

AEROBLADE: Training-Free Detection of Latent Diffusion Images Using Autoencoder Reconstruction Error

Jonas Ricker

Denis Lukovnikov

Asja Fischer

Ruhr University Bochum

{jonas.ricker, denis.lukovnikov, asja.fischer}@rub.de

Abstract

With recent text-to-image models, anyone can generate deceptively realistic images with arbitrary contents, fueling the growing threat of visual disinformation. A key enabler for generating high-resolution images with low computational cost has been the development of latent diffusion models (LDMs). In contrast to conventional diffusion models, LDMs perform the denoising process in the low-dimensional latent space of a pre-trained autoencoder (AE) instead of the high-dimensional image space. Despite their relevance, the forensic analysis of LDMs is still in its infancy. In this work we propose AEROBLADE, a novel detection method which exploits an inherent component of LDMs: the AE used to transform images between image and latent space. We find that generated images can be more accurately reconstructed by the AE than real images, allowing for a simple detection approach based on the reconstruction error. Most importantly, our method is easy to implement and does not require any training, yet nearly matches the performance of detectors that rely on extensive training. We empirically demonstrate that AEROBLADE is effective against state-of-the-art LDMs, including Stable Diffusion and Midjourney. Beyond detection, our approach allows for the qualitative analysis of images, which can be leveraged for identifying inpainted regions. We release our code and data at <https://github.com/jonasricker/aeroblade>.

1. Introduction

The emergence of powerful and easy-to-use text-to-image models, like Stable Diffusion [3] and Midjourney [18], marked a turning point in the history of generative AI. While generating high-quality images previously required technological expertise and significant computational resources, hyperrealistic scenes with arbitrary contents are now only a few keystrokes away. These models open up countless creative possibilities, but may also have harmful consequences, like generated pictures winning art con-

tests [36] or fake photos of the Pope going viral [48]. While the ability to generate realistic images from prompts carries the inherent danger of malicious acts, the greater risk lies in the growing *erosion of trust* in legitimate sources due to the flood of synthetic media [49]. Developing effective and efficient detection methods is therefore of utmost importance.

Scaling the generation process to high resolutions while keeping the computational cost low was mainly made possible by so-called latent diffusion models (LDMs) [34]. Despite their ubiquity, LDMs have not yet been sufficiently studied from the perspective of forensics. Whereas standard diffusion models (DMs) operate directly in the image space, LDMs use the low-resolution latent space of a pre-trained autoencoder (AE). The model first generates a small latent representation, which is then transformed to a high-resolution image by the AE’s decoder.

In this work, we demonstrate that this property allows for a simple yet highly effective approach for detecting images generated by LDMs. Our proposed method, AEROBLADE (**a**utoencoder **r**econstruction-**b**ased **l**atent **d**iffusion **d**etection), distinguishes real and generated images by computing their AE reconstruction error, which is the distance between the original image and its reconstruction obtained by passing it through the AE’s encoder and decoder (see Sec. 4). While generated images can be accurately reconstructed, real images exhibit deficiencies, especially in complex regions (see Fig. 1). By computing the reconstruction error for the AEs of multiple LDMs, our method is effective against a wide range of models and can be easily extended to new ones. We find that this simple approach is able to reliably detect generated images, achieving a mean **average precision (AP)** of **0.992** on various state-of-the-art models including Stable Diffusion [3], Kandinsky [32], and Midjourney [18] (see Sec. 5.2). Notably, our method **does not require any training**, yet performs almost as well as extensively trained classifiers (see Sec. 5.3). In contrast to most existing detectors, which usually output only a score denoting how likely it is an image is generated, AEROBLADE additionally provides rich qualitative information, giving insights into how well certain regions can be reconstructed.

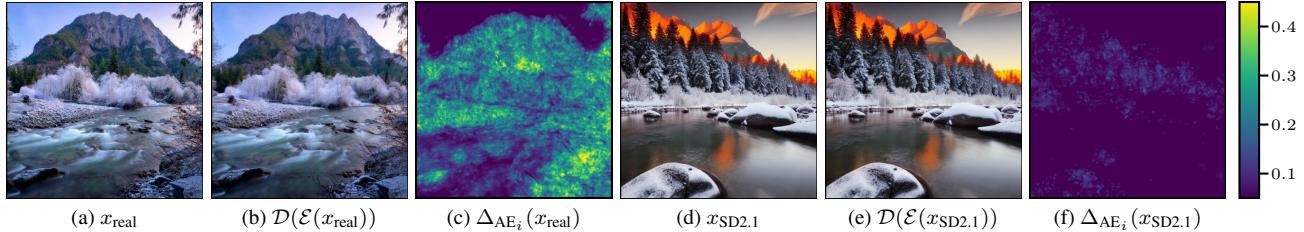


Figure 1. Example illustrating the idea behind AEROBLADE. (a) shows a real image from LAION-5B [39] and (d) is generated by Stable Diffusion 2.1 [34]. (b) and (e) are the corresponding reconstructions obtained by passing the original images through the AE of Stable Diffusion 2. (c) and (f) visualize the error between original and reconstruction measured using the LPIPS [60] distance. The reconstruction error is significantly lower for the generated image $x_{\text{SD2.1}}$ than for the real image x_{real} , which AEROBLADE leverages for detection.

We show that this information can be utilized for localizing inpainted regions within real images (see Sec. 5.4).

In summary, we make the following contributions:

- We present AEROBLADE, a simple and training-free method for detecting LDM-generated images based on the AE reconstruction error.
- We empirically show that our approach can effectively distinguish real images from images generated by seven state-of-the-art LDMs.
- Moreover, we thoroughly study the properties of AE reconstruction errors and demonstrate the qualitative insights AEROBLADE provides, which can help to identify inpainted regions.

2. Related Work

Detection of Generated Images In consequence of the rapid evolution of generative AI, different approaches for detecting synthetic images have been explored in the recent past. One class of detection methods exploits visual artifacts, like incorrect illumination [10, 24], inconsistent eye reflections [16], or irregular pupil shapes [14]. Another successful strategy is to analyze images in the frequency domain, where generated images exhibit distinguishable artifacts [1, 4, 12, 40]. Instead of using handcrafted features, a variety of learning-based methods has been proposed [5, 6, 13, 26, 37, 50]. Wang et al. [50] make the observation that a deep classifier trained to distinguish generated images by a single GAN from real images generalizes surprisingly well to images from other GANs. Gragnaniello et al. [13] show that using more extensive augmentation and omitting early downsampling improves the detection performance. Moreover, GAN inversion [55] has been leveraged for identifying generated images [27], which is related to the idea of reconstruction-based detection.

Most existing detectors are trained and evaluated on GAN-generated images, while the forensic analysis of DMs is still in an early stage [5, 23, 33, 41, 53]. Corvi et al. [5] show that a deep classifier trained on images generated by the originally proposed LDM [34] is able to detect images

by this model, but has limited generalization capabilities. Aiming towards universal detection, Ojha et al. [26] propose to train a simple linear classifier on top of features from a pre-trained CLIP-ViT [9, 30]. They show that using a feature space not explicitly trained on real and fake images provides better generalization to new model classes, including DMs. Most related to the method proposed here is DIRE [53]. Similar to previous works, DIRE makes use of a deep network for classifying images as real or fake, however, it uses the differences between images and their reconstructions obtained from a pre-trained ADM [7] as input data. Specifically, they use the deterministic forward and reverse process from DDIM [44] to map images to Gaussian noise and back. Our method differs considerably from DIRE, since AEROBLADE does not require training a deep classifier and obtains the reconstruction error from just the AE, without going through the costly forward and backward processes. To the best of our knowledge, the only other training-free detection method for DMs is SeDID_{Stat} [23], which can be seen as a refined version of DIRE. Instead of using the difference between original and reconstruction, it exploits the difference between individual steps during the denoising process. Similar to DIRE and unlike our approach, SeDID_{Stat} needs to perform computationally expensive diffusion and denoising operations. Moreover, it was only tested on low-resolution images (32×32), and has shown to be sensitive to the choice of hyperparameters, in particular the step at which the error is computed.

Diffusion Models for Visual Anomaly Detection A different task for which the reconstruction capabilities of DMs can be used is visual anomaly detection. The idea is that anomalous regions of an image can be identified by reconstructing the image with a DM trained or conditioned on nominal images. Since the model can only generate data from the learned distribution of nominal samples, the difference between original and reconstruction reveals anomalies. This idea has been explored for general out-of-distribution detection [22], medical images [54], industrial applications [25], and unsupervised video anomaly detection [46].

3. Preliminaries

In this section we provide background information on LDMs and how they differ from standard DMs. We also explain the details of the LPIPS [60] distance metric, which we use to estimate the reconstruction error.

Latent Diffusion Models (LDMs) DMs are a class of generative models based on nonequilibrium thermodynamics [43] that have been shown to be capable of high-fidelity image synthesis [7, 15, 44, 45, 56]. In the forward (or diffusion) process, an image is gradually disturbed by adding Gaussian noise. The model, which typically uses the U-Net [35] architecture, then learns to recover a slightly less noisy version of the image at different steps. During the reverse (or denoising) process, new images are generated from pure Gaussian noise by reversing the diffusion process until a clean image is reached. Since predictions operate in the high-dimensional image space, both training and inference require excessive computational resources. To tackle this limitation, Rombach et al. [34] propose to perform the denoising process in an optimized latent space of lower dimensionality. The key idea is to first generate the semantics of an image in latent space, which are then transformed to the image space, adding high-frequency details. Dividing the generation process into these two phases makes both training and sampling much more efficient. Several powerful text-to-image models are based on the concept of LDMs, including the popular Stable Diffusion [3] and Midjourney [18].¹

To map images from image space to latent space and back, LDMs use a pre-trained AE. That is, during the forward process, an image x is mapped to the latent representation $z = \mathcal{E}(x)$ using the encoder \mathcal{E} . During the reverse process, the final image $\tilde{x} = \mathcal{D}(\tilde{z})$ is obtained by passing the denoised latents \tilde{z} through the decoder \mathcal{D} .

Learned Perceptual Image Patch Similarity (LPIPS) The LPIPS metric, proposed by Zhang et al. [60], aims to solve the problem of perceptual similarity, i.e., estimating how similar two images are according to human judgement. The authors make the observation that the internal activations of a classifier trained on ImageNet [38] provide an embedding that corresponds surprisingly well with human perception, even across different architectures (SqueezeNet [17], AlexNet [20], and VGG16 [42]). They find that LPIPS outperforms specialized distance metrics like SSIM [52] or MS-SSIM [51].

To obtain the LPIPS distance between two images, both are fed through the network to extract the activations from

¹If not stated otherwise, we use the term ‘‘LDM’’ to describe the class of DMs that perform denoising in latent space, not the particular LDM proposed by Rombach et al. [34].

certain layers. For VGG16, which we mainly use in this work, these are the five convolutional layers before the pooling layers. For each layer, the activations are scaled channel-wise using learned linear weights. The similarity is computed as the ℓ_2 difference between both activations. The activations from each layer are then spatially averaged (along width and height), and the final similarity score is given as the sum of these averages.

4. Methodology

We first introduce the general framework for reconstruction-based fake detection. Subsequently, we provide the details and formal definition of AEROBLADE.

Reconstruction-based Fake Image Detection The idea of reconstruction-based detection builds upon two assumptions. The first is that, given a generative model G_i and an image x , we can compute a reconstruction $\tilde{x} = \phi_i(x)$, with ϕ_i denoting a reconstruction method that is based on G_i . The second assumption is that for an image x_i , which is generated by model G_i , the distance d between the original image and its reconstruction, $d(x_i, \tilde{x}_i)$, is small. In contrast, a real image x_r cannot be reconstructed as accurately, i.e., $d(x_r, \tilde{x}_r) > d(x_i, \tilde{x}_i)$.

AEROBLADE The most crucial component of a reconstruction-based detection technique is the choice of the reconstruction method ϕ . We find that in the case of LDMs, there is a straightforward choice that is easy to implement and efficient. Our proposed method, AEROBLADE (**a**utoencoder **r**econstruction-**b**ased **l**atent **d**iffusion **d**etection), is based on the observation that the model’s AE is better at reconstructing generated images than reconstructing real images. An intuitive explanation is that AEs use a latent representation z that maps to a constrained data manifold. Generated images lie within it and can therefore be accurately reconstructed, while real images are mapped to the closest point in the manifold, leading to a higher error. Therefore, the distance between an image and its reconstruction allows for simple threshold-based detection. In contrast to previous works [23, 53], our method does neither require performing the costly deterministic denoising process, nor any additional training.

We define the reconstruction error of an image x based on the AE of an LDM G_i (denoted by AE_i) as

$$\Delta_{\text{AE}_i}(x) := d(x, \tilde{x}) = d(x, \mathcal{D}_i(\mathcal{E}_i(x))), \quad (1)$$

with \mathcal{E}_i and \mathcal{D}_i being the encoder and decoder of the AE, respectively, and d being some distance metric. We provide an illustration of Eq. (1) in Fig. 2. In our experiments we find that LPIPS [60] is a suitable choice for d , but we investigate alternative distance metrics in Sec. 5.5.

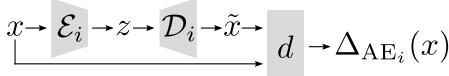


Figure 2. Graphical representation of Eq. (1). The reconstruction error $\Delta_{\text{AE}_i}(x)$ is defined as the distance between an image x and its reconstruction \tilde{x} obtained from passing it through the encoder \mathcal{E}_i and decoder \mathcal{D}_i of an LDM’s AE.

Eq. (1) computes the reconstruction error using the AE from a single model, assuming that images are either real or generated by this particular model. In practice, however, with a constantly growing pool of generative models available, this assumption does not hold. To account for this, we compute Δ_{AE_i} for a set of LDMs, $\{G_i, i \in \mathcal{I}\}$, and use the smallest reconstruction error for classification. Formally, we define the minimum reconstruction error as

$$\Delta_{\text{Min}}(x) := \min_{i \in \mathcal{I}} \Delta_{\text{AE}_i}(x) = \min_{i \in \mathcal{I}} d(x, \mathcal{D}_i(\mathcal{E}_i(x))). \quad (2)$$

Based on the assumption that the “correct” AE provides the best reconstruction, using Δ_{Min} is more suitable for detection than, e.g., taking the average from multiple AEs.

5. Experiments

In this section we experimentally demonstrate the effectiveness of AEROBLADE. After introducing the datasets and AEs we use in our evaluation, we report the detection performance and compare it to several state-of-the-art baselines. We finally demonstrate that our approach can be used to spot inpainted regions within real images and conduct further analyses regarding the properties of AEROBLADE.

5.1. Setup

Data We evaluate our method on images from seven text-to-image LDMs, among them four open-source and three proprietary models. We include three different versions of Stable Diffusion [3], the initially released version 1.1 (SD1.1), 1.5 (SD1.5) which is widely used as a base for custom models by the generative AI community, and the more recent 2.1 (SD2.1). We additionally test on images generated by Kandinsky 2.1 (KD2.1) [32], which builds upon Stable Diffusion and DALL-E 2 [31]. To allow for a fair evaluation, we attempt to generate images whose contents match those of the real images (taken from LAION-5B [39]). To achieve this, we extract prompts from real images using CLIP interrogator [29], which combines CLIP [30] and BLIP [21] to optimize a prompt towards a target image. Finally, we include data from three different versions of the popular image generation service Midjourney [18], namely versions 4 (MJ4), 5 (MJ5), and 5.1 (MJ5.1). Since these models are proprietary, we cannot extract matching prompts. Instead, we use a diverse set of images from the publicly available Midjourney Discord server.

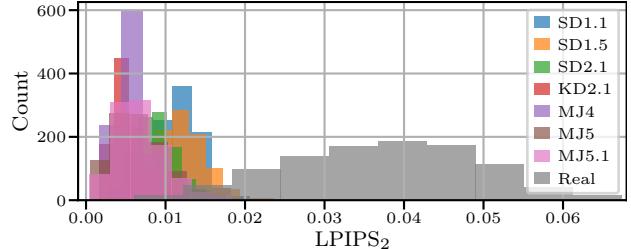


Figure 3. Distributions of reconstruction error Δ_{Min} using LPIPS₂ for different datasets. We provide results for all LPIPS variants in Sec. 8.1 in the supplementary material.

We collect 1 000 images for each generative model plus 1 000 real images. We provide more details on the data collection in Sec. 7.1 in the supplementary material.

Autoencoders We compute reconstructions using three different AEs: SD1 and SD2, which are used by Stable Diffusion models with the corresponding versions, and the AE from Kandinsky 2.1 (KD2.1). It should be noted that while all AEs have the same task, their architectures differ. Stable Diffusion uses a variational autoencoder (VAE) [19] with Kullback-Leibler regularization, while Kandinsky uses a discrete vector quantized-VAE (VQ-VAE) [47]. We emphasize that the AE used by Midjourney is not publicly available, which allows us to test the applicability of our method to closed-source generators.

5.2. Evaluation of Detection Performance

We initially inspect the distributions of reconstruction errors for real and generated images. For each dataset, we compute the reconstruction errors from 1 000 images using all three AEs and choose the minimum reconstruction error (Δ_{Min}) per sample. Besides using the standard definition of LPIPS (summation of the Euclidean distances of the activations from all five layers), we also evaluate using the distances based on activations from a single layer only. We denote these as LPIPS_i, $i \in \{1, \dots, 5\}$. The reconstruction errors for LPIPS₂ are shown in Fig. 3. We make the promising observation that generated images have a consistently lower reconstruction error than real images.

To obtain a quantitative evaluation of AEROBLADE, we follow previous works [26, 50, 53] and report the performance in average precision (AP) in Tab. 1. We first compare the results from different LPIPS variants. Overall, our results suggest that the second LPIPS layer (LPIPS₂) captures the most meaningful differences between original images and reconstructions. The AP decreases towards higher layers (with a larger receptive field), indicating that fine-grained details cause higher reconstruction errors. We further analyze the relation between image complexity and reconstruction error in Sec. 5.4.

Distance	AE	SD1.1	SD1.5	SD2.1	KD2.1	MJ4	MJ5	MJ5.1
LPIPS	SD1	0.989	0.988	0.827	0.879	0.967	0.930	0.930
	SD2	0.763	0.771	0.992	0.878	0.999	0.994	0.994
	KD2.1	0.593	0.606	0.665	0.997	0.919	0.878	0.860
	Min	0.959	0.957	0.991	0.996	0.998	0.993	0.994
LPIPS ₁	SD1	0.954	0.951	0.667	0.785	0.894	0.835	0.842
	SD2	0.574	0.593	0.967	0.790	0.995	0.979	0.980
	KD2.1	0.481	0.502	0.550	0.994	0.874	0.804	0.790
	Min	0.882	0.884	0.961	0.994	0.993	0.975	0.976
LPIPS ₂	SD1	0.992	0.991	0.777	0.878	0.971	0.933	0.924
	SD2	0.716	0.722	0.996	0.879	0.999	0.998	0.997
	KD2.1	0.543	0.552	0.623	0.999	0.927	0.882	0.858
	Min	0.979	0.978	0.994	0.999	0.999	0.997	0.996
LPIPS ₃	SD1	0.989	0.987	0.874	0.904	0.974	0.944	0.949
	SD2	0.828	0.832	0.991	0.905	0.997	0.993	0.995
	KD2.1	0.615	0.622	0.695	0.998	0.929	0.894	0.881
	Min	0.969	0.965	0.991	0.997	0.997	0.993	0.995
LPIPS ₄	SD1	0.981	0.979	0.881	0.897	0.966	0.945	0.943
	SD2	0.841	0.845	0.983	0.896	0.997	0.989	0.991
	KD2.1	0.685	0.691	0.742	0.994	0.916	0.896	0.877
	Min	0.926	0.924	0.983	0.990	0.996	0.989	0.991
LPIPS ₅	SD1	0.960	0.960	0.838	0.813	0.923	0.904	0.898
	SD2	0.800	0.814	0.965	0.815	0.989	0.976	0.978
	KD2.1	0.662	0.670	0.700	0.969	0.861	0.850	0.820
	Min	0.867	0.869	0.964	0.944	0.989	0.976	0.978

Table 1. Detection performance of AEROBLADE for different LPIPS variants (first column) and AEs (second column), measured in AP. “Min” refers to the minimum reconstruction error Δ_{Min} , as defined in Eq. (2). The remaining columns contain the results for individual datasets. The best result for each dataset is highlighted in **bold**.

Distance	AE	SD1.1	SD1.5	SD2.1	KD2.1	MJ4	MJ5	MJ5.1
LPIPS ₂	SD1	1.000	1.000	0.000	0.000	0.000	0.000	0.000
	SD2	0.000	0.000	1.000	0.000	0.999	0.997	0.995
	KD2.1	0.000	0.000	0.000	1.000	0.001	0.003	0.005

Table 2. Fraction of samples for which an AE has the smallest reconstruction error using LPIPS₂. The highest fraction for each dataset is highlighted in **bold**. We provide results for all LPIPS variants in Sec. 8.2 in the supplementary material.

Second, we compare the performance of different AEs to obtain the reconstructions. We observe that, as was to be expected, using the “matching” AE for a given dataset (e.g., SD1 for SD1.1) yields the best results, with the AP ranging from 0.991 to 0.999 (with LPIPS₂). However, as the results for Midjourney demonstrate, accurate detection is possible even without access to the generator’s AE. While the AP is relatively high for all three AEs, it appears that the AE used by Midjourney is similar to the one used by Stable Diffusion 2. Across all datasets, we only observe a small performance drop when using Δ_{Min} , instead of Δ_{AE_i} with the matching AE, which is crucial for applying AEROBLADE in real-world scenarios. As a sanity check, we determine for each sample which AE provides the most accurate reconstruction, i.e., for which AE the reconstruction error Δ_{AE_i} is equal to the minimum reconstruction error Δ_{Min} .

The results in Tab. 2 do not only confirm the assumption that the correct AE achieves the lowest reconstruction error, but also demonstrate that AEROBLADE can be used to attribute images to a specific generative model. However, attribution is limited by the fact that some generative models use the same AE, e.g., Stable Diffusion 1.1 and 1.5.

5.3. Comparison to Baselines

We compare AEROBLADE against several state-of-art baselines (see Sec. 2) using code and pre-trained models provided by the original authors. An exception is SeDID_{Stat} [23], which has only been evaluated on conventional DMs. To allow for a fair comparison, we adapt their approach to text-to-image LDMs. We also allow SeDID_{Stat} to “cheat”, by using the matching model for each dataset (except for Midjourney, for which we use SD2.1). We provide more details on how we evaluate the baselines in Sec. 7.2 in the supplementary material.

The results in Tab. 3 show the AP and the true positive rate (TPR) at a fixed false positive rate (FPR). The latter is better suited to estimate the usefulness of a detector in realistic settings, in which only a certain FPR (5 % in our case) is tolerable. Among all methods, only AEROBLADE and the method by Corvi et al. [5] can reliably detect generated images from all models. Despite involving no training, our method achieves almost the same performance as the deep classifier directly trained on generated images. The only other training-free approach, SeDID_{Stat} [23], achieves mediocre performance for some generative models, but fails completely to detect images from other generators. The universal detection approach by Ojha et al. [26] performs decent on most models, especially SD1.1 and 1.5, but by taking the more realistic TPR@5%FPR metric into account, its deficiencies become apparent. We suppose the performance drop, in comparison to the results in the original publication, stem from the fact that the classifier was trained on smaller images (256 × 256). Surprisingly, we obtain very poor results with DIRE [53], which contradicts the results reported by the authors, who claim that their method generalizes to images generated by Stable Diffusion 1.5. We took great care to use their public code and pre-trained models as specified in the documentation. In our analysis, almost all images (both real and generated) were classified as being generated. Upon close inspection of code, data, and models provided by the authors, we suspect that the DIRE representations used to train the classifiers were saved inconsistently. In particular, the DIREs of real images were compressed using JPEG while the DIREs of generated images were stored in lossless PNG format. This unwanted bias would explain why all of our uncompressed DIREs are classified as being generated. We provide a thorough analysis in Sec. 9 in the supplementary material.

Training Samples	AP							TPR@5%FPR							
	SD1.1	SD1.5	SD2.1	KD2.1	MJ4	MJ5	MJ5.1	SD1.1	SD1.5	SD2.1	KD2.1	MJ4	MJ5	MJ5.1	
Gragnaniello [13]	720 000	0.715	0.701	0.629	0.526	0.664	0.666	0.598	0.149	0.151	0.107	0.043	0.160	0.163	0.108
Corvi [5]	400 000	1.000	0.999	1.000	1.000	0.999									
Ohja [26]	720 000	0.895	0.835	0.732	0.744	0.756	0.713	0.682	0.596	0.416	0.256	0.287	0.274	0.236	0.205
DIRE [53]	80 000	0.457	0.457	0.480	0.513	0.503	0.498	0.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000
SeDID _{Stat} [23]	-	0.484	0.783	0.713	0.728	0.376	0.401	0.403	0.049	0.308	0.205	0.259	0.005	0.011	0.012
AEROBLADE	-	0.979	0.978	0.994	0.999	0.999	0.997	0.996	0.981	0.963	0.995	0.999	0.997	0.993	0.989

Table 3. Detection performance of AEROBLADE and baselines, measured in AP and TPR@5%FPR. All methods in the upper section require training, while those in the lower section are training-free. The results for our method are obtained with LPIPS₂ and Δ_{Min} (corresponding to the 12th row in Tab. 1). The best result for each dataset (per section) is highlighted in **bold**.

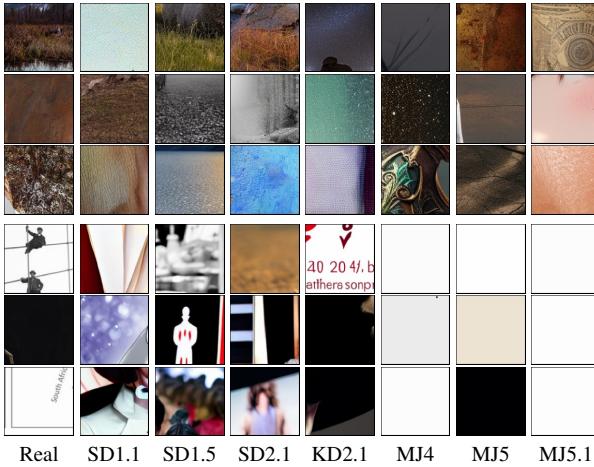


Figure 4. Example patches of size 128×128 with high (upper half) and low (lower half) reconstruction error. The reconstruction error is computed using LPIPS₂ and Δ_{Min} , and the patches are selected from the top and bottom 1% of each dataset.

5.4. Qualitative Image Analysis with AEROBLADE

In this section we first examine how the reconstruction error of real and generated images is influenced by image complexity. We then illustrate how AEROBLADE can be leveraged to identify inpainted regions within real images.

Relation Between Image Complexity and Reconstruction Error During our experiments we make the observation that simple parts of images, like monochrome areas, can be reconstructed more accurately than complex parts. The examples in Fig. 4 illustrate this behavior. However, we find that the relation between reconstruction error and image complexity differs between real and generated images. In Fig. 5 we plot Δ_{Min} against the image complexity for overlapping patches of size 128×128 with stride 64. We estimate the complexity of a patch by its file size after JPEG compression with quality 50, which approximates the Kolmogorov complexity [2, 57]. For real images, we observe that the reconstruction error is positively correlated

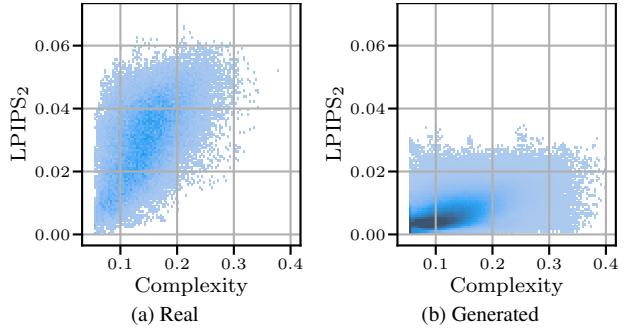


Figure 5. Reconstruction error against complexity for real and generated images. (b) contains generated images from all datasets, we provide individual plots in Sec. 8.3 in the supplementary material. The color map is clipped at 1 000 samples for better visibility.

with image complexity. For generated images, this trend is significantly less pronounced. In particular, patches with high complexity can be reconstructed considerably better compared to real images. These results suggest that if the fine-grained details of an image can be accurately reconstructed, it is likely generated by an LDM. This allows for a qualitative image analysis, which can explain AEROBLADE’s predictions. In the next section we demonstrate how this analysis can help to localize inpainted regions. It should be noted that this property makes generated images with low complexity, e.g., logos, harder to detect with our method. However, we argue that these are less harmful than complex, photorealistic images.

Localizing Inpainted Regions In Fig. 6 we show that inspecting the reconstruction error *map* of an image can provide hints to identify inpainted regions within an authentic image. To illustrate this, we take real images and inpaint a randomly generated rectangular mask using the inpainting variant of Stable Diffusion 1.5. We then compute Δ_{AE} , with the AE from Stable Diffusion 1 but omit spatial averaging to obtain the heatmaps shown in the third row of Fig. 6. The inpainted regions can be identified due to their significantly lower reconstruction error, which is clearly no-

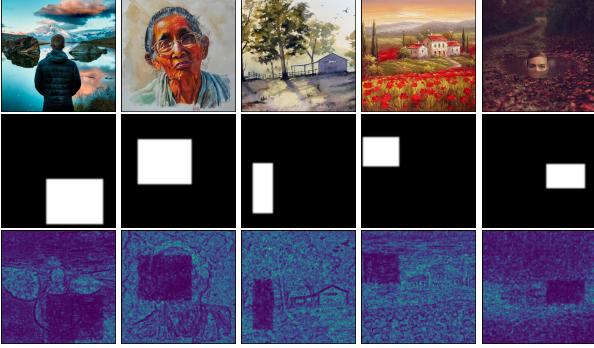


Figure 6. Examples illustrating the localization of inpainted regions using the reconstruction error. We show the images inpainted with Stable Diffusion 1.5 (top), the masks used for inpainting (center) and the reconstruction error maps (bottom), computed with LPIPS_2 and the AE from Stable Diffusion 1.5. We provide more examples in Sec. 8.4 in the supplementary material.

ticeable in regions with higher complexity. In future work we aim to investigate how the reconstruction error can be used to predict the precise locations of inpainted regions.

5.5. Additional Analyses

Finally, we conduct additional experiments to better understand the properties of AEROBLADE. We analyze its robustness to common image perturbations, evaluate the influence of the used distance metric, and explore computing the errors from deeper reconstructions.

Robustness to Perturbations In real-world scenarios, images are often processed, e.g., during the upload to social media platforms, which may affect detection performance. We therefore evaluate how robust our method is to common image perturbations. Following previous works [12, 59] we use JPEG compression (with quality q), center cropping (with crop factor f and subsequent resizing to the original size), Gaussian blur, and Gaussian noise (both with standard deviation σ). In Fig. 7 we report the performance of AEROBLADE and several baselines on a set of 250 real and 250 generated images (we omit DIRE [53] and SeDID_{Stat} [23] due to the low performance on clean images and high computational cost). We find that our method outperforms the detectors from Gragnaniello et al. [13] and Ojha et al. [26] in most settings. The results for individual datasets and layers (see Sec. 8.5 in the supplementary material) also reveal that the robustness strongly depends on the dataset and selected LPIPS layer. For instance, LPIPS₄ appears to be significantly less affected by perturbations, possibly due to the larger receptive field. While the method proposed by Corvi et al. [5] is more robust, we emphasize that it was directly trained on perturbed, LDM-generated images. Our method, in contrast, is completely training-free. In future work, we

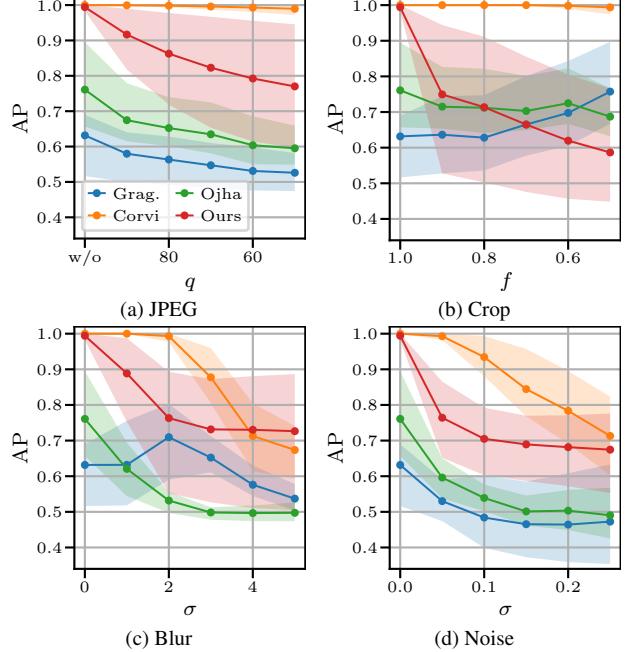


Figure 7. Detection performance of AEROBLADE (with LPIPS_2 and Δ_{\min}) and baselines on perturbed images, measured in AP. Results are averaged over all datasets, with shaded areas indicating the minimum and maximum. We provide extended results in Sec. 8.5 in the supplementary material.

Distance	SD1.1	SD1.5	SD2.1	KD2.1	MJ4	MJ5	MJ5.1
LPIPS (VGG16)	0.959	0.957	0.991	0.996	0.998	0.993	0.994
LPIPS (AlexNet)	0.755	0.765	0.904	0.960	0.981	0.953	0.953
LPIPS (SqueezeNet)	0.848	0.847	0.950	0.976	0.992	0.973	0.974
DISTS [8]	0.866	0.860	0.962	0.928	0.998	0.991	0.995
MSE	0.534	0.571	0.792	0.882	0.886	0.863	0.866
SSIM [52]	0.808	0.812	0.924	0.981	0.984	0.957	0.955
MS-SSIM [51]	0.842	0.854	0.955	0.984	0.989	0.977	0.977

Table 4. Detection performance of AEROBLADE using different distance metrics, measured in AP.

plan to explore whether using a more robust distance metric or adding a simple classifier trained on the reconstruction errors can increase the robustness of AEROBLADE.

Exploring Different Distance Metrics In Tab. 4 we compare how the distance metric influences the performance of AEROBLADE. Beside VGG16 [42], which we use in previous experiments, we consider SqueezeNet [17] and AlexNet [20] as backbones for LPIPS (using its standard definition). Moreover, we test alternative similarity metrics like mean squared error (MSE), structural similarity index (SSIM) [52], multiscale structural similarity index (MS-SSIM) [51], and DISTS [8]. The results show that LPIPS (VGG16) achieves the best overall performance, while other metrics achieve high AP only on some datasets.

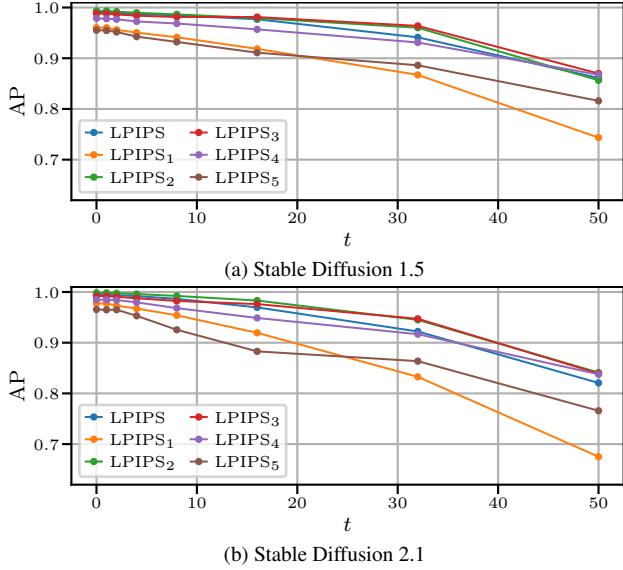


Figure 8. Detection performance using deeper reconstructions, measured in AP. Reconstructions are computed using the corresponding AE and t steps of the DM’s denoising process.

Using Deeper Reconstructions Finally, we experiment with reconstructing images using not only the AE, but also (parts of) the denoising process in latent space to study the effect of deeper reconstructions on training-free detection performance. Here, we first compute latent (noisy) images that would generate the given image by inverting the DDIM sampler [44] for several steps and then reconstruct the original image by denoising with the DDIM sampler as usual. The hypothesis is again that generated images should be reconstructed more accurately using DDIM inversion and denoising. We set the total number of steps to 50 and guide the deterministic denoising process by a prompt extracted using BLIP [21]. Fig. 8 depicts the detection performance for Stable Diffusion 1.5 and 2.1 using different amounts of reconstruction steps (out of 50) based on 250 samples per dataset. Note that we use the matching AE and U-Net weights for both datasets, respectively, and that $t = 0$ corresponds to our previously described approach using just Δ_{AE_i} . With one or two steps, the AP is almost equal to the setting with $t = 0$ across both datasets and all variants of LPIPS. Increasing the number of steps causes a notable decrease in detection performance, especially for higher LPIPS layers. Note that the reconstructions obtained with the full denoising process (i.e., $t = 50$) correspond to the reconstructions that DIRE [53] uses. We conclude that including the denoising process to compute the reconstruction distance does not benefit the detection performance, especially given the added computational complexity. Another disadvantage is that it requires the weights of the corresponding U-Net, which could be more difficult to obtain than just the AE.

6. Discussion and Conclusion

With LDMs being a key enabler for high-resolution image synthesis, their forensic analysis and the development of specialized detection methods is long overdue. In this work, we attempt to close this gap by proposing AEROBLADE, a simple and training-free approach which can reliably detect images from state-of-the-art LDMs like Stable Diffusion and Midjourney. We find that, despite being training-free, our method achieves a detection performance which is comparable to deep classifiers that are directly trained on LDM-generated images, making it a promising alternative.

A limitation of our method is that, to get the best results, access to the AE of the LDM which actually generated the image is required (see Sec. 5.2). However, we argue that this does not substantially restrict the usability of AEROBLADE in real-world settings. The most widely used LDMs, especially Stable Diffusion, have become so popular due to being publicly available. Furthermore, the fact that the open-source community contributes major innovations to existing models [28] results in free and proprietary models sharing the same characteristics. Our observation that images generated by Midjourney can be detected almost perfectly by using the AE from Stable Diffusion 2 is a prime example for this. We are therefore convinced that with a sufficiently large pool of AEs, AEROBLADE is effective against a wide range of practically relevant LDMs. The modularity of our approach is actually a key benefit, since extending it to new models is trivial and does not require expensive re-training.

We realize that this property can also be used as a simple means for model inventors to responsibly disclose new generative models. Existing approaches for responsible model disclosure [11, 58, 59] usually require changes to either the training data or the model itself, which might be undesirable or even infeasible. With AEROBLADE, model inventors only have to publish their custom AE, whereas the backbone operating in latent space, which typically is the most valuable asset, remains private. By doing so, model inventors can alleviate potential negative consequences of their work with little to no additional overhead.

We hope that our work offers a novel perspective on the detection of images generated by modern text-to-image models. We believe that AEROBLADE and potential follow-up works can contribute to mitigating the threats of these models to our digital society.

Acknowledgements

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy - EXC 2092 CASA - 390781972.

References

- [1] Keshigeyan Chandrasegaran, Ngoc-Trung Tran, and Ngai-Man Cheung. A closer look at Fourier spectrum discrepancies for CNN-generated images detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2
- [2] Rudi Cilibrasi and Paul Vitanyi. Clustering by compression. *IEEE Transactions on Information Theory*, 2005. 6
- [3] CompVis. Stable diffusion. <https://github.com/CompVis/stable-diffusion>, 2023. 1, 3, 4, 12
- [4] Riccardo Corvi, Davide Cozzolino, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. Intriguing properties of synthetic images: From generative adversarial networks to diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2023. 2
- [5] Riccardo Corvi, Davide Cozzolino, Giada Zingarini, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. On the detection of synthetic images generated by diffusion models. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023. 2, 5, 6, 7, 12, 17
- [6] Davide Cozzolino, Diego Gragnaniello, Giovanni Poggi, and Luisa Verdoliva. Towards universal GAN image detection. In *IEEE International Conference on Visual Communications and Image Processing (VCIP)*, 2021. 2
- [7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3, 12, 20
- [8] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P. Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022. 7
- [9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. 2
- [10] Hany Farid. Lighting (in)consistency of paint by text. *arXiv Preprint*, 2022. 2
- [11] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 8
- [12] Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. Leveraging frequency analysis for deep fake image recognition. In *International Conference on Machine Learning (ICML)*, 2020. 2, 7
- [13] D. Gragnaniello, D. Cozzolino, F. Marra, G. Poggi, and L. Verdoliva. Are GAN generated images easy to detect? A critical analysis of the state-of-the-art. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2021. 2, 6, 7, 12, 17
- [14] Hui Guo, Shu Hu, Xin Wang, Ming-Ching Chang, and Siwei Lyu. Eyes tell all: Irregular pupil shapes reveal GAN-generated faces. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022. 2
- [15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3
- [16] Shu Hu, Yuezun Li, and Siwei Lyu. Exposing GAN-Generated faces using inconsistent corneal specular highlights. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021. 2
- [17] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv Preprint*, 2016. 3, 7
- [18] Midjourney Inc. Midjourney. <https://www.midjourney.com>, 2023. 1, 3, 4, 12
- [19] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 4
- [20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 3, 7
- [21] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 4, 8
- [22] Zhenzhen Liu, Jin Peng Zhou, Yufan Wang, and Kilian Q. Weinberger. Unsupervised out-of-distribution detection with diffusion inpainting. In *International Conference on Machine Learning (ICML)*, 2023. 2
- [23] RuiPeng Ma, Jinhao Duan, Fei Kong, Xiaoshuang Shi, and Kaidi Xu. Exposing the fake: Effective diffusion-generated images detection. In *Workshop on New Frontiers in Adversarial Machine Learning (AdvML)*, 2023. 2, 3, 5, 6, 7, 12
- [24] Falko Matern, Christian Riess, and Marc Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, 2019. 2
- [25] Arian Mousakhan, Thomas Brox, and Jawad Tayyub. Anomaly detection with conditioned denoising diffusion models. *arXiv Preprint*, 2023. 2
- [26] Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 4, 5, 6, 7, 12, 17
- [27] Cecilia Pasquini, Francesco Laiti, Davide Lobba, Giovanni Ambrosi, Giulia Boato, and Francesco De Natale. Identifying synthetic faces through GAN inversion and biometric traits analysis. *Applied Sciences*, 2023. 2
- [28] Dylan Patel and Ahmad Afzal. Google "We have no moat, and neither does OpenAI". *SemiAnalysis*, 2023. 8
- [29] pharmapsychotic. Clip-interrogator. <https://github.com/pharmapsychotic/clip-interrogator>, 2023. 4
- [30] 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 *International Conference on Machine Learning (ICML)*, 2021. 2, 4, 12
- [31] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *arXiv Preprint*, 2022. 4
- [32] Anton Razzhigaev, Arseniy Shakhmatov, Anastasia Maltseva, Vladimir Arkhipkin, Igor Pavlov, Ilya Ryabov, Angelina Kuts, Alexander Panchenko, Andrey Kuznetsov, and Denis Dimitrov. Kandinsky: An improved text-to-image synthesis with image prior and latent diffusion. *arXiv Preprint*, 2023. 1, 4, 12
- [33] Jonas Ricker, Simon Damm, Thorsten Holz, and Asja Fischer. Towards the detection of diffusion model deepfakes. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2024. 2
- [34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3
- [35] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015. 3
- [36] Kevin Roose. An A.I.-generated picture won an art prize. Artists aren't happy. *The New York Times*, 2022. 1
- [37] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Niessner. FaceForensics++: Learning to detect manipulated facial images. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2
- [38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 2015. 3
- [39] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 4, 12
- [40] Katja Schwarz, Yiyi Liao, and Andreas Geiger. On the frequency bias of generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2
- [41] 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 *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2023. 2
- [42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 3, 7
- [43] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015. 3
- [44] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2022. 2, 3, 8, 20
- [45] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3
- [46] Anil Osman Tur, Nicola Dall'Asen, Cigdem Beyan, and Elisa Ricci. Exploring diffusion models for unsupervised video anomaly detection. In *IEEE International Conference on Image Processing (ICIP)*, 2023. 2
- [47] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 4
- [48] James Vincent. The swagged-out pope is an AI fake — and an early glimpse of a new reality. *The Verge*, 2023. 1
- [49] Gian Volpicelli. AI and the end of photographic truth. *POLITICO*, 2023. 1
- [50] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A. Efros. CNN-generated images are surprisingly easy to spot... for now. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 4
- [51] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *Asilomar Conference on Signals, Systems & Computers*, 2003. 3, 7
- [52] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing (TIP)*, 2004. 3, 7
- [53] Zhendong Wang, Jianmin Bao, Wengang Zhou, Weilun Wang, Hezhen Hu, Hong Chen, and Houqiang Li. DIRE for diffusion-generated image detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2, 3, 4, 5, 6, 7, 8, 12, 20, 21
- [54] Julia Wolleb, Florentin Bieder, Robin Sandkühler, and Philippe C. Cattin. Diffusion models for medical anomaly detection. In *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2022. 2
- [55] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN inversion: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2023. 2
- [56] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 2023. 3
- [57] Honghai Yu and Stefan Winkler. Image complexity and spatial information. In *International Workshop on Quality Multimedia Experience (QoMEX)*, 2013. 6

[58] Ning Yu, Vladislav Skripniuk, Sahar Abdelnabi, and Mario Fritz. Artificial fingerprinting for generative models: Rooting deepfake attribution in training data. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

8

[59] Ning Yu, Vladislav Skripniuk, Dingfan Chen, Larry S. Davis, and Mario Fritz. Responsible disclosure of generative models using scalable fingerprinting. In *International Conference on Learning Representations (ICLR)*, 2022. 7, 8

[60] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3

AEROBLADE: Training-Free Detection of Latent Diffusion Images Using Autoencoder Reconstruction Error

Supplementary Material

7. Implementation Details

7.1. Data Collection

Real To obtain real images of high visual quality, we resort to LAION-Aesthetics², which defines several subsets from LAION-5B [39] based on image aesthetics. The authors train a linear model on top of CLIP [30] features to predict the aesthetics of an image, which is then used to create multiple collections. For our experiments we choose images with an aesthetics score of 6.5 or higher. Since the images come in various resolutions, we only use images whose smaller side has at least 512 pixels but whose total number of pixels is less or equal to 768². We then take the center crop of size 512×512 , ensuring that crops contain meaningful content while avoiding resizing operations, which could potentially distort the results.

Stable Diffusion [3] and Kandinsky [32] We use the Diffusers³ library to generate images using the prompts extracted from real images. All images are generated with the default settings and have size 512×512 . We use the same library to compute the reconstructions based on the AEs.

Midjourney [18] We take images generated by Midjourney from a dataset available on Kaggle⁴. It contains the URLs to images from the official Midjourney Discord server, together with some metadata, including the used version. We filter the dataset by version (v4, v5, and v5.1) and only select images which have size 1024×1024 .

7.2. Baselines

Gragnaniello et al. [13] and Corvi et al. [5] We use the code and model checkpoints from the official repository⁵ provided by Corvi et al. [5], which also contains the detector from Gragnaniello et al. [13].

Ojha et al. [26] We use the code and model checkpoints from the official repository⁶. According to the code, a center crop of size 224×224 is used as input. Since the authors evaluate their method on smaller images (256×256) than we do, we also try resizing images before cropping. However, this does not significantly alter the results.

DIRE [53] We use the code and model checkpoints from the official repository⁷. In particular, we compute the DIRE representations using ADM [7] trained on LSUN Bedroom and classify with the corresponding detector. This setting corresponds to the setting in Table 3 in the original paper [53], where the authors report good generalization to Stable Diffusion 1.5.

SeDID_{Stat} [23] At the time of writing, no code is publicly available, which is why we reimplement the method based on the authors' definitions. To guide the denoising process we use the same technique as in our experiment with deeper reconstructions (see Sec. 5.5). We experiment with different values for the total number of steps and T_{SE} and observe that 50 steps in total and $T_{SE} = 25$ achieves the best results.

²<https://laion.ai/blog/laion-aesthetics>

³<https://github.com/huggingface/diffusers>

⁴<https://kaggle.com/datasets/iraklip/modjourney-v51-cleaned-data>

⁵<https://github.com/grip-unina/DMimageDetection>

⁶<https://github.com/Yuheng-Li/UniversalFakeDetect>

⁷<https://github.com/ZhendongWang6/DIRE>

8. Additional Results

8.1. Reconstruction Error Histograms

Fig. 9 is the extended version of Fig. 3 that includes the other variants of LPIPS. For LPIPS_2 the distributions of reconstruction errors for real and generated samples are most separated. This confirms our results in Tab. 1, according to which LPIPS_2 achieves the highest APs. We observe that for LPIPS_1 , the distributions appear to be shifted to the left, indicating that the reconstruction errors are lower for both real and generated images. For higher layers, the results suggest that the error becomes lower for real images but higher for generated images, making the distributions less separable.

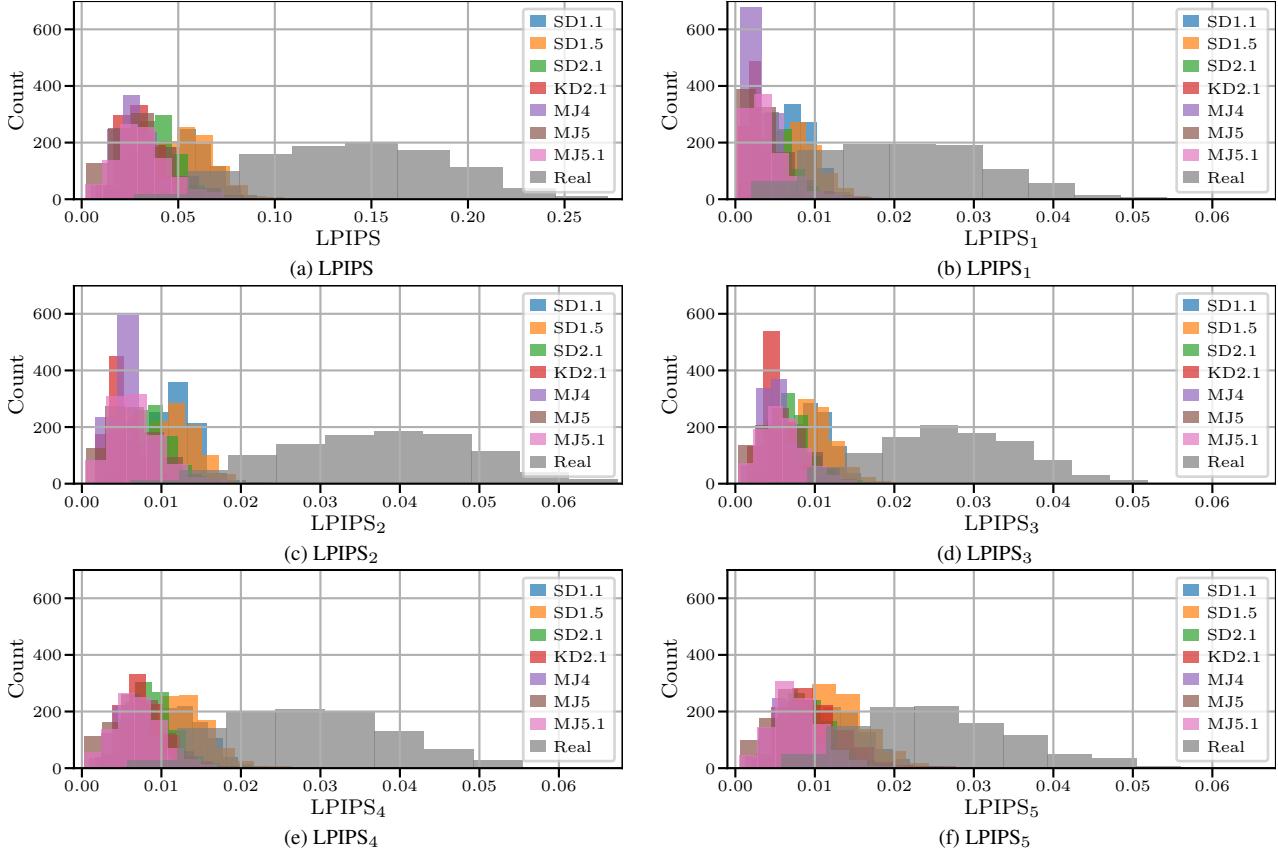


Figure 9. Distributions of reconstruction errors Δ_{\min} using different LPIPS variants. (c) is identical to Fig. 3 and is included here for completeness. The x-axis in (a) differs because LPIPS is defined as the sum of all layers.

8.2. Attribution Based on Minimal Reconstruction Error

Tab. 5 is the extended version of Tab. 2 that includes all variants of LPIPS. We observe that for LPIPS, LPIPS₁, and LPIPS₂, almost all images are correctly attributed (based on the minimal reconstruction error) to the AE that actually generated them. While this is technically not the case for Midjourney, since the AE is not publicly available, the results strongly indicate that the AE is similar to that of Stable Diffusion 2. Towards higher layers, especially images from Stable Diffusion 1.1 and 1.5 tend to be attributed to the AE of Stable Diffusion 2.

Distance	AE	SD1.1	SD1.5	SD2.1	KD2.1	MJ4	MJ5	MJ5.1
LPIPS	SD1	1.000	0.999	0.000	0.000	0.000	0.000	0.000
	SD2	0.000	0.001	1.000	0.000	0.999	0.993	0.997
	KD2.1	0.000	0.000	0.000	1.000	0.001	0.007	0.003
LPIPS ₁	SD1	1.000	0.999	0.000	0.000	0.000	0.000	0.000
	SD2	0.000	0.001	1.000	0.000	0.999	0.999	0.999
	KD2.1	0.000	0.000	0.000	1.000	0.001	0.001	0.001
LPIPS ₂	SD1	1.000	1.000	0.000	0.000	0.000	0.000	0.000
	SD2	0.000	0.000	1.000	0.000	0.999	0.997	0.995
	KD2.1	0.000	0.000	0.000	1.000	0.001	0.003	0.005
LPIPS ₃	SD1	0.998	0.995	0.000	0.000	0.000	0.001	0.000
	SD2	0.002	0.005	1.000	0.000	0.997	0.991	0.995
	KD2.1	0.000	0.000	0.000	1.000	0.003	0.008	0.005
LPIPS ₄	SD1	0.975	0.954	0.000	0.000	0.000	0.000	0.000
	SD2	0.025	0.046	0.998	0.003	0.998	0.982	0.993
	KD2.1	0.000	0.000	0.002	0.997	0.002	0.018	0.007
LPIPS ₅	SD1	0.835	0.772	0.001	0.001	0.000	0.000	0.000
	SD2	0.162	0.228	0.995	0.013	0.997	0.977	0.990
	KD2.1	0.003	0.000	0.004	0.986	0.003	0.023	0.010

Table 5. Fraction of samples for which an AE has the smallest reconstruction error using different LPIPS variants. The highest fraction for each dataset is highlighted in **bold**. The results for LPIPS₂ are identical to Tab. 2 and are included here for completeness.

8.3. Reconstruction Error Against Complexity

Fig. 10 is the extended version of Fig. 5 which contains plots for individual datasets. We find that the relation between complexity and reconstruction error is relatively similar for different generative models, with the variants of Stable Diffusion and Kandinsky yielding more compact distributions than Midjourney.

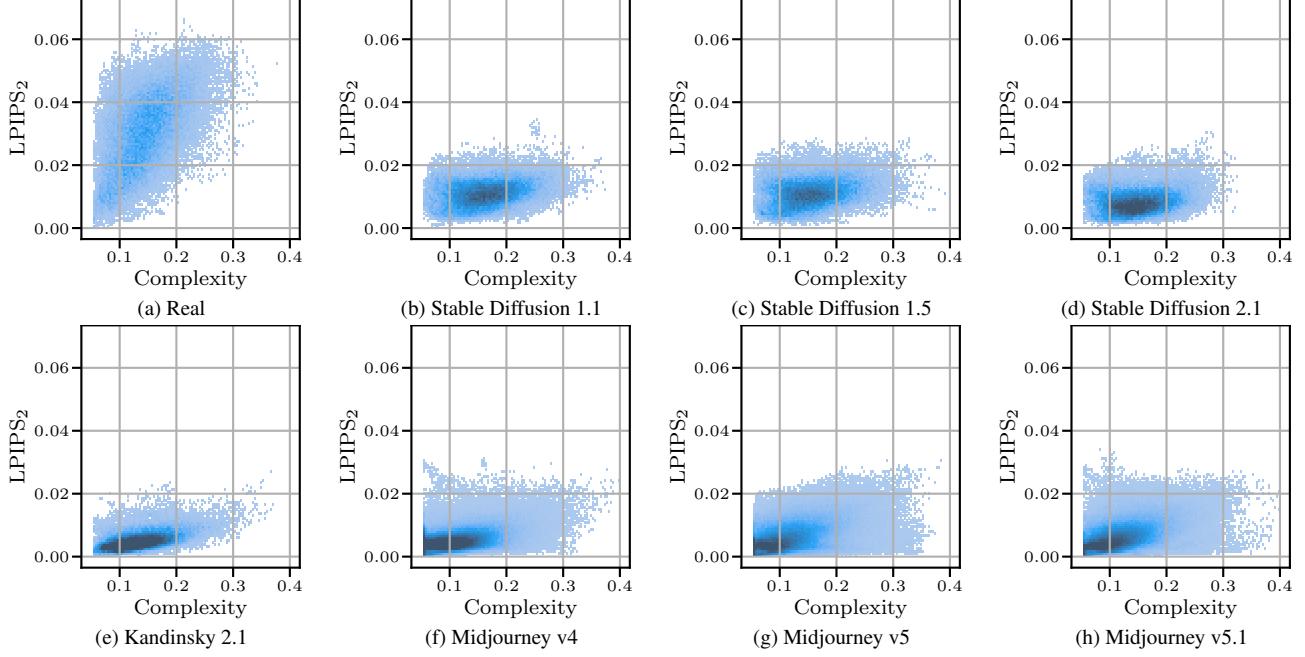


Figure 10. Reconstruction error against complexity for different datasets. (a) is identical to Fig. 5a and is included here for completeness. The color map is clipped at 1 000 samples for better visibility.

8.4. Inpainting Localization

In Fig. 11 we provide additional examples for reconstruction maps of real images with inpainted regions (corresponding to Fig. 6). Across different scenes, the reconstruction error provides a good indication of where the inpainted region is located.



Figure 11. Additional examples illustrating the localization of inpainted regions using the reconstruction error. We show the images inpainted with Stable Diffusion 1.5 (top), the masks used for inpainting (center) and the reconstruction error maps (bottom), computed with LPIPS₂ and the AE from Stable Diffusion 1.5.

8.5. Robustness to Perturbations

In Fig. 12 we first provide examples that illustrate how the perturbed images evaluated in Sec. 5.5 look like. Note that for JPEG and cropping, a smaller value (q or f) leads to a stronger perturbation, while for the addition of noise and blurring a larger value (σ) has a stronger effect. For blurring we use a kernel size of 9.

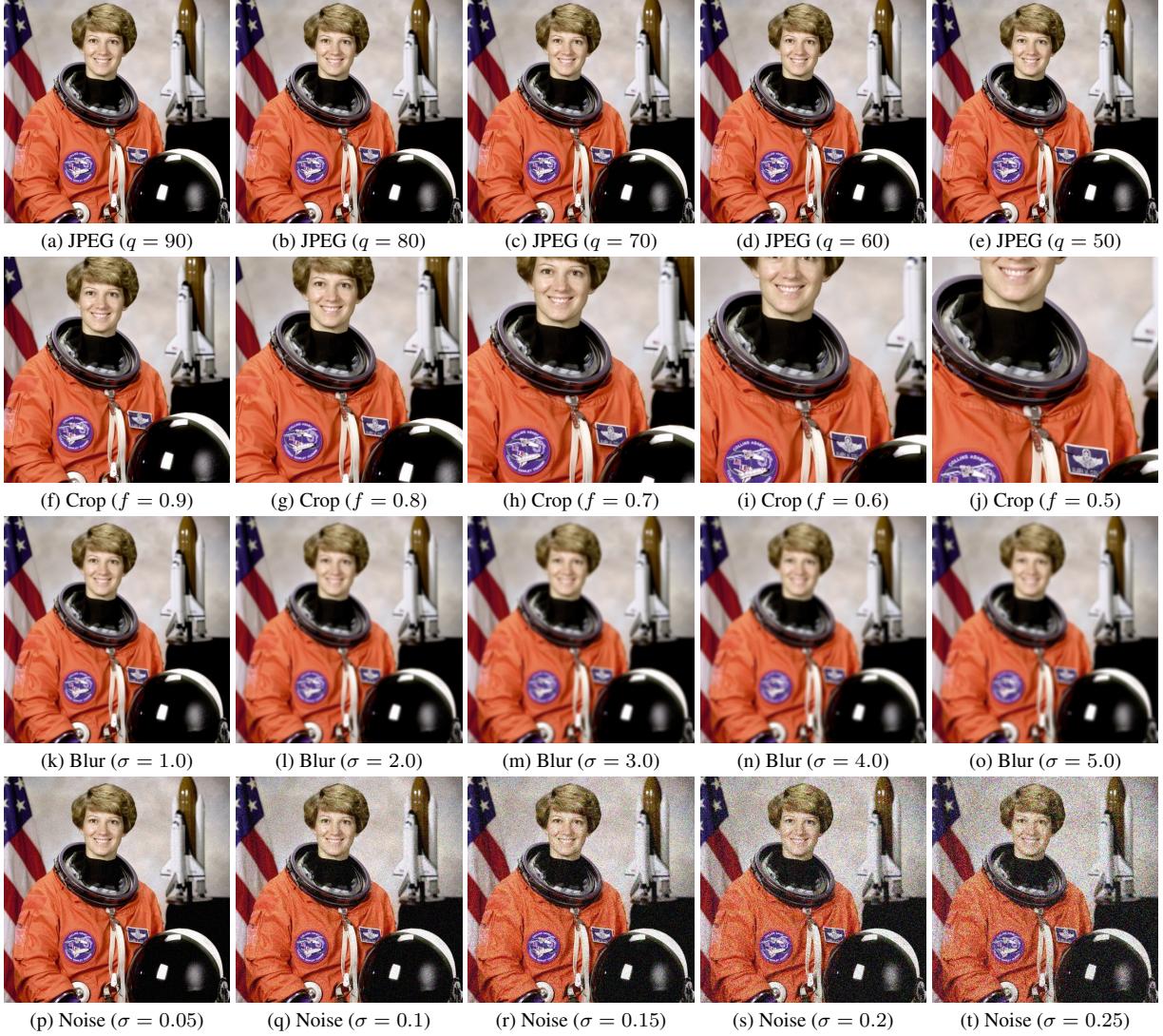


Figure 12. Visualization of different perturbations. In (a)-(e), q denotes the JPEG quality factor. In (f)-(j), f denotes the crop factor. In (k)-(t), σ denotes the standard deviation of the Gaussian blur and noise, respectively.

In addition to Fig. 7, we provide the results on individual datasets for the baselines (see Figs. 13 to 15) as well as all variants of LPIPS (see Figs. 16 to 21). We observe that in some settings, higher layers perform better than LPIPS₂. We suppose that higher layers are less affected by the loss of details caused by the perturbations. This is illustrated by an alternative version of Fig. 7 where we select the optimal LPIPS layer for each perturbation (see Fig. 22). While this of course gives an unfair advantage, it shows the potential robustness of AEROBLADE.

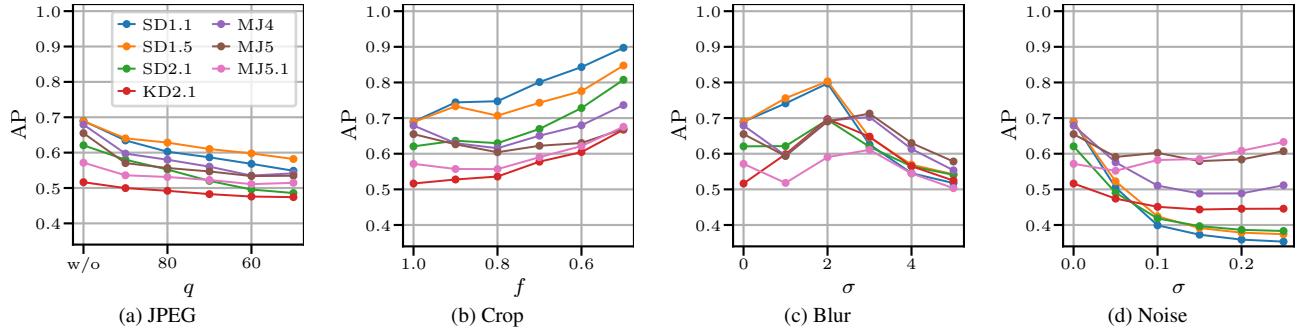


Figure 13. Detection performance of the detector from Gragnaniello et al. [13] on perturbed images, measured in AP.

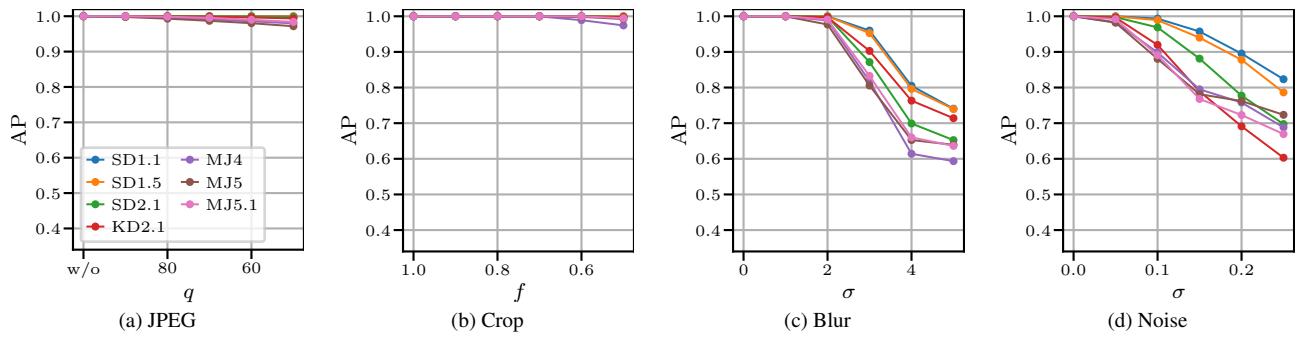


Figure 14. Detection performance of the detector from Corvi et al. [5] on perturbed images, measured in AP.

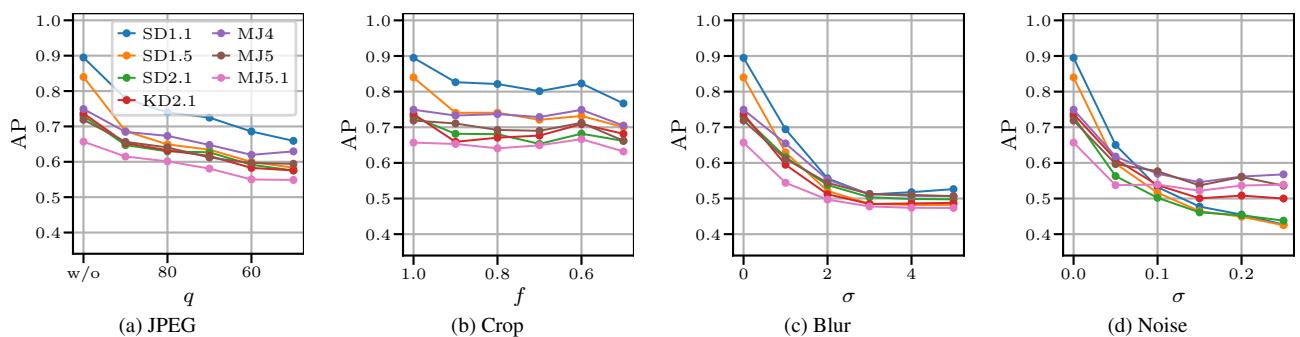


Figure 15. Detection performance of the detector from Ojha et al. [26] on perturbed images, measured in AP.

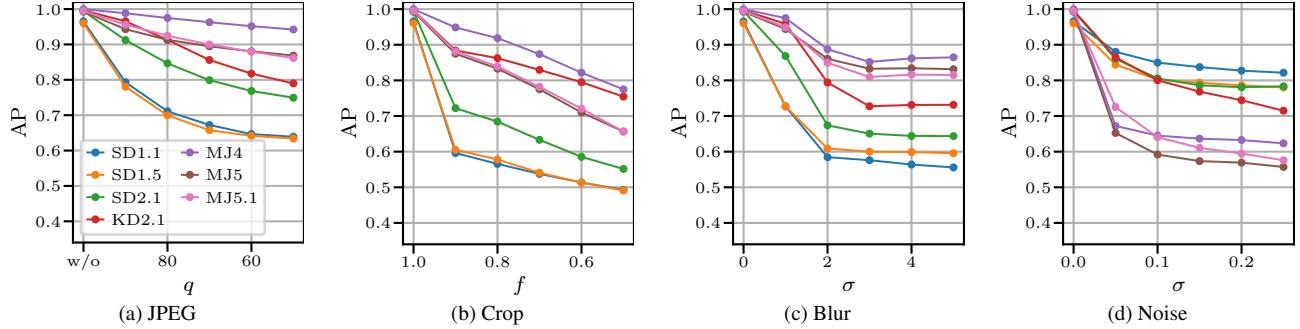


Figure 16. Detection performance of AEROBLADE (with LPIPS and Δ_{Min}) on perturbed images, measured in AP.

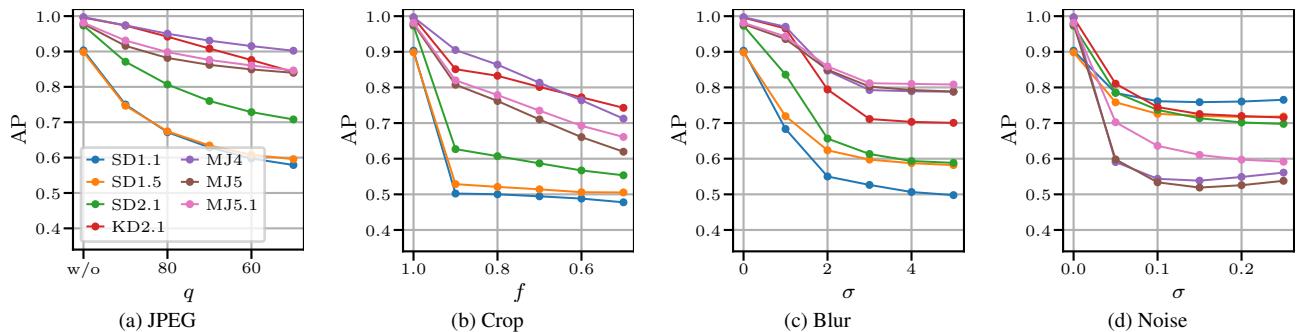


Figure 17. Detection performance of AEROBLADE (with LPIPS₁ and Δ_{Min}) on perturbed images, measured in AP.

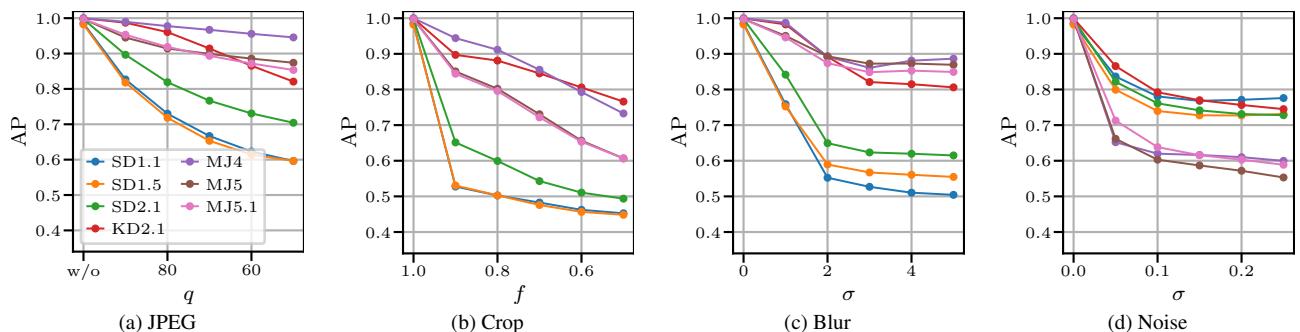


Figure 18. Detection performance of AEROBLADE (with LPIPS₂ and Δ_{Min}) on perturbed images, measured in AP.

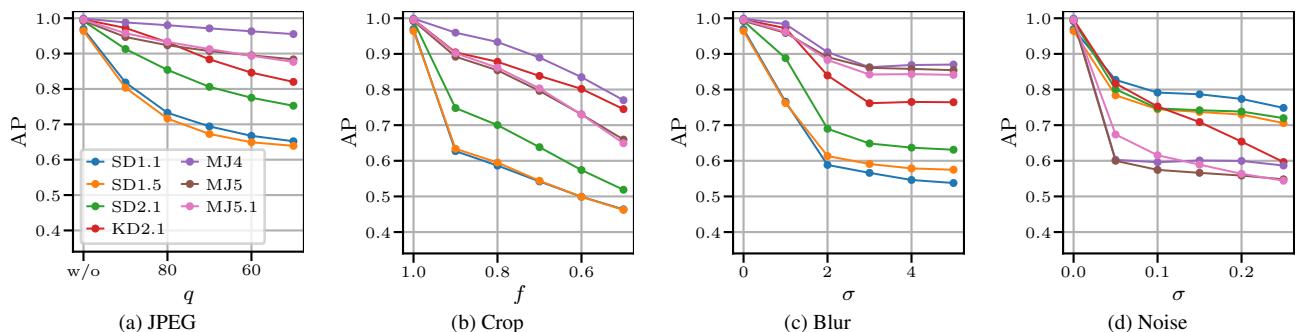


Figure 19. Detection performance of AEROBLADE (with LPIPS₃ and Δ_{Min}) on perturbed images, measured in AP.

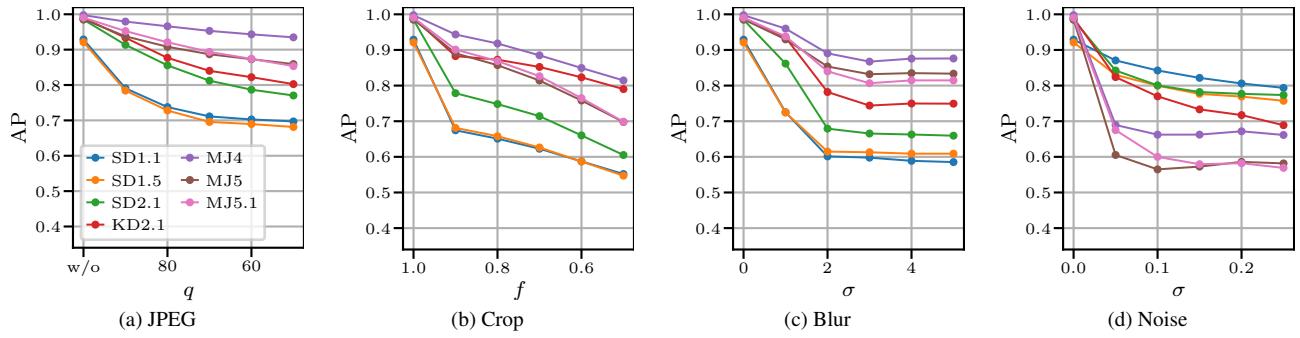


Figure 20. Detection performance of AEROBLADE (with LPIPS_4 and Δ_{Min}) on perturbed images, measured in AP.

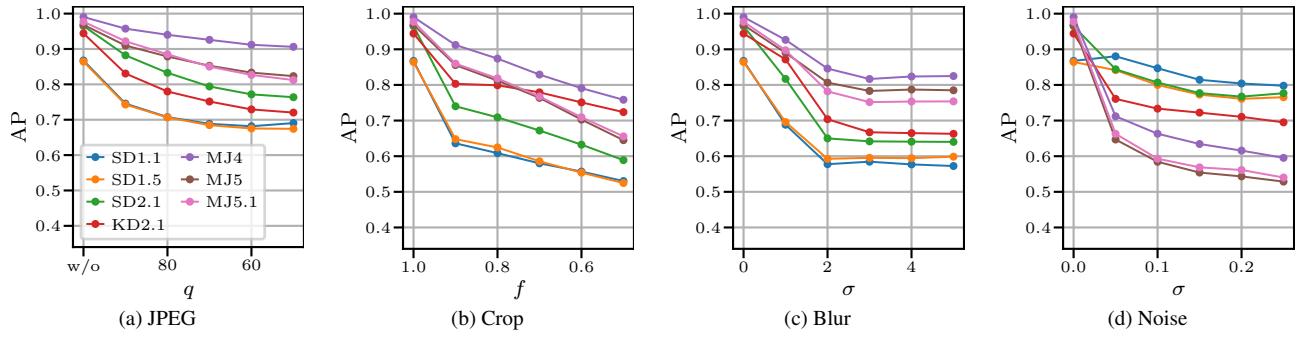


Figure 21. Detection performance of AEROBLADE (with LPIPS_5 and Δ_{Min}) on perturbed images, measured in AP.

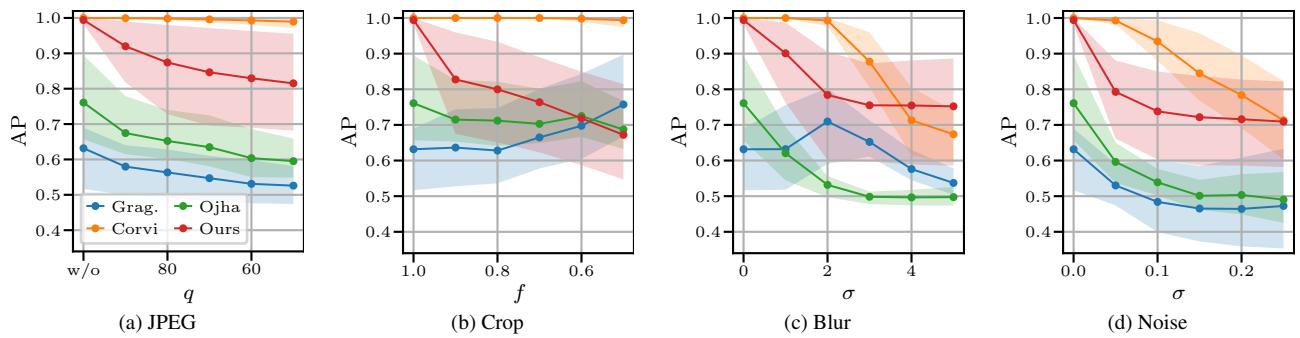


Figure 22. Detection performance of AEROBLADE (with optimal LPIPS variant for each setting and Δ_{Min}) and baselines on perturbed images, measured in AP. Results are averaged over all datasets, with shaded areas indicating the minimum and maximum.

9. Analysis of DIRE

While trying to reproduce the results from Wang et al. [53], we found that the experiments conducted by the authors might have been affected by an unwanted bias in the data, as we briefly mention in Sec. 5.3. In this section, we provide details on the experiments and results that led us to our findings.

9.1. DIRE

Before proceeding, we briefly summarize the approach proposed by Wang et al. [53]. The idea behind DIRE is that a diffusion model should be able to reconstruct an image more accurately (compared to the original image), if that image is generated by a diffusion model, as opposed to being a real image. In their analysis, the authors use a diffusion model (ADM [7] in most of their experiments) to invert a given image x_0 to its initial time step x_T (corresponding to Gaussian noise) using the inverted DDIM sampler [44], and then denoise back to x'_0 using DDIM as usual. The authors hypothesize that the pixel-wise difference between x_0 and x'_0 , the DIRE representation, has different characteristics for real and generated images. To exploit this for classification, they train a binary classifier on the pixel-wise reconstruction error images $|x_0 - x'_0|$ (from real images and images generated using some diffusion model). The authors provide code, data and pre-trained models⁸, which we use according to the official instructions in the following experiments.

In the following tables, we test different pre-trained classifiers, which all use the ResNet-50 architecture but are trained on different datasets. These classifiers are ADM-B (trained on LSUN Bedroom vs images generated by ADM trained on LSUN Bedroom), ADM-IN (trained on ImageNet vs images generated by ADM trained on ImageNet), PNDM-B, and IDDPB-B. Further, when referring to datasets (such as in the second and third column in the tables), LSUN-B stands for LSUN Bedroom and ADM-B and ADM-IN stand for images generated using ADM trained on LSUN Bedroom and ImageNet, respectively.

9.2. Experiment 1

We first experiment with the LSUN Bedroom test set provided by the authors. We obtain the images by downloading and extracting `adm.tar.gz` (images generated by ADM trained on LSUN Bedroom) and `real.tar.gz` (real LSUN Bedroom images) from `dire/test/lsun_bedroom/lsun_bedroom`. It should be noted that the authors provide original images, reconstructions, and the DIRE representations (i.e., the differences between originals and reconstructions). The script `test.py`, which is used to evaluate the classification performance, directly uses the DIRE representations.

The key observation we make is that the DIREs of fake images are stored as PNG files (which uses lossless compression), while the DIREs of real images are stored as JPEG files (compressed with quality factor 95). We suspect the reason for this is in `guided_diffusion/compute_dire.py`, where the DIRE files inherit the extension of the corresponding original input image file and the image library automatically applies the corresponding compression. Since the original real images were provided in JPEG, their DIRE images are also automatically stored with JPEG compression. We stress that the authors could not have avoided the compression of the original real images, since the dataset simply comes in this format. However, saving the DIRE in the lossy JPEG format introduces an unwanted bias, as our experiments show. A possible solution to this would have been to store the generated images using the same JPEG compression as the original real images (since uncompressed real images are not available) before computing their DIREs.

Variant 1 We first run `test.py` according to the authors' instructions given in the repository. We compare the performance of different pre-trained classifiers and report the results in Tab. 6a. The "Classifier" column denotes on which kind of generated images (together with the corresponding real images) the classifier was trained on. In this setting, we are able to reproduce the authors' results: all classifiers achieve very good detection and even generalize to other datasets.

Variant 2 We repeat the previous experiment, but this time we ensure that the DIREs of real and generated images are saved consistently. Since we do not have access to real DIREs in their uncompressed form, we instead convert the fake DIREs to JPEG with the same quality level (95) as the real DIREs provided by the authors.

As the results in Tab. 6b show, the performance of all classifiers drops from almost perfect detection (see Tab. 6a) to random guessing. In particular, all images are classified as being real. We emphasize that the only change we made (compared to the previous evaluation) was to convert the uncompressed DIREs of generated images to the same format as the DIREs of real images.

⁸<https://github.com/ZhendongWang6/DIRE>

Classifier	Real (Test)	Fake (Test)	Acc	AP	Acc_R	Acc_F
ADM-B	LSUN-B	ADM-B	1.000	1.000	1.000	1.000
PNDM-B	LSUN-B	ADM-B	1.000	1.000	1.000	1.000
IDDPM-B	LSUN-B	ADM-B	0.996	1.000	1.000	0.991
ADM-IN	ImageNet	ADM-IN	0.999	1.000	1.000	0.998
ADM-B	ImageNet	ADM-IN	0.998	1.000	0.999	0.999

(a) real DIREs: JPEG, fake DIREs: PNG

Classifier	Real (Test)	Fake (Test)	Acc	AP	Acc_R	Acc_F
ADM-B	LSUN-B	ADM-B	0.501	0.660	1.000	0.200
PNDM-B	LSUN-B	ADM-B	0.500	0.660	1.000	0.000
IDDPM-B	LSUN-B	ADM-B	0.502	0.670	1.000	0.400
ADM-IN	ImageNet	ADM-IN	0.500	0.670	1.000	0.100
ADM-B	ImageNet	ADM-IN	0.500	0.520	0.999	0.000

(b) real DIREs: JPEG, fake DIREs: JPEG

Table 6. Performance of different pre-trained classifiers on test data provided by the authors. The only difference between (a) and (b) is that in (b) the reconstructions of fake images are stored in the JPEG format (quality level 95), like the real images.

9.3. Experiment 2

In a second experiment we further demonstrate that the format in which the DIREs are saved in has a significant influence on the detection. We take 1 000 DIREs of images generated by ADM (trained on LSUN Bedroom) (`dire/test/lsun_bedroom/lsun_bedroom/adm.tar.gz`) and split them in two sets A and B, each containing 500 images.

Variant 1 All DIREs are stored in the lossless PNG format. We pretend that set A is actually a collection of real images, while set B is considered to be fake. Again, we use `test.py` provided by the authors to evaluate the detection performance. The results in Tab. 7a are as expected, since all images are actually generated and half of them have the wrong label, the classifiers achieve an accuracy of approximately 0.5.

Variant 2 We repeat the previous experiment, but convert all DIREs in set A (which we label as being real) to JPEG with quality 95. All other parameters remain unchanged. The results in Tab. 7b show that set A (containing DIREs from generated images saved as JPEGs) is now classified as being real. Thus, the almost perfect classification accuracy (compared to Tab. 7a) is caused exclusively by the fact that we converted the DIREs in set A to the JPEG format.

Classifier	Real* (Test)	Fake (Test)	Acc	AP	Acc_R	Acc_F
ADM-B	ADM-B	ADM-B	0.500	0.500	0.000	1.000
PNDM-B	ADM-B	ADM-B	0.500	0.490	0.000	1.000
IDDPM-B	ADM-B	ADM-B	0.503	0.490	0.012	0.994

(a) real* DIREs: PNG, fake DIREs: PNG

Classifier	Real* (Test)	Fake (Test)	Acc	AP	Acc_R	Acc_F
ADM-B	ADM-B	ADM-B	0.999	1.000	0.998	1.000
PNDM-B	ADM-B	ADM-B	0.997	1.000	0.994	1.000
IDDPM-B	ADM-B	ADM-B	0.997	1.000	1.000	0.994

(b) real* DIREs: JPEG, fake DIREs: PNG

Table 7. Performance of different pre-trained classifiers on test data provided by the authors. “Real*” indicates that the images are labeled as being real but are actually generated by ADM. The only difference between (a) and (b) is that in (b) the reconstructions of images that are labeled as being real are stored in the JPEG format (quality level 95).

9.4. Discussion

In both experiments we show that the classifiers provided by Wang et al. [53] suffer from an unwanted bias in the training data. The DIREs of real images were saved as JPEGs, while those of generated images were saved as lossless PNGs. As a consequence, the detector is highly sensitive to the presence of compression artifacts. In other words, the format in which the DIREs are saved controls whether it is classified as being real or fake, not whether the image is actually real or fake. Based on our findings, we believe that the authors’ experiments should be re-evaluated without the bias caused by the inconsistent file formats.