with our threat model: a constrained convolutional layer suppresses unrelated semantic content and highlights forensic traces, patch-level examination and top-k aggregation scoring localize spatially sparse artifacts, and certified-derived "hard" positives sharpen the one-class classification boundary against close substitutions and adversarially modified inputs. Together, these choices enhance sensitivity to generator substitutions that induce little semantic shifts. The empirical results in the following sections support this intuition.

5 Empirical Evaluation

Data splits and calibration. For each certified generator G_{cert}, we sample disjoint generated-image splits for training, calibration, and verification. Unless stated otherwise, we use 100,000 certified training images, 10,000 certified calibration images, and 10,000 verification images, as detailed in Appendix B, Table B.1. The calibration split is used only to set τ_α following Eq. (1); FPR/TPR values are computed on the held-out verification split. Appendix E.2 quantifies finite-sample uncertainty at the stricter operating points, and Appendix E.5 reports a training-set-size check. All task-specific fitting and calibration use *only* images from G_{cert}, with no auxiliary natural-image negatives, non-certified generator samples, or generator weights introduced during enrollment. Patch-based baselines share the same patch tiling $\mathcal{P}(x)$; their respective score aggregation rules are specified in Appendix A.2.

Baselines. We compare against single-source verifiers under the same *certified-only* enrollment protocol. Generic one-class baselines include **Deep SVDD** [23], **OC-SVM** [22], and **CLIP + OC-SVM** using frozen CLIP image embeddings [29]. We also adapt three existing forensic representations as attribution baselines: **DCT fingerprints** with an OC-SVM [30], and one-class models fitted to frozen **Forensic Self-Description (FSD)** [31] and **Neighboring Pixel Relationships (NPR)** features [32]. Task-specific fitting and calibration use only certified-generator outputs, with no natural-image or known non-certified samples introduced during enrollment; this excludes methods such as OCC-CLIP. Finally, **FARE ablations** remove components within the same patch-verification pipeline and compare against DROCC/DOC3-style boundary tightening [24, 25].

Evaluation scenarios. All scenarios verify at t_1 against non-certified deployments $G_{\text{dep}} \notin \mathcal{C}$ ($\mathcal{C}$ is the contract). We describe each scenario below; detailed case definitions are in Appendix A.

(i) Generator swaps (Table 1). We evaluate open-world rejection under cross-family swaps and within-family near-swaps. Cross-family uses different architecture/family pools, while within-family uses the same generator family or version lineage; Appendix A, Table A.1 gives exact cases. Cross-family cases include: (i) a pool of 12 generators pre-trained on FFHQ [33] (6 GANs, 3 VAEs, 3 diffusion), certifying each model in turn and testing against the remaining 11; and (ii) a Stable Diffusion (SD) [34] certified generator (SD1.4/1.5/2.0/2.1/3.5) [35, 36], evaluated against the diffusion-dominated CommunityForensics pool [37]. Within-family cases include: (i) StyleGAN2/3 training-configuration variants on FFHQ [38, 39]; and (ii) SD version swaps among the above checkpoints. For SD experiments, prompts are sourced from DiffusionDB [40], split disjointly across training, calibration, and verification, and matched between certified and non-certified deployments within each case. We also report a prompt-diversity check for SD in Appendix F, Table F.1.

(ii) Cost-motivated deployment variants (Table 2). We construct same-checkpoint variants motivated by possible resource savings. These include simulated weight compression through quantize–dequantize operations or dense weight pruning, and inference-time changes using fewer diffusion

Detection Method	Cross-Family Swaps		Within-Family Swaps	
	CF-FFHQ	CF-Diff	WF-GANTrain	WF-SDVer
	TPR@1%FPR	TPR@1%FPR	TPR@1%FPR	TPR@1%FPR
Deep SVDD	12.46±29.05	0.00±0.00	1.23±0.62	1.21±1.06
OC-SVM	6.89±15.51	0.00±0.00	0.76±0.49	2.79±4.31
CLIP + OC-SVM	13.91±31.48	0.00±0.00	1.34±0.52	1.73±1.66
DCT Fingerprint + OC-SVM	11.46±26.84	0.00±0.00	0.89±0.40	3.07±4.40
FSD	8.45±13.60	18.40±3.23	1.02±0.23	11.23±3.57
NPR + OC-SVM	10.92±15.15	4.60±1.12	0.77±0.31	18.32±5.19
FARE w/o Constraint	80.57±27.93	7.95±1.85	31.26±21.14	8.24±4.91
FARE w/o Patch	77.26±30.38	10.38±3.48	11.23±10.51	7.36±4.71
FARE w/o Adv	76.60±33.05	28.77±5.18	79.34±24.26	35.33±18.23
FARE w/o Contradiction	68.06±34.01	65.78±8.87	88.92±10.25	72.08±20.75
FARE w/o Adv, Contradiction	9.91±22.28	0.97±0.55	1.74±1.01	1.56±2.04
FARE (Full)	**99.43**±1.19	**99.45**±0.88	**96.38**±2.77	**92.56**±3.06
FARE (Full, under attack)	99.28±1.35	99.38±0.91	95.73±2.69	92.38±3.42
Conditional PGD ASR (%)	0.15	0.07	0.67	0.19

Table 1: Single-image swap detection performance under cross-family and within-family swapping scenarios, compared with baselines. Except for the ASR row, each entry reports TPR@1%FPR across all evaluated cases within the scenario (details in Appendix A). Cross-family covers CF-FFHQ (FFHQ-256 model pool) and CF-Diff (diffusion-dominated CommunityForensics pool). Within-family covers WF-GANTrain (StyleGAN training-configuration variants) and WF-SDVer (Stable Diffusion versioning). We also report performance under exact-model white-box PGD against the pretrained FARE network, with $\|\delta\|_\infty \leq 0.025$ and LPIPS < 0.05. The final row reports conditional ASR: the percentage of initially detected non-certified images changed to accepted under these constraints. All task-specific fitting and calibration use only outputs from the certified generator.

steps, across GAN (StyleGAN3 [39]), VAE (VDVAE [41]), and diffusion (NCSN++ [42], SD2.1 [35]) models. The simulated compression experiments measure detectability, not realized runtime or memory savings; Appendix E.6 reports measured resource use for reduced diffusion steps.

(iii) Single-image vs. batch verification (Table 3). Our primary aim is *single-image* auditing at a 1% FPR. For lower FPR budgets $(0.5\%, 0.1\%)$, we additionally report *batch verification* by aggregating per-image scores over $n \in \{5, 10\}$ images and calibrating batch thresholds on certified batches.

Adversarial forgery attacks. We evaluate exact-model white-box PGD [7] and decision-only HSJA [9]. PGD directly minimizes the image score $s(x')$ under an ℓ_∞ budget, while HSJA observes only the final binary decision. Appendix E.1 gives both protocols, attack success rates, and their scope.

Metrics. We report TPR (%) at fixed FPR α, where true positives are detected non-certified outputs and false positives are certified outputs flagged as non-certified. For batch verification with scores $s_i = s(x_i)$, we use $S_{\text{mean}} = \frac{1}{n}\sum_{i=1}^{n} s_i$ and $S_{\max} = \max_i s_i$, rejecting when $S_{\text{mean}} > \tau^{\text{mean}}_{n,\alpha}$ or $S_{\max} > \tau^{\max}_{n,\alpha}$. The thresholds are calibrated on certified batches to satisfy FPR α.

Implementation notes. All generators are sourced from official GitHub or Hugging Face repositories. FARE training and calibration hyperparameters are specified in Appendix B, with ablations on key hyperparameters such as patch size d and top-k aggregation in Appendix D. The core FARE implementation is publicly available at https://github.com/kaikaiyao/FARE; Appendix G describes the release and implementation details.

6 Results

6.1 Swap Detection under Generator Substitution

Table 1 reports FARE's *single-image* swap detection under generator substitution at a strict operating point (TPR@1%FPR), averaged over all certified–swapped pairs within each scenario. All

baselines remain far below FARE in this low-FPR regime. FSD reaches its highest mean TPR on CF-Diff at 18.40%, while NPR + OC-SVM reaches its highest mean on WF-SDVer at 18.32%; both remain below 1.1% on WF-GANTrain. Standard deviations are computed across heterogeneous certified–swapped cases within each scenario; large values indicate that a baseline detects some easy substitutions but fails on many close or diffusion-heavy cases, rather than providing reliable scenario-level performance.

In contrast, FARE achieves near-perfect cross-family detection (99.43% on CF-FFHQ and 99.45% on CF-Diff) and remains strong on harder within-family near-swaps (96.38% on WF-GANTrain and 92.56% on WF-SDVer). Appendix E.3 reports all within-family pairs; the weakest SD pair, SD1.4 $\rightarrow$ SD1.5, reaches 85.85% TPR. Under the exact-model white-box PGD stress test with LPIPS < 0.05, drops are limited (e.g., 99.45%$\rightarrow$99.38% on CF-Diff; 92.56%$\rightarrow$92.38% on WF-SDVer). Table 1 also reports conditional attack success rate (ASR), the fraction of initially detected non-certified images flipped to accepted while satisfying LPIPS < 0.05. On WF-GANTrain, decision-only HSJA reaches 0.14% ASR at 5,000 queries per image, compared with 0.67% for PGD. PGD reaches 77.64% ASR without the LPIPS constraint, showing that the low ASR depends on the perceptual criterion. Appendix E.1 gives the full query-budget and LPIPS analyses. These results cover the evaluated objectives and budgets, not black-box attacks in general.

The ablations in Table 1 clarify the role of each component. Removing the constrained layer sharply degrades performance, especially on diffusion-heavy and within-family evaluations (e.g., CF-Diff drops to 7.95% and WF-SDVer to 8.24%), supporting the benefit of the constrained residual representation at low FPR. Replacing patch-based scoring with full-image processing also breaks the verifier on the hardest settings (CF-Diff 10.38%; WF-GANTrain 11.23%), supporting the benefit of local scoring over full-image processing. Finally, hard positives are essential for a tight boundary: adversarial near-boundary positives and contradiction positives both contribute substantially, and removing both reduces FARE to near chance across scenarios (e.g., CF-Diff 0.97%; WF-GANTrain 1.74%). Overall, these results support FARE's design of tightening a content-suppressed one-class acceptance region using certified-derived hard positives. Additional hyperparameter ablations are in Appendix D.

6.2 Detection of Cost-Motivated Deployment Variants

Table 2 evaluates a stricter non-compliance setting than Sec. 6.1: instead of swapping to a different generator, the provider deploys *cost-motivated variants of the same checkpoint* (e.g., simulated model compression or fewer inference steps). This is inherently harder because generator-specific traces may persist when the model identity is unchanged; mild modifications can therefore induce only small distribution shifts, making 1% FPR detection more challenging.

Weight compression. FARE remains highly sensitive to *aggressive* compression, achieving near-perfect detection under INT4 quantization and heavy pruning across architectures (e.g., 97–99% TPR on StyleGAN3/VDVAE/NCSN++ and 98.27% / 98.12% on SD2.1). In contrast, *milder* compression is harder: INT6 and mild pruning reduce TPR to the low 90s on StyleGAN3/VDVAE/NCSN++ (e.g., 90–93%) and the high 80s on SD2.1 (86.42% and 87.36%). This matches the threat model: stronger compression induces larger shifts, while milder compression remains closer to certified output statistics. We exclude INT8 because this strict-contract benchmark focuses on clearly non-default compression variants, while INT8 is commonly a default and typically preserves fidelity.

Diffusion inference-time changes. Reducing denoising steps shows a similar severity trend. A moderate reduction ($20 \rightarrow 10$) is hardest, especially for SD2.1 (80.86%), and remains challenging for NCSN++ (88.93%). A more aggressive reduction ($20 \rightarrow 5$) becomes easy to detect (99.12% on NCSN++ and 98.29% on SD2.1), indicating that substantial inference-time shortcuts shift outputs away from the certified acceptance region. Overall, FARE reliably flags deployment variants that substantially change the output distribution, while milder same-checkpoint deviations remain harder at strict operating points. This is a limitation of single-image auditing under a strict false-alarm budget: when a deployment change preserves most generator-specific traces, the shift may remain close to the calibrated certified tail.

Deployment Variant	Detection Performance (TPR@1%FPR)			
	StyleGAN3	VDVAE	NCSN++	SD2.1
Weight compression				
Quantization (INT6)	92.57±0.91	91.03±1.02	90.50±1.10	86.42±1.35
Quantization (INT4)	98.89±0.34	97.14±0.48	98.39±0.29	98.27±0.31
Mild Pruning	92.42±0.88	93.44±0.77	91.57±0.95	87.36±1.28
Heavy Pruning	98.44±0.41	99.32±0.22	97.53±0.52	98.12±0.36
Inference-time changes				
Diffusion steps: 20→10	–	–	88.93±1.24	80.86±1.89
Diffusion steps: 20→5	–	–	99.12±0.21	98.29±0.33

Table 2: Single-image detection performance of FARE (TPR@1%FPR) under cost-motivated deployment variants across different generator architectures. For each model, the verifier is trained and calibrated on the default certified deployment, then evaluated on modified variants of the *same* model. Quantization and pruning are simulated weight transformations; reduced diffusion steps change the executed inference schedule.

Rule	n	WF-GANTrain (TPR@αFPR)			WF-SDVer (TPR@αFPR)		
		$\alpha = 1\%$	$\alpha = 0.5\%$	$\alpha = 0.1\%$	$\alpha = 1\%$	$\alpha = 0.5\%$	$\alpha = 0.1\%$
Single-image	1	96.38±2.77	89.86±2.64	71.53±4.87	92.56±3.06	85.49±3.84	62.39±5.27
Batch (Mean)	5	99.81±0.24	99.67±0.31	99.03±0.46	99.46±0.35	99.12±0.43	98.27±0.68
Batch (Mean)	10	99.92±0.17	99.85±0.23	99.47±0.32	99.73±0.26	99.41±0.36	98.84±0.49
Batch (Max)	5	99.48±0.37	99.16±0.45	98.29±0.63	98.93±0.42	98.57±0.58	97.24±0.86
Batch (Max)	10	99.65±0.26	99.37±0.34	98.74±0.47	99.21±0.33	98.86±0.46	97.92±0.67

Table 3: Batch verification performance of FARE at given operating points on within-family near-swaps. Entries report mean ± std TPR@αFPR at $\alpha \in \{1\%, 0.5\%, 0.1\%\}$ for WF-GANTrain (StyleGAN training-configuration variants) and WF-SDVer (Stable Diffusion versioning).

6.3 Batch Verification at Low FPR

Table 3 evaluates verification at up to one order of magnitude lower FPR and how much aggregation over multiple outputs from the same deployment recovers performance. On the harder scenarios, FARE is strong for single-image verification at α=1% (96.38% on WF-GANTrain; 92.56% on WF-SDVer), but drops at α=0.1% to 71.53% and 62.39%, respectively, because the boundary must be calibrated to the certified tail, leaving less margin for subtle swaps. Batching largely closes this gap: with n=5 at α=0.1%, mean aggregation reaches 99.03% (WF-GANTrain) and 98.27% (WF-SDVer), while max reaches 98.29% and 97.24%; with n=10, mean reaches 99.47% and 98.84%, while max reaches 98.74% and 97.92%. The reduced variance relative to single-image verification indicates that aggregation also stabilizes decisions.

Mean aggregation is consistently stronger: averaging reduces certified-batch score variance while accumulating small but consistent substitution-induced shifts. Max aggregation also benefits from batching, but remains slightly weaker at the strictest operating point because fixed-FPR calibration makes the max threshold increasingly conservative as n grows. Overall, Table 3 suggests that single-image auditing is effective at α=1% but insufficient at extreme budgets, while small-batch verification (e.g., $n \in \{5, 10\}$) achieves high TPR in the evaluated swap scenarios at α=0.1%. Applications requiring α=0.1%-level false-alarm control should therefore prefer small-batch auditing when repeated queries from the same deployment are available.

7 Discussion and Conclusion

FARE shows that output-only integrity auditing of black-box image generators is viable under certified-only enrollment. A one-class acceptance region built on forensic residual features, patch-level localization, and certified-derived hard positives achieves near-perfect single-image detection for cross-family substitutions and strong performance on harder within-family near-swaps at 1%

FPR. Table 1 shows that these gains require the full design: removing the constrained convolution, patch scoring, adversarial positives, or contradiction positives each causes substantial degradation. Exact-model white-box PGD and decision-only HSJA have low attack success under the evaluated budgets and the LPIPS constraint of 0.05, although these experiments do not cover every adaptive attack.

Several limitations remain. Mild same-checkpoint variants are hardest because they stay closest to the certified distribution, making small shifts difficult to detect under strict false-alarm budgets. In the evaluated model-swap settings, aggregating five images improves detection at lower FPRs; we have not established the same gain for every compression or inference variant.

We study a strict contract that permits no shift beyond the enrolled deployment. FARE checks whether an image is consistent with that contract; it cannot infer whether an identical output change arose from a benign or malicious pipeline. More permissive contracts could include expected compression, resizing, prompt rewriting, or safety filtering, but we have not evaluated them. We also do not evaluate a versioned commercial API because an opaque service may change its checkpoint without providing an independently verifiable version label. The evaluated generators span GAN, VAE, and diffusion families, but broader commercial deployments remain future work.

Patch-level visualizations and normalized spatial entropy expose the evidence entering FARE's decision and show that the highest-scoring patches are not consistently confined to one fixed image region (Appendix E.7); they do not exclude semantic shortcuts within a patch. Overall, FARE provides evidence for output-only post-certification auditing within this deployment and attack scope. Future directions include broader contracts that tolerate pre-agreed inference variants, hybrid provenance with active watermarking or cryptographic attestation, and extensions to temporal modalities such as video generation, where artifact structures and distribution-shift signals may differ from the spatial setting studied here.

Acknowledgments and Disclosure of Funding

We thank our anonymous reviewers for their valuable feedback, which substantially improved the paper. This work was supported by the Edinburgh International Data Facility (EIDF) and the Data-Driven Innovation Programme at the University of Edinburgh. Access to EIDF was facilitated through the University of Edinburgh's Generative AI Laboratory GAIL Fellow scheme. Marc Juarez is a GAIL fellow and the recipient of a Google Research Scholar Award in Security.

The authors declare no competing interests.

References

[1] Ella P. Fokkinga, Thijs A. Eker, Jan Erik van Woerden, Jean-Michel Witon, Simon O. B. Stallinga, Arnoud Visser, Klamer Schutte, and Friso G. Heslinga. Generative AI methods for synthesis of image data to train AI for automated scene understanding in a military context: a review of opportunities. In *Synthetic Data for Artificial Intelligence and Machine Learning: Tools, Techniques, and Applications III*, volume 13459 of *Proceedings of SPIE*, page 1345905. SPIE, 2025. 1

[2] Lennart R. Koetzier, Jie Wu, Domenico Mastrodicasa, Aline Lutz, Matthew Chung, W. Adam Koszek, Jayanth Pratap, Akshay S. Chaudhari, Pranav Rajpurkar, Matthew P. Lungren, and Martin J. Willemink. Generating synthetic data for medical imaging. *Radiology*, 312(3):e232471, 2024. 1

[3] European Parliament and Council of the European Union. Regulation (EU) 2024/1689 (Artificial Intelligence Act). Official Journal of the European Union, L, 2024/1689, 12 July 2024, 2024. 1

[4] OpenAI. New models and developer products announced at DevDay. https://openai.com/index/new-models-and-developer-products-announced-at-devday/, November 2023. Published November 6, 2023. Accessed: 2026-09-25. 1

[5] Adobe. Adobe Firefly API: Firefly Services. https://developer.adobe.com/firefly-services/docs/firefly-api/. Accessed: 2026-09-25. 1

[6] Francesco Marra, Diego Gragnaniello, Luisa Verdoliva, and Giovanni Poggi. Do GANs leave artificial fingerprints? In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 506–511, 2019. 2, 3

[7] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 7

[8] Kai Yao and Marc Juarez. Smudged fingerprints: A systematic evaluation of the robustness of AI image fingerprints. In *2026 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 284–302, 2026. 3

[9] Jianbo Chen, Michael I. Jordan, and Martin J. Wainwright. HopSkipJumpAttack: A query-efficient decision-based attack. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1277–1294, 2020. 3, 7, 18

[10] Sven Gowal, Rudy Bunel, Florian Stimberg, David Stutz, Guillermo Ortiz-Jimenez, Christina Kouridi, Mel Vecerik, Jamie Hayes, Sylvestre-Alvise Rebuffi, Paul Bernard, Chris Gamble, Miklós Z. Horváth, Fabian Kaczmarczyck, Alex Kaskasoli, Aleksandar Petrov, Ilia Shumailov, Meghana Thotakuri, Olivia Wiles, Jessica Yung, Zahra Ahmed, Victor Martin, Simon Rosen, Christopher Savčak, Armin Senoner, Nidhi Vyas, and Pushmeet Kohli. SynthID-Image: Image watermarking at internet scale. *arXiv preprint arXiv:2510.09263*, 2025. 3

[11] Matthew Tancik, Ben Mildenhall, and Ren Ng. StegaStamp: Invisible hyperlinks in physical photographs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2117–2126, 2020. 3

[12] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. In *Advances in Neural Information Processing Systems*, volume 36, pages 58047–58063, 2023. 3

[13] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22466–22477, 2023.

[14] Sam Gunn, Xuandong Zhao, and Dawn Song. An undetectable watermark for generative image models. In *International Conference on Learning Representations (ICLR)*, 2025. 3

[15] Ning Yu, Larry S. Davis, and Mario Fritz. Attributing fake images to GANs: Learning and analyzing GAN fingerprints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7556–7566, 2019. 3

[16] Tarik Dzanic, Karan Shah, and Freddie Witherden. Fourier spectrum discrepancies in deep network generated images. In *Advances in Neural Information Processing Systems*, volume 33, pages 3022–3032. Curran Associates, Inc., 2020. 3

[17] Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. Leveraging frequency analysis for deep fake image recognition. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pages 3247–3258. PMLR, 2020.

[18] Ricard Durall, Margret Keuper, and Janis Keuper. Watch your up-convolution: CNN based generative deep neural networks are failing to reproduce spectral distributions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7890–7899, 2020. 3

[19] Sharath Girish, Saksham Suri, Sai Saketh Rambhatla, and Abhinav Shrivastava. Towards discovery and attribution of open-world GAN generated images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14094–14103, 2021. 3

[20] Fengyuan Liu, Haochen Luo, Yiming Li, Philip Torr, and Jindong Gu. Which model generated this image? a model-agnostic approach for origin attribution. In *Computer Vision – ECCV 2024*, volume 15120 of *Lecture Notes in Computer Science*, pages 282–301. Springer, 2024. 3

[21] Mike Laszkiewicz, Jonas Ricker, Johannes Lederer, and Asja Fischer. Single-model attribution of generative models through final-layer inversion. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 26007–26042. PMLR, 2024. 3

[22] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001. 3, 6

[23] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 4393–4402. PMLR, 2018. 3, 6

[24] Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. DROCC: Deep robust one-class classification. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 3711–3721. PMLR, 2020. 3, 6

[25] Sauptik Dhar and Bernardo Gonzalez-Torres. DOC^3: Deep one class classification using contradictions. *Machine Learning*, 113(8):5109–5150, 2024. 3, 5, 6

[26] Belhassen Bayar and Matthew C. Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec)*, pages 5–10, 2016. 4, 5

[27] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2002. 4

[28] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2127–2136. PMLR, 2018. 4

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 8748–8763. PMLR, 2021. 6

[30] Oliver Giudice, Luca Guarnera, and Sebastiano Battiato. Fighting deepfakes by detecting GAN DCT anomalies. *Journal of Imaging*, 7(8):128, 2021. 6, 15

[31] Tai D. Nguyen, Aref Azizpour, and Matthew C. Stamm. Forensic self-descriptions are all you need for zero-shot detection, open-set source attribution, and clustering of AI-generated images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3040–3050, 2025. 6, 15

[32] Chuangchuang Tan, Huan Liu, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Rethinking the up-sampling operations in CNN-based generative network for generalizable deepfake detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28130–28139, 2024. 6, 15

[33] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019. 6

[34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 6

[35] Joshua Lopez. Stable Diffusion v2.1 and DreamStudio updates 7-dec 22. Stability AI. https://stability.ai/news-updates/stablediffusion2-1-release7-dec-2022, 2022. Accessed: 2026-09-25. 6, 7

[36] Stability AI. Introducing Stable Diffusion 3.5. https://stability.ai/news-updates/introducing-stable-diffusion-3-5, October 2024. Published October 22, 2024. Accessed: 2026-09-25. 6

[37] Jeongsoo Park and Andrew Owens. Community forensics: Using thousands of generators to train fake image detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8245–8257, 2025. 6

[38] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119, 2020. 6

[39] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 852–863, 2021. 6, 7

[40] Zijie J. Wang, Evan Montoya, David Munechika, Haoyang Yang, Benjamin Hoover, and Duen Horng Chau. DiffusionDB: A large-scale prompt gallery dataset for text-to-image generative models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 893–911. Association for Computational Linguistics, 2023. 6

[41] Rewon Child. Very deep VAEs generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations (ICLR)*, 2021. 7

[42] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 7

Supplementary Material

This appendix provides the implementation and experimental details deferred from the main paper. It describes the evaluation scenarios, baseline implementations, FARE hyperparameters and architecture, additional attack and calibration analyses, within-family pairwise results, ablations, sample efficiency, resource use under reduced diffusion steps, patch-level evidence, the Stable Diffusion prompt-diversity check, and reproducibility details.

A Detailed Scenario Definition for Generator Swaps

This section provides detailed definitions for the generator-swap evaluations referenced in Table 1 of the main text. In all experiments, the contract is the strict singleton contract $\mathcal{C} = \{(M_{\text{cert}}, \Pi_{\text{cert}})\}$, meaning any undeclared change in either the checkpoint or the inference pipeline is treated as non-certified. Unless otherwise noted, unconditional generators are queried with a fixed latent seed, and conditional text-to-image models are queried with a fixed prompt list. Certified and non-certified deployments are compared under matched seeds or prompts. All images are converted to RGB and, when necessary, resized to the stated evaluation resolution before patch extraction.

This strict contract permits no deployment shift beyond the enrolled checkpoint and pipeline. FARE checks consistency with that contract, not the provider's intent. An image-only auditor cannot distinguish benign and malicious pipelines that produce the same compression, resizing, prompt-rewriting, or safety-filtering artifacts. Such expected changes could be enrolled under a more permissive contract, which we do not evaluate here.

A.1 Scenario definitions

Table A.1 summarizes the definition of each scenario referenced in the generator-swap evaluations in Table 1.

For the FFHQ-256 pool, all models are treated as unconditional face generators. For Stable Diffusion experiments, prompts are sampled from DiffusionDB and deduplicated before being split into train, calibration, and verification subsets.

For the CommunityForensics diffusion benchmark, we retain only the diffusion-dominated evaluation subset because it serves as the most relevant open-world stress test for SD-family certification.

The evaluated generators span GAN, VAE, and diffusion families, but we do not evaluate a versioned commercial API. An opaque service may change the served checkpoint without exposing a stable, independently verifiable version label, which would prevent a controlled version-swap experiment.

Table A.1: Detailed definitions of the evaluation scenarios from Table 1 in the main paper. Abbreviations: SG2 = StyleGAN2, SG3-T = StyleGAN3-T (translation equiv.), SG3-R = StyleGAN3-R (translation and rotation equiv.), SD = Stable Diffusion, ADA = Adaptive Discriminator Augmentation, FFHQU = Unaligned FFHQ dataset

Scenario	Certified deployment(s)	Non-certified deployment(s)	Resolution
CF-FFHQ	SG2, SG3-T, GANformer, StyleSwin, R3GAN, CIPS, VDVAE, VQVAE, NVAE, NCSN++, LDM, ADM	For each certified model, the remaining models in the pool	256^2
CF-Diff	SD1.4, SD1.5, SD2.0, SD2.1, SD3.5-Medium	Evaluation images from CommunityForensics	512^2
WF-GANTrain (SG2)	SG2-30k, SG2-70k, SG2-140k, SG2-30k-ADA, SG2-70k-ADA, SG2-140k-ADA	For each certified model, the remaining models in the pool	256^2
WF-GANTrain (SG3)	SG3-T, SG3-R, SG3-T-FFHQU, SG3-R-FFHQU	For each certified model, the remaining models in the pool	256^2
WF-SDVer	SD1.4, SD1.5, SD2.0, SD2.1, SD3.5-Medium	For each certified model, the remaining models in the pool	512^2

A.2 Baseline implementation

Table A.2 details the one-class and forensic baselines. Their task-specific fitting and threshold calibration use only certified images from the enrolled generator. Patch-based baselines share FARE's patch extraction, with method-specific image-level aggregation reported in the table.

Table A.2: Implementation details of the one-class and forensic baselines in Table 1

Method	Implementation details
Deep SVDD	The same convolutional backbone as FARE, but without the constrained front-end and with a 256-dimensional embedding head. The center is initialized from 5,000 certified patches. Training uses 32×32 patches and batch size 16. The image score is the top-10 mean of squared distances to the learned center.
OC-SVM (Patch)	A one-class SVM trained on frozen ResNet-50 patch descriptors. Certified 32×32 patches are resized to 224×224. The RBF kernel uses $\nu = 0.01$ and $\gamma = \texttt{scale}$. Image scores are the top-5 mean of patch anomaly scores.
CLIP + OC-SVM	A one-class SVM trained on frozen CLIP ViT-B/32 image embeddings of a full image. The RBF kernel uses $\nu = 0.01$ and $\gamma = \texttt{scale}$.
DCT Feature + OC-SVM	A one-class SVM trained on our adapted 63-dimensional DCT fingerprints, motivated by [30], computed from the mean absolute AC coefficients of each full image. The RBF kernel uses $\nu = 0.01$ and $\gamma = \texttt{scale}$. Because this baseline is image-level, no patch aggregation is applied.
FSD	A one-class model fitted to the frozen Forensic Self-Description representation [31]. Task-specific fitting and threshold calibration use only certified-generator outputs.
NPR + OC-SVM	A one-class SVM fitted to frozen Neighboring Pixel Relationships features [32]. Task-specific fitting and threshold calibration use only certified-generator outputs.

B Hyperparameters of FARE

This section details the hyperparameters used in FARE (Section 4). Table B.1 lists those for network training and calibration, Table B.2 describes the network architecture, and Table B.3 specifies the exact-model white-box attack.

In line with the Bayar–Stamm design principle, we do not place a non-linearity directly after the constrained layer. All remaining convolutions use GroupNorm and ReLU, and the classifier head outputs a single scalar anomaly logit for each patch.

Table B.1: FARE network training and calibration hyperparameters. Unless stated otherwise, the same settings are used across all scenarios

Hyperparameter	Value
Num certified training images	100,000
Num certified calibration images	10,000
Num verification images	10,000
Image resolution	256×256 for FFHQ pool (i.e. GAN/VAE/diffusion pool); 512×512 for the conditioned SD family
Patch size d	$d = 32$ for both training and verification
Top-k parameter	$k = 10$
Optimizer	AdamW
Learning rate	1×10^{-3}
Batch size	32
Max training iterations	1,000,000
Warm-up iterations T_{warm}	2,000
Loss weights (μ, λ)	$(0.1, 0.1)$
Adversarial shell radius r	0.03
Adversarial shell thickness γ	1.2
PGD steps for adv. positives	5
PGD step size for adv. positives	0.005 before projection onto the ℓ_2 shell
Contradiction transforms $\mathcal{T}$	Gaussian blur with $\sigma = 1.0$ and kernel size 5; Gaussian noise with std $= 2/255$; JPEG quality $= 75$; and crop-resize that keeps 90% of the area and then resizes back

Table B.2: FARE network architecture

Stage	Implemented setting
Constrained conv layer	Bayar–Stamm constrained depthwise convolution, kernel size 5, eight filters per RGB channel, constraint scale 1.0
Channel mixing	1×1 conv from 24 channels to width 64, followed by ReLU
Backbone	Six 3×3 convolutional blocks with widths $(64, 128, 128, 256, 256, 256)$ and strides $(1, 2, 1, 2, 1, 1)$
Normalization	GroupNorm throughout the backbone
Head	Global average pooling, then a linear map from 256 features to one logit

Table B.3: Hyperparameters of the exact-model white-box PGD evaluation. Thresholds are kept fixed at the values calibrated on certified images

Attack hyperparameter	Value
Attack type	PGD on image pixels as a white-box stress test against the deployed detector
Optimization objective	Minimize $s(x')$, i.e., the image-level detector score after top-k aggregation
Perturbation budget $\|\delta\|_\infty$	$\epsilon = 0.025$ with images normalized to $[0, 1]$
Number of PGD steps	50 with early stopping after successful acceptance
Step size	0.001
Random restarts	5, retaining the best adversarial example across restarts
Attack success criterion	$s(x') \leq \tau_\alpha$ and $\mathrm{LPIPS}(x', x) < 0.05$
Perceptual fidelity check	LPIPS is evaluated on the original and adversarial images

C Algorithmic Pseudocodes

This section details the pseudocodes referenced from the method section (i.e., Section 4). The mathematical objective and verification rule remain in Section 4; the algorithms below summarize and specify the training loop and the PGD routine used to generate near-boundary adversarial positives.

Algorithm C.1 FARE Enrollment: Training

Require: Certified images $\mathcal{D}_{\mathrm{cert}}$ sampled from marginal $P_{G_{\mathrm{cert}}}$; patch size d; transform set $\mathcal{T}$; weights (μ, λ); warm-up iterations T_{warm}; total training iterations $T_{\max}$.
Ensure: Trained verifier network f_θ.

Initialize f_θ with the Bayar–Stamm constrained first layer described in Section 4.
for $t = 1$ **to** $T_{\max}$ **do**
  Sample a minibatch of images $x \in \mathcal{D}_{\mathrm{cert}}$.
  Extract patches $\mathcal{P}(x)$ for each x.
  Compute $\mathcal{L}_{\mathrm{cert}}$ from Eq. (3).
  if $t \leq T_{\mathrm{warm}}$ **then**
    Set $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{cert}}$ ▷ warm-up: certified negatives only
  else
    Compute $\mathcal{L}_{\mathrm{adv}}$ (Eq. (5)) by generating $\tilde{p}$ via Alg. C.2 for each patch.
    Compute $\mathcal{L}_{\mathrm{con}}$ (Eq. (6)) by applying all $\varphi \in \mathcal{T}$ to each x.
    Set $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{cert}} + \mu\mathcal{L}_{\mathrm{adv}} + \lambda\mathcal{L}_{\mathrm{con}}$.
  end if
  Update θ by descending $\nabla_\theta \mathcal{L}$.
  Project the constrained kernels back onto the Bayar–Stamm constraint set.
end for

Algorithm C.2 Adversarial Patch Generation via PGD

Require: Certified patch p; classifier f_θ; shell $\mathcal{S}$ defined by (r, γ); PGD steps T; step size η.
Ensure: Adversarial patch $\tilde{p}$.

```
Initialize p̃ = p.
Initialize random perturbation δ and apply Euclidean projection Π_S(δ).
for j = 1 to T do
    Compute gradients: g ← ∇_δ BCEWithLogits(f_θ(clip(p̃ + δ)), 0)
    Update and project: δ ← Π_S(δ − η · g)
end for
return p̃ = clip(p + δ)
```

D Hyperparameter Ablation Study

This section evaluates the hyperparameters that are key to FARE's performance. Tables D.1, D.2, and D.3 present the ablation results for patch size d, top-k aggregation, and warm-up length T_{warm}, respectively. For brevity, the WF-GANTrain column reports the average over its SG2 and SG3 variants.

We find that a medium patch size ($d = 32$) consistently outperforms both smaller and larger configurations. Regarding aggregation, values of k between 10 and 50 yield comparable performance, while smaller or larger values of k perform visibly worse; we fix $k = 10$ for all reported results to maintain consistency. Finally, a warm-up period exceeding 1,000 iterations proves beneficial by stabilizing the certified region before adversarial and contradiction positives are introduced, but further increase does not bring significant performance gains. We set $T_{\text{warm}} = 2{,}000$ for all experiments.

Table D.1: Ablation on patch size d. Results are reported as TPR@1%FPR.

d	CF-FFHQ	CF-Diff	WF-GANTrain	WF-SDVer
8	91.24±1.25	99.17±0.67	92.42±2.43	90.35±5.42
32	**99.43**±1.19	**99.45**±0.88	**96.38**±2.77	**92.56**±3.06
128	89.93±2.51	85.02±4.53	70.11±6.12	72.84±9.43

Table D.2: Ablation on the top-k image aggregation parameter. Results are reported as TPR@1%FPR.

k	CF-FFHQ	CF-Diff	WF-GANTrain (avg.)	WF-SDVer
5	98.87±1.12	96.08±2.34	94.86±1.89	88.73±3.45
10	99.43±1.19	**99.45**±0.88	**96.38**±2.77	**92.56**±3.06
50	**99.49**±0.74	99.11±1.02	95.74±1.72	91.88±2.41
100	95.54±2.67	94.27±3.11	92.91±3.48	86.92±4.92

E Additional Evaluation Analyses

E.1 Attack Evaluation

We report attack success rate (ASR) as the fraction of initially detected non-certified images that the attack changes to accepted at the fixed calibrated threshold. The exact-model white-box attack uses PGD with an ℓ_∞ budget of $\epsilon = 0.025$ for pixels in $[0, 1]$. An attack succeeds only when it changes the decision and satisfies $\text{LPIPS}(x_{\text{adv}}, x) < 0.05$. Under this criterion, conditional ASR is 0.15% on CF-FFHQ, 0.07% on CF-Diff, 0.67% on WF-GANTrain, and 0.19% on WF-SDVer.

The LPIPS condition substantially changes the PGD result. On WF-GANTrain, ASR is 0.05%, 0.40%, 0.67%, 5.53%, 23.86%, and 58.10% at LPIPS thresholds of 0.01, 0.025, 0.05, 0.1, 0.25, and 0.5, respectively; without an LPIPS constraint, it is 77.64%. The higher success rates therefore require progressively larger changes to the image.

Table D.3: Ablation on the certified-only warm-up length T_{warm}. Results are reported as TPR@1%FPR.

T_{warm}	CF-FFHQ	CF-Diff	WF-GANTrain	WF-SDVer
0	96.26±2.14	98.71±1.05	93.54±2.88	89.92±2.12
1,000	98.90±1.45	99.23±0.82	95.81±1.94	**92.84**±1.56
2,000	**99.43**±1.19	99.45±0.88	**96.38**±2.77	92.56±3.06
10,000	99.40±1.02	**99.68**±0.54	96.33±1.68	92.13±2.04

We also evaluate decision-only HopSkipJumpAttack (HSJA) [9] on WF-GANTrain under the same LPIPS threshold of 0.05. The attacker observes only FARE's binary accept-or-reject decision. ASR is 0.00% at 100 and 500 queries per image, 0.05% at 1,000 queries, and 0.14% at 5,000 queries, compared with 0.67% for exact-model PGD in the same scenario. These results cover the evaluated exact-model and decision-only attacks. They do not establish robustness to surrogate-transfer, generation-pipeline, cross-detector, or other adaptive attacks.

E.2 Finite-Sample Calibration at Strict Operating Points

The calibrated false-positive rate varies across finite calibration samples even when the scorer is fixed. Let $Z_1, \ldots, Z_m$ be independent and identically distributed certified single-image scores with continuous cumulative distribution function F, and write their ascending order statistics as $Z_{(1)} \le \cdots \le Z_{(m)}$. For an upper-tail rank r_α, distinct from the patch-aggregation parameter k, set $\tau_\alpha = Z_{(m+1-r_\alpha)}$. The population false-positive rate at this random threshold is $p_T = 1 - F(\tau_\alpha)$ and follows

$$p_T \sim \text{Beta}(r_\alpha, m + 1 - r_\alpha). \tag{E.1}$$

This follows because the probability integral transform makes $F(Z_i)$ independent uniform variables, so $F(\tau_\alpha)$ is their $(m + 1 - r_\alpha)$th order statistic. With $m = 10{,}000$, the nominal 0.5% and 0.1% targets use $r_\alpha = 50$ and $r_\alpha = 10$. Their exact central 95% ranges are $[0.371\%, 0.647\%]$ and $[0.048\%, 0.171\%]$, and their one-sided 95% upper bounds are 0.621% and 0.157%. Thus, a nominal target does not fix the population FPR for every realized calibration set. These intervals assume continuous scores; tied scores require a specified tie-breaking rule and corresponding analysis. They quantify finite-calibration uncertainty for the fixed single-image scorer and the same certified distribution. They do not include scorer or distribution changes, or automatically provide a guarantee for batch calibration.

E.3 Pairwise Results for Within-Family Swaps

Tables E.1, E.2, and E.3 report every certified–substituted pair in the within-family scenarios. Rows identify the certified generator and columns identify the substituted generator. The weakest pairs are SG2-70k-ADA → SG2-140k-ADA at 89.29%, SG3-R-FFHQU → SG3-R at 95.18%, and SD1.4 → SD1.5 at 85.85% TPR at 1% FPR.

Certified	30k	70k	140k	30k-A	70k-A	140k-A
SG2-30k	–	97.40	98.04	96.81	98.28	98.12
SG2-70k	98.45	–	91.67	98.08	93.49	93.82
SG2-140k	99.20	92.52	–	96.71	93.67	93.01
SG2-30k-ADA	97.72	97.14	97.54	–	94.44	94.78
SG2-70k-ADA	99.48	94.31	94.56	95.36	–	89.29
SG2-140k-ADA	99.53	94.73	93.82	95.83	90.11	–

Table E.1: Pairwise WF-GANTrain results for StyleGAN2 training variants. Entries are TPR@1%FPR (%); A denotes ADA training.

Certified	SG3-T	SG3-R	SG3-T-FFHQU	SG3-R-FFHQU
SG3-T	–	99.01	96.93	99.93
SG3-R	99.83	–	99.93	96.11
SG3-T-FFHQU	96.02	99.19	–	98.98
SG3-R-FFHQU	99.20	95.18	99.82	–

Table E.2: Pairwise WF-GANTrain results for StyleGAN3 variants. Entries are TPR@1%FPR (%).

Certified	SD1.4	SD1.5	SD2.0	SD2.1	SD3.5-M
SD1.4	–	85.85	90.29	91.02	93.32
SD1.5	89.53	–	89.58	90.54	93.96
SD2.0	94.06	93.97	–	87.63	93.40
SD2.1	94.01	94.12	90.87	–	92.88
SD3.5-M	97.03	95.95	96.57	96.62	–

Table E.3: Pairwise WF-SDVer results. Entries are TPR@1%FPR (%).

E.4 Contradiction Transform Ablation

We remove each contradiction transform in turn from the four-transform set used during training. In the CF-FFHQ pool, the StyleGAN2-certified detector reaches 98.84% average TPR at 1% FPR with all four transforms. Removing any transform reduces performance, although the effect is not uniform: crop–resize and Gaussian blur have the largest effects in this setting.

Contradiction transform set	Mean TPR@1%FPR
All four transforms	98.84
Without Gaussian blur	86.76
Without Gaussian noise	94.28
Without JPEG compression	93.90
Without crop–resize	80.63

Table E.4: Leave-one-out ablation of the contradiction transforms for a StyleGAN2-certified detector in CF-FFHQ (%).

E.5 Training-Set Size

The main experiments use 100,000 training images as a consistent protocol across settings, not as a universal minimum. In CF-FFHQ, FARE reaches 99.27% TPR at 1% FPR with 1,000 training images, compared with 99.43% using 100,000. In WF-SDVer, 10,000 training images approach the performance obtained with 100,000, while only a few thousand produce a noticeable reduction. The required training-set size may therefore depend on how close the substituted generators are to the certified generator. The calibration set remains fixed at 10,000 images throughout this study; calibration-set size is not separately ablated.

E.6 Resource Use under Fewer Diffusion Steps

Reducing the number of diffusion steps changes the executed computation directly. Table E.5 reports the measured latency, estimated GPU board energy, peak memory, and CLIP image–text similarity for the evaluated 20-, 10-, and 5-step configurations. Relative to 20 steps, 10 steps reduce latency by 46.5% and estimated energy by 42.3%; 5 steps reduce them by 70.7% and 68.6%. Peak memory is unchanged, while lower CLIP similarity indicates weaker prompt alignment. We do not infer monetary savings because hardware and provider pricing differ.

The quantization and pruning variants in Table 2 do not provide corresponding resource measurements. Quantization is simulated through quantize–dequantize operations while retaining the original tensor storage and computation. Pruning zeros the specified proportion of smallest-magnitude weights without sparse storage or kernels. These experiments measure detectability after simulated compression, not realized latency, memory, or energy savings.

Steps	Latency (s/image)	Energy (J/image)	Peak memory (GiB)	CLIP similarity
20	1.311	354.9	3.10	0.2956
10	0.701	204.9	3.10	0.2796
5	0.384	111.4	3.10	0.2409

Table E.5: Resource and prompt-alignment measurements for the reduced diffusion-step experiment. Energy is estimated GPU board energy.

E.7 Patch-Level Evidence

FARE divides each 256×256 image into an 8×8 grid of non-overlapping 32×32 patches. Its image score averages the ten largest raw patch anomaly logits. Figure E.1 shows one representative image from the enrolled StyleGAN2 generator and from each of three substituted generators. The white boxes identify the same ten patches used by the decision rule; the visualization is therefore a direct view of the evidence entering the score rather than a separate post-hoc attribution method.

To test whether these selected patches repeatedly occupy one fixed image region, we separately evaluate 100 images from each generator and count how often every grid cell enters the top ten. Normalized spatial entropy ranges from 0.965 to 0.992 across the four generators, indicating that the selected patches are broadly distributed over the grid. This rules out consistent reliance on one fixed patch location in this experiment. It does not rule out semantic shortcuts within a patch or establish that every selected feature is causally forensic.

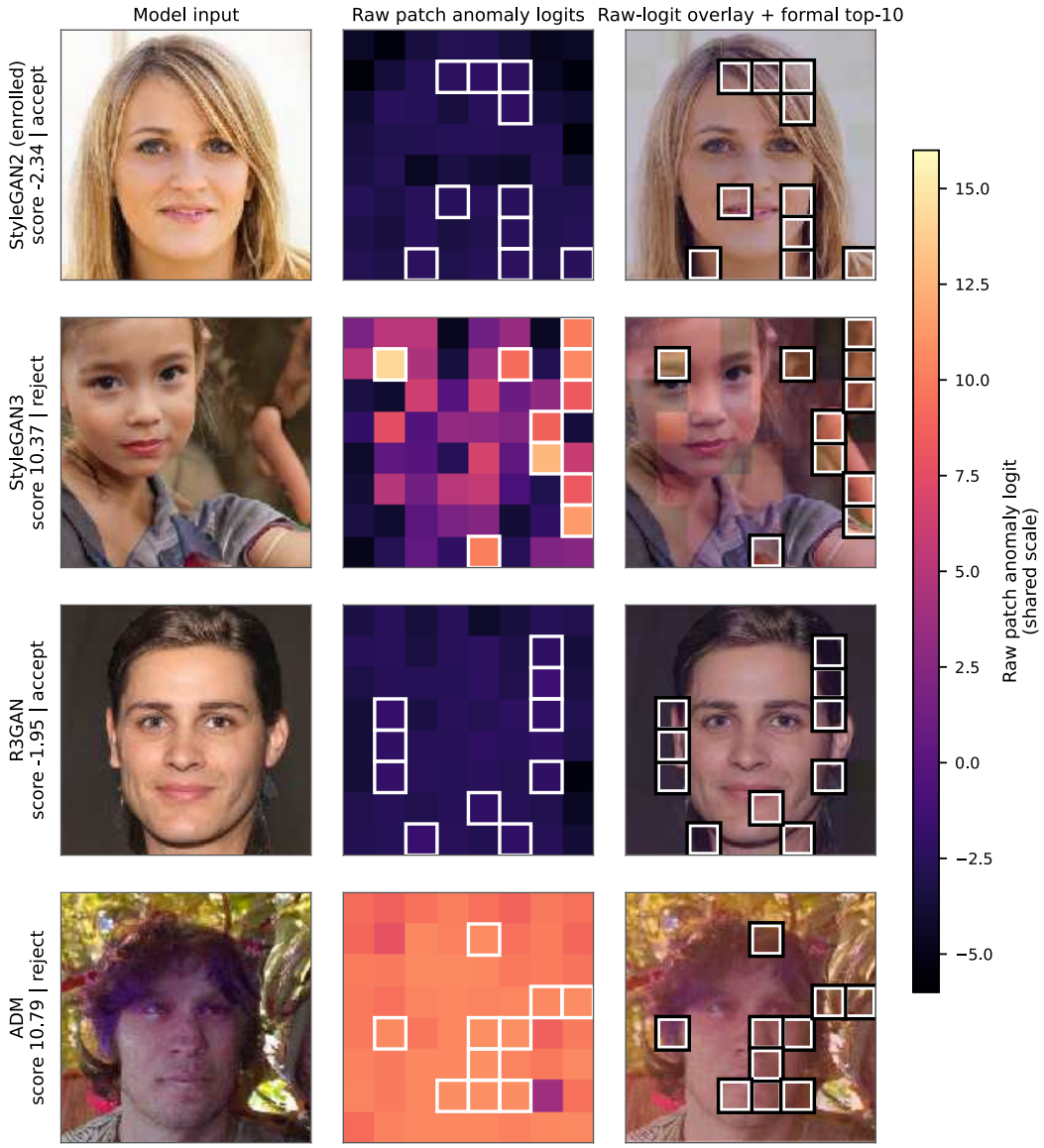

Figure E.1: Representative FARE patch evidence. Each row shows the model input, raw patch anomaly logits on a shared scale, and the logits overlaid on the input. White boxes mark the ten patches averaged by the image-level decision rule. The examples include the enrolled StyleGAN2 generator and substituted StyleGAN3, R3GAN, and ADM generators. The R3GAN image is accepted despite being a substitute, illustrating a false negative.

F Prompt-Diversity Check for Stable Diffusion

The prompt-diversity check evaluates whether FARE remains effective when the prompt distribution at verification time differs from the distribution used during enrollment and calibration. We draw prompts from PartiPrompts and split them into disjoint buckets based on the dataset's `Category` field. For each run, training and calibration use prompts from a single category, while verification uses prompts drawn from all remaining categories. Table F.1 reports four such single-category enrollment settings: "Animal", "Artifacts", "World-knowledge", and "People". Results are averaged over the Stable Diffusion version-pair substitutions used in the WF-SDVer scenario.

Across the four held-out category splits, FARE reaches 97.74–99.27% TPR at 1% FPR. These results show that FARE remains effective under the evaluated category-level prompt shifts. They do not cover more extreme content such as text-heavy images, diagrams, medical-style images, abstract art, or unusual textures, and they do not by themselves establish that the detector is independent of image semantics.

Table F.1: Prompt-diversity robustness for Stable Diffusion. Results are reported as TPR@1%FPR under held-out prompt distributions

Train/cal. prompt split	Verification prompt split	TPR@1%FPR
"Animal"	Other prompts	$97.83^{\pm 2.25}$
"Artifacts"	Other prompts	$99.27^{\pm 1.03}$
"World-knowledge"	Other prompts	$97.74^{\pm 1.40}$
"People"	Other prompts	$98.68^{\pm 1.79}$

G Reproducibility

Our codebase for reproducing FARE is publicly available at https://github.com/kaikaiyao/FARE. It provides the verifier, training, calibration, scoring and PGD attack utilities, method configurations, unit tests, and a training and scoring example. The README provides installation and data preparation instructions, together with links to the publicly available FFHQ, DiffusionDB, and CommunityForensics resources. Generator choices, data splits, hyperparameters, and evaluation protocols are specified in this paper and appendix.

Table G.1: Reproduction details: software, hardware, and code publication

Item	Value
Software stack	Python 3.11.10, PyTorch 2.5.1, CUDA 12.1, torchvision 0.20.1, diffusers 0.37.0, transformers 5.3.0, scikit-learn 1.8.0, pandas 3.0.1
GPU	NVIDIA H200
Code release	https://github.com/kaikaiyao/FARE (core implementation)

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the post-certification auditing problem, the output-only verifier, the low-FPR/adversarial evaluation scope, and the main empirical claims; these are supported by Sections 2–6 and the limitations discussed in Section 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 discusses the main limitations, including subtle same-checkpoint variants, the statistical nature of verification, reliance on batch verification at stricter FPRs, and the limited coverage of evaluated exact-model and decision-only attacks.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions. The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs.
   - The authors should reflect on the factors that influence the performance of the approach.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Appendix E.2 states the fixed-scorer, independent-sample, and continuous-score assumptions for the finite-sample calibration law and derives it using uniform order statistics. The remaining mathematical content defines the method and algorithms.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.

- Any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The method, calibration rule, algorithms, evaluation scenarios, baseline implementations, hyperparameters, architecture, attack settings, and reproducibility details are specified in Sections 4–5 and Appendices A–G.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. In general, releasing code and data is often one good way, but reproducibility can also be provided via detailed instructions, access to a hosted model, release of a checkpoint, or other appropriate means.
   - NeurIPS does not require releasing code, but it does require all submissions to provide some reasonable avenue for reproducibility, depending on the nature of the contribution.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our codebase for reproducing FARE is publicly available at `https://github.com/kaikaiyao/FARE`, with installation, data preparation, training, calibration, and scoring instructions. FFHQ, DiffusionDB, and CommunityForensics are publicly available from their original providers. The paper and supplementary material specify the generator choices, data splits, hyperparameters, and evaluation protocols.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines for more details.
   - While release of code and data is encouraged, [No] is an acceptable answer when release is not possible. Papers cannot be rejected simply for not including code, unless this is central to the contribution.
   - The instructions should contain the exact command and environment needed to reproduce the results.
   - The authors should provide instructions on data access and preparation, including raw data, preprocessed data, intermediate data, and generated data, as applicable.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines, or state which ones are omitted and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions, if applicable.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

Justification: Sections 5 and Appendices A–B specify the data splits, calibration protocol, baselines, model architecture, optimizer, learning rate, batch size, training iterations, patch size, aggregation parameter, and attack hyperparameters.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Tables 1 and 3 report means with standard deviations for the main results, and the appendix reports mean and standard deviation for ablations and prompt-diversity checks.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated.
- The method for calculating the error bars should be explained.
- The assumptions made should be given.
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- If error bars are reported in tables or plots, the authors should explain how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix G and Table G.1 report the reproduction software stack and the GPU worker used for the experiments. The public repository linked in Appendix G provides the core FARE implementation and method configurations.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for individual experimental runs and estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper.

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: The research is framed as an auditing method for accountable deployment of generative AI, uses generated images and existing public/open model assets, and does not describe human-subject experiments or any deviation from the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity, e.g., if there is a special consideration due to laws or regulations in their jurisdiction.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Sections 1, 2, and 7 discuss positive accountability/governance impacts as well as security-relevant risks, including malicious provider behavior, adversarial attempts to evade verification, and limitations of the statistical verifier.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses, fairness considerations, privacy considerations, and security considerations.
    - If there is a direct path to negative applications, the authors should point it out.
    - The authors should consider harms that could arise when the technology is used as intended and functioning correctly, when it is used as intended but gives incorrect results, and from intentional or unintentional misuse.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new high-risk image generator, language model, or scraped dataset; the described release is code for an auditing verifier.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model.
    - Datasets scraped from the Internet could pose safety risks; authors should describe how they avoided releasing unsafe images.
    - Providing effective safeguards can be challenging, but authors are encouraged to make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The manuscript credits the existing generators, datasets, forensic baselines, and implementation sources used in the evaluation through the corresponding citations in Sections 3 and 5 and Appendix A.2. The public repository links to the underlying methods and dependencies, whose licenses and terms continue to apply; it does not redistribute third-party datasets or pretrained weights.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license should be included for each asset.
- For scraped data from a particular source, the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new asset is the public core FARE implementation at https://github.com/kaikaiyao/FARE. Its README documents installation, input preparation, training, calibration, scoring, and release limitations. Appendix G describes the release and experimental environment.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates, including details about training, license, limitations, and related metadata.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets, if applicable. You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not report crowdsourcing experiments or new research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution involves human subjects, as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not report crowdsourcing experiments or new research with human subjects, so IRB or equivalent review is not applicable based on the manuscript contents.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If approval was obtained, it should be clearly stated in the paper.
- Procedures may vary significantly between institutions and locations, and authors are expected to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity, such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not described as an important, original, or non-standard component of the core method; the experiments concern image-generator auditing and forensic verification. LLMs and AI agents were used only for language editing and basic coding assistance.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to the LLM policy in the NeurIPS handbook for what should or should not be described.