MANIFOLD INDUCED BIASES FOR ZERO-SHOT AND FEW-SHOT DETECTION OF GENERATED IMAGES

Anonymous authors
Paper under double-blind review

ABSTRACT

Distinguishing between real and AI-generated images presents a timely and significant challenge. Despite extensive research in the (semi-)supervised regime, only recently, zero-shot and few-shot solutions have emerged as promising approaches to this task: They alleviate the ongoing data maintenance, which quickly becomes outdated due to advances in generative technologies. We identify two main gaps: (1) a lack of theoretical grounding for the methods, and (2) significant room for performance improvements in zero-shot and few-shot regimes. Our approach is founded on understanding and quantifying the biases inherent in generated content, where we use these quantities as criteria for characterizing generated images. Specifically, we explore the biases induced by the implicitly learned manifold of a pre-trained diffusion model: Through score-function analysis, curvature and gradient of the probability manifold are approximated in the zero-shot setting - yielding a scalar criterion for classification. We further extend our contribution to the few-shot setting by employing a mixture-of-experts methodology. Empirical results across 20 generative models demonstrate that our method outperforms current approaches in both zero-shot and few-shot settings. This work advances the theoretical understanding and practical usage of generated content biases through the lens of manifold analysis.

1 INTRODUCTION

Recent advancements in generative models, particularly diffusion-based techniques, have resulted in the creation of synthesized images that are increasingly difficult to distinguish from authentic ones. This poses significant challenges in content verification, security, and combating disinformation, driving the demand for reliable mechanisms to detect AI-generated images.

A wide array of contemporary research has focused on this task, employing methods ranging from standard convolutional neural networks (CNNs) (Wang et al., 2020; Baraheem & Nguyen, 2023; Epstein et al., 2023; Bird & Lotfi, 2024) to approaches that distinguish hand-crafted and learned characteristics (Bammey, 2023; Martin-Rodriguez et al., 2023; Zhong et al., 2023; Wang et al., 2023; Tan et al., 2024; Chen et al., 2024). Despite these efforts, there is a consensus on the critical importance of generalization to unseen generative techniques in this field (Bontcheva et al., 2024): These techniques evolve quickly, presenting substantial challenges in maintaining up-to-date generated datasets, which are crucial for supervised detection methods. Methods that generalize well to unseen generative techniques alleviate the need for constant data collection and retraining.

Targeted efforts to enhance such generalization have been actively pursued in a specific semi-supervised setting, where models are trained on one generative technique and evaluated on another (Ojha et al., 2023; Sha et al., 2023). Notably, these methods still require data consisting of hundreds of thousands of diverse generated images. In recent months, zero-shot and few-shot techniques have emerged for this task (Cozzolino et al., 2024; Ricker et al., 2024; He et al., 2024). Zero-shot methods use pre-trained models to solve tasks they were not trained for without designated (or any additional) training. Few-shot methods employ a similar tactic but involve minimal data to adapt the pre-trained model (e.g. incorporating a lightweight classifier). While these methods hold great potential by eliminating the need for extensive training and data maintenance, there remains significant room for improvement in terms of operating on a wide array of (unseen) generative techniques. Moreover, current methods lack theoretical grounding and rely primarily on empirical experimentation - a

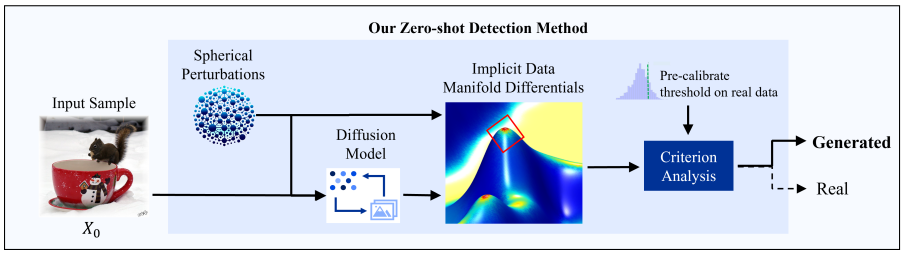

Figure 1: The proposed zero-shot detection pipeline, which circumvents the need for generated data. An input image x_0 is subjected to a pre-trained diffusion model and spherical perturbations. This sets the stage for our mathematical characterization of x_0, resulting in a criterion for the detection task.

surprising gap given the success of generated text detection methods like the widely recognized Mitchell et al. (2023), which provides theoretical motivation through curvature-based criteria.

This work advances solutions to these gaps. It is grounded in theoretical understanding - capitalizing on the diffusion model's implicitly learned probability manifold p to quantify inherent biases of its generated outputs. **Our main hypothesis is that such bias-driven quantities can serve as criteria to detect generated content**. We present novel derivations aimed at quantifying the stability of output images along the diffusion models' generation process, yielding a new stability criterion. By design, these stable points correspond to images the model is biased to produce. We evaluate this criterion for detecting generated images in both zero-shot and few-shot settings. For our criterion, we leverage diffusion models' ability to approximate the score function, expressed as

$$S = \nabla \log p, \tag{1}$$

to explore novel ways of analyzing the manifold defined by a function surface $\log p$. To this end, we consider the (hyper) surface curvature H [1]

$$H \propto \nabla \cdot \left(\frac{\nabla \log p}{|\nabla \log p|} \right), \tag{2}$$

see details on this formulation and its relation to the (negative) subgradient of the total-variation energy in Kimmel et al. (1997); Aubert et al. (2006). Note that unlike S, it is not straightforward to access H. In this work, we develop mathematically-founded ways to access such properties, and devise a first-of-its-kind curvature-based zero-shot framework for generated image detection (Fig. 1). Extended capabilities to the few-shot regime are provided as well.

Key Contributions:

- We establish a theoretical framework by integrating manifold analysis with diffusion model score functions, introducing a novel, bias-driven criterion for distinguishing real and generated images. This sets the foundation for further theoretical investigations in the domain.
- We propose the first zero-shot analysis of pre-trained diffusion models for generated image detection. Remarkably - this analysis demonstrates excellent generalization to unseen generative techniques.
- Our method demonstrates superior performance over existing approaches in both zero-shot and few-shot settings, validated through comprehensive experiments on a diverse dataset of approximately 200,000 images across 20 generative models.

These contributions advance both the theory and application of the detection capabilities in image generation technologies. To reproduce our results, see our official implementation[2].

2 Related Works

The evolution of detecting AI-generated image technologies has primarily relied on supervised learning methodologies. The common approach utilizes standard CNNs trained on a mix of real and

[1] Various high-dimensional curvature definitions exist, this is our choice.
[2] `https://tinyurl.com/zeroshotimplementation`

generated images to distinguish between them, such as Wang et al. (2020); Baraheem & Nguyen (2023); Epstein et al. (2023); Bird & Lotfi (2024). Building on this, subsequent research by Bammey (2023); Martin-Rodriguez et al. (2023); Zhong et al. (2023); Wang et al. (2023); Tan et al. (2024); Chen et al. (2024) advanced these techniques by identifying and integrating key phenomenological features that enhance the distinction between real and generated images. Yet, these approaches rely on extensive generated image datasets from various generation techniques, limiting their generalizability to generated images from unseen generative techniques Ojha et al. (2023).

Recent studies have proposed alternative methods to enhance generalization in detecting images generated by unseen generative techniques. Those unsupervised and semi-supervised methods Zhang et al. (2022a;b); Qiao et al. (2024); Cioni et al. (2024) aim to reduce the reliance on extensive labeled datasets; however, they still rely on access to generative methods during training, leading to biases towards those generation techniques. Ojha et al. (2023) and Sha et al. (2023) introduced a notable approach to improving generalizability by using CLIP's feature space to analyze content from a limited set of generative techniques, achieving unmatched generalization to unseen methods. Cozzolino et al. (2024) extends this approach and capitalizes on generalization in the low-data regime. Using few-shot analysis with pre-trained CLIP, they outperform existing methods.

To eliminate data maintenance and training altogether, zero-shot approaches emerged for generated image detection. To the best of our knowledge, Ricker et al. (2024) and He et al. (2024) are the only such methods. The former employed a pre-trained auto-encoder (AE) for out-of-distribution analysis - a well-established technique, e.g. An & Cho (2015). Pre-trained on real images, the AE is expected to encode-decode them better than generated images. Therefore, the reconstruction error is their criterion for detection. He et al. (2024) compares the image representation similarity between an image and its noise-perturbed counterpart. They offer novel insights into the desired qualities of pre-trained representations for the detection task. Our proposed detection method is the first to analyze diffusion models in the zero-shot setting for the detection task, showing unprecedented generalizability to unseen techniques and outperforming both Ricker et al. (2024) and He et al. (2024) in terms of operating on a wide array of (unseen) generative techniques.

3 Preliminaries

3.1 Diffusion Model Setting

Diffusion models are generative models that produce high-quality samples in various domains. They operate by an iterative generation process of noise reduction based on a pre-set noise schedule. Let the data manifold be $\Omega \subset \mathbb{R}^d$, where d is the data dimensionality, and denote a sample $x \in \Omega$. Each iteration t involves denoising a noised signal x_t via a neural network $f(x_t, t; \theta)$ (θ are the tuneable weights), subsequently progressing to x_{t-1}. This sequence begins at $t = T$ by sampling a tensor of iid normally distributed entries i.e. $x_T \sim \mathcal{N}(0, I)$, and terminates at $x_0 \in \Omega$, representing the final output. In our setting x_0 is an image. This generation process is known as *reverse diffusion*, where during training, f is optimized to reverse a *forward diffusion* process, defined using scheduling parameter $\alpha_t \in \mathbb{R}^+ \, \forall t$, and noise $\epsilon \sim \mathcal{N}(0, I)$ as follows

$$x_t = \sqrt{1 - \alpha_t} x_0 + \sqrt{\alpha_t} \epsilon. \tag{3}$$

3.2 Score-Function in Diffusion Models

The score-function is defined as $\nabla \log p(x)$, where $p(x)$ is the probability of x. Let $p_{\alpha_t}(x_t)$ be the probability of x_t considering Equation (3). The founding works of today's diffusion models Song & Ermon (2019; 2020); Kadkhodaie & Simoncelli (2021) capitalize on Equation (3), analyzing it from a score-function perspective based on the following seminal result by Miyasawa et al. (1961)

$$\nabla \log p_{\alpha_t}(x_t) = \frac{1}{\alpha_t} \left(\sqrt{1 - \alpha_t} \mathbf{E}_x[x_0 | x_t] - x_t \right), \tag{4}$$

where $\mathbf{E}_x[x_0|x_t]$ is the Minimum Mean-Squared-Error (MMSE) denoiser of x_t. It is replaced with the output of a denoising model $\hat{x}_0 = f(x_t, t; \theta)$, for which

$$\nabla \log p_{\alpha_t}(x_t) \approx \frac{1}{\alpha_t} \left(\sqrt{1 - \alpha_t} f(x_t, t; \theta) - x_t \right). \tag{5}$$

Often, $f(x_t, t; \theta)$ predicts noise, i.e. $\hat{x}_0 = \frac{1}{\sqrt{1-\alpha_t}}\left(x_t - \sqrt{\alpha_t} f(x_t, t; \theta)\right)$. Finally, replacing $f(x_t, t; \theta)$ with the true x_0 we have $\nabla \log p_{\alpha_t}(x) = -\frac{1}{\sqrt{\alpha_t}}\epsilon$, i.e. Equation (5) approximates ϵ up to a known factor. It is straightforward that, despite Ω being a zero-measure of $\mathbb{R}^d$ (Ω is assumed to have a dimension much lower than d), the probability of x_t is non-zero on the entire $\mathbb{R}^d$ space.

With $\nabla \log p$, the generation process can be simulated as Itô's SDE Ito et al. (1951),

$$\dot{x}(\tau) = \nabla_x \log p(x(\tau)) + \sqrt{2}\mathbf{w}_\tau, \tag{6}$$

where $\dot{x}(\tau)$ is the time derivative of $x(\tau)$, and $\mathbf{w}_\tau$ the time derivative of Brownian motion $\mathbf{w}(\tau)$, i.e. it injects noise to the process. In Song & Ermon (2019), a generative process that accounts for a p_{α_t} that changes in time was introduced - generalizing Equation (6). With that said, this paper employs a fixed-point analysis of p_{α_t} for a fixed α_t. Remark: The commonly used time here is reversed, i.e. large τ implies small t. The consensus in diffusion models research and this paper is to reliably use t.

4 Method

Here we present our mathematical perspective, derivations and practical choices for detection. For analyzeable and illustrative cases, demonstrating derivations and assumptions - see Figs. 3, 4(b-c).

4.1 Key Quantities and Criterions in our Fixed-point Geometric Analysis

In the setting of Sec. 3, a generated sample x_0, following Equation (6), is expected to be near a stable local maximum in the learnt log probability manifold - and x_0 to be a point of positive curvature and low gradient. Conversely, real data points that are unlikely to be generated will not exhibit these characteristics. Essentially, the learned manifold is expected to be "bumpier" than the actual data manifold, with generated data appearing as peaks on this bumpy surface. For a graphical illustration of this idea, refer to Fig. 2. To test x_0 for these characteristics, we work in its local neighbourhood. Let us employ a fixed-point analysis, "freezing" the generative process at a small fixed t. We assume α_t is small enough for p_{α_t} to approximate the data distribution and large enough for p_{α_t} to be smooth. Since t is fixed, we use α (without t) from now on. Relying on the smoothness of p_α, we can use $\log p_\alpha$ to construct a d-manifold embedded in $\mathbb{R}^{d+1}$ as a parametric hyper-surface of the form $(x, \log p_\alpha(x))$, for which the total-variation curvature of Equation (2) applies.

Denote B_0 the local neighbourhood of x_0 and ∂B_0 as its boundary, and their respective volumes $|B_0|$, $|\partial B_0|$. Let $\langle \cdot, \cdot \rangle$, $\| \cdot \|_2$ denote the Euclidean inner product and norm. A gradient criterion

$$D(x_0) := \frac{1}{|\partial B_0|} \int_{\partial B_0} \|\nabla \log p_\alpha(x)\|_2 dx, \tag{7}$$

will be employed, as well as a curvature criterion

$$\kappa(x_0) := \frac{-1}{|B_0|} \int_{B_0} \nabla \cdot \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2} dx. \tag{8}$$

Note the minus sign, which ensures that an inward pointing gradient (negative divergence) is associated with positive curvature, and vice versa. We choose B_0 to be the ball

$$B_0 = \{x : \|\sqrt{1-\alpha}x_0 - x\|_2 < \sqrt{d\alpha}\}. \tag{9}$$

This B_0 is practical: Its spherical boundary ∂B_0 (and its close neighbourhood) is highly probable under p_α, and the score function on ∂B_0 can be approximated via the diffusion model at a fixed t (Equation (5))[3] [4]. Additionally, ∂B_0 is easily sampled as

$$\tilde{x} = \sqrt{1-\alpha}x_0 + \sqrt{\alpha}u_d, \tag{10}$$

where $u_d \sim \text{Unif}(\mathcal{S}^{d-1}(\sqrt{d}))$ - a uniform distribution on the $(d-1)$-dimensional sphere centered at $\vec{0}$ with radius $\sqrt{d}$. Clarification: $\nabla \log p_{\alpha_t}(\cdot)$ can be applied to samples of $\tilde{x}$, yet the score function will be calculated for a sampled x_t, namely Equation (4), Equation (5) still hold, i.e.

$$\nabla \log p_\alpha(\tilde{x}) = \frac{1}{\alpha}\left(\sqrt{1-\alpha}\mathbf{E}_x[x_0|\tilde{x}] - \tilde{x}\right) \approx \frac{1}{\alpha}\left(\sqrt{1-\alpha}f(\tilde{x}, t; \theta) - \tilde{x}\right) := h(\tilde{x}), \tag{11}$$

[3] This approximation is the *concentration of measure*, further reading: Giannopoulos & Milman (2000).

[4] To ensure $x_0 \in B_0$ we require $1 - \sqrt{1-\alpha} < \sqrt{d\alpha}\|x_0\|_2$.

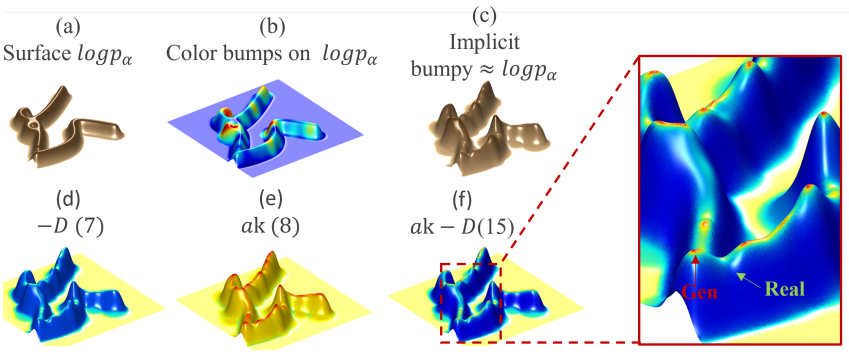

Figure 2: Toy probability surface: Simulation of toy data probability in a two-dimensional space ($d = 2$), structured along a one-dimensional manifold (Ω is a curve). (a) The log probability surface of perturbed samples, considering a uniform probability on the Ω curve. (b) A simulation of the hypothesis that generative models learn a bumpy version of the manifold: Bumps are randomly assigned to the manifold and visualized in color on the original surface. (c) The resulting bumpy surface. (d) Gradient magnitude of the bumpy manifold. (e) Total-variation curvature of the bumpy manifold. (f) Demonstrates the differential property derived from our analysis, highlighting locally maximal regions of the bumps which correspond to likely generated data points. We mathematically establish a way to capture this property through a zero-shot analysis of the diffusion model.

where $h(\tilde{x})$ denotes the diffusion-model approximation of the score function. Notice the relation between $\tilde{x}$, constructed with $\alpha \leftarrow \alpha_t$ and x_t: As d increases, the probability of $\|\epsilon\|_2$ (ϵ of Equation (3)) is concentrated around its mean $\sqrt{d}$, reducing the norm's stochasticity - making u_d and ϵ (and as a consequence $\tilde{x}$ and x_t) interchangeable in high dimension d Laurent & Massart (2000), see Fig. 4. [5]

4.2 Mathematical Claims: Accessing Key Quantities and Criterions

Claim 1. *Given an image x_0 and a sample $x \sim \tilde{x}|x_0$, drawn according to Equation (10), we denote $u_d(x) = \frac{x-\sqrt{1-\alpha}x_0}{\sqrt{\alpha}}$ [6] and set $\gamma = \frac{|\partial B_0|}{\sqrt{d}|B_0|}$. Then the following relation holds:*

$$-\mathbf{E}_{x\sim\tilde{x}|x_0}\langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \gamma u_d(x) + \nabla \log p_\alpha(x)\rangle = \kappa(x_0) - D(x_0). \tag{12}$$

This provides a characterization of x_0 as a stable maximal point under the backward diffusion process, Equation (6), quantifying both gradient magnitude (should be low) and curvature (should be high) aspects.

Proof Outline (full proof in the Appendix D)

First we use Gauss divergence theorem for the curvature term

$$-|B_0|\kappa(x_0) = \int_{B_0} \nabla \cdot \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2} dx = \int_{\partial B_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \hat{n}\rangle dx, \tag{13}$$

where $\hat{n}$ denote the outward-pointing normals to the sphere ∂B_0. Using the properties of $\tilde{x}_t|x_0$, which is uniformly distributed on ∂B_0, we have that $\mathbf{E}_{x\sim\tilde{x}|x_0}(\cdot) = \frac{1}{|\partial B_0|}\int_{\partial B_0}(\cdot)$. Moreover, by construction $\hat{n} = \frac{x-\sqrt{1-\alpha}x_0}{\|\sqrt{1-\alpha}x_0 - x\|_2} = \frac{u_d(x)}{\sqrt{d}}$, $\forall x \in \partial B_0$. Hence we get

$$\kappa(x_0) = -\frac{|\partial B_0|}{\sqrt{d}|B_0|}\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, u_d(x)\rangle. \tag{14}$$

[5] Thus, model trained to denoise x_t, will perform well for $\nabla \log p_{\alpha_t}(\tilde{x}_t)$ in the sense of Equation (5), since $\tilde{x}_t$ samples from the highest-probability sub-sphere of $p(x_t)$. Infact, in our high-dimensional setting, sampling x_t that is far away from ∂B_0 is highly improbable.

[6] The notation of u_d as a function of x, i.e. $u_d(x)$, expresses the fact that $u_d(x)$, x are constructed as a pair drawing the same noise. We use this when explicitly formulating expectation as an integral, where it is crucial to decide on one variable (u_d or x) to integrate upon

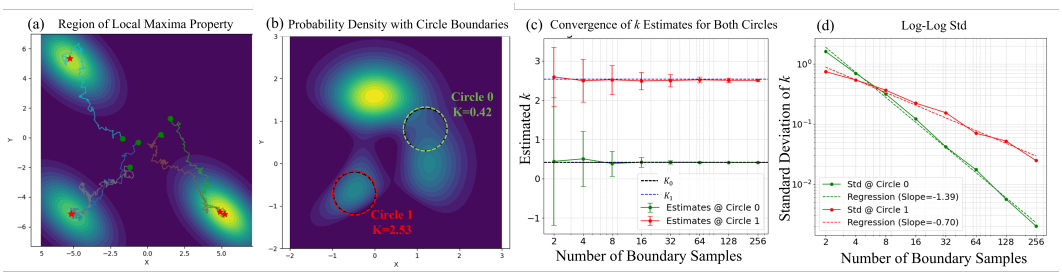

Figure 3: **a) The Local Maxima Region Property.** We trained a diffusion model on a 3-modal Gaussian Mixture Model (GMM) (details in Appendix B). The colormap shows the learned PDF, with reverse diffusion trajectories overlaid. Starting points (green circles) converge toward local maxima of the probability, confirming our assumption that the generation process ends near stable local maxima (red stars). For statistics at scale see Fig. 7 **b) Expected Behavior of the Curvature Criterion κ.** We compute κ on two marked circles centered at local maxima and saddle points of a differentiable analytic function (details in Appendix A). As expected, κ is higher for the local maxima. **c) Error Analysis of κ Estimators**. We experiment with both κ values from b), and approximate them with increasing no. of spherical samples based on (Equation (13)). We average 100 runs and show std as error bars. Results confirm reliability: 1) The mean remains close to the true value even with few samples (unbiased estimator). 2) Separation between maxima and saddle points is maintained, even with as few as 4 samples. **d) Consistency and Convergence of κ Estimators.** The standard deviation of κ estimators is plotted against the number of spherical boundary samples in a log-log plot. Linear regression is applied to quantify the rate of convergence, showing a good fit with negative regression slopes, confirming exponential convergence. Combined with the empirical unbiasedness demonstrated in b), this establishes that the κ estimators are empirically consistent.

We then use the same properties of the uniform distribution $\tilde{x}_t|x_0$, and obtain $D(x_0) = \mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \nabla\log p_\alpha(x)\rangle$. Finally - by linearity, and since $\gamma = \frac{|\partial B_0|}{\sqrt{d}|B_0|}$, we have

$$-\mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \gamma a u(x) - \nabla\log p_\alpha(x)\rangle = a\kappa(x_0) - D(x_0), \tag{15}$$

where $a = 1$ for the choice of that γ.

Corollary 2. *In the setting of claim 1, we furthermore have the following approximation*

$$\frac{-\sqrt{1-\alpha}}{\alpha}\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \hat{x}_0\rangle \approx a\kappa(x_0) - D(x_0). \tag{16}$$

Proof Outline (full proof in the Appendix D)

By linearity we decompose the expectation to summands via Equation (11). We have

$$\mathbf{E}_{x\sim\tilde{x}|x_0}\left(\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}\right) \approx 0, \tag{17}$$

since integration of normals over the sphere is zero, and $\nabla\log p_\alpha(x)$ approximates the uniform spherical noise. Thus, $\mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \sqrt{1-\alpha}x_0\rangle \approx 0$, since x_0 is deterministic. From here, dividing the (remaining two) summands by α leads to Equation 16, with $\gamma a = \frac{1}{\sqrt{\alpha}}$, and we are done.

Corollary 3. *Let b_0 represent the statistical bias of predictor $\hat{x}_0$, defined as $b_0 = x_0 - \mathbf{E}_{x\sim\tilde{x}|x_0}(\hat{x}_0)$. Transitioning from score-function to the denoising perspective of diffusion models, the summand which is approximately zero in Corollary 2, given by $\mathbf{E}_{x\sim\tilde{x}|x_0}\left\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, x_0\right\rangle$, satisfies*

$$\frac{\alpha}{\sqrt{1-\alpha}}\mathbf{E}_{x\sim\tilde{x}|x_0}\left\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, x_0\right\rangle \approx -\langle b_0, x_0\rangle, \tag{18}$$

see proof in the Appendix. This is yet another quantity, capturing bias towards generating x_0, where:

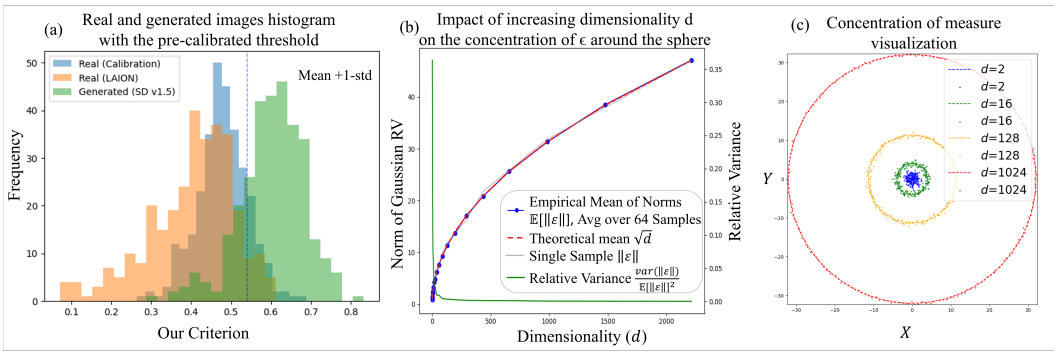

Figure 4: (a) We calibrate a decision threshold based on the mean and standard deviation of 1,000 real image criteria, ensuring it's free from generated data influence. Criteria from another dataset's real and generated images are also displayed. (b) As a result of $\|\epsilon\| \sim \chi$-distribution, as d increases, ϵ concentrates around a spherical *thin shell*, with radii $\sqrt{d}$. This demonstrates interchangeable use of x_t and $\tilde{x}$ in high-dimensional space. (c) 2D Visualization of the Concentration of Measure: For each d, the radii and samples' are set as the d-dimensional $\mathbb{E}\|\epsilon\|$ and $var(\|\epsilon\|)$ respectively. Correspondingly, the radii increases while the variance converges with d, effectively simulating the phenomenon in 2D.

1. *This summand equals zero for unbiased noise predictors.*
2. *If not set to zero, higher values occur when b is (anti-)correlated with x_0, indicating a bias in the (negative) noise prediction towards the clean image. It is intuitive that the noise prediction steers the iteratively denoised generated image towards its inherent biases - expressed by this summand.*
3. *In score function analysis settings, the noise predictor is approximated by an MMSE denoiser, which is unbiased, indeed leading to this summand being zero.*

4.3 Numerical Formulation and Best Practices

Given x_0, we approximate the criterion $a\kappa(x_0) - D(x_0)$ as expressed in Corr. 2. The implementation, illustrated in Fig. 1, involves the following steps: **1) Sampling Perturbations:** We generate s spherical perturbations to x_0 to produce samples $\{\tilde{x}^{(i)}\}_{i=1}^{s}$ according to Eq. Equation (10). These perturbations simulate variations around x_0 and are crucial for approximating the expectation. **2) Noise predictions:** Feed each perturbed sample $\tilde{x}^{(i)}$ to the diffusion model of choice to obtain noise predictions $h(\tilde{x}^{(i)})$ of Eq. Equation (11). **3) Criterion:** Compute:

$$C(x_0) := \frac{\sqrt{1-\alpha}}{\alpha} \frac{1}{s} \sum_{i=1}^{s} \left\langle \frac{-h(\tilde{x}^{(i)})}{\|h(\tilde{x}^{(i)})\|_2}, \hat{x}_0 \right\rangle \approx a\kappa(x_0) - D(x_0),$$

Practical choices. We map data and noise predictions to CLIP Radford et al. (2021) before calculating $C(x_0)$. When using stable diffusion, this mapping has 2 stages: 1) map from latent space to image space using stable diffusoin's decoder, 2) map to CLIP. For threshold calibration we recommend using real images only to avoid bias to certain generative techniques. We use as threshold the Empirical mean plus one-standard-deviation - as shown in Fig. 4. x_0 is classified as generated if $C(x_0)$ exceeds this threshold, and real otherwise. We can omit $\frac{\sqrt{1-\alpha}}{\alpha}$ - it does not depend on x_0.

Important take-away: $C(x_0)$ approximates manifold-bias criteria. Another surprising perspective is that it measures similarity between the predictions of noise and data. Nevertheless, this is a result of our mathematical derivations and is supported hereafter by thorough evidence.

5 Evaluation

In this section, we empirically validate our hypothesis: Bias-driven quantities can serve as a robust criterion for detecting generated images. Specifically, we test our mathematically derived criteria for stable points in the diffusion model's generation process.

Table 1: Comparison of zero-shot detection methods across various metrics. We report average AUC, AP, and Accuracy. Additionally, we include a top-10 generative technique performance, where for each detection method the 10 best-performing cases are selected, and all detection methods are compared on them. Our method significantly surpasses existing methods.

Model	AUC	AP	Accuracy	RIGID Top 10 Accuracy	AEROBLADE Top 10 Accuracy	Ours Top 10 Accuracy
RIGID	0.439	0.519	0.555	0.666	0.569	0.482
AEROBLADE	0.444	0.492	0.464	0.492	0.565	0.438
Ours	**0.835**	**0.832**	**0.741**	**0.678**	**0.739**	**0.839**

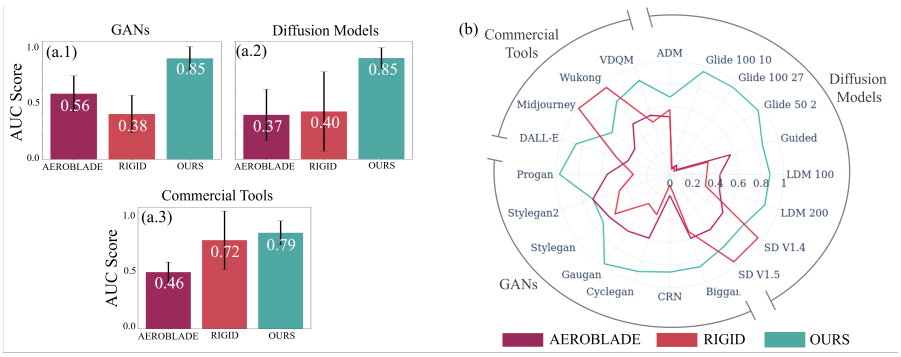

Figure 5: **Zero-shot comparison.** Plots $a.1$-$a.3$ demonstrate the superior AUC performance of our method across the three main generative technique groups. Error bars represent variability in AUC between techniques within each group, with our method showing the least variation. Plot b details AUC per technique, where our method achieves the highest scores in most cases. Although our criterion originates from a zero-shot analysis of an LDM model, we demonstrate good generalization to other techniques. Competitors show sensitivity to changes in technique, which hampers their generalization capabilities (clarified by detailed histograms in the Appendix, Fig. 8).

5.1 Experimental Settings

Datasets. To ensure a diverse representation of the generative techniques, our method is evaluated across three benchmark datasets within the domain of generated image detection. The **CNNSpot** Wang et al. (2020) dataset comprises real and generated images from 20 categories of the LSUN Yu et al. (2015) dataset, featuring images produced by over ten generative models, primarily GANs. The **Universal Fake Detect** Ojha et al. (2023) dataset extends CNNSpot with generated images from newer models, primarily diffusion models. The **GenImage** Zhu et al. (2023) dataset features images produced by commercial generative tools, including Midjourney. In total, our aggregated dataset consists of $100K$ of real images and additional $100K$ images produced from 20 different generation techniques Karras et al. (2017); Zhu et al. (2017); Karras et al. (2019); Dhariwal & Nichol (2021); Ramesh et al. (2021); Rombach et al. (2022); Midjourney (2024). For the complete list of generative models used in our evaluation, see Appendix F.1.

Implementation Details.We used Stable Diffusion 1.4 Rombach et al. (2022) as our diffusion model and LLaVA 1.5 Liu et al. (2023) for generating text captions required as input by this model. Criterion hyper-parameters were set as follows: 1) No. of spherical noises s was set to 64; 2) Perturbation strength $\alpha\sqrt{d} = 1.28$, determining B_0 radii and 3) A small scalar $\delta = 10^{-8}$ was added to the criterion denominator to ensure it is strictly positive. Code and datasets are detailed in Appendix C.

5.2 Empirical Cases and Results

To demonstrate the practicality of our method, we compare it against state-of-the-art zero-shot detection methods and test its effectiveness in a mixture-of-experts (MoE) setting combined with the leading few-shot technique. Ablation and sensitivity studies are provided as well.

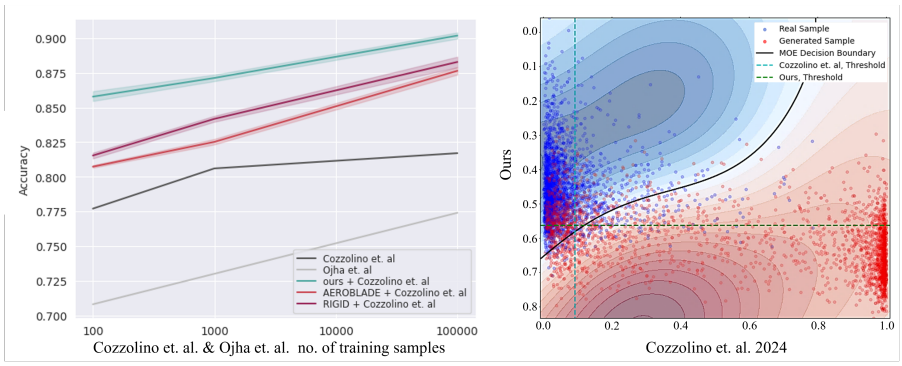

Figure 6: **Few-shot Performance**. Left - performance improvement in the MoE setting with Cozzolino et al. (2024), in a few-shot regime. We report results of MoE with ours vs other zero-shot methods, and the original Ojha et al. (2023); Cozzolino et al. (2024). Our efficacy proves to be the best. Right - scatter plot of our criterion and Cozzolino et al. (2024). The decision boundary was obtained in our MoE setting using SVM. Significant improvement of separability can be observed.

Zero-shot Comparison Across 20 **Generation Techniques**. In this experiment, we benchmark our method against two leading zero-shot image detectors: AEROBLADE Ricker et al. (2024) and RIGID He et al. (2024) under zero-shot settings. AEROBLADE Ricker et al. (2024) uses the reconstruction error for a given image obtained from a pre-trained variational autoencoder as the criterion for the detection task. RIGID He et al. (2024) compares the image representation of the original image and its noise-perturbed counterpart in a pre-trained feature space and uses their similarity as the detection criterion. Our implementation strictly adheres to the specifications detailed in their publications, utilizing their publicly available code. All methods used the same calibration set of $1K$ real images for threshold calibration and were evaluated with the test set described in Sec. 5.1, covering 20 diverse generation techniques to assess generalizability. More details in Appendix F.2.

Table 1 presents a comparison of zero-shot detection methods across key performance metrics. The results show the average score among all generative techniques with our method outperforms existing methods by a significant margin. Fig. 5 provides an in-depth analysis featuring bar plots that summarize outcomes across various groups of generative techniques, including GANs, diffusion models, and commercial tools like Midjourney and DALL-E. This is complemented by a high-resolution polar plot comparing per-generative technique performance. Note: Table 1 reports on the balanced test-set of Sec. 5.1. For the per-technique evaluation of Fig. 5 we have re-balanced the test sets per generative technique, making sure that each technique is evaluated on an equal number of real and generated images. Our method consistently outperforms AEROBLADE and RIGID across all groups and in the majority of generative techniques. Our competitors exhibit low AUC in generative techniques like Glide and CRN due to variations in their criteria across different generative techniques, as demonstrated in the per-technique histograms provided in the appendix, Fig. 8.

Mixture of Experts with Few-shot Approaches. While zero-shot scenarios often reflect real-world situations where data is unavailable or not worth managing, some cases may justify handling small amounts of generated data for significant performance gains. Recently, Cozzolino et al. (2024) introduced a few-shot detection method leveraging a pre-trained CLIP. They established new benchmarks for generalization to unseen techniques in data-limited scenarios, reducing data maintenance costs yet not eliminating it altogether. To prove the applicability of zero-shot methods in the few-shot regime, we integrate them into a mixture-of-experts (MoE) framework alongside few-shot approaches, enhancing performance while remaining in the few-shot bounds. Utilizing an extra small set of examples, we trained a lightweight classifier to combine the outputs from a zero-shot method and Cozzolino et al. (2024), forging a hybrid approach. In all MoE experiments, additional $1K$ labeled samples where used to train the light-weight classifier - these where randomly selected in an additional train-test split, implemented on the dataset initially used for zero-shot testing.

Several MoE cases where tested, each using a different light-weight classification algorithm. The random forest classifier excell ed, hence it is used for our quantitative analysis. For visualizing decision boundaries, we employ an SVM for its smoother and clearer geometric representation.

Table 2: Sensitivity and ablation analysis results. The table highlights the method's robustness across various configurations, including base models, number of perturbations S, spherical noise levels α (radii=$\alpha\sqrt{d}$), and image corruption techniques. The results show consistent performance with slight variations, highlighting the method's robustness. Base settings are: SD v1.4 Model, $S = 64$, $\alpha = 0.01$ ($\alpha\sqrt{d} = 1.28$) and without image corruption.

Exp.	**Number of Perturbations S (SD v1.4 Model)**					**Level of Spherical Noises α (SD v1.4 Model)**		
Setting	**4**	**8**	**16**	**32**	**64**	**α=0.01**	**α=0.1**	**α=1**
AUC	0.828	0.829	0.8308	0.833	**0.835**	**0.835**	0.82	0.815
Exp.	**Image Corruption Techniques (SD v1.4 Model)**					**Various Base Models**		
Setting	**Without**	**Jpeg compression**		**Gaussian blur**		**Sd v1.4**	**SD v2 base**	**Kandinsky 2.1**
AUC	**0.835**	0.79		0.822		**0.835**	0.831	0.826

Fig. 6 demonstrates that while all zero-shot methods enhance performance, our method consistently outperforms others. This also implies that, for users willing to invest in managing a small amount of data, our method serves as an easy-to-integrate plugin that enhances few-shot frameworks, offering a flexible trade-off between data availability and performance improvement.

Sensitivity and Ablation Analysis. We conducted evaluations to verify the robustness of our method across diverse configurations. This section provides key insights from these evaluations with the main results summarized in Table 2 and further details provided in the Appendix G.2. *Various stable diffusion models*. While our method focused on Stable Diffusion v1.4, here we also evaluated Stable Diffusion v2 Base and Kandinsky 2.1, which differ in size and technique. Results showed consistent performance with slight AUC decreases. *Various no. of perturbations S*. We explore different perturbation No. to assess their impact on the detection performance. The results reveal that increasing S consistently enhances detection performance, which aligns with our research thesis. *Various spherical noise levels*. We varied the spherical noise levels (i.e., radii). These adjustments resulted in AUC decreases of 1.5% and 2%. *Image corruption techniques*. In real-world scenarios, adversaries may use compression or blurring to obscure traces of image generation. To address this, we evaluated our method under JPEG compression (AUC drop of 3.45%) and Gaussian blur (Kernel Size = 3, AUC drop of 1.2%), demonstrating its robustness to such techniques.

6 Limitations

Our theoretical derivations and the resulting quantities are induced by the learned manifold biases of *the analyzed diffusion model*. While detection of images generated by this model is expected, interestingly, our method shows good detection of generations from other generative techniques, including completely different generative groups (Fig. 5). This cross-technique capability is a notable main strength, however there is no comprehensive theory to explain it. We hypothesize that different models, especially those trained on similar datasets, might exhibit similar characteristics within their probability manifolds and generated images Nalisnick et al. (2018); Kornblith et al. (2019). However, further research is necessary to substantiate such hypotheses.

7 Conclusion

This paper introduces a novel zero-shot framework for detecting AI-generated images, capitalizing on biases inherent to the implicitly learned manifold of a pre-trained diffusion model. By combining score function analysis with non-Euclidean manifold geometry, we advance the theoretical understanding of manifold biases, utilizing this knowledge to quantify discrepancies between real and generated images. Our main hypothesis - that such bias-driven quantities can effectively detect generated content - has proven viable: Empirical evaluations confirm that our method outperforms state-of-the-art zero-shot methods across benchmark datasets. Furthermore, we enhance few-shot performance without violating the few-shot regime - showing superior performance here as well. This work establishes a foundation for further research into detection via diffusion-model manifold biases. We also advocate for broader theoretical and practical investigations into other bias-driven quantities.

REFERENCES

Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, 2(1):1–18, 2015.

Gilles Aubert, Pierre Kornprobst, and Giles Aubert. *Mathematical problems in image processing: partial differential equations and the calculus of variations*, volume 147. Springer, 2006.

Quentin Bammey. Synthbuster: Towards detection of diffusion model generated images. *IEEE Open Journal of Signal Processing*, 2023.

Samah S Baraheem and Tam V Nguyen. Ai vs. ai: Can ai detect ai-generated images? *Journal of Imaging*, 9(10):199, 2023.

Jordan J Bird and Ahmad Lotfi. Cifake: Image classification and explainable identification of ai-generated synthetic images. *IEEE Access*, 2024.

Kalina Bontcheva, Symeon Papadopoulous, Filareti Tsalakanidou, Riccardo Gallotti, Noémie Krack, Denis Teyssou, Agent France-Presse, Luca Cuccovillo, and Luisa Verdoliva. Generative ai and disinformation: Recent advances, challenges, and opportunities. *European Digital Media Observatory*, 2024.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3291–3300, 2018.

Jiaxuan Chen, Jieteng Yao, and Li Niu. A single simple patch is all you need for ai-generated image detection. *arXiv preprint arXiv:2402.01123*, 2024.

Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1511–1520, 2017.

Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797, 2018.

Dario Cioni, Christos Tzelepis, Lorenzo Seidenari, and Ioannis Patras. Are clip features all you need for universal synthetic image origin attribution? *arXiv preprint arXiv:2408.09153*, 2024.

Davide Cozzolino, Giovanni Poggi, Riccardo Corvi, Matthias Nießner, and Luisa Verdoliva. Raising the bar of ai-generated image detection with clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4356–4366, 2024.

Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. Second-order attention network for single image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11065–11074, 2019.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

David C. Epstein, Ishan Jain, Oliver Wang, and Richard Zhang. Online detection of ai-generated images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 382–392, October 2023.

Apostolos A Giannopoulos and Vitali D Milman. Concentration property on probability spaces. Technical report, SCAN-0005095, 2000.

Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10696–10706, 2022.

Zhiyuan He, Pin-Yu Chen, and Tsung-Yi Ho. Rigid: A training-free and model-agnostic framework for robust ai-generated image detection. *arXiv preprint arXiv:2405.20112*, 2024.

Kiyosi Ito, Kiyosi Itô, Kiyosi Itô, Japon Mathématicien, Kiyosi Itô, and Japan Mathematician. *On stochastic differential equations*, volume 4. American Mathematical Society New York, 1951.

Zahra Kadkhodaie and Eero Simoncelli. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. *Advances in Neural Information Processing Systems*, 34:13242–13254, 2021.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

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*, pp. 4401–4410, 2019.

Kimmel, Sochen, and Malladi. From high energy physics to low level vision. In *International Conference on Scale-Space Theories in Computer Vision*, pp. 236–247. Springer, 1997.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMLR, 2019.

Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of statistics*, pp. 1302–1338, 2000.

Ke Li, Tianhao Zhang, and Jitendra Malik. Diverse image synthesis from semantic layouts via conditional imle. 2019 ieee. In *CVF International Conference on Computer Vision (ICCV)*, pp. 4219–4228, 2019.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023.

Fernando Martin-Rodriguez, Rocio Garcia-Mojon, and Monica Fernandez-Barciela. Detection of ai-created images using pixel-wise feature extraction and convolutional neural networks. *Sensors*, 23(22):9037, 2023.

Midjourney. Midjourney: An independent research lab exploring new mediums of thought. `https://www.midjourney.com/`, 2024. Accessed: 2024-05-18.

MindSpore. Wukong: A pre-trained model for chinese text-to-image generation. `https://xihe.mindspore.cn/modelzoo/wukong`, 2024. Accessed: 2024-05-18.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*, pp. 24950–24962. PMLR, 2023.

Koichi Miyasawa et al. An empirical bayes estimator of the mean of a normal population. *Bull. Inst. Internat. Statist*, 38(181-188):1–2, 1961.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*, 2018.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24480–24489, 2023.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2337–2346, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Tong Qiao, Hang Shao, Shichuang Xie, and Ran Shi. Unsupervised generative fake image detector. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pp. 8821–8831. Pmlr, 2021.

Jonas Ricker, Denis Lukovnikov, and Asja Fischer. Aeroblade: Training-free detection of latent diffusion images using autoencoder reconstruction error. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9130–9140, 2024.

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*, pp. 10684–10695, 2022.

Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1–11, 2019.

Zeyang Sha, Zheng Li, Ning Yu, and Yang Zhang. De-fake: Detection and attribution of fake images generated by text-to-image generation models. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pp. 3418–3432, 2023.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

Chuangchuang Tan, Ping Liu, RenShuai Tao, Huan Liu, Yao Zhao, Baoyuan Wu, and Yunchao Wei. Data-independent operator: A training-free artifact representation extractor for generalizable deepfake detection. *arXiv preprint arXiv:2403.06803*, 2024.

Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. Cnn-generated images are surprisingly easy to spot... for now. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8695–8704, 2020.

Zhendong Wang, Jianmin Bao, Wengang Zhou, Weilun Wang, Hezhen Hu, Hong Chen, and Houqiang Li. Dire for diffusion-generated image detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22445–22455, 2023.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Mingxu Zhang, Hongxia Wang, Peisong He, Asad Malik, and Hanqing Liu. Exposing unseen gan-generated image using unsupervised domain adaptation. *Knowledge-Based Systems*, 257: 109905, 2022a.

Mingxu Zhang, Hongxia Wang, Peisong He, Asad Malik, and Hanqing Liu. Improving gan-generated image detection generalization using unsupervised domain adaptation. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, 2022b.

Nan Zhong, Yiran Xu, Zhenxing Qian, and Xinpeng Zhang. Rich and poor texture contrast: A simple yet effective approach for ai-generated image detection. *arXiv preprint arXiv:2311.12397*, 2023.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

Mingjian Zhu, Hanting Chen, Qiangyu YAN, Xudong Huang, Guanyu Lin, Wei Li, Zhijun Tu, Hailin Hu, Jie Hu, and Yunhe Wang. Genimage: A million-scale benchmark for detecting ai-generated image. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 77771–77782. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/f4d4a021f9051a6c18183b059117e8b5-Paper-Datasets_and_Benchmarks.pdf.

APPENDIX

In this Appendix, we provide additional information relevant to proofs, experimental settings, and experimental results. This document is presented as follows:

- **Reproducibility and Code** – links to our code project and dataset, as well as additional details regarding the implementation of our zero-shot method in our evaluation procedure.
- **Full Proofs** – detailed steps and justifications of our theoretical findings, which further support the claims presented in the main paper.
- **Experimental Settings Additional Information** – description regarding our used dataset and implementations.
- **Experimental Results Additional Information** – An extended analysis of the experimental results from the main manuscript.

A EXPERIMENTAL DETAILS FOR κ ERROR ANALYSIS EXPERIMENTS

In this appendix, we provide the full experimental details to re-produce Fig. 3(b-d) for estimating the curvature criterion κ using spherical boundary samples. The experiments are designed to validate the approximation of κ through averaging over the sphere, leveraging the Gauss Divergence Theorem. The results demonstrate the effectiveness of our method in distinguishing between local maxima and saddle points of a differentiable analytic function.

We define a two-dimensional differentiable analytic function $f : \mathbb{R}^2 \to \mathbb{R}$, inspired by MATLAB's `peaks` function, which exhibits multiple peaks and valleys. The function is given by:

$$f(x, y) = \frac{1}{C}\left[3(1-x)^2 e^{-x^2-(y+1)^2} - 10\left(\frac{x}{5} - x^3 - y^5\right) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2}\right],$$

where C is a normalization constant ensuring that the integral of f over the domain is 1. Values of f below a threshold (e.g., 1×10^{-5}) are set to zero to maintain non-negativity.

We select two circles centered at specific points to evaluate κ. The first circle is centered at a local maximum $(1.2, 0.8)$, and the second is centered at a saddle point $(-0.475, -0.7)$. Both circles have a radius of $R = 0.5$. These points represent distinct features of the function f, allowing us to assess the sensitivity of κ to curvature differences.

For each circle, the true value of κ is computed using the volume integral of Equation (8). Let $B_R(\mathbf{c})$ denote the circle of radius R centered at $\mathbf{c}$. Numerical integration is performed by summing over the grid points inside each circle.

We approximate κ using discrete samples along the boundary of each circle, discretizing Equation (13). The number of boundary samples N_{boundary} is varied as $2, 4, 8, 16, 32, 64, 128, 256$. To reduce sampling bias, a random angular offset is introduced to the uniformly spaced boundary points in each run.

Boundary points are computed as:

$$(x_i, y_i) = (x_{\text{center}} + R\cos\theta_i, y_{\text{center}} + R\sin\theta_i),$$

where θ_i are the sampled angles. Gradients are interpolated at these boundary points, and the normalized gradient components are used to compute the dot product with the inward-pointing normal vector:

$$\mathbf{n}_{\text{in}} = -\frac{1}{R}\begin{pmatrix} x_i - x_{\text{center}} \\ y_i - y_{\text{center}} \end{pmatrix},$$

$$\left(\frac{\nabla f}{\|\nabla f\|} \cdot \mathbf{n}_{\text{in}}\right)_i = \left(\frac{\partial f}{\partial x}\right)_i^{\text{norm}} n_{x,i} + \left(\frac{\partial f}{\partial y}\right)_i^{\text{norm}} n_{y,i}.$$

The approximate κ is then computed as:

$$\kappa_{\text{approx}} = \sum_{i=1}^{N_{\text{boundary}}} \left(\frac{\nabla f}{\|\nabla f\|} \cdot \mathbf{n}_{\text{in}}\right)_i \Delta s, \quad \Delta s = \frac{2\pi R}{N_{\text{boundary}}}.$$

For each N_{boundary}, 100 independent runs are performed to compute the mean and standard deviation of κ_{approx}. Error bars represent the standard deviation.

The results validate the approximation, showing that the mean of κ_{approx} aligns with the true κ, demonstrating unbiasedness. The variance of κ_{approx} decreases as N_{boundary} increases, indicating improved accuracy with more samples. Even with low N_{boundary}, the estimated κ values for the local maximum and saddle point are distinctly separated within error margins, supporting the reliability of the method.

B Experimental Detals for the Local Probability Maxima Property Verification.

In this section, we present a the experimental details of Fig. 3 (a) to illustrate how generated samples from a diffusion model tend to converge to stable local maxima on the learned probability manifold. The learned manifold is approximated using Kernel Density Estimation (KDE) of the generated samples. Statistics at scale are provided below, Fig. 7.

We constructed a synthetic dataset by sampling from a Gaussian Mixture Model (GMM) in 2D with the following parameters: means $= \left\{\begin{pmatrix}-5\\-5\end{pmatrix}, \begin{pmatrix}0\\-5\end{pmatrix}, \begin{pmatrix}-5\\0\end{pmatrix}\right\}$., covariances $= \left\{\begin{pmatrix}0.1 & 0\\0 & 0.1\end{pmatrix}, \begin{pmatrix}0.1 & 0\\0 & 0.1\end{pmatrix}, \begin{pmatrix}0.1 & 0\\0 & 0.1\end{pmatrix}\right\}$. and weights $= \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$. For the training set we produce 1000 training data points from the defined GMM.

We trained a diffusion model on this dataset to learn the underlying data distribution. The key components of the training process we used: $T = 100$ diffusion steps and define a linear noise schedule with β_t linearly spaced between 1×10^{-4} and 0.02. The denoising model is a simple neural network consisting of fully connected layers with ReLU activation. Standard diffusion model training is done for $n_{\text{epochs}} = 1000$ epochs, were a a simple MSE loss between predicted and true noise is used.

We use the trained diffusion model to generate $n_{\text{gen_samples}} = 1000$ new samples. For a random subset of $n_{\text{trajectories}} = 5$ samples, we record their trajectories during the reverse diffusion process to analyze their paths towards convergence. Furthermore, to approximate the learned probability manifold, we apply Kernel Density Estimation (KDE) on the generated samples. In Fig. 4 (a) it is indeed observed - that the generation trajectories terminate at local maximas

C Reproducibility and Code

To ensure reproducibility, we provide our code and a detailed description of our computational environment.

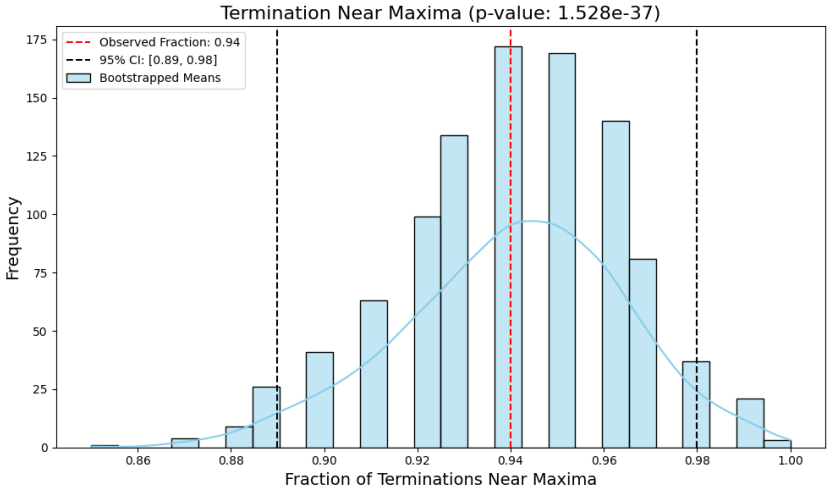

Figure 7: **Termination Analysis of Diffusion Trajectories Near Local Maxima.** The plot shows the fraction of 100 diffusion trajectories terminating near one of the maxima of the Gaussian mixture model (GMM). A trajectory is considered to terminate near a local maximum if its final point lies within a Mahalanobis distance of 2.45 from any of the GMM component means, corresponding to approximately 95% of the mass of a 2D Gaussian. The red dashed line represents the observed termination-near-maxima fraction (0.94), while the black dashed lines indicate the 95% confidence interval (0.89 to 0.98), derived via bootstrapping. In the bootstrapping process, the termination data (binary values indicating whether each of the 100 trajectories terminates near a maximum) was resampled with replacement 1,000 times to compute the distribution of termination fractions. The histogram and KDE curve illustrate this bootstrapped distribution. The p-value from a binomial test ($p = 1.528 \times 10^{-37}$) confirms that the observed termination fraction significantly deviates from random chance, supporting strong convergence of trajectories toward maxima.

Code. The implementation of our zero-shot detection method, as well as the train and test sets are available at the following link: `https://tinyurl.com/zeroshotimplementation`. To reproduce our results follow the readme file. **Hardware and Programs.** All of the experiments were conducted on the Ubuntu 20.04 Linux operating system, equipped with a Standard NC48ads A100 v4 configuration, featuring 4 virtual GPUs and 440 GB of memory. The experimental code base was developed in Python 3.8.2, utilizing PyTorch 2.1.2 and the NumPy 1.26.3 package for computational tasks.

D MATHEMATICAL FULL PROOFS

Claim 1. *Given x_0, consider Equation (10) , for which samples $x \sim \tilde{x}|x_0$ are drawn uniformly from the sphere ∂B_0, and each x is drawn with a corresponding $u_d(x) = \frac{x-\sqrt{1-\alpha}x_0}{\sqrt{\alpha}}$.[7] Then we can use $\nabla log p_\alpha(\cdot)$, and a tunable parameter $\gamma \in \mathbb{R}^+$ to obtain*

$$-\mathbf{E}_{x\sim\tilde{x}|x_0}\langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \gamma u_d(x) + \nabla \log p_\alpha(x)\rangle = \kappa(x_0) - D(x_0). \tag{19}$$

This provides a characterization of x_0 as a stable maximal point under the backward diffusion process, Equation (6), quantifying both gradient magnitude (should be low) and curvature (should be high) aspects.

Proof. Let us begin with the curvature term. By Gauss divergence Thm. we have

$$-|B_0|\kappa(x_0) = \int_{B_0} \nabla \cdot \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2} dx = \int_{\partial B_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \hat{n}\rangle dx, \tag{20}$$

where $\hat{n}$ are the outward-pointing normals to the sphere ∂B_0. By construction we have $\hat{n} = \frac{\sqrt{1-\alpha}x_0 - x}{\|\sqrt{1-\alpha}x_0 - x\|_2} = \frac{u_d(x)}{\sqrt{d}}$, $\forall x \in \partial B_0$, thus

$$\int_{\partial B_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \hat{n}\rangle dx = \frac{1}{\sqrt{d}} \int_{\partial B_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, u_d(x)\rangle dx. \tag{21}$$

Using the properties of $\tilde{x}_t|x_0$, which is uniformly distributed on ∂B_0, we have

$$\frac{1}{\sqrt{d}} \int_{\partial B_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, u_d(x)\rangle dx = \frac{|\partial B_0|}{\sqrt{d}} \mathbf{E}_{x\sim\tilde{x}_t|x_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, u_d(x)\rangle, \tag{22}$$

where $|\partial B_0|$ denotes the volume of ∂B_0. Tracing this back to κ (Equation (20)) we have

$$\kappa(x_0) = -\frac{|\partial B_0|}{\sqrt{d}|B_0|} \mathbf{E}_{x\sim\tilde{x}_t|x_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, u_d(x)\rangle. \tag{23}$$

Let us now analyze the gradient term. We similarly use the properties of the uniform distribution $\tilde{x}_t|x_0$, and get

$$D(x_0) = \frac{1}{|\partial B_0|} \int_{\partial B_0} \|\nabla \log p_\alpha(x)\|_2 dx = \mathbf{E}_{x\sim\tilde{x}_t|x_0} \|\nabla \log p_\alpha(x)\|_2. \tag{24}$$

For convenience, let us plug $\|\nabla \log p_\alpha(x)\|_2 = \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \nabla \log p_\alpha(x)\rangle$ and get

$$D(x_0) = \mathbf{E}_{x\sim\tilde{x}_t|x_0} \langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, \nabla \log p_\alpha(x)\rangle. \tag{25}$$

Finally, we can obtain any linear combination $a\kappa(x_0) + bD(x_0)$ as

[7] The notation of u_d as a function of x, i.e. $u_d(x)$, expresses the fact that they are constructed using the same noise. We use this under the integration sign when expressing explicitly expectations, where it is crucial to decide on one variable to integrate upon

$$a\kappa(x_0) + bD(x_0) = a\frac{-|\partial B_0|}{\sqrt{d}|B_0|}\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, u_d(x)\rangle + b\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \nabla\log p_\alpha(x)\rangle \tag{26}$$

$$= \mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, a\frac{-|\partial B_0|}{\sqrt{d}|B_0|}u_d(x) + b\nabla\log p_\alpha(x)\rangle. \tag{27}$$

Setting $b = -1$, and tuning $\gamma = \frac{|\partial B_0|}{\sqrt{d}|B_0|}$, we have

$$-\mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \gamma a u(x) - \nabla\log p_\alpha(x)\rangle = a\kappa(x_0) - D(x_0), \tag{28}$$

where we can choose $a = 1$.

□

Corollary 2. *In the setting of claim 1, we furthermore have the following approximation*

$$\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \frac{\sqrt{1-\alpha}}{\alpha}\hat{x}_0\rangle \approx \mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \frac{1}{\sqrt{\alpha}}u_d(x) + \nabla\log p_\alpha(x)\rangle, \tag{29}$$

where we can tune parameter a s.t.

$$-\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \frac{\sqrt{1-\alpha}}{\alpha}\hat{x}_0\rangle \approx a\kappa(x_0) - D(x_0). \tag{30}$$

Proof. By Equation (11) we have

$$\sqrt{1-\alpha}\mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \hat{x}_0\rangle \approx \mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \sqrt{1-\alpha}x_0 + \sqrt{\alpha}u_d(x) + \alpha\nabla\log p_\alpha(x)\rangle \tag{31}$$

Notice that

$$\mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \sqrt{1-\alpha}x_0\rangle = \langle\mathbf{E}_{x\sim\tilde{x}|x_0}\left(\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}\right), \sqrt{1-\alpha}x_0\rangle = 0, \tag{32}$$

since integration of normals over the sphere is zero, and $\nabla\log p_\alpha(x)$ approximates the uniform spherical noise. Thus by linearity of the expectation

$$\mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \sqrt{1-\alpha}x_0 + \sqrt{\alpha}u_d(x) + \alpha\nabla\log p_\alpha(x)\rangle = \mathbf{E}_{x\sim\tilde{x}|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \sqrt{\alpha}u_d(x) + \alpha\nabla\log p_\alpha(x)\rangle. \tag{33}$$

Tracing back to Equation (31) and dividing by α, we get

$$\frac{\sqrt{1-\alpha}}{\alpha}\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \hat{x}_0\rangle \approx \mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \frac{1}{\sqrt{\alpha_t}}u_d(x) + \nabla\log p_\alpha(x)\rangle. \tag{34}$$

Finally, similarly to Equation (26), Equation (28), setting $a = \frac{1}{\sqrt{\alpha}\gamma}$, results with

$$\frac{\sqrt{1-\alpha}}{\alpha}\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, \hat{x}_0\rangle \approx a\kappa(x_0) - D(x_0). \tag{35}$$

□

Corollary 3. *Let b_0 represent the bias of predictor $\hat{x}_0$ in the statistical sense, defined as $b_0 = x_0 - \mathbf{E}_{x\sim\tilde{x}|x_0}(\hat{x}_0)$. Transitioning from score-function back to the denoising perspective of diffusion models, the zeroized summand of Corollary 2, given by $\mathbf{E}_{x\sim\tilde{x}|x_0}\left\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, x_0\right\rangle$, can be traced back to:*

$$\mathbf{E}_{x\sim\tilde{x}|x_0}\left\langle\frac{\nabla\log p_\alpha(x)}{\|\nabla\log p_\alpha(x)\|_2}, x_0\right\rangle \approx -\langle b_0, x_0\rangle, \tag{36}$$

which is yet another quantity that captures bias towards generating x_0, where:

1. *This summand is effectively zeroized for unbiased noise predictors.*
2. *If not zeroized, higher values occur when b is (anti-)correlated with x_0, indicating a bias in the (minus) noise prediction towards the clean image. Since diffusion models denoise pure noise - it is intuitive that the noise prediction steers the resulting generated image towards its inherent biases - as is captured by this summand.*
3. *In score function analysis settings, the noise predictor is approximated by an MMSE denoiser, which is assumed to be unbiased, indeed leading to the zeroizing of this summand.*

Proof.

$$
\begin{aligned}
-\mathbf{E}_{x\sim\tilde{x}|x_0}\left\langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, x_0 \right\rangle &\approx \mathbf{E}_{x\sim\tilde{x}|x_0}(\langle x - \sqrt{1-\alpha}\hat{x}_0, x_0\rangle) \\
&= \frac{1}{\alpha}\langle \mathbf{E}_{x\sim\tilde{x}|x_0}(\sqrt{1-\alpha}x_0 + u_d - \sqrt{1-\alpha}\hat{x}_0), x_0\rangle \\
&= \frac{1}{\alpha}\langle \sqrt{1-\alpha}x_0 + \mathbf{E}_{x\sim\tilde{x}|x_0}(u_d) - \sqrt{1-\alpha}\mathbf{E}_{x\sim\tilde{x}|x_0}(\hat{x}_0), x_0\rangle \\
&= \frac{\sqrt{1-\alpha}}{\alpha}\langle x_0 + 0 - \mathbf{E}_{x\sim\tilde{x}|x_0}(\hat{x}_0), x_0\rangle \\
&= \frac{\sqrt{1-\alpha}}{\alpha}\langle x_0 - \mathbf{E}_{x\sim\tilde{x}|x_0}(\hat{x}_0), x_0\rangle \\
&= \frac{\sqrt{1-\alpha}}{\alpha}\langle b_0, x_0\rangle
\end{aligned}
$$

First transition follows Equation (4), Equation (5) and plugs Equation (10). The other transitions use linearity of the expectation, and the zero mean if the spherical noise u_d.

□

E Approximation Discussion

Reminder:

$$
D(x_0) = \frac{1}{|\partial B_0|}\int_{\partial B_0} \|\nabla \log p_\alpha(x)\|_2 dx = \mathbf{E}_{x\sim\tilde{x}_t|x_0}\|\nabla \log p_\alpha(x)\|_2. \tag{37}
$$

$$
\kappa(x_0) = -\frac{|\partial B_0|}{\sqrt{d}|B_0|}\mathbf{E}_{x\sim\tilde{x}_t|x_0}\langle \frac{\nabla \log p_\alpha(x)}{\|\nabla \log p_\alpha(x)\|_2}, u_d(x)\rangle. \tag{38}
$$

$$
\nabla \log p_\alpha(\tilde{x}) = \frac{1}{\alpha}\left(\sqrt{1-\alpha}\mathbf{E}_x[x_0|\tilde{x}] - \tilde{x}\right) \approx \frac{1}{\alpha}\left(\sqrt{1-\alpha}f(\tilde{x}, t; \theta) - \tilde{x}\right) := h(\tilde{x}), \tag{39}
$$

$$
C(x_0) := \frac{\sqrt{1-\alpha}}{\alpha}\frac{1}{s}\sum_{i=1}^{s}\left\langle \frac{-h(\tilde{x}^{(i)})}{\|h(\tilde{x}^{(i)})\|_2}, \hat{x}_0 \right\rangle \approx a\kappa(x_0) - D(x_0),
$$

F Experimental Settings Additional Information

F.1 Datasets

As mention in the main manuscript, the evaluation of our proposed method incorporates three benchmark datasets, namely, CNNSpot Wang et al. (2020), Universal Fake Detect Ojha et al. (2023) and GenImage Zhu et al. (2023) datasets. In our evaluation, we extracted a subset from each dataset, containing real images and fake images generated from the following generative models: ProGAN Karras et al. (2017), StyleGAN Karras et al. (2019), BigGAN Brock et al. (2018),GauGAN Park et al. (2019), CycleGAN Zhu et al. (2017), StarGAN Choi et al. (2018), Cascaded Refinement Networks

(CRN) Chen & Koltun (2017), Implicit Maximum Likelihood Estimation (IMLE) Li et al. (2019), SAN Dai et al. (2019), seeing-dark Chen et al. (2018), deepfake Rossler et al. (2019), Midjourney Midjourney (2024), Stable Diffusion V1.4 Rombach et al. (2022), Stable Diffusion V1.5 Rombach et al. (2022), ADM Dhariwal & Nichol (2021), Wukong MindSpore (2024), VQDM Gu et al. (2022), LDM Rombach et al. (2022) and Glide Nichol et al. (2021). In Figure 5 in the main manuscript, we divided our dataset into three groups: images generated by GANs (produces by ProGAN, StyleGAN, StyleGan2, BigGAN, GauGAN, CycleGAN, and CRN models), diffusion models(produces by LDM, Glide, Stable Diffusion V1.4, Stable Diffusion V1.5 and Guided diffusion models), and commercial tools (produces by Midjourney, Wukong, VQDM and DALL-E tools). Additionally, for the real images we used the LSUN, MSCOCO, ImageNet and LAION datasets.

To construct the calibration set for the zero-shot methods, we extracted 1,000 real samples from the datasets. For the test set, we selected 200,000 samples, ensuring a representative volume from each generation technique.

F.2 Methods for Comparison

We benchmarked our method against two recent and leading image detection zero-shot methods Ricker et al. (2024); He et al. (2024). These state-of-the-art methods are designed to enhance generalization by detecting generated images in a zero-shot settings. The implementations closely follow the specifications outlined in their respective publications. Specifically, we applied Ricker et al. (2024) directly by employing the code provided in their published paper, selecting the parameters leading to the highest performance according to their report (such as using the Kandinsky 2.1 model with the LPIPS similarity metric). Since He et al. (2024) was not available at the time of paper submission, we carefully reconstructed their implementation based on the details they provided, applying identical parameters, such as applying the DINO model with perturbation noise level of 0.05 and threshold value of 95%.

In the mixture-of-expert (MoE) experiment we utilized two additional leading image detection methods Cozzolino et al. (2024); Ojha et al. (2023). These few-shot and semi-supervised methods are designed to enhance generalization in detecting images created by unseen generative techniques. The implementations closely follow the specifications outlined in their respective publications. Specifically, the detection models are trained on images generated by a single model (ProGAN Karras et al. (2017) from the CNNSpot Wang et al. (2020) dataset) and tested on images from various other models. In implementing both methods, we initially employed the CLIP embedder Radford et al. (2021) using the open-source "clip-vit-large-patch14" model. For Ojha et al. (2023), we utilized a KNN model with $k = 9$ and cosine similarity, as this configuration was reported to achieve the best results in their paper. For Cozzolino et al. (2024), we employed a standard SVM model Pedregosa et al. (2011). Training of both methods was conducted with 10 different seeds $(1, 5, 9, 16, 17, 24, 43, 54, 59, 65)$, and the final detection results were averaged to ensure robustness.

G Experimental Results Additional Information

G.1 Complementary Results

In Fig. 8 statistics are gathered from all zero-shot methods to shed light on the variability of the different criteria (ours, RIGID, and AEROBLADE) across generative techniques. The competitors exhibit high variability, demonstrating their lack of generalizability. In the main text, under 5 in some cases the competitors have surprisingly low AUCs . This is due to their criteria being overly sensitive - as shown in the attached histograms. While in some techniques generated images obtain higher criteria values compared to real ones (e.g., Wukong), others (e.g., Glide 50) demonstrate a reverse trend. This makes the setting of a global threshold, such as the proposed 95% percentile of real images in RIGID, ineffective in some generative techniques it became - reducing their overall AUC. While incorporating information about the specific generative technique could improve their performance, it would compromise real-world practicality, and violate the standard testing for generalization to unseen techniques.

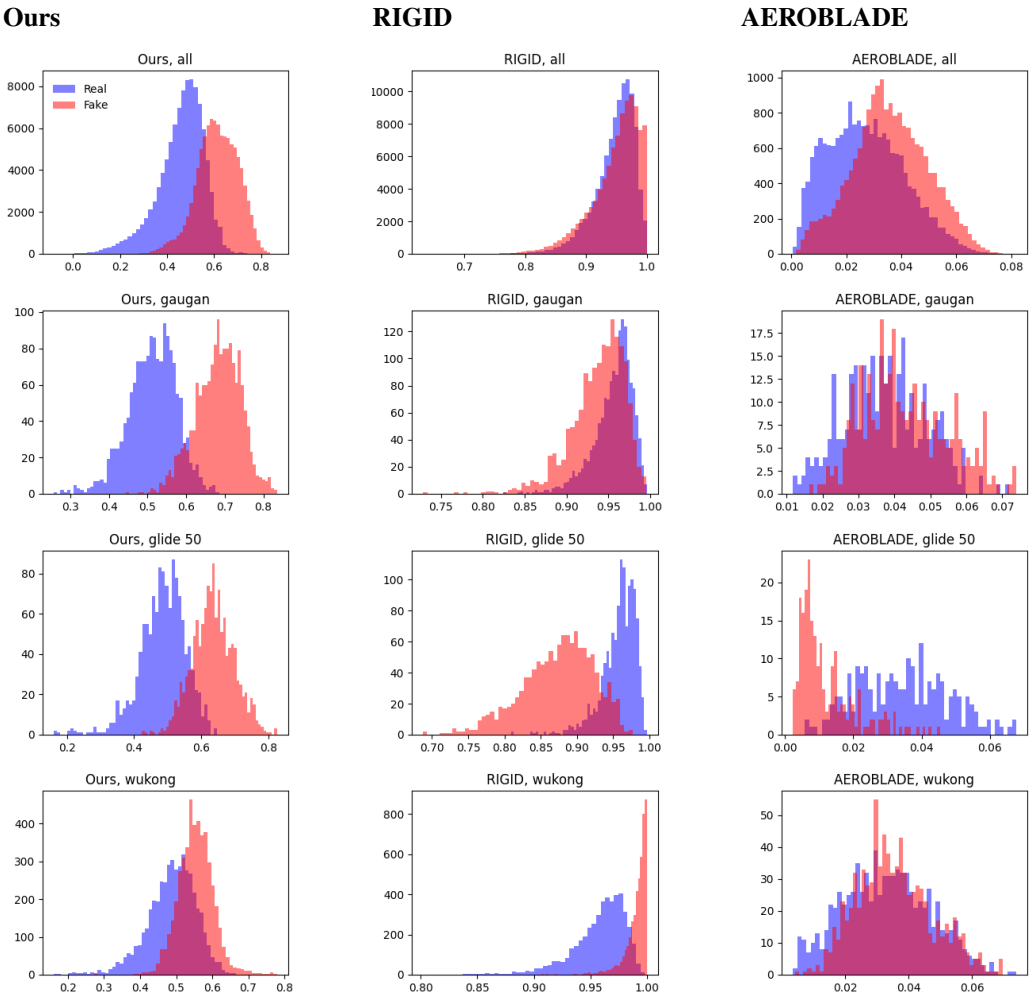

Figure 8: Histograms of the criterion proposed by our method and the competitors, for all the data as well as per-generative-technique cases. Even when the competitors exhibit high separability between real and generated images, they cannot set a global threshold that will capture all techniques - which is problematic since good generalization to unseen techniques is desired.

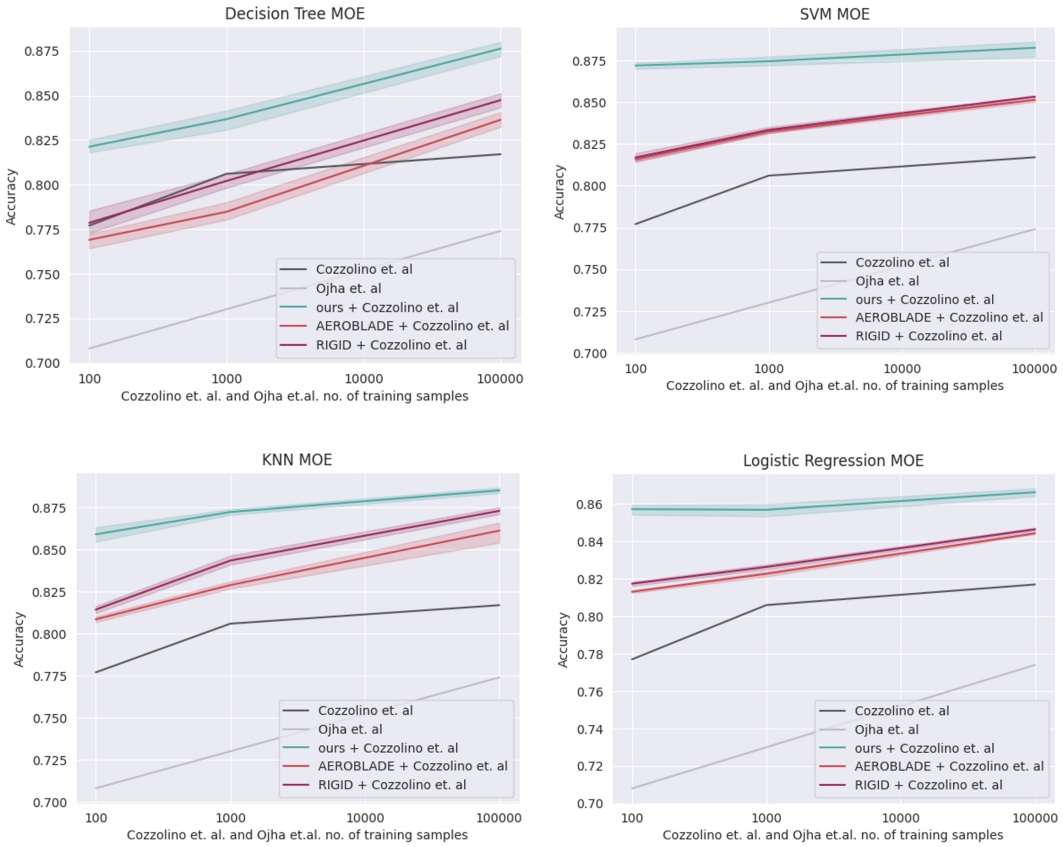

Figure 9: **Few-shot MoE**. Results of our MoE experiment using different classification algorithms. All combinations outperform Cozzolino significantly. Our method's efficacy proves to be the best.

G.2 Sensitivity and Ablation Analysis Additional Information

Various stable diffusion models. Our approach exploits the implicitly learned probability manifold of diffusion models to distinguish AI-generated images. While we demonstrated the robustness of our methodology using the Stable Diffusion v1.4 model, it is critical to verify that our results are not unduly influenced by specific characteristics of the used model. Consequently, we broadened our assessment scope to include newer models such as Stable Diffusion v2 Base and Kandinsky 2.1, noting these versions exhibit variations in architecture scale and generative algorithms. The methodology deployed in these expanded evaluations was the same as our original experiments (described in Sec. 5.1 of our manuscript), maintaining consistency in the calibration and test sets employed. This consistent experimental framework ensures that any observed performance variations are attributable solely to the model differences rather than experimental conditions. Preliminary findings indicate a modest performance decrement of approximately 2% with both Stable Diffusion v2 Base and Kandinsky 2.1, suggesting slight variances in generative fidelity and stability across model versions.

Various classification algorithms for MoE settings. In the main manuscript, we present the results of our MoE experiment when using a random forest algorithm to combine the results of the methods used in the MoE. Fig. 9 presents the results when different classification algorithms are used: decision tree, support-vector-machine (SVM), K-nearest-neighbors (KNN), and logistic regression. All models were set using the default parameters of the scikit-learn Python library. As can be seen in Fig. 9, our method's efficacy proves to be the best regardless of the algorithm used.

Various no. of perturbations S. One of the hyperparameters of our method is the volume of perturbations. In this experiment, we explore different perturbation sizes to assess their impact on the detection performance. For consistency, we maintained the same experiment methodologies

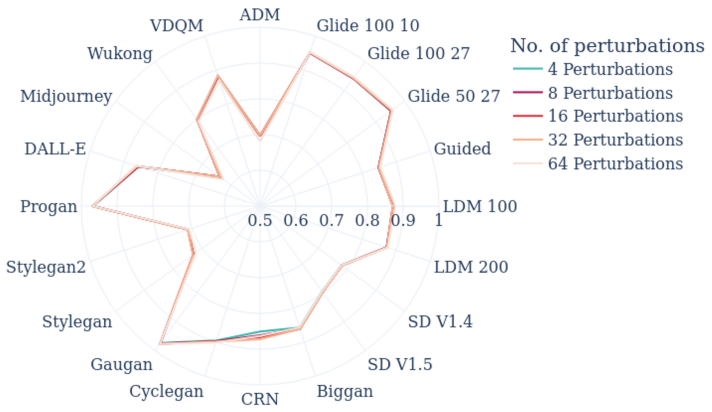

Figure 10: **AUC Detection Performance Analysis Across Different Numbers of Perturbations and Generation Techniques.** The number of perturbations shows minimal impact on performance.

as described in Sec. 5.1 of our manuscript, focusing solely on variations in perturbation size. We tested perturbation sizes of 4, 8, 16, 32, and 64, as implemented in our method. The corresponding average AUC detection results were 0.828, 0.829, 0.830, 0.833, and 0.835 respectively. The results reveal a clear trend: increasing S consistently enhances detection performance, which aligns with our research thesis. These findings reveal a clear trend: as the number of perturbations S increases, so does the detection performance. This progression supports our research theory since higher volume of perturbations provide better approximations of the numerical quantities.

Various spherical noise levels. Another hyperparameter in our method involves the levels of spherical noise, specifically the radii. In this experiment, we systematically adjusted the noise by reducing the radii, first by a factor of 10 and subsequently by a factor of 100. Additionally, we scaled down the test set to $16,000$ images for this small-scale study, though the variety of generative techniques assessed remained constant. These modifications led to performance decreases of 1.5% and 2%, respectively. These results help us understand the impact of noise adjustments on the robustness of our detection capabilities.

Robustness to JPEG. In real-world scenarios, images are frequently compressed to JPEG format for easier storage and transmission. To assess our method's practicality, we evaluated its performance on both real and generated JPEG-compressed images. For this small-scale study, we reduced the test set to 16,000 images, while maintaining the same variety of generative techniques. The results indicated a modest decrease in accuracy of 3.45%, demonstrating minimal impact on the method's detection effectiveness

G.3 Run-time Analysis

We performed a detailed runtime analysis to evaluate the computational efficiency of our method compared to existing approaches. Using a single A100 GPU, we observed a runtime of **2.1 seconds per sample** with 16 perturbations and 6.9 seconds per sample with 64 perturbations. Importantly, the 16-perturbation setup can be fully parallelized on a single A100 GPU. A more powerful GPU would reduce the runtime for 64 perturbations to 2.1 seconds per sample as well. For comparison, our primary competitor, AEROBLADE, on the same A100 GPU requires **5.4 seconds per sample**, making our method significantly more computationally efficient in terms of runtime.

G.4 Detailed Zero-shot Comparison

Tables 3 shows the entire zero-shot comparison for each detection method, over each detection technique using the accuracy, area-under-the-curve (AUC) and average precision (AP) metrics.

Table 3: Performance metrics across different models and methods

Model	**Accuracy**			**AUC**			**AP**		
	RIGID	**AEROBLADE**	**Ours**	**RIGID**	**AEROBLADE**	**Ours**	**RIGID**	**AEROBLADE**	**Ours**
ADM	0.5144	0.5065	**0.5727**	0.5715	0.5079	**0.6811**	0.5635	0.5701	**0.6541**
BigGan	0.5298	0.5831	**0.7756**	0.5300	0.5932	**0.8575**	0.5400	0.6132	**0.8680**
CRN	0.5000	0.3065	**0.7302**	0.0945	0.1875	**0.8637**	0.3202	0.3411	**0.8597**
CycleGan	0.4640	0.5611	**0.7342**	0.3760	0.5911	**0.9013**	0.4119	0.5617	**0.9052**
DALL-E	0.5071	0.4016	**0.7772**	0.4951	0.3852	**0.8660**	0.5152	0.4104	**0.8810**
GauGan	0.4804	0.5759	**0.8794**	0.3181	0.6212	**0.9770**	0.3924	0.6315	**0.9792**
Glide 100 10	0.4591	0.2939	**0.8831**	0.0484	0.0588	**0.9534**	0.3120	0.3142	**0.9588**
Glide 100 27	0.4594	0.3026	**0.8713**	0.0662	0.1017	**0.9462**	0.3139	0.3252	**0.9533**
Glide 50 27	0.4591	0.3000	**0.8843**	0.0471	0.0754	**0.9588**	0.3114	0.3159	**0.9625**
Guided	0.4803	0.5421	**0.7390**	0.3513	0.5590	**0.8540**	0.4135	0.5883	**0.8594**
IMLE	0.5000	0.3316	**0.7311**	0.0502	0.2310	**0.8699**	0.3128	0.3524	**0.8763**
LDM 100	0.4685	0.4295	**0.8232**	0.3241	0.4489	**0.8781**	0.3890	0.4423	**0.9096**
LDM 200	0.4669	0.4353	**0.8268**	0.3284	0.4587	**0.8758**	0.3923	0.4449	**0.9081**
Midjourney	**0.9407**	0.4020	0.5545	**0.9890**	0.3808	0.6278	**0.9906**	0.4057	0.6086
SAN	0.3746	0.3485	**0.6000**	0.3516	0.3985	**0.6131**	0.4359	0.4159	**0.5738**
SD v1.4	**0.8696**	0.5264	0.6196	**0.9555**	0.5747	0.7818	**0.9535**	0.5403	0.7230
SD v1.5	**0.8718**	0.5512	0.6300	**0.9543**	0.6027	0.7955	**0.9514**	0.5764	**0.7339**
Stylegan	0.5335	0.6081	0.7045	0.5932	0.6496	0.7225	0.5770	0.6388	**0.7649**
Stylegan2	0.4705	**0.6467**	0.6422	0.4641	**0.7125**	0.7100	0.4565	0.6836	**0.7275**
ProGan	0.4769	0.5306	**0.9032**	0.3216	0.5493	**0.9689**	0.3935	0.5450	**0.9738**
VDQM	0.5217	0.5282	**0.7686**	0.4852	0.5486	**0.8744**	0.5245	0.5939	**0.8783**
Wukong	**0.8777**	0.5135	0.6539	**0.9471**	0.5502	0.7935	**0.9552**	0.5249	0.7662
Average	0.5557	0.4648	**0.7411**	0.4392	0.4448	**0.8350**	0.5194	0.4925	**0.8330**